HW 1: Math Review and Plotting

Due Date: Tuesday June 25, 11:59 PM

Collaboration Policy

Data science is a collaborative activity. While you may talk with others about the homework, we ask that you **write your solutions individually**. If you do discuss the assignments with others please **include their names** at the top of your notebook.

Collaborators: *list collaborators here*

This Assignment

One of the purposes of this homework is to help you diagnose your preparedness for the course. The rest of this course will assume familiarity with the programming and math concepts covered in this homework. If you struggle with this homework, please consider reviewing prerequisite material.

Score Breakdown

Question	Points
1	1
2a	1
2b	1
2c	1
2d	1
3	4
4a	3
4b	2
4c	2
4d	2
4e	2
5	2
6	2
Total	24

Here are some useful Jupyter notebook keyboard shortcuts. To learn more keyboard shortcuts, go to **Help -> Keyboard Shortcuts** in the menu above.

Here are a few we like:

1. `ctrl+return` : *Evaluate the current cell*
2. `shift+return`: *Evaluate the current cell and move to the next*
3. `esc` : *command mode* (may need to press before using any of the commands below)
4. `a` : *create a cell above*
5. `b` : *create a cell below*
6. `dd` : *delete a cell*
7. `m` : *convert a cell to markdown*
8. `y` : *convert a cell to code*

Initialize your environment

This cell should run without error if you're using the course Jupyter Hub or you have set up your personal computer correctly (http://www.ds100.org/sp19/setup).

In [1]:

```
import numpy as np
import matplotlib
%matplotlib inline
import matplotlib.pyplot as plt
plt.style.use('fivethirtyeight')
```

Python

Question 1

Recall the formula for population variance below:

$$\sigma^2 = \frac{\sum_{i=1}^{N}(x_i - \mu)^2}{N}$$

Complete the functions below to compute the population variance of `population`, an array of numbers. For this question, **do not use built in NumPy functions; we will use NumPy to verify your code.**

```
BEGIN QUESTION
name: q1
points: 1
```

In [2]:

```
def mean(population):
    """
    Returns the mean of population (mu)

    Keyword arguments:
    population -- a numpy array of numbers
    """
    # Calculate the mean of a population
    return sum(population) / len(population) # SOLUTION

def variance(population):
    """
    Returns the variance of population (sigma squared)

    Keyword arguments:
    population -- a numpy array of numbers
    """
    # Calculate the variance of a population
    # BEGIN SOLUTION
    m = mean(population)
    return sum((xi - m) ** 2 for xi in population) / len(population)
    # END SOLUTION
```

In [3]:

```
# TEST
population_0 = np.random.randn(100)
np.isclose(mean(population_0), np.mean(population_0), atol=1e-6)
```

Out[3]:

```
True
```

In [4]:

```
# HIDDEN TEST
population_1 = 3 * np.random.randn(100) + 5
np.isclose(mean(population_1), np.mean(population_1), atol=1e-6)
```

Out[4]:

```
True
```

In [5]:

```
# TEST
population_0 = np.random.randn(100)
np.isclose(variance(population_0), np.var(population_0), atol=1e-6)
```

Out[5]:

```
True
```

In [6]:

```
# HIDDEN TEST
population_1 = 3 * np.random.randn(100) + 5
np.isclose(variance(population_1), np.var(population_1), atol=1e-6)
```

Out[6]:

```
True
```

NumPy

You should be able to understand the code in the following cells. If not, review the following:

- DS100 NumPy Review (http://ds100.org/fa17/assets/notebooks/numpy/Numpy_Review.html)
- Condensed NumPy Review (http://cs231n.github.io/python-numpy-tutorial/#numpy)
- The Official NumPy Tutorial (https://docs.scipy.org/doc/numpy-dev/user/quickstart.html)
- The Data 8 Textbook Chapter on NumPy (https://www.inferentialthinking.com/chapters/05/1/Arrays)

Jupyter pro-tip: Pull up the docs for any function in Jupyter by running a cell with the function name and a `?` at the end:

In [7]:

```
np.arange?
```

You can close the window at the bottom by pressing `esc` several times.

Another Jupyter pro-tip: Pull up the docs for any function in Jupyter by typing the function name, then `<Shift>-<Tab>` on your keyboard. This is super convenient when you forget the order of the arguments to a function. You can press `<Tab>` multiple times to expand the docs and reveal additional information.

