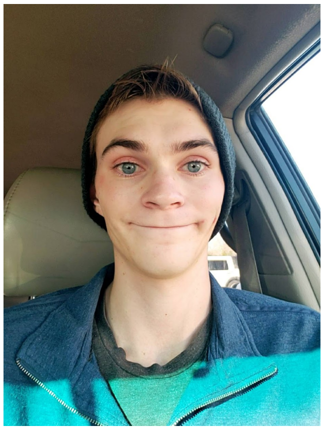
Short Bio

 My name is Carson Stevens and this will be my last semester at School of Mines. Grew up in Lakewood on the other side of Green Mountain, but want to move somewhere 'just different' after I graduate. Some of my hobbies include photography, 4wheeling, and reefkeeping. I love to play and tinker with Raspberry Pis and computer vision is a great tool to be able to use when programming with them. I also have yet to play in detail with Deep Learning and Computer Vision, so I am also very excited about that! Here is one of my reef tanks under 420nm light:

Here are my bunnies Pax and Bella:

Math Review Questions[35 pts]

Answers Below

1. Consider the matrix A=

$$\begin{bmatrix} 4 & -2 \\ 1 & 1 \end{bmatrix}$$

A = [4 − 2 1 1] a. Compute the determinant of the matrix, |A|. b. Compute the trace of the matrix. c. Which of the following matrices is the inverse of A? (i) LaTeX: A^{-1} =

$$\begin{bmatrix} 1/4 & -1/2 \\ 1 & 1 \end{bmatrix}$$

A − 1 = [1 / 4 − 1 / 2 1 1] (ii) LaTeX: A^{-1} =

$$\begin{bmatrix} 4 & 1 \\ -2 & 1 \end{bmatrix}$$

A − 1 = [4 1 − 2 1] (iii) LaTeX: A^{-1} =

$$\begin{bmatrix} 1/6 & 1/3 \\ -1/6 & 2/3 \end{bmatrix}$$

A − 1 = [1 / 6 1 / 3 − 1 / 6 2 / 3] d. Which of the following vectors is the eigenvector of A? (i) x = (-1 2)T (ii) x = (2 1)T (iii) x = (0 1)T (iv) x = (1 0)T e. What is the corresponding eigenvalue? 2. Consider the matrix LaTeX: B\:=\:

$$\begin{bmatrix} 3 & 4 \\ 5 & -1 \end{bmatrix}$$

B = [3 4 5 − 1] a. Compute (AB)T b. Compute BTAT 3. Consider the vectors x = (1 2 3)T and y = (-1 2 -3)T. a. Compute the inner (dot) product LaTeX: x\cdot yx · y b. Compute the vector (cross) product LaTeX: x \times yx × y 4. The faces of a 10-sided die are numbered 0 through 9. a. If the die is rolled, what is the probability that the value of the roll is a prime number? b. What is the expected value of the roll? c. If the die is rolled twice, what is the probability that the same number is obtained both times? 5. Pull a card at random from a deck of cards. What is the conditional probability that the card is the ace of clubs, given that it is a black card? 6. A company makes widgets from three machines. Machine M1 makes 3000/hour, and 80% are good. Machine M2 makes 4000/hour, and 90% are good. Machine M3 makes 3000/hour, and 60% are good. All widgets are mixed together. What is the probability that a widget drawn at random is good (hint: use marginalization)? 7. A medical test shows that a person has a disease. What is the probability that the person actually has the disease (hint: use Bayes' rule)? Here's what we know about the disease and the test: 1 in 100 people have the disease. That is, if D is the event that a randomly selected individual has the disease, then p(D) = 0.01. If H is the event that a randomly selected individual is healthy, then p(H) = 0.99. If a person has the disease, then the probability that the blood test comes back positive is 0.95. That is, p(T+ | D) = 0.95. If a person is healthy, then the probability that the diagnostic test comes back negative is 0.95. That is, p(T− | H) = 0.95.

Math Review Answers

1a. (4*1)-(-2*1) ===> 6

1b. (4+1) ===> 5

1c. iii

1d.

(-x+4 * -x+1) + 2 = x^2 - 5x + 6 = (x - 3)(x -2) = 3,2

```
(4x - 2y) = 3x   => 1x - 2y = 0

(1x + 1y) = 3y   => (1x - 2y = 0)

=> [2

    1]



(4x -2y) = 2x   => 2x - 2y = 0

(1x + 1y) = 2y   => (1x - 1y = 0)

=> [1

    1]



===> ii
```

1e. 3

2a.

[(4*3)+(-2*5) (4*4)+(-2*-1)]

[(1*3)+(1*5) (1*4)+(-1*1)]

```
[2  18]       [2  8]
[8  3 ] ===>  [18 3]
```

2b.

[3 5] [4 1] [(3*4)+(5*-2) (3*1)+(5*1)]

[4 -1] . [-2 1] => [(4*4)+(-1*-2) (4*1)+(-1*1)]

===> [2 8]

```
     [18 3]
```

3a.

[1 [-1

2 . 2 => -1+4-9 ===> -6

3] -3]

3b.

[1 [-1

2 x 2 ===> [-12 0 4]

3] -3]

4a. Primes: 2,3,5,7 => (4/10) ===> 40%

4b. (0+1+2+3+4+5+6+7+8+9)/10 ===> 4.5

4c. 1/10 * 1/10 = 1/100 ===> 1%

1. (1/26)

1. ((3000*0.8)+ (4000*0.9) + (3000*0.6))/(3000+4000+3000)

=> 7800/10000

===> 78%

1. (0.01)(0.95)/((0.01*0.95) + (1-0.01)*0.95)

=> 0.0095/(0.0095+0.9405)

=> 0.0095/0.95 = 0.01

===> 0.0001%

Python Programming

In [11]:

```python
import numpy as np
import math as m

A = np.array([[4,-2],[1,1]])
B = np.array([[3,4],[5,-1]])
x = np.transpose(np.array([1,2,3]))
y = np.transpose(np.array([-1,2,-3]))
w, v = np.linalg.eig(A)

# In[2]:


def guassianFunction(kernel, sigma, method="loops"):
    size = kernel.shape[0]

    if method == "numpy":
        # numpy.fromfunction() referenced from: https://stackoverflow.com/questions/47369579/how-to-get
-the-gaussian-filter
        return np.fromfunction(lambda x, y: (1/(2*np.pi*sigma**2)) * np.e ** ((-1*((x-(size-1)/2)**2+(y
-(size-1)/2)**2))/(2*sigma**2)), (size,size))

    elif method =="loops":
        for x,row in enumerate(kernel):
            for col, y in enumerate(row):
                kernel[x][col]=(1/(2*np.pi*sigma**2)) * np.e ** ((-1*((x-(size-1)/2)**2+(y-(size-1)/2)*
*2))/(2*sigma**2))
        return kernel

def guassianKernel(mu, sigma,size=9,method="loops",normalize=True, rounding=None):
    x1 = np.linspace(-mu,mu,size)
    y1 = np.linspace(-mu,mu,size)
    kernel = np.meshgrid(x1, y1)[0]

    kernel = guassianFunction(kernel,sigma,method)
    if normalize: kernel /= np.sum(kernel)
    if rounding: kernel = np.round(kernel,rounding)

    return kernel


# In[3]:


print("A:\n",A,"\nB:\n",B)
print("\nx: ",x,"\ny: ",y)
print("\n\nDeterminant of A: {}".format(np.linalg.det(A)))
print("\nTrace of A: {}".format(np.trace(A)))
print("\nTranspose of A:\n {}".format(np.transpose(A)))
print("\nInverse of A:\n {}".format(np.linalg.inv(A)))
print("\nEigenvalues of A: {}".format(np.linalg.eigvals(A)))
print("\nEigenvector from Eigenvalue 2: {}".format(v[:,1]))#[1,1]^T)
# print(v[:,0]) #[2,1]^T for eigenvalue 3
print("\n(AB)^T:\n {}".format(np.transpose(np.dot(A,B))))
print("\n(B^T)(A^T):\n {}".format(np.dot(np.transpose(B), np.transpose(A))))
print("\nx . y: {}".format(np.dot(x,y)))
print("\nx X y: {}".format(np.cross(x,y)))
print("\n\nSymmetric Guassian Kernel:\n", guassianKernel(1,1,size=9,method="numpy",rounding=7))
print("\n\nKernel Sum: ", guassianKernel(1,1,size=9,method="numpy").sum())

try:
    print("\n\tNumpy fromfunction method time: ")
    get_ipython().run_line_magic('timeit', 'guassianKernel(1,1,size=9,method="numpy")')
except:
    print("\t\t\tTimeit magic functionality only in Jupyter")

try:
    print("\n\t2 For loop method time: ")
    get_ipython().run_line_magic('timeit', 'guassianKernel(1,1,size=9,method="loops")')
except:
    print("\t\t\tTimeit magic functionality only in Jupyter")
```

```
A:
 [[ 4 -2]
 [ 1  1]]
B:
 [[ 3  4]
 [ 5 -1]]

x:  [1 2 3]
y:  [-1  2 -3]


Determinant of A: 6.0

Trace of A: 5

Transpose of A:
 [[ 4  1]
 [-2  1]]

Inverse of A:
 [[ 0.16666667  0.33333333]
 [-0.16666667  0.66666667]]

Eigenvalues of A: [3. 2.]

Eigenvector from Eigenvalue 2: [0.70710678 0.70710678]

(AB)^T:
 [[ 2  8]
 [18  3]]

(B^T)(A^T):
 [[ 2  8]
 [18  3]]

x . y: -6

x X y: [-12   0   4]


Symmetric Guassian Kernel:
 [[0.000000e+00 6.000000e-07 7.200000e-06 3.240000e-05 5.340000e-05
  3.240000e-05 7.200000e-06 6.000000e-07 0.000000e+00]
 [6.000000e-07 1.960000e-05 2.393000e-04 1.072400e-03 1.768100e-03
  1.072400e-03 2.393000e-04 1.960000e-05 6.000000e-07]
 [7.200000e-06 2.393000e-04 2.915000e-03 1.306430e-02 2.153940e-02
  1.306430e-02 2.915000e-03 2.393000e-04 7.200000e-06]
 [3.240000e-05 1.072400e-03 1.306430e-02 5.855020e-02 9.653290e-02
  5.855020e-02 1.306430e-02 1.072400e-03 3.240000e-05]
 [5.340000e-05 1.768100e-03 2.153940e-02 9.653290e-02 1.591559e-01
  9.653290e-02 2.153940e-02 1.768100e-03 5.340000e-05]
 [3.240000e-05 1.072400e-03 1.306430e-02 5.855020e-02 9.653290e-02
  5.855020e-02 1.306430e-02 1.072400e-03 3.240000e-05]
 [7.200000e-06 2.393000e-04 2.915000e-03 1.306430e-02 2.153940e-02
  1.306430e-02 2.915000e-03 2.393000e-04 7.200000e-06]
 [6.000000e-07 1.960000e-05 2.393000e-04 1.072400e-03 1.768100e-03
  1.072400e-03 2.393000e-04 1.960000e-05 6.000000e-07]
 [0.000000e+00 6.000000e-07 7.200000e-06 3.240000e-05 5.340000e-05
  3.240000e-05 7.200000e-06 6.000000e-07 0.000000e+00]]


Kernel Sum:  1.0000000000000002

        Numpy fromfunction method time:
145 µs ± 1.44 µs per loop (mean ± std. dev. of 7 runs, 10000 loops each)

        2 For loop method time:
435 µs ± 3.55 µs per loop (mean ± std. dev. of 7 runs, 1000 loops each)
```