CSCI576 Homework#1

Yifan Wang

3038184983

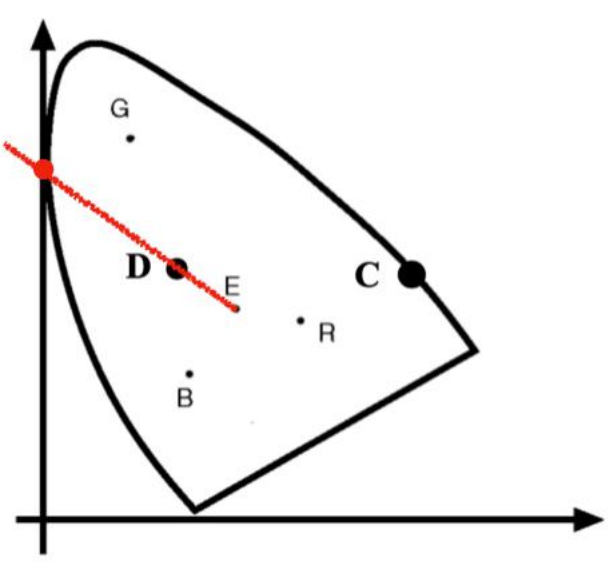
wang608@usc.edu

Question1: Color Theory – 10 points

One of uses of chromaticity diagrams is to find the gamut of colors given the primaries. It can also be used to find dominant and complementary colors –
Dominant color of a given color D (or dominant wavelength in a color D) is defined as the spectral color which can be mixed with white light in order to reproduce the desired D color. *Complementary colors* are those which when mixed in some proportion create the color white. Using these definitions and the understanding of the chromaticity diagram that you have, answer the following.

- In the image alongside find the dominant wavelength of color D. Show this wavelength. *(2 points)*

The dominat wavelength is the red point's value in the following figure;

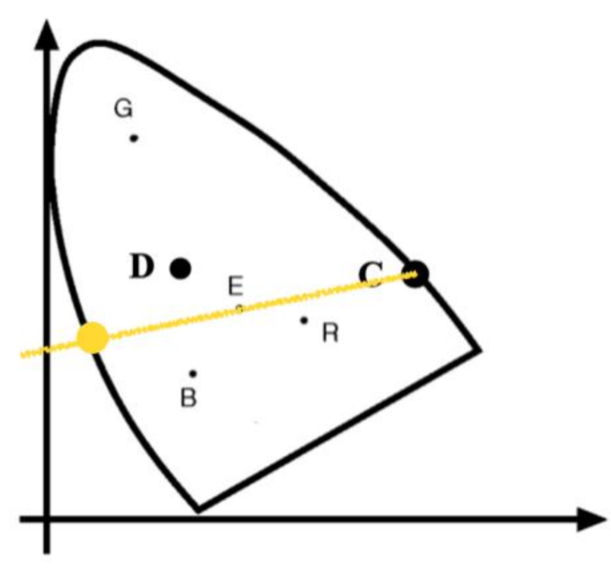


- Do all colors have a dominant wavelength? Explain your reasoning. *(3 points)*

Not all color would have domaint color, like purplr which is generated by mixing severl other lights since the intersection of the rays hit the boundary in the flat part;

- Find the color which is complimentary to the color C. Show this color *(2 points)*

The compimentarty color to C lies on line CE, as the right figure shows following (yellow line);



- What colors in RGB color space map to the equiluminous point E upon projection into the chromaticity space. *(3 points)*

Equiluminous point E is the white color in 8-bit RGB color space it is $(255, 255, 255)$

Question 2: Color Theory (10 points)

- The chromaticity diagram in (*x*, *y*) represents the normalized color matching functions X, Y and Z. Prove that *(2 points)*

  Z = [(1-*x*-*y*)/*y*] Y

We have:

$$x = \frac{X}{X+Y+Z}$$
$$y = \frac{Y}{X+Y+Z}$$
$$z = \frac{Z}{X+Y+Z}$$

then

$$x + y = \frac{X+Y}{X+Y+Z}$$
$$1 - x - y = 1 - \frac{X+Y}{X+Y+Z} = \frac{Z}{X+Y+Z}$$
$$\frac{1-x-y}{y} = \frac{Z}{y(X+Y+Z)} = \frac{Z}{(X+Y+Z)\frac{Y}{X+Y+Z}} = \frac{Z}{Y}$$
$$Z = \frac{1-x-y}{y}Y$$