Try it on the function below:

In [8]:

```
np.linspace
```

Out[8]:

```
<function numpy.core.function_base.linspace(start, stop, num=50, end
point=True, retstep=False, dtype=None)>
```

Now, let's go through some linear algebra coding questions with NumPy. In this question, we'll ask you to use your linear algebra knowledge to fill in NumPy matrices. To conduct matrix multiplication in NumPy, you should write code like the following:

In [9]:

```
# A matrix in NumPy is a 2-dimensional NumPy array
matA = np.array([
    [1, 2, 3],
    [4, 5, 6],
])

matB = np.array([
    [10, 11],
    [12, 13],
    [14, 15],
])

# The notation B @ v means: compute the matrix multiplication Bv
matA @ matB
```

Out[9]:

```
array([[ 76,  82],
       [184, 199]])
```

You can also use the same syntax to do matrix-vector multiplication or vector dot products. Handy!

In [10]:

```
matA = np.array([
    [1, 2, 3],
    [4, 5, 6],
])

# A vector in NumPy is simply a 1-dimensional NumPy array
some_vec = np.array([ 10, 12, 14, ])

another_vec = np.array([ 10, 20, 30 ])

print(matA @ some_vec)
print(some_vec @ another_vec)
```

```
[ 76 184]
760
```

Question 2a

Joey, Deb, and Sam are shopping for fruit at Berkeley Bowl. Berkeley Bowl, true to its name, only sells fruit bowls. A fruit bowl contains some fruit and the price of a fruit bowl is the total price of all of its individual fruit.

Berkeley Bowl has apples for $2.00, bananas for \$1.00, and cantaloupes for $4.00 (expensive!). The price of each of these can be written in a vector:

$$\vec{v} = \begin{bmatrix} 2 \\ 1 \\ 4 \end{bmatrix}$$

Berkeley Bowl sells the following fruit bowls:

1. 2 of each fruit
2. 5 apples and 8 bananas
3. 2 bananas and 3 cantaloupes
4. 10 cantaloupes

Create a 2-dimensional numpy array encoding the matrix B such that the matrix-vector multiplication

$$B\vec{v}$$

evaluates to a length 4 column vector containing the price of each fruit bowl. The first entry of the result should be the cost of fruit bowl #1, the second entry the cost of fruit bowl #2, etc.

```
BEGIN QUESTION
name: q2a
points: 1
```

In [11]:

```
v = np.array([2,1,4])

# BEGIN SOLUTION NO PROMPT
B = np.array([
    [2, 2, 2],
    [5, 8, 0],
    [0, 2, 3],
    [0, 0, 10]
])
# END SOLUTION
""" # BEGIN PROMPT
B = ...
"""; # END PROMPT

# The notation B @ v means: compute the matrix multiplication Bv
B @ v
```

Out[11]:

```
array([14, 18, 14, 40])
```

In [12]:

```
# TEST
B.shape
```

Out[12]:

```
(4, 3)
```

In [13]:

```
# TEST
np.allclose(B @ v, np.array([14, 18, 14, 40]))
```

Out[13]:

```
True
```

Question 2b

Joey, Deb, and Sam make the following purchases:

- Joey buys 2 fruit bowl #1s and 1 fruit bowl #2.
- Deb buys 1 of each fruit bowl.
- Sam buys 10 fruit bowl #4s (he really like cantaloupes).

Create a matrix A such that the matrix expression

$$AB\vec{v}$$

evaluates to a length 3 column vector containing how much each of them spent. The first entry of the result should be the total amount spent by Joey, the second entry the amount sent by Deb, etc.

