1. (25 points) **Linear algebra refresher.**

   (a) (12 points) Let **Q** be a real orthogonal matrix.

   i. (3 points) Show that $\mathbf{Q^T}$ and $\mathbf{Q^{-1}}$ are also orthogonal.

   ii. (3 points) Show that **Q** has eigenvalues with norm 1.

   iii. (3 points) Show that the determinant of **Q** is either +1 or -1.

   iv. (3 points) Show that **Q** defines a length preserving transformation.

i. For an orthogonal matrix Q, $Q \cdot Q^T = I$

Q^T is an orthogonal matrix since $Q^T(Q^T)^T = Q^TQ = I$

$(Q^T)^TQ^T = QQ^T = I$ ✓

Q^{-1} is orthogonal because $Q^{-1} = Q^T$ since $Q \cdot Q^T = I$,
and multiplying a matrix by its inverse is the identity matrix

$Q^{-1}(Q^{-1})^T = Q^{-1}(Q^T)^{-1} = (Q^TQ)^{-1} = I^{-1} = I$ ✓

ii. λ is the eigenvalue of Q, $Qx = \lambda x$

$(Qx)^T(Qx) = (\lambda x)^T(\lambda x)$

$x^TQ^TQx = |\lambda|^2 x^Tx$

$\therefore |\lambda|^2 = 1$ since $Q^TQ = I$

Thus Q has eigenvalues of norm 1

iii. $\det(Q) = \pm 1$

Q is orthogonal

$\det(I) = 1$

$\rightarrow \det(Q^T \cdot Q) = 1$

$= \det(Q^T) \cdot \det(Q)$

$1 = (\det(Q))^2$ $\quad \det(Q) = \det(Q^T)$

$\therefore$ $\det(Q) = \pm 1$

iv. Q preserves lengths $\rightarrow \|Qv\|^2 = \|v\|^2$

$\|Qv\|^2 = Qv \cdot Qv = (Qv)^TQv = v^TQ^TQv = v^TIv = v \cdot v = \|v\|^2$

(b) (8 points) Let $\mathbf{A}$ be a matrix.

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

i. $A = U\Sigma V^T$

$AA^T = (U\Sigma V^T)(U\Sigma V^T)^T$
$= U\Sigma V^T(V\Sigma^T U^T)$
$= U\Sigma\Sigma^T U^T$

$A^TA = (U\Sigma V^T)^T(U\Sigma V^T)$
$= V\Sigma^T U^T U\Sigma V^T$
$A^TA = V\Sigma^T\Sigma V^T$

The columns of U are the left singular vectors of A and they are the orthonormal eigenvectors of AA^T

The columns of V are the right singular vectors of A and they are the orthonormal eigenvectors of A^TA

ii. The singular values of A are the square roots of the eigenvalues of A^TA and AA^T

$$\sigma_i(A) = \lambda_i^{1/2}(A^TA) = \lambda_i^{1/2}(AA^T)$$

(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 $\mathbf{A}$ is an eigenvector.

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

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

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

i. False. should be at most n distinct eigenvalues

$I_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ but all eigenvalues are 1 (only 1 distinct eigenvalue)

ii False, only if they have the same eigenvalue

Example $A = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$ $\lambda_1 = 1$, $\lambda_2 = -1$

$Av = \lambda_1 v_1$ $Av_2 = \lambda_2 v_2$ $v_1 + v_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$, which is not an eigenvector of A

$Av_1 = v_1$ $Av_2 = -v_2$

$v_1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ $v_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$

iii $x^TAx \geq 0$ for all x, then eigenvalues are nonnegative

True

$Ax = \lambda x$

$x^TAx = x^T\lambda x$

$x^TAx = x^Tx\lambda \geq 0$

x^Tx is positive, so λ must be positive or 0

iv rank(A) can exceed number of distinct nonzero λ

True the number of nonzero eigenvalues must be $\leq$ rank

v. True

2. (25 points) **Probability refresher.**

(a) (5 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. (1 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. (2 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. (2 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?

a) Event T is the coin lands tails, H for heads

Two coins H50, H60

$P(H|H50) = 0.5$

$P(H|H60) = 0.6$

$P(T) = P(T|H50)P(H50) + P(T|H60)P(H60)$

$= 0.5 \cdot 0.5 + 0.4 \cdot 0.5 = 0.45$

i. $P(H50|T) = ?$

$= \frac{P(T|H50)P(H50)}{P(T)}$

$= \frac{0.5 \cdot 0.5}{0.45} = \boxed{5/9}$

ii $P(H50|THHH) = \frac{P(THHH|H50)P(H50)}{P(THHH)}$

$= \frac{(0.5)^4 \cdot 0.5}{0.5(0.5)^4 + 0.5(0.4 \cdot 0.6^3)}$

$\approx \boxed{0.4197}$

iii $P(H50 \mid 9H, 1T) = ?$

$$= \frac{P(9H, 1T \mid H50) \cdot P(H50)}{P(9H, 1T)}$$

$$P(9H, 1T) = P(9H, 1T \mid H50)\,P(H50) + P(9H, 1T \mid H55)\,P(H55) + P(9H, 1T \mid H60)\,P(H60)$$

$$= 0.5^{10} \cdot \tfrac{1}{3} + 0.55^{9} \cdot 0.45 \cdot \tfrac{1}{3} + 0.6^{9} \cdot 0.4 \cdot \tfrac{1}{3}$$

$$\approx 0.00236$$

$$P(H50 \mid 9H, 1T) = \frac{0.5^{10} \cdot \frac{1}{3}}{P(9H, 1T)} \approx \boxed{0.1379}$$

$$P(H55 \mid 9H, 1T) = \frac{0.55^{9} \cdot 0.45 \cdot \frac{1}{3}}{P(9H, 1T)} \approx \boxed{0.2927}$$

$$P(H60 \mid 9H, 1T) = \frac{0.6^{9} \cdot 0.4 \cdot \frac{1}{3}}{P(9H, 1T)} = \boxed{0.5694}$$

(b) (5 points) Students at UCLA are from these disciplines: 15% Science, 21% Healthcare, 24% Liberal Arts, and 40% Engineering. (Each student belongs to a unique discipline.) The students attend a lecture and give feedback. Suppose 90% of the Science students liked the lecture, 18% of the Healthcare students liked it, none of the Liberal Arts students liked it, and 10% of the Engineering students liked it. If a student is randomly chosen, and the student liked the lecture, what is the conditional probability that the student is from Science?

Event L = liked the lecture

$$P(\text{Science}|L) = \frac{P(L|\text{Science})\,P(\text{Science})}{P(L)} = ?$$

S

$P(\text{Science}) = 0.15$

$P(L|\text{Science}) = 0.9$

$$= \frac{0.9 \cdot 0.15}{0.2128} = \boxed{0.6344}$$

$$P(L) = P(L|S)\,P(S) + P(L|H)\,P(H) + P(L|LA)\,P(LA) + P(L|E)\,P(E)$$

$$= 0.9 \cdot 0.15 + 0.18 \cdot 0.21 + 0 + 0.1 \cdot 0.4$$

$$= 0.2128$$

(c) (5 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; give an explanation of the result that you find.

"1" is positive, "0" is negative test

P is pregnant

$$P(P|1) = \frac{P(1|P)P(P)}{P(1)} = \frac{0.99 \cdot 0.01}{P(1)} = 0.09$$

$P(P|1) =$ 0.09

$$P(1) = P(1|\text{Not P})\,P(\text{Not P}) + P(1|P)\,P(P)$$
$$= 0.1 \cdot 0.99 + 0.99 \cdot 0.01$$
$$= 0.1089$$

The test sucks (only 9% correct given positive test) There are way many more nonpregnant women than pregnant women (99% not pregnant), and it still fails 10% of the time on nonpregnant women. It fails more on nonpregnant women (10% of 99%) than it gets correct given it is positive (9%)

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

$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$ $\quad E(Ax+b) = ?$

$$E(Ax_i) = E\left(\sum_{j=1}^{n} A_{ij} x_j\right)$$

$$= \sum_{j=1}^{n} A_{ij} E(x)_j$$

$$= [A \cdot E(x)]_i$$

Expectation is linear, so $E(Ax) = AE(x)$

$$\boxed{E(Ax+b) = AE(x)+b}$$

(e) (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?

$$cov(x) = E((x-Ex)(x-Ex)^T)$$

$cov(Ax+b) = ?$

$$= E((Ax+b-E(Ax+b))(Ax+b-E(Ax+b))^T)$$

$$= E((Ax+b-AE(x)-b)(Ax+b-AE(x)-b)^T)$$

$$= E(A(x-E(x))(A(x-E(x)))^T)$$

$$= E(A(x-E(x))((x-E(x))^T A^T)$$

$$= AE((x-E(x))(x-E(x))^T)A^T$$

$$= \boxed{A\,cov(x)\,A^T}$$

3. (10 points) **Multivariate derivatives.**

(a) (1 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) (1 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 $\nabla_{\mathbf{y}} \mathbf{x}^T \mathbf{A} \mathbf{y}$?