Here you are tasked with mapping the gamut of a printer to that of a color CRT monitor. Assume that gamuts are not the same, that is, there are colors in the printer's gamut that do not appear in the monitor's gamut and vice versa. So in order to print a color seen on the monitor you choose the nearest color in the gamut of the printer. Answer the following questions

- Discuss (giving reasons) whether this algorithm will work effectively? *(2 points)*

Yes, it will work effectively, since each color unseem can find a neareast color during the print step. The nearest color looks almost the same as the original one;

- You have two images – a cartoon image with constant color tones and a real image with varying color tones? Which image will this algorithm perform better – give reasons. *(2 points)*

The real image would perform better. Because, in cartoon, the transation between object and background would be sharp and a color would occupy a large range of area with less change on object/backgrounds. If this color is modified to the nearest color, it would be obvious to see the difference. While the natrual image would have much smoother transation, so that slightly change the color of some pixel would not influence a lot.

- Can you suggest improvements rather than just choosing the nearest color? *(4 points)*

We can choose nearest available the color on the liner start from the color point (x,y) to the white point. This would give better color appromixation.

Entropy Coding – 20 points

Consider a communication system that gives out only two symbols X and Y. Assume that the parameterization followed by the probabilities are $P(X) = x^2$ and $P(Y) = (1-x^2)$.

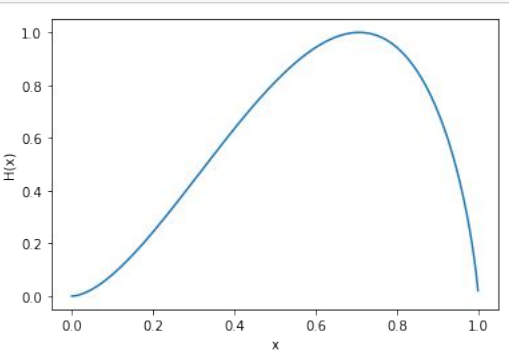
- Write down the entropy function and plot it as a function of *x*. *(1 + 3 points)*

$$H(x) = -\left[x^2 \log x^2 + \left(1-x^2\right)\log\left(1-x^2\right)\right]$$

In [7]:
```python
import numpy as np
import matplotlib.pyplot as plt

def func(x):
    return -(x**2*np.log2(x**2) + (1-x**2)*np.log2(1-x**2))

x = np.arange(1, 1000)/1000
plt.plot(x, func(x))
plt.xlabel('x')
plt.ylabel('H(x)')
plt.show()
```



- From your plot, for what value of *x* does the Entropy become a minimum? At what values of *x* is the Entropy a maximum? *.(2 points)*

The minimum entropy is 0 when $x = 0$ or $x = 1$. The maximum value by observing the plot is around $x \approx 0.7$

- Although the plot visually gives you the value of *x* for which the entropy in maximum, can you now mathematically find out the value(s) for which the entropy is a maximum? *(6 points)*

The maximum entropy is when $x^2 = 1 - x^2$, $x = \frac{\sqrt{2}}{2}$.

- Can you do the same for the minimum, that is can you find mathematically prove the value(s) of *x* for which the entropy is a minimum? *(8 points)*

For the minimum, if we want to approch the smallext entropy, the symbol we got should have no information, so that X would occur for sure or never occur meet this condition.

Generic Compression – (20 points)

The following sequence of real numbers has been obtained sampling a signal:
5.8, 6.2, 6.2, 7.2, 7.3, 7.3, 6.5, 6.8, 6.8, 6.8, 5.5, 5.0, 5.2, 5.2, 5.8, 6.2, 6.2, 6.2, 5.9, 6.3, 5.2, 4.2, 2.8, 2.8, 2.3, 2.9, 1.8, 2.5, 2.5, 3.3, 4.1, 4.9
This signal is then quantized using the interval [0,8] and dividing it into 32 uniformly distributed levels.