Note that the tests for this question do not tell you whether your answer is correct. That's up to you to determine.

```
BEGIN QUESTION
name: q2b
points: 1
```

In [14]:

```
A = np.array([
    [2, 1, 0, 0],
    # Finish this!
    # BEGIN SOLUTION
    [1, 1, 1, 1],
    [0, 0, 0, 10]
    # END SOLUTION
])

A @ B @ v
```

Out[14]:

```
array([ 46,  86, 400])
```

In [15]:

```
# TEST
A.shape
```

Out[15]:

```
(3, 4)
```

In [16]:

```
# HIDDEN TEST
np.allclose(A @ B @ v , np.array([ 46,  86, 400]))
```

Out[16]:

```
True
```

Question 2c

Who spent the most money? Assign `most` to a string containing the name of this person.

```
BEGIN QUESTION
name: q2c
points: 1
```

In [17]:

```
most = "Sam" # SOLUTION
```

In [18]:

```
# TEST
most in ["Joey", "Deb", "Sam"]
```

Out[18]:

```
True
```

In [19]:

```
# HIDDEN TEST
most
```

Out[19]:

```
'Sam'
```

Question 2d

Let's suppose Berkeley Bowl changes their fruit prices, but you don't know what they changed their prices to. Joey, Deb, and Sam buy the same quantity of fruit baskets and the number of fruit in each basket is the same, but now they each spent these amounts:

$$\vec{x} = \begin{bmatrix} 80 \\ 80 \\ 100 \end{bmatrix}$$

Use `np.linalg.inv` and the above final costs to compute the new prices for the individual fruits as a vector called `new_v`.

```
BEGIN QUESTION
name: q2d
points: 1
```

In [20]:

```
new_v = np.linalg.inv(A @ B) @ np.array([80, 80, 100]) # SOLUTION
new_v
```

Out[20]:

```
array([5.5       , 2.20833333, 1.        ])
```

In [21]:

```
# TEST
new_v.shape
```

Out[21]:

```
(3,)
```

In [22]:

```
# HIDDEN TEST
np.allclose(new_v, np.array([ 5.5,  2.20833333,  1.]))
```

Out[22]:

```
True
```

Multivariable Calculus, Linear Algebra, and Probability

The following questions ask you to recall your knowledge of multivariable calculus, linear algebra, and probability. We will use some of the most fundamental concepts from each discipline in this class, so the following problems should at least seem familiar to you.

For the following problems, you should use LaTeX to format your answer. If you aren't familiar with LaTeX, not to worry. It's not hard to use in a Jupyter notebook. Just place your math in between dollar signs:

$ f(x) = 2x \$ becomes $f(x) = 2x$.

If you have a longer equation, use double dollar signs to place it on a line by itself:

$\$ \sum_{i=0}^n i^2 $\$ becomes:

$$\sum_{i=0}^{n} i^2$$

.

Here is some handy notation:

Output	Latex
x^{a+b}	`x^{a + b}`
x_{a+b}	`x_{a + b}`
$\frac{a}{b}$	`\frac{a}{b}`
$\sqrt{a+b}$	`\sqrt{a + b}`
$\{\alpha, \beta, \gamma, \pi, \mu, \sigma^2\}$	`\{ \alpha, \beta, \gamma, \pi, \mu, \sigma^2  \}`
$\sum_{x=1}^{100}$	`\sum_{x=1}^{100}`
$\frac{\partial}{\partial x}$	`\frac{\partial}{\partial x}`
$\begin{bmatrix} 2x + 4y \\ 4x + 6y^2 \end{bmatrix}$	`\begin{bmatrix} 2x + 4y \\ 4x + 6y^2 \\ \end{bmatrix}`

For more about basic LaTeX formatting, you can read this article. (https://www.sharelatex.com/learn/Mathematical_expressions)

If you have trouble with these topics, we suggest reviewing:

- Khan Academy's Multivariable Calculus (https://www.khanacademy.org/math/multivariable-calculus)
- Khan Academy's Linear Algebra (https://www.khanacademy.org/math/linear-algebra)
- Khan Academy's Statistics and Probability (https://www.khanacademy.org/math/statistics-probability)

Recall that summation (or sigma notation) is a way of expressing a long sum in a concise way. Let $a_1, a_2, \ldots, a_n \in \mathbb{R}$ and $x_1, x_2, \ldots, x_n \in \mathbb{R}$ be collections of real numbers. When you see x_i, you can think of the i as an index for the i^{th} x. For example x_2 is the second x value in the list $x_1, x_2, \ldots, x_n$. We define sigma notation as follows:

$$\sum_{i=1}^{n} a_i x_i = a_1 x_1 + a_2 x_2 + \ldots + a_n x_n$$

We commonly use sigma notation to compactly write the definition of the arithmetic mean (commonly known as the `average`):

$$\bar{x} = \frac{1}{n}(x_1 + x_2 + \ldots + x_n) = \frac{1}{n} \sum_{i=1}^{n} x_i$$

Question 3

For each of the statements below, either prove that it is true by using definitions or show that it is false by providing a counterexample.

Statement I

$$\frac{\sum_{i=1}^{n} a_i x_i}{\sum_{i=1}^{n} a_i} = \sum_{i=1}^{n} x_i$$

```
BEGIN QUESTION
name: q3a
manual: true
points: 1
```

SOLUTION: False. Let $a_1 = a_2 = x_1 = x_2 = 1$. Then

$$\frac{\sum_{i=1}^{2} a_i x_i}{\sum_{i=1}^{2} a_i} = \frac{1 \times 1 + 1 \times 1}{1 + 1} = \frac{2}{2} = 1$$

but

$$\sum_{i=1}^{2} x_i = 1 + 1 = 2$$

Statement II

$\sum_{i=1}^{n} x_1 = nx_1$

```
BEGIN QUESTION
name: q3b
manual: true
points: 1
```

SOLUTION: True.

$$\sum_{i=1}^{n} x_1 = \underbrace{x_1 + x_1 + \ldots + x_1}_{n \text{ times}} = nx_1$$

Statement III

$\sum_{i=1}^{n} a_3 x_i = n a_3 \bar{x}$

```
BEGIN QUESTION
name: q3c
manual: true
points: 1
```

SOLUTION: True.

$$\sum_{i=1}^{n} a_3 x_i = a_3 \sum_{i=1}^{n} x_i = a_3 n \frac{1}{n} \sum_{i=1}^{n} x_i = n a_3 \left(\frac{1}{n} \sum_{i=1}^{n} x_i \right) = n a_3 \bar{x}$$

Statement IV

$\sum_{i=1}^{n} a_i x_i = n \bar{a} \bar{x}$

```
BEGIN QUESTION
name: q3d
manual: true
points: 1
```

SOLUTION: False. Consider $a_1 = x_2 = 0$ and $a_2 = x_1 = 1$. Then

$$\sum_{i=1}^{2} a_i x_i = 0 \times 1 + 1 \times 0 = 0$$

but

$$n\bar{a}\bar{x} = 2 \times \frac{0+1}{2} \times \frac{1+0}{2} = 2 \times \frac{1}{2} \times \frac{1}{2} = \frac{1}{2}$$

Note:

We can also generalize the summation concepts above to multiple indices: consider an array of values x_{ij}

$$\begin{bmatrix} x_{1,1} & x_{1,2} & \dots & x_{1,n} \\ x_{2,1} & x_{2,2} & \dots & x_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n,1} & x_{n,2} & \dots & x_{n,n} \end{bmatrix}$$

By convention, the first index refers to the row and the second index references the column. e.g. $x_{2,4}$ is the value in the second row and the fourth column. For multi-indexed arrays like this, we can write down the sum of all the values by evoking sigma notation multiple times:

$$\begin{aligned} \sum_{i=1}^{n} \sum_{j=1}^{n} x_{i,j} &= \sum_{i=1}^{n} \left(\sum_{j=1}^{n} x_{i,j} \right) \\ &= \sum_{i=1}^{n} \left(x_{i,1} + x_{i,2} + \dots + x_{i,n} \right) \\ &= \sum_{i=1}^{n} x_{i,1} + \sum_{i=1}^{n} x_{i,2} + \dots + \sum_{i=1}^{n} x_{i,n} \\ &= \left(x_{1,1} + x_{1,2} + \dots + x_{1,n} \right) + \left(x_{2,1} + x_{2,2} + \dots + x_{2,n} \right) + \dots + \left(x_{n,1} + x_{n,2} + \dots + x_{n,n} \right) \\ &= x_{1,1} + x_{1,2} + \dots + x_{1,n} + x_{2,1} + x_{2,2} + \dots + x_{2,n} + \dots + \dots + x_{n,1} + x_{n,2} + \dots + x_{n,n} \end{aligned}$$

Question 4a

Suppose we have the following scalar-valued function on x and y:

$$f(x, y) = x^2 + 4xy + 2y^3 + e^{-3y} + \ln(2y)$$

Compute the partial derivative of $f(x, y)$ with respect to x.

```
BEGIN QUESTION
name: q4ai
manual: true
points: 1
```

SOLUTION:

$$\frac{\partial}{\partial x} f(x, y) = 2x + 4y$$

Now compute the partial derivative of $f(x, y)$ with respect to y:

```
BEGIN QUESTION
name: q4aii
manual: true
points: 1
```

SOLUTION:

$$\frac{\partial}{\partial y} f(x, y) = 4x + 6y^2 - 3e^{-3y} + \frac{2}{2y}$$

Finally, using your answers to the above two parts, compute $\nabla f(x, y)$ (the gradient of $f(x, y)$) and evaluate the gradient at the point ($x = 2, y = -1$).

```
BEGIN QUESTION
name: q4aiii
manual: true
points: 1
```

SOLUTION:

$$\nabla f(x,y) = [2x + 4y, 4x + 6y^2 - 3e^{-3y} + \frac{2}{2y}]^T$$

SOLUTION:

$$\nabla f(2,-1) = [0, 13 - 3e^3]^T$$

Question 4b

Find the value(s) of x which minimizes the expression below. Justify why it is the minimum.

$\sum_{i=1}^{10}(i-x)^2$

```
BEGIN QUESTION
name: q4b
manual: true
points: 2
```

SOLUTION: To find the critical points, we need to find the first derivative with respect to x

$$\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}x}\sum_{i=1}^{10}(i-x)^2 &= \sum_{i=1}^{10}\frac{\mathrm{d}}{\mathrm{d}x}(i-x)^2 \\
&= \sum_{i=1}^{10} -2(i-x) \\
&= -2\left(\sum_{i=1}^{10} i - \sum_{i=1}^{10} x\right) \\
&= -2\left(\frac{10\times(1+10)}{2} - 10x\right) \\
&= -2\,(55 - 10x)
\end{aligned}$$

We then set the derivative equal to zero and solve for x.

$$0 = -2\,(55 - 10x) \implies x = 5.5$$

Make sure the second derivative is positive (shows that the curve is convex) to verify that this is indeed a minimum:

$$\frac{\mathrm{d}}{\mathrm{d}x}-2\,(55 - 10x) = 20 > 0$$

Question 4c

Let $\sigma(x) = \dfrac{1}{1+e^{-x}}$. Show that $\sigma(-x) = 1 - \sigma(x)$.

```
BEGIN QUESTION
name: q4c
manual: true
points: 2
```

SOLUTION:

$$\sigma(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{e^x+1}$$
$$\sigma(-x) = \frac{e^{-x}}{e^{-x}+1} = \frac{e^{-x}+1-1}{1+e^{-x}} = \frac{1+e^{-x}}{1+e^{-x}} - \frac{1}{1+e^{-x}} = 1-\sigma(x)$$

Question 4d

Show that the derivative can be written as:

$$\frac{d}{dx}\sigma(x) = \sigma(x)(1-\sigma(x))$$

```
BEGIN QUESTION
name: q4d
manual: true
points: 2
```

SOLUTION:

$$\sigma(x) = \frac{1}{1+e^{-x}} \implies \sigma(x)(1+e^{-x}) = 1$$

Taking the derivative with respect to x on both sides yields:

$$(1+e^{-x})\frac{d}{dx}\sigma(x) + \sigma(x)\frac{d}{dx}(1+e^{-x}) = (1+e^{-x})\frac{d}{dx}\sigma(x) + \sigma(x)(-e^{-x}) = 0$$

Solving for $\dfrac{d}{dx}\sigma(x)$:

$$\begin{aligned}\frac{d}{dx}\sigma(x) &= \sigma(x)\frac{e^{-x}}{1+e^{-x}}\\ &= \sigma(x)\sigma(-x)\\ &= \sigma(x)(1-\sigma(x))\end{aligned}$$

Question 4e

Write code to plot the function $f(x) = x^2$, the equation of the tangent line passing through $x = 8$, and the equation of the tangent line passing through $x = 0$.

Set the range of the x-axis to (-15, 15) and the range of the y axis to (-100, 300) and the figure size to (4,4).

Your resulting plot should look like this:

You should use the `plt.plot` function to plot lines. You may find the following functions useful:

- `plt.plot(..)` (https://matplotlib.org/api/_as_gen/matplotlib.pyplot.plot.html)
- `plt.figure(figsize=..)` (https://stackoverflow.com/questions/332289/how-do-you-change-the-size-of-figures-drawn-with-matplotlib)
- `plt.ylim(..)` (https://matplotlib.org/api/_as_gen/matplotlib.pyplot.ylim.html)
- `plt.axhline(..)` (https://matplotlib.org/api/_as_gen/matplotlib.pyplot.hlines.html)

```
BEGIN QUESTION
name: q4e
manual: true
points: 2
```

In [23]:

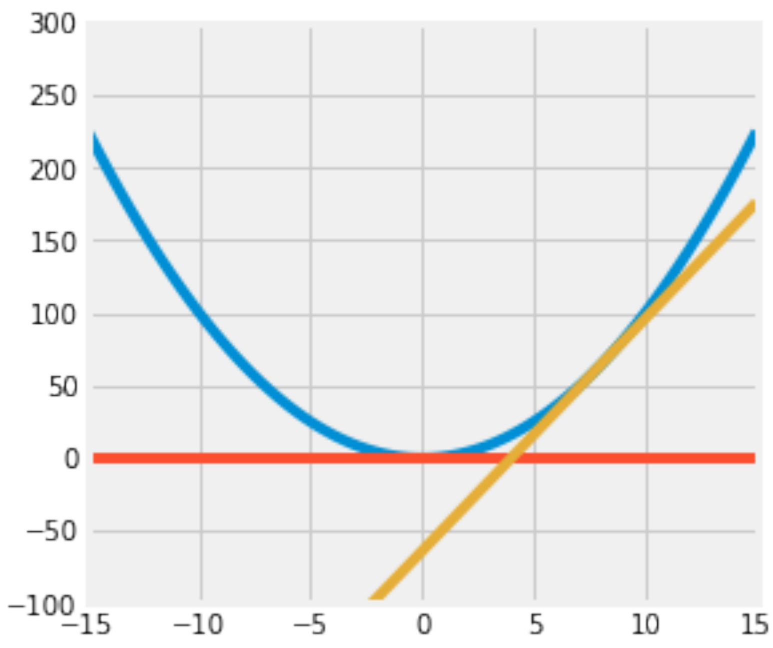
```
def f(x):
    return x**2 # SOLUTION

def df(x):
    return 2*x # SOLUTION

def plot(f, df):
    # BEGIN SOLUTION
    x = np.linspace(-15, 15, 100)

    plt.figure(figsize=(4, 4))
    plt.plot(x, f(x))
    plt.plot(x, np.zeros(x.shape[0]))
    plt.plot(x, df(8) * (x-8) + f(8))
    plt.xlim(-15, 15)
    plt.ylim(-100,300)
    plt.show()
    # END SOLUTION

plot(f, df)
```



Question 5

Consider the following scenario:

Only 1% of 40-year-old women who participate in a routine mammography test have breast cancer. 80% of women who have breast cancer will test positive, but 9.6% of women who don't have breast cancer will also get positive tests.

Suppose we know that a woman of this age tested positive in a routine screening. What is the probability that she actually has breast cancer?

Hint: Use Bayes' rule.

```
BEGIN QUESTION
name: q5
manual: true
points: 2
```

SOLUTION: Again, we want to know what percentage of women with positive tests actually have breast cancer.

We can directly apply Bayes' rule:

$$P(\text{Has Cancer} \mid \text{Tested Positive}) = \frac{P(\text{Tested Positive} \mid \text{Has Cancer}) \times P(\text{Has Cancer})}{P(\text{Tested Positive})}$$

We'll split the denominator into two mutually exclusive cases: a) testing positive with cancer and b) testing positive with no cancer. Then we'll use some basic probability rules to calculate the probabilities we need.

$$\begin{aligned} P(\text{Tested Positive}) &= P(\text{Tested Positive AND Cancer}) + P(\text{Tested Positive AND No Cancer}) \\ &= P(\text{Tested Positive} \mid \text{Cancer}) \times P(\text{Cancer}) \\ &\quad + P(\text{Tested Positive} \mid \text{No Cancer}) \times P(\text{No Cancer}) \\ &= P(\text{Tested Positive} \mid \text{Cancer}) \times P(\text{Cancer}) \\ &\quad + P(\text{Tested Positive} \mid \text{No Cancer}) \times (1 - P(\text{Cancer})) \end{aligned}$$

So ultimately we have the form:

$$P(\text{Has Cancer} \mid \text{Tested Positive}) = \frac{P(\text{Tested Positive} \mid \text{Has Cancer}) \times P(\text{Has C}}{P(\text{Tested Positive} \mid \text{Cancer}) \times P(\text{Cancer}) + P(\text{Tested Positive} \mid \text{N}}$$

Into which we can plug in the values to get our answer:

$$P(\text{Has Cancer} \mid \text{Tested Positive}) = \frac{0.80 \times 0.01}{0.80 \times 0.01 + 0.096 \times (1 - 0.01)} \approx 0.078$$

So the patient has a 7.8% chance of having cancer.

Plotting

Question 6

We should also familiarize ourselves with looking up documentation and learning how to read it. Below is a section of code that plots a basic wireframe. Replace each `# Your answer here` with a description of what the line above does, what the arguments being passed in are, and how the arguments are used in the function. For example,

```
np.arange(2, 5, 0.2)
# This returns an array of numbers from 2 to 5 with an interval size of 0.
2
```

Hint: The `Shift` + `Tab` tip from earlier in the notebook may help here. Remember that objects must be defined in order for the documentation shortcut to work; for example, all of the documentation will show for method calls from `np` since we've already executed `import numpy as np`. However, since `z` is not yet defined in the kernel, `z.reshape()` will not show documentation until you run the line `z = np.cos(squared)`.

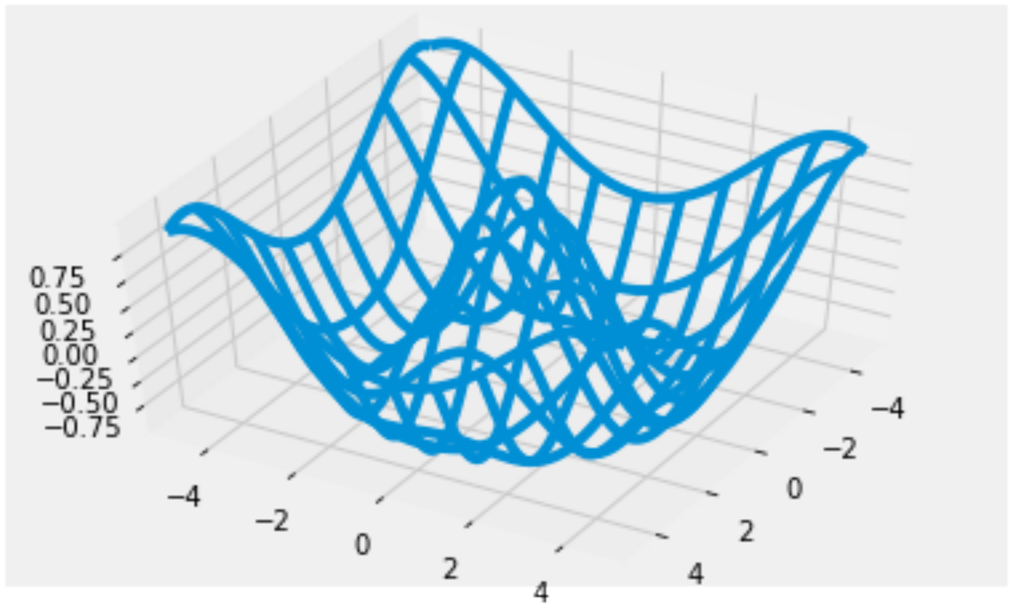
```
BEGIN QUESTION
name: q7
manual: true
points: 2
```

In [24]:

```
from mpl_toolkits.mplot3d import axes3d

u = np.linspace(1.5*np.pi, -1.5*np.pi, 100)
# Your answer here
[x,y] = np.meshgrid(u, u)
# Your answer here
squared = np.sqrt(x.flatten()**2 + y.flatten()**2)
z = np.cos(squared)
# Your answer here
z = z.reshape(x.shape)
# Your answer here

fig = plt.figure()
ax = fig.add_subplot(111, projection='3d')
# Your answer here
ax.plot_wireframe(x, y, z, rstride=10, cstride=10)
# Your answer here
ax.view_init(elev=50., azim=30)
# Your answer here
plt.savefig("figure1.png")
# Your answer here
```


0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−4
−2
0
2
4
−4
−2
0
2
4