(c) (1 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 $\nabla_{\mathbf{A}} \mathbf{x}^T \mathbf{A} \mathbf{y}$?

(d) (1 points) Let $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{A} \in \mathbb{R}^{n \times n}$, and let $f = \mathbf{x}^T \mathbf{A} \mathbf{x} + \mathbf{b}^T \mathbf{x}$. What is $\nabla_{\mathbf{x}} f$?

a) $\nabla_x x^T A y = \frac{\partial x^T A y}{\partial x} = \boxed{Ay}$ ($n \times 1$)

b) $\nabla_y x^T A y = \frac{\partial x^T A y}{\partial y} = (x^T A)^T = \boxed{A^T x}$ ($m \times 1$)

c) $\nabla_A x^T A y = ?$

$x^T A y = \sum_{i=1}^{} \sum_{j=1}^{} x_i y_j A_{ij}$ $\quad \nabla_A \sum_{i=1}^{} \sum_{j=1}^{} x_i y_j A_{ij} = \boxed{x y^T}$

$n \times 1 \cdot 1 \times m$

$n \times m$ ✓

d) $f = x^T A x + b^T x$ From class $\nabla_x x^T A x = (A + A^T) x$

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

(e) (1 points) Let $\mathbf{A} \in \mathbb{R}^{n \times n}$, $\mathbf{B} \in \mathbb{R}^{n \times n}$ and $f = \text{tr}(\mathbf{AB})$. What is $\nabla_{\mathbf{A}} f$?

(f) (2 points) Let $\mathbf{A} \in \mathbb{R}^{n \times n}$, $\mathbf{B} \in \mathbb{R}^{n \times n}$ and $f = \text{tr}(\mathbf{BA} + \mathbf{A}^T\mathbf{B} + \mathbf{A}^2\mathbf{B})$. What is $\nabla_{\mathbf{A}} f$?

e) $A \in R^{n \times n}$, $B \in R^{n \times n}$, $f = tr(AB)$

$\nabla_A f = ?$

$tr(AB) = \sum_{i=1}^{N} (AB)_{ii} = \sum_{i=1}^{N} a_{1i} b_{i1} + \sum_{i=1}^{N} a_{2i} b_{i2} + \ldots$

$\frac{\partial tr(AB)}{\partial a_{ij}} = b_{ji}$, so $\boxed{\nabla_A f = B^T}$ ← matrix cookbook eq. 100

f) $f = tr(BA + A^TB + A^2B)$, $\nabla_A f = ?$

$tr(BA) = \sum_{i=1}^{N} (BA)_{ii} = \sum_{i=1}^{N} b_{1i} a_{i1} + \sum_{i=1}^{N} b_{2i} a_{i2} + \ldots$

$\frac{\partial tr(BA)}{\partial a_{ij}} = b_{ji}$, so $\nabla_A tr(BA) = B^T$ ✓, since $tr(AB) = tr(BA)$

$tr(A^TB) = \sum_{i}^{N} \sum_{j}^{M} a_{ij} b_{ij}$

$\frac{\partial tr(A^TB)}{\partial a_{ij}} = b_{ij} = B$ Matrix cookbook eq 103

$\frac{\partial tr(A^2B)}{\partial A} = (AB + BA)^T$ matrix cookbook equation 107

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

$= tr(BA) + tr(A^TB) + tr(A^2B)$

$\boxed{\nabla_A f = B^T + B + (AB + BA)^T}$

(g) (3 points) Let $\mathbf{A} \in \mathbb{R}^{n \times n}$, $\mathbf{B} \in \mathbb{R}^{n \times n}$ and $f = \|\mathbf{A} + \lambda \mathbf{B}\|_F^2$. What is $\nabla_{\mathbf{A}} f$?

$f = \|A + \lambda B\|_F^2 \quad \nabla_A f = ?$

$f = tr[(A+\lambda B)^T (A+\lambda B)]$

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

Drop terms with no dependence on A

$f = tr(A^TA) + \lambda tr(A^TB) + \lambda tr(B^TA)$

$tr(A^TB) = tr(B^TA)$

$f = tr(A^TA) + 2\lambda tr(A^TB)$

$\nabla_A f = \frac{\partial tr(A^TA)}{\partial A} + 2\lambda \frac{\partial tr(A^TB)}{\partial A}$ matrix cookbook eq 103

$= 2A + 2\lambda B = \boxed{2(A+\lambda B)}$

↑ matrix cookbook eq 115

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 $\mathbf{W}$.

Where $\mathbf{W}$ is a matrix, and for each example in the training set, both $\mathbf{x}^{(i)}$ and $\mathbf{y}^{(i)}$ $\forall i = 1, \ldots, n$ are vectors.
Hint: you may find the following derivatives useful:

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

4. $\min_W \frac{1}{2}\sum_{i=1}^{n} ||y^{(i)} - Wx^{(i)}||^2$ $\qquad \frac{\partial tr(WA)}{\partial W} = A^T$

$W = ?$ $\qquad \frac{\partial tr(WAW^T)}{\partial W} = WA^T + WA$

$||A||^2 = tr(A^TA)$

$f(W) = \frac{1}{2}\sum_{i=1}^{n} tr((y_i - Wx_i)^T(y_i - Wx_i))$ Ignore terms w/ not W

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

$= \frac{1}{2} tr(-2Y^TWX + X^TW^TWX)$

$= -tr(Y^TWX) + \frac{1}{2} tr(X^TW^TWX)$ $\qquad tr(AB) = tr(BA)$

$= -tr(WXY^T) + \frac{1}{2} tr(WXX^TW^T)$

$f(W) = -tr(WXY^T) + \frac{1}{2} tr(WXX^TW^T)$

$\frac{\partial f(W)}{\partial W} = -YX^T + \frac{1}{2}(WXX^T + WXX^T) = 0$ minimize

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

$WXX^T = YX^T$

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

5. (10 points) **Regularized least squares**

In lecture, we worked through the following least squares problem

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

However, the least squares has a tendency to overfit the training data. One common technique used to address the overfitting problem is regularization. In this problem, we work through one of the regularization techniques namely ridge regularization which is also known as the regularized least squares problem. In the regularized least squares we solve the following optimization problem

$$\arg\min_{\theta} \frac{1}{2}\sum_{i=1}^{N}(y^{(i)} - \theta^T \hat{\mathbf{x}}^{(i)})^2 + \frac{\lambda}{2}\|\theta\|_2^2$$

where λ is a tunable regularization parameter. From the above cost function it can be observed that we are seeking least squares solution with a smaller 2-norm. Derive the solution to the regularized least squares problem, i.e Find θ^*.

5. $\underset{\theta}{\operatorname{argmin}} \frac{1}{2} \sum_{i=1}^{n} (y^{(i)} - \theta^T x^{(i)})^2 + \frac{\lambda}{2} \|\theta\|_2^2$

$\theta^* = ?$ From lecture/discussion

$\frac{1}{2} \sum_{i=1}^{n} (y^{(i)} - \theta^T x^{(i)})^2 = \frac{1}{2} (Y - X\theta)^T (Y - X\theta)$

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

$X \in \mathbb{R}^{n \times 2}$

$\underset{\theta}{\operatorname{argmin}} \frac{1}{2} (Y - X\theta)^T (Y - X\theta) + \frac{\lambda}{2} \|\theta\|_2^2$

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

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

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

$\frac{\partial L(\theta)}{\partial \theta} = -X^T Y + (X^T X + \lambda I)\theta = 0$

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

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 2023, Prof. J.C. Kao, TAs: T.M, P.L, R.G, K.K, N.V, S.R, S.P, M.E

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt

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

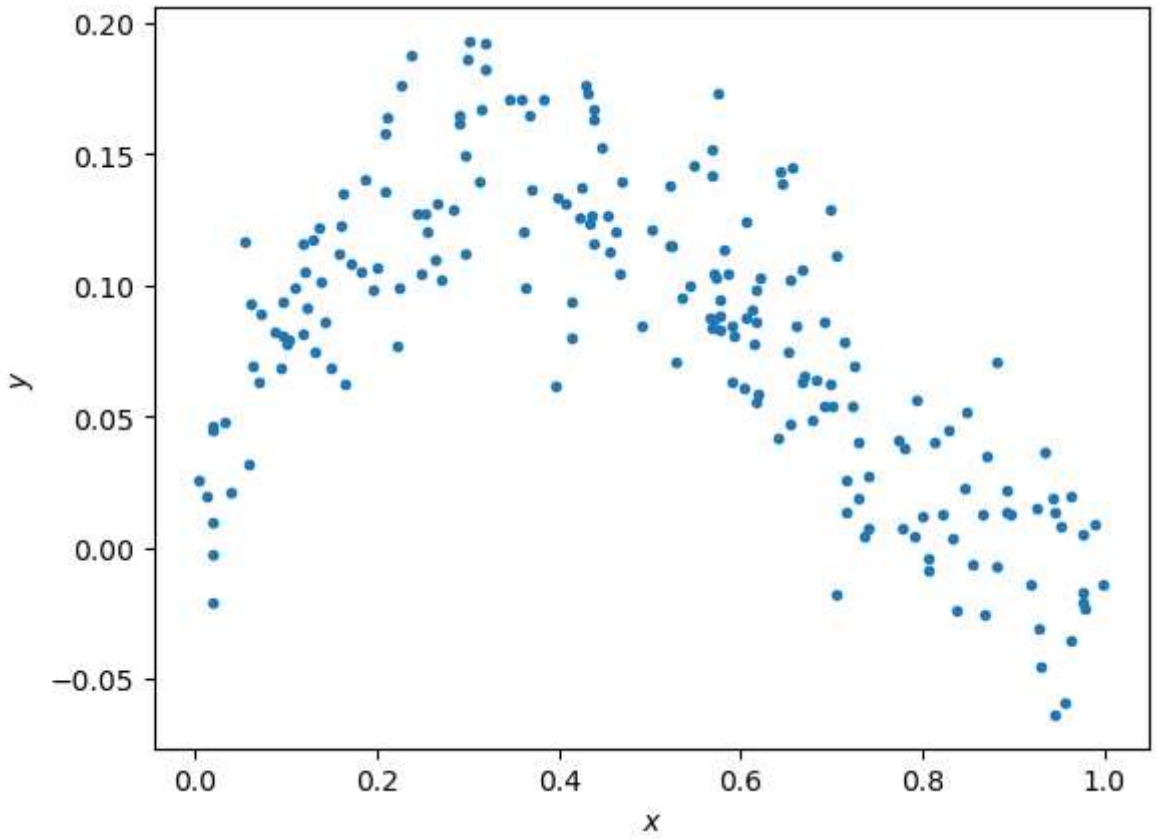
Data generation

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

In [2]:
```
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('$y$')
```

Out[2]:
```
Text(0, 0.5, '$y$')
```



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?

ANSWERS:

(1) The generating distribution of x is a uniform distribution from 0 to 1.

(2) The distribution of the additive noise ϵ is normal (Gaussian) distribution; the mean is 0 and standard deviation is 0.03.

Fitting data to the model (5 points)

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

In [3]:
```
# 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]
```

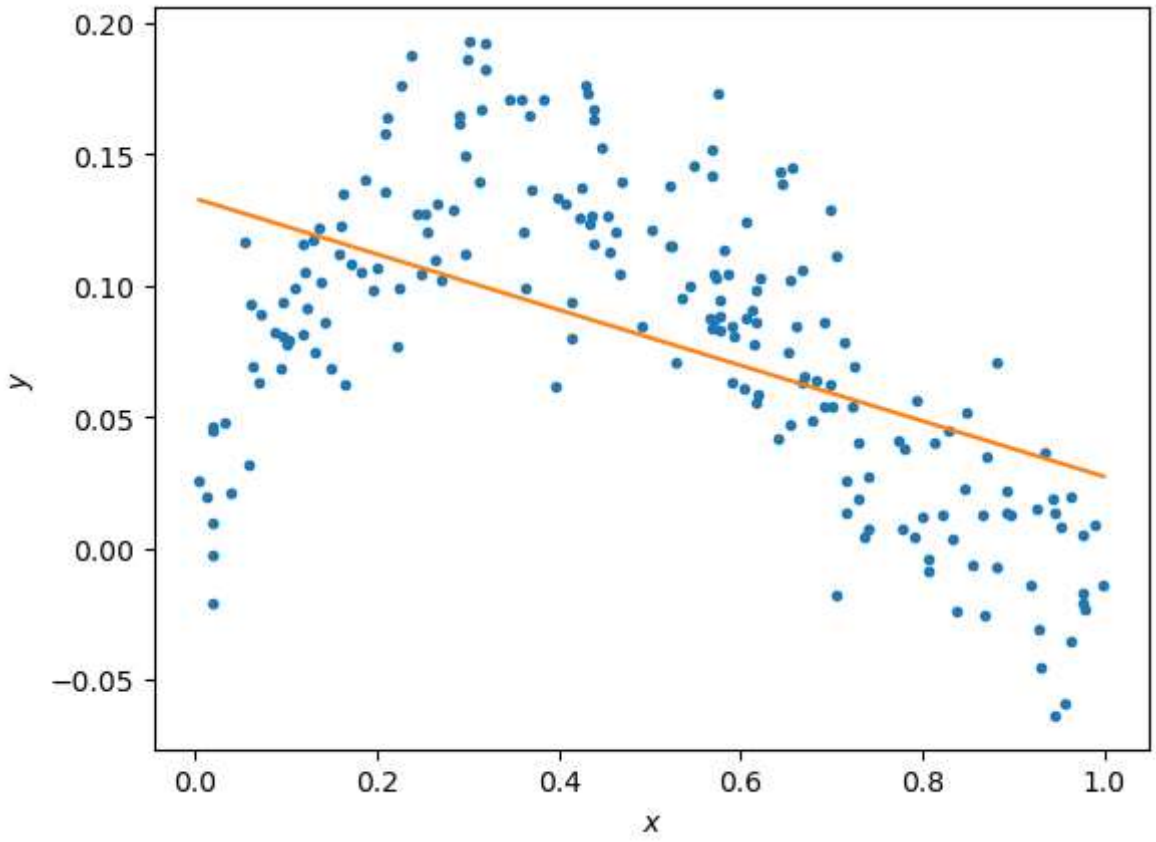
```
theta = np.linalg.inv(xhat @ xhat.T) @ (xhat @ y)

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

In [4]:
```
# 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))
```

Out[4]:
```
[<matplotlib.lines.Line2D at 0x240a907b160>]
```



QUESTIONS

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

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

ANSWERS

(1) This model underfits the data.

(2) We need to use a model of higher order; this means adding more parameters to theta to increase the complexity of the model.

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.

In [5]:
```
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
#   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,
#   ... etc.

xhats = [xhat]
thetas = [theta]

for i in range(1, N):
  xhat = np.vstack((x**(i+1), xhat))
  theta = np.linalg.inv(xhat @ xhat.T) @ (xhat @ y)
  xhats.append(xhat)
  thetas.append(theta)

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

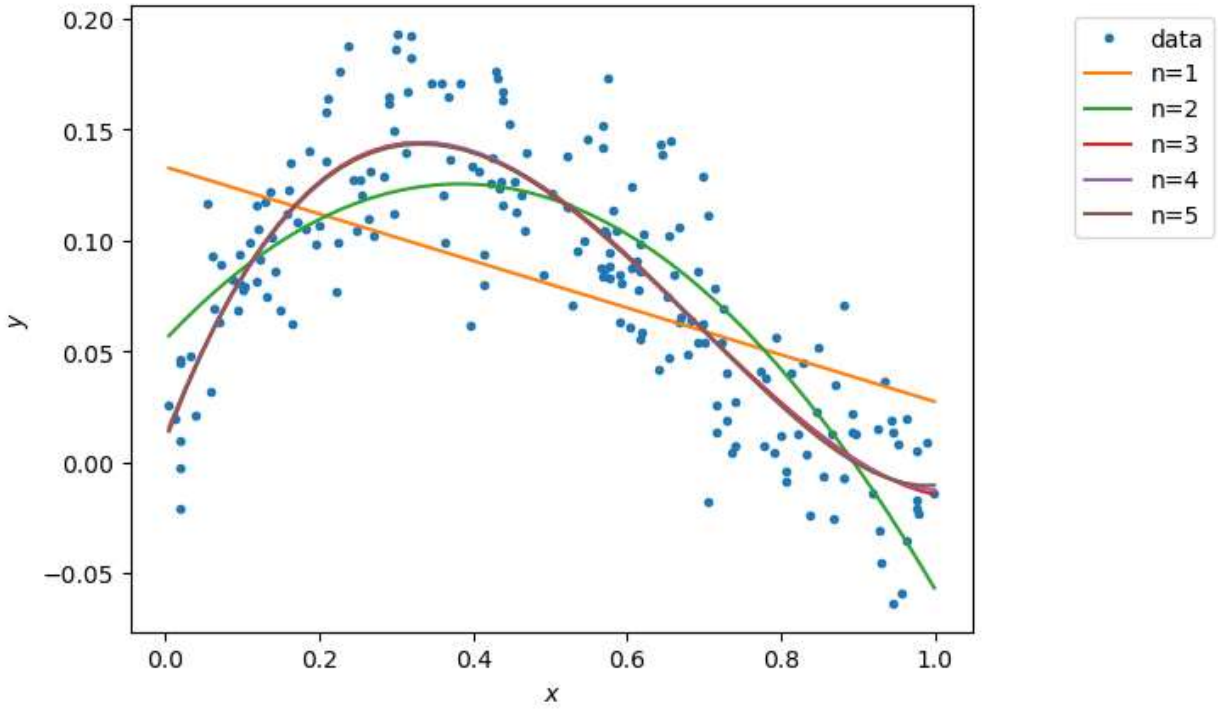
In [6]:
```
# Plot the data
f = plt.figure()
ax = f.gca()
ax.plot(x, y, '.')
ax.set_xlabel('$x$')
ax.set_ylabel('$y$')

# 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)
```



Calculating the training error (5 points)

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

In [7]:
```
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(N):
  y_pred = y - thetas[i] @ xhats[i]
  training_errors.append(y_pred.T @ y_pred / num_train)

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

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

```
Training errors are:
 [0.0023799610883627007, 0.0010924922209268528, 0.0008169603801105372, 0.000816535373
529698, 0.0008161479195525292]
```

QUESTIONS

(1) What polynomial has the best training error?

(2) Why is this expected?

ANSWERS

(1) The polynomial of degree 5 has the lowest training error (best).

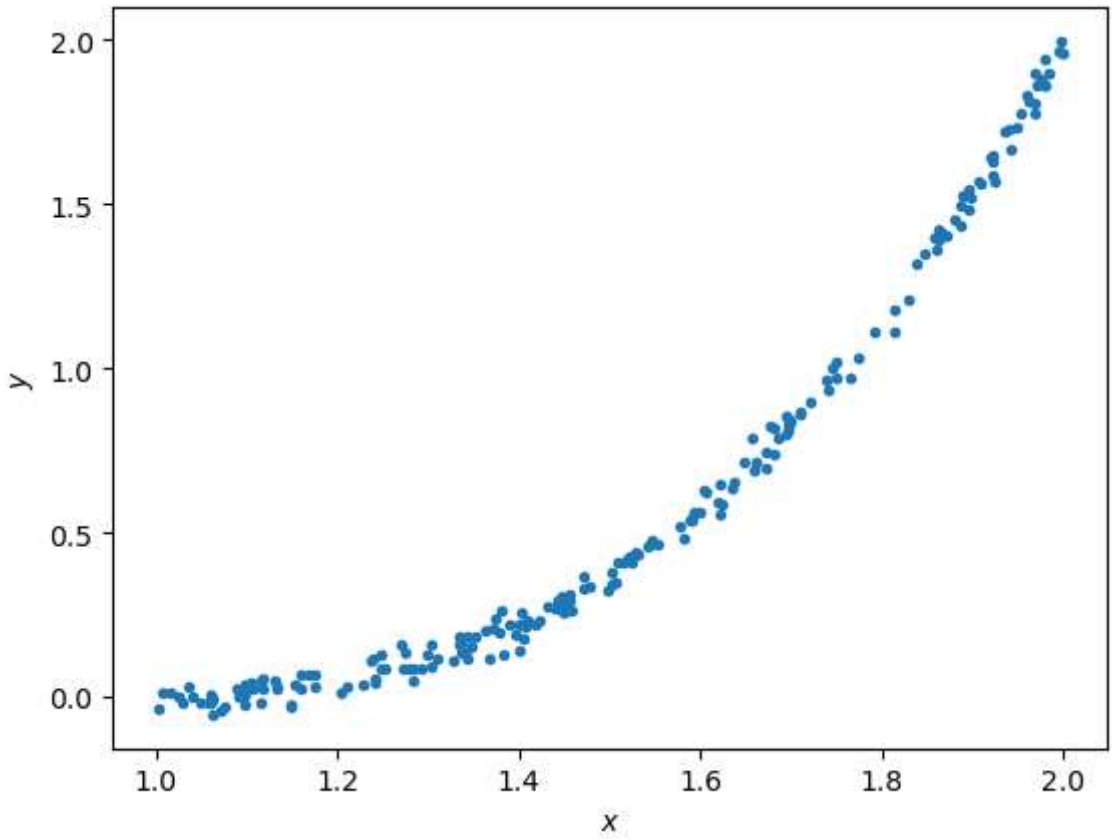
(2) This is expected because higher-order polynomials have more parameters and dimensions to fit the data best (barring any overfitting). Higher order models perform at least as well as lower order models, so we expect there to be lower training error with more complex models.

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.

In [8]:
```
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('$y$')
```

Out[8]:
```
Text(0, 0.5, '$y$')
```



In [9]:
```
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)
```

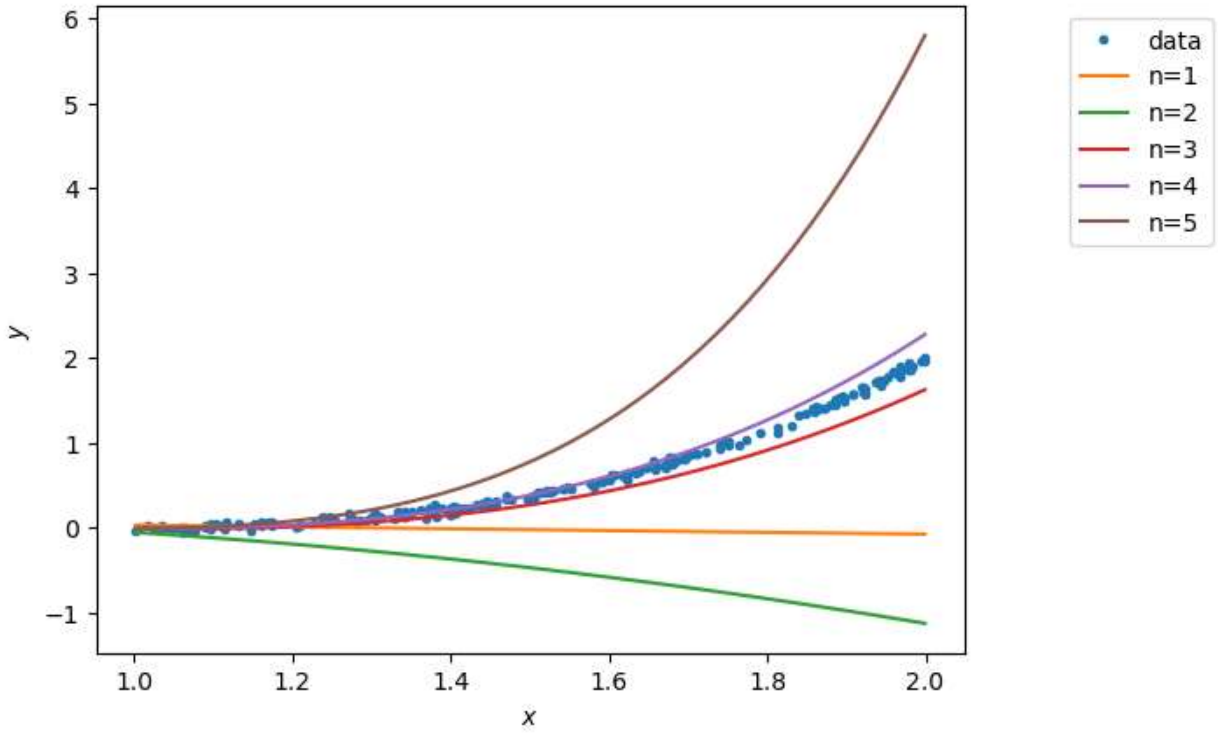
In [10]:
```
# Plot the data
f = plt.figure()
ax = f.gca()
ax.plot(x, y, '.')
ax.set_xlabel('$x$')
ax.set_ylabel('$y$')

# 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)
```



In [11]:
```
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(N):
  y_pred = y - thetas[i] @ xhats[i]
  testing_errors.append(y_pred.T @ y_pred / num_train)

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

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

```
Testing errors are:
 [0.8086165184550589, 2.131919244505802, 0.03125697108274475, 0.011870765189429009,
2.149102183459647]
```

QUESTIONS

(1) What polynomial has the best testing error?

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

ANSWERS

(1) The fourth order polynomial (degree 4) has the lowest testing error (best).

(2) The fifth order polynomial does not generalize well because it overfits the data. It fits very well to the training data, but given new unseen data (the test data), it does not generalize well because it learns too much from the noise of the training data.