- What does the quantized sequence look like? For ease of computation, assume that you placed the level 0 at 0.25, the level 1 at 0.5P, level 2 at 0.75, level 3 at 1.0 and so on. This should simplify your calculations. Round off any fractional value to the nearest integral levels *(4 points)*

Quantized value:

```
[22, 24, 24, 28, 28,
 28, 25, 26, 26, 26,
 21, 19, 20, 20, 22,
 24, 24, 24, 23, 24,
 20, 16, 10, 10,  8,
 11,  6,  9,  9, 12,
 15, 19]
```

- How many bits do you need to transmit the entire signal? *(2 points)*

$$32 * \log 32 = 32 * 5 = 160\, bits$$

- If you need to encode the quantized output using DPCM. Compute the successive differences between the values – what is the maximum and minimum value for the difference? Assuming that this is your range (ie, ignore first value), how many bits are required to encode the sequence now? *(4 points)*

DPCM

```
[22,  2,  0,  4,  0,
  0, -3,  1,  0,  0,
 -5, -2,  1,  0,  2,
  2,  0,  0, -1,  1,
 -4, -4, -6,  0, -2,
  3, -5,  3,  0,  3,
  3,  4]
```

The maximum and minimum value is 4 and -6 which needs $\lceil \log 11 \rceil = 4$ bits to represent one level.

$$31 * 4 = 124\, bits$$

Plus 5 bits for saving the first value;

- What is the compression ratio you have achieved (ignoring first value)? *(1 point)*

$$155 : 124 \approx 1.25 : 1$$

- Instead of transmitting the differences, you use Huffman coded values for the differences. How many bits do you need now to encode the sequence? Show all your work and how you arrived at the final answer *(5+3 points)*

Huffman

```
symbol:   -6 | -5 | -4 | -3 | -2 | -1 |  0 |  1 |  2 |  3 |  4
freq:      1 |  2 |  2 |  1 |  2 |  1 | 10 |  3 |  3 |  4 |  2

step |        (symbol, freq)
  1  | ({-6, -3}, 2), ({-5}, 2), ({-4}, 2), ({-2}, 2), ({-1}, 1), ({0}, 10), ({1}, 3), ({2}, 3), ({3}, 4), ({4}, 2)
  2  | ({-6, -3, -1}, 3), ({-5}, 2), ({-4}, 2), ({-2}, 2), ({0}, 10), ({1}, 3), ({2}, 3), ({3}, 4), ({4}, 2)
  3  | ({-6, -3, -1}, 3), ({-5, -4}, 4), ({-2}, 2), ({0}, 10), ({1}, 3), ({2}, 3), ({3}, 4), ({4}, 2)
  4  | ({-6, -3, -1}, 3), ({-5, -4}, 4), ({-2, 4}, 4), ({0}, 10), ({1}, 3), ({2}, 3), ({3}, 4)
  5  | ({-6, -3, -1, 1}, 6), ({-5, -4}, 4), ({-2, 4}, 4), ({0}, 10), ({2}, 3), ({3}, 4)
  6  | ({-6, -3, -1, 1}, 6), ({-5, -4, 2}, 7), ({-2, 4}, 4), ({0}, 10), ({3}, 4)
  7  | ({-6, -3, -1, 1}, 6), ({-5, -4, 2}, 7), ({-2, 4, 3}, 8), ({0}, 10)
  8  | ({-6, -3, -1, 1, -5, -4, 2}, 13), ({-2, 4, 3}, 8), ({0}, 10)
  9  | ({-6, -3, -1, 1, -5, -4, 2}, 13), ({-2, 4, 3, 0}, 18)
 10  | ({-6, -3, -1, 1, -5, -4, 2, -2, 4, 3, 0}, 31)


                          31
                     /          \
                    /            \
                   /              \
                  /                \
                13                  18
               /  \                /  \
              /    \              /    \
             /      \            8      (0)
            /        \          / \
           /          \        /   \
          6            7      4     (3)
         / \          / \    / \
        /   \        /   \  /   \
       3    (1)     4    (2) (-2) (4)
      / \          / \
     /   \        /   \
    2    (-1)   (-5)  (-4)
   / \
  /   \
(-6)  (-3)

The Huffman tree is shown above:
symbol  | bitstream
-6      | 00000
-5      | 0100
-4      | 0101
-3      | 00001
-2      | 1000
-1      | 0001
 0      | 11
 1      | 001
 2      | 011
 3      | 101
 4      | 1001
```

The number of bits for save the difference is

$$1*5+2*4+2*4+1*5+2*4+1*4+10*2+3*3+3*3+4*3+2*4 = 96\, bits$$

- What is the compression ratio you have achieved now (ignoring first value)? *(2 points)*

$$155 : 96 \approx 1.61 : 1$$

In [ ]: