ECE 637: Lab 6

Rahul Deshmukh
deshmuk5@purdue.edu

March 16, 2021

Section 2 Report

1. Plot of $x_0(\lambda), y_0(\lambda)$ and $z_0(\lambda)$:

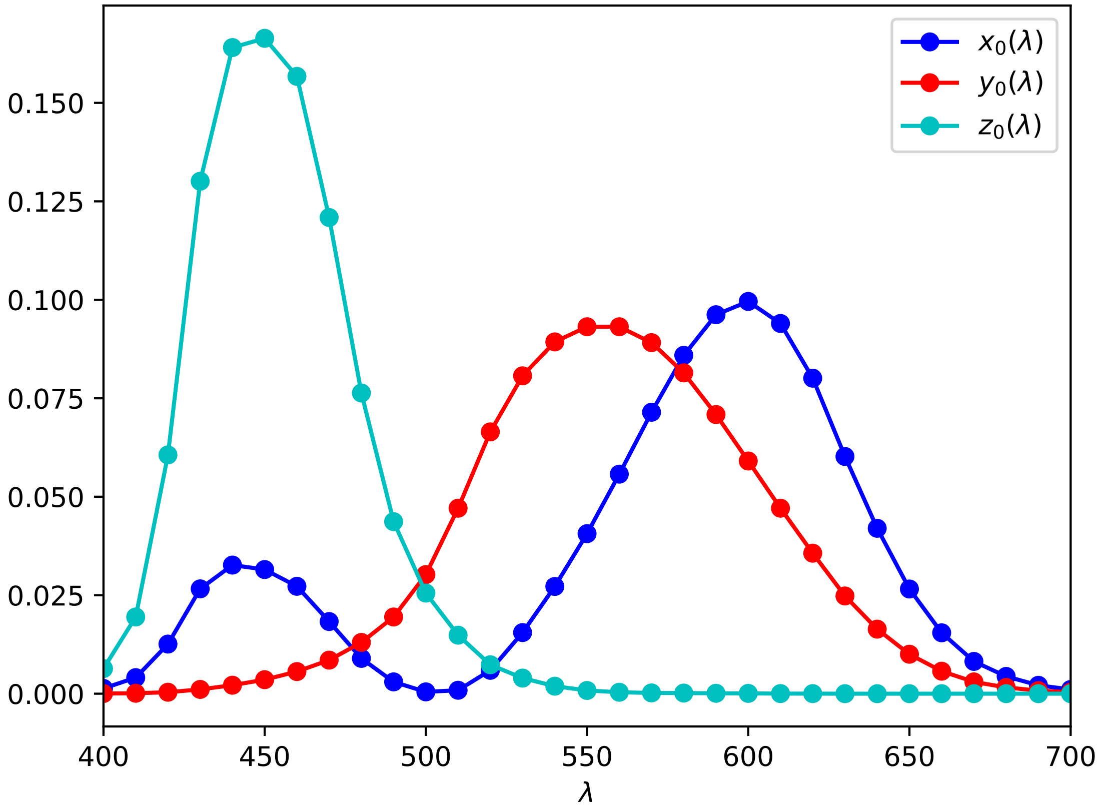


Figure 1: Plot of $x_0(\lambda), y_0(\lambda)$ and $z_0(\lambda)$

2. Plot of $l_0(\lambda), m_0(\lambda)$ and $s_0(\lambda)$

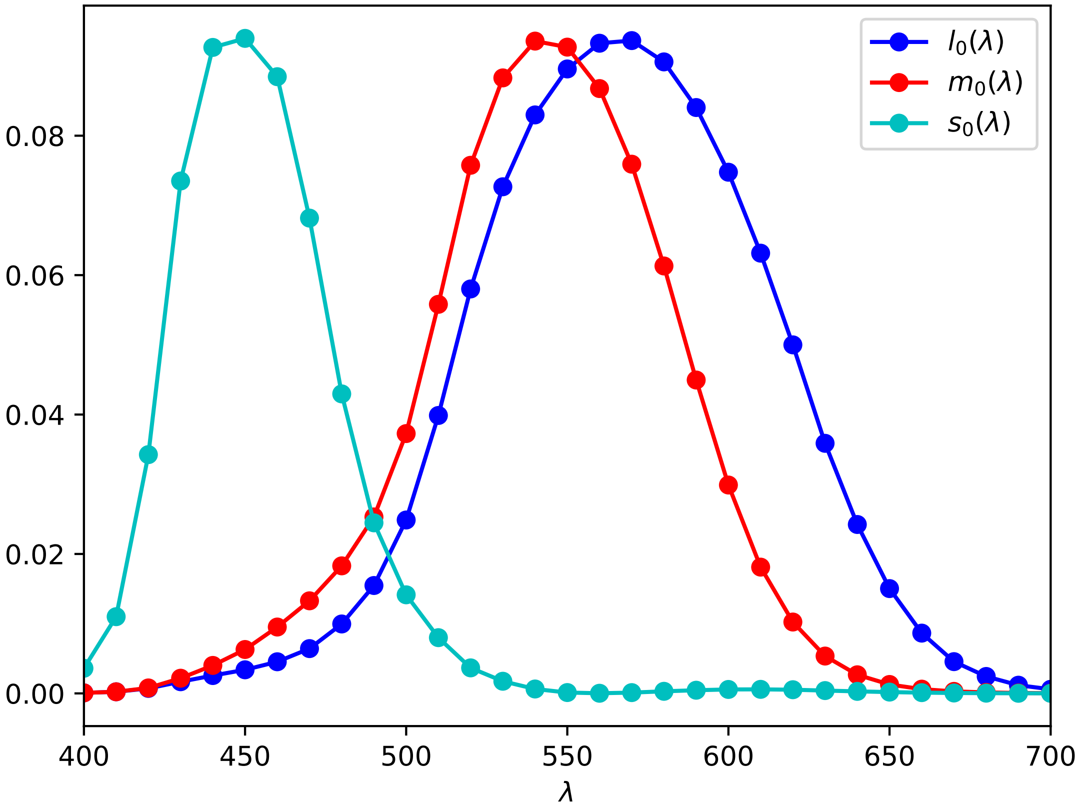


Figure 2: Plot of $l_0(\lambda), m_0(\lambda)$ and $s_0(\lambda)$

3. Plot of D_{65} and fluorescent illuminants

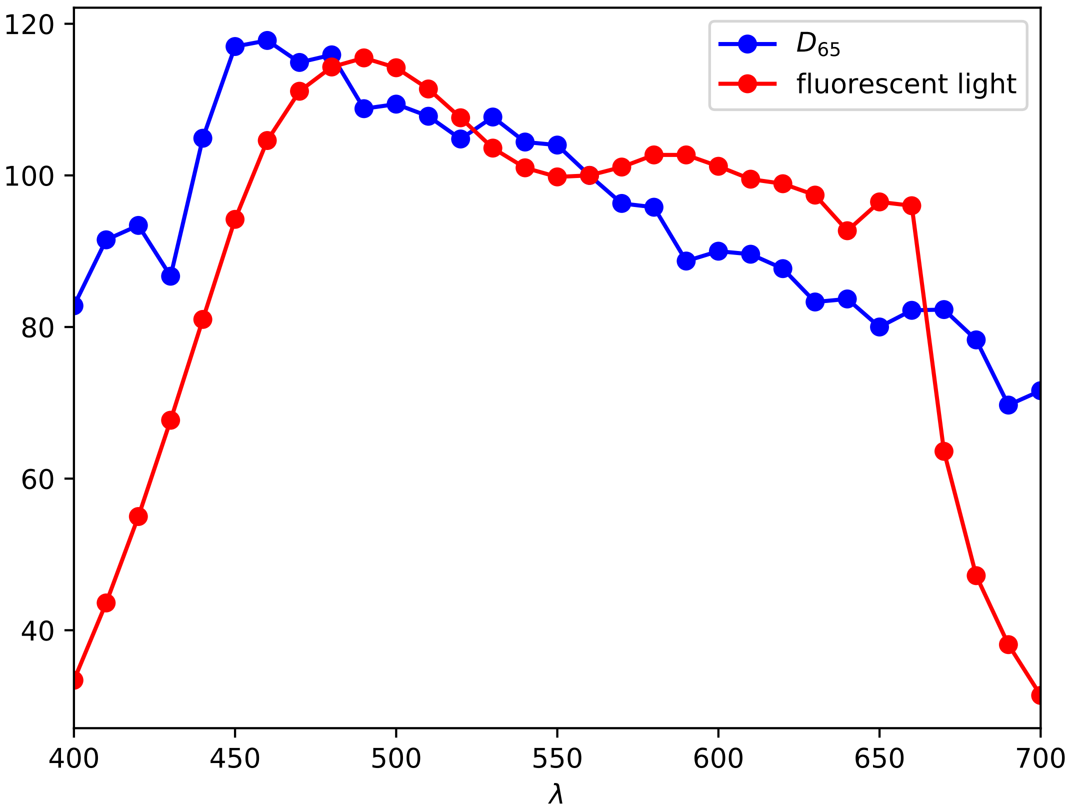


Figure 3: Plot of D_{65} and fluorescent illuminants

For python code refer to Listing 2 at page 7.

Section 3 Report

Chromaticity diagram: For python code refer to Listing 3 at page 8.

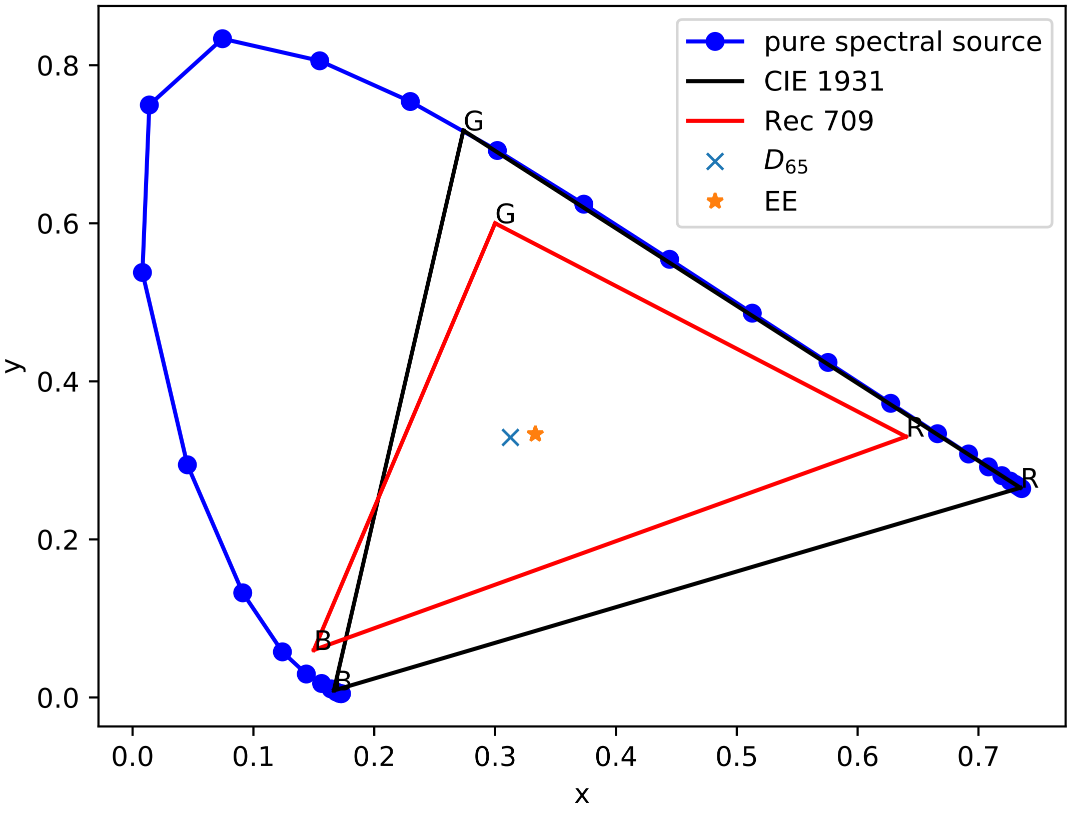


Figure 4: Chromaticity diagram

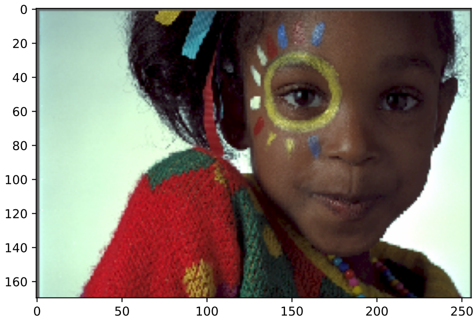
Section 4 Report

1. The Matrix M_{709_D65}:

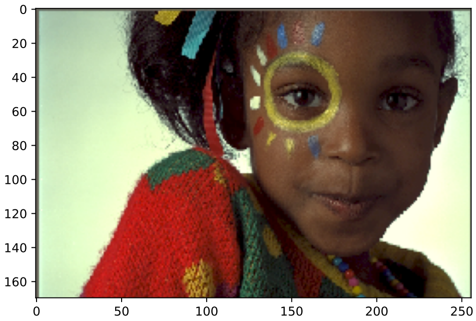
Listing 1: output log showing M_{709_D65}

```
M709_d65:
[[0.4123908  0.35758434 0.18048079]
 [0.21263901 0.71516868 0.07219232]
 [0.01933082 0.11919478 0.95053215]]
```

2. Images obtained from D_{65} and fluorescent light sources:



(a) D_{65} light source image



(b) fluorescent light source image

3. Qualitative description of the difference between the two images: Based on visual comparison we can see that the fluorescent image has a stronger yellow tint. This can be explained using Figure 3 where fluorescent illumination response crosses D_{65} close to $\lambda = 550$ which is the yellow color ($\lambda = 580$). Fluorescent illuminant also cuts out the voilet color($\lambda = 450$) compared to D_{65}.

For python code refer to Listing 4 at page 9 and Listing 6 at page 11

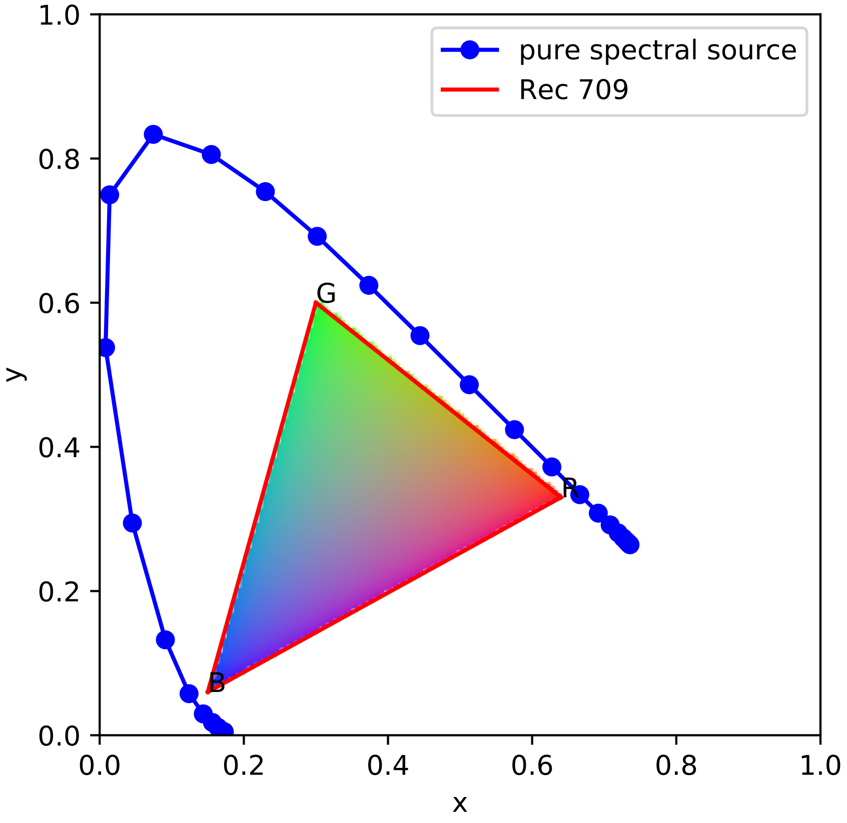


Figure 6: Color chromaticity diagram

For python code refer to Listing 5 at page 10

Appendix

Got to git repo for complete code.

Listing 2: Python code for section 2

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Tue Mar 16 14:26:23 2021

@author: rahul
ECE 637 DIP-1
Lab 6: Intro to colorimetry Ex 2
"""
import sys
import numpy as np
import matplotlib.pyplot as plt

lam = np.linspace(400,700,31)
A_inv = np.array([[ 0.2430,  0.8560, -0.0440],
                  [-0.3910,  1.1650,  0.0870],
                  [ 0.0100, -0.0080,  0.5630]])


def main(data):
    x = data['x'][0]
    y = data['y'][0]
    z = data['z'][0]
    # plot of x,y,z_0 wrt lam
    plt.subplots()
    plt.plot(lam, x,'-ob',label='$x_0(\lambda)$')
    plt.plot(lam, y,'-or',label='$y_0(\lambda)$')
    plt.plot(lam, z,'-oc',label='$z_0(\lambda)$')
    plt.xlim([lam[0], lam[-1]])
    plt.xlabel('$\lambda$')
    plt.legend()
    plt.savefig('xyz.pdf')
    plt.close()

    #plot of l,m,s wrt lam
    lms = np.dot(A_inv,np.vstack((x,y,z)))
    plt.subplots()
    plt.plot(lam, lms[0,:],'-ob',label='$l_0(\lambda)$')
    plt.plot(lam, lms[1,:],'-or',label='$m_0(\lambda)$')
    plt.plot(lam, lms[2,:],'-oc',label='$s_0(\lambda)$')
    plt.xlim([lam[0], lam[-1]])
    plt.xlabel('$\lambda$')
    plt.legend()
    plt.savefig('lms.pdf')
    plt.close()

    #plot of D65 and fluor illum
    plt.figure()
    plt.plot(lam, data['illum1'][0],'-ob',label='$D_{65}$')
    plt.plot(lam, data['illum2'][0],'-or',label='fluorescent light')
    plt.xlim([lam[0], lam[-1]])
    plt.xlabel('$\lambda$')
    plt.legend()
```

```python
    plt.savefig('illums_spectrum.pdf')
    plt.close()

if __name__=="__main__":
    data_path = sys.argv[1]
    data = np.load(data_path,allow_pickle=True)[()]
    data.keys()
    main(data)
```

Listing 3: Python code for section 3

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Tue Mar 16 14:26:23 2021

@author: rahul
ECE 637 DIP-1
Lab 6: Intro to colorimetry Ex 3
"""
import sys
import numpy as np
import matplotlib.pyplot as plt

lam = np.linspace(400,700,31)
A_inv = np.array([[ 0.2430,  0.8560, -0.0440],
                  [-0.3910,  1.1650,  0.0870],
                  [ 0.0100, -0.0080,  0.5630]])

RGB_cie = np.array([[0.73467, 0.26533, 0.0],
                    [0.27376, 0.71741, 0.00883],
                    [0.16658, 0.00886, 0.82456]])

RGB_709 = np.array([[0.640, 0.330, 0.030],
                    [0.300, 0.600, 0.100],
                    [0.150, 0.060, 0.790]])
name=['R','G','B']

D_65_wp = np.array([0.3127, 0.3290, 0.3583])
EE_wp = 0.3333*np.ones(3)

def main(data):
    X = data['x'][0]
    Y = data['y'][0]
    Z = data['z'][0]

    #plot chromacity (x,y)
    x = X/(X+Y+Z)
    y = Y/(X+Y+Z)

    plt.figure()
    plt.plot(x,y,'-ob',label='pure spectral source')
    plt.xlabel('x')
    plt.ylabel('y')
    plt.plot(RGB_cie[:2,0], RGB_cie[:2,1],'k',label = 'CIE 1931')
    plt.plot(RGB_cie[1:3,0], RGB_cie[1:3,1],'k')
    plt.plot(RGB_cie[[0,2],0], RGB_cie[[0,2],1],'k')
    for i in range(3): plt.text(RGB_cie[i,0], RGB_cie[i,1], name[i])

    plt.plot(RGB_709[:2,0], RGB_709[:2,1],'r', label='Rec 709')
    plt.plot(RGB_709[1:3,0], RGB_709[1:3,1],'r')
    plt.plot(RGB_709[[0,2],0], RGB_709[[0,2],1],'r')
    for i in range(3): plt.text(RGB_709[i,0], RGB_709[i,1], name[i])

    plt.plot(D_65_wp[0],D_65_wp[1], 'x',label='$D_{65}$')
    plt.plot(EE_wp[0],EE_wp[1], '*', label='EE')

    plt.legend()
    plt.savefig('chrom_diag.pdf')


if __name__=="__main__":
    data_path = sys.argv[1]
    data = np.load(data_path,allow_pickle=True)[()]
    main(data)
```

Listing 4: Python code for section 4

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Tue Mar 16 14:26:23 2021

@author: rahul
ECE 637 DIP-1
Lab 6: Intro to colorimetry Ex 4
"""
import sys
import numpy as np
from PIL import Image

lam = np.linspace(400,700,31)
A_inv = np.array([[ 0.2430,  0.8560, -0.0440],
                  [-0.3910,  1.1650,  0.0870],
                  [ 0.0100, -0.0080,  0.5630]])

RGB_cie = np.array([[0.73467, 0.26533, 0.0],
                    [0.27376, 0.71741, 0.00883],
                    [0.16658, 0.00886, 0.82456]])

RGB_709 = np.array([[0.640, 0.330, 0.030],
                    [0.300, 0.600, 0.100],
                    [0.150, 0.060, 0.790]])
name=['R','G','B']

D_65_wp = np.array([0.3127, 0.3290, 0.3583])
EE_wp = 0.3333*np.ones(3)

def main(data, reflect, source, source_name):
    X = data['x'][0]
    Y = data['y'][0]
    Z = data['z'][0]

    m,n,p = reflect.shape
    I = np.zeros_like(reflect)
    for i in range(m):
        for j in range(n): I[i,j,:] = reflect[i,j,:]*source

    XYZ = np.zeros((m,n,3))
    for i in range(m):
        for j in range(n): XYZ[i,j,:] = np.dot(np.vstack((X,Y,Z)),I[i,j,:])

    D_65_wp_scaled = D_65_wp/D_65_wp[1]
    scaling_coefs = np.dot(np.linalg.inv(RGB_709.T),D_65_wp_scaled)
    M = np.dot(RGB_709.T, np.diag(scaling_coefs))
    M_inv = np.linalg.inv(M)
    print('M709_d65:')
    print(M)

    rgb = np.zeros((m,n,3))
    for i in range(m):
        for j in range(n): rgb[i,j,:] = np.dot(M_inv,XYZ[i,j,:])
    rgb = np.clip(rgb,0,1)
    #print(rgb.shape)
    im = Image.fromarray((rgb*255).astype(np.uint8))
    im.save('rgb_'+source_name+'.tif','tiff')


if __name__=="__main__":
    data_path = sys.argv[1]
    reflection_path = sys.argv[2]
    source_name = sys.argv[3]

    data = np.load(data_path,allow_pickle=True)[()]
    reflect = np.load(reflection_path,allow_pickle=True)[()]
    reflect = reflect['R']
    if source_name=='d65': source = data['illum1'][0]
    elif source_name=='ee': source = data['illum2'][0]

    main(data, reflect, source, source_name)
```

Listing 5: Python code for section 5

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Tue Mar 16 20:00:46 2021

@author: rahul
ECE 637 DIP-1
Lab 6: Intro to colorimetry Ex 4
"""
import sys
import numpy as np
import matplotlib.pyplot as plt

RGB_709 = np.array([[0.640, 0.330, 0.030],
                    [0.300, 0.600, 0.100],
                    [0.150, 0.060, 0.790]])
name=['R','G','B']
D_65_wp = np.array([0.3127, 0.3290, 0.3583])

def main(step, gamma, x_data, y_data):
    u = np.arange(0,1,step)
    N = u.shape[0]
    x,y = np.meshgrid(u,u)
    z= 1-x-y
```

```python

    #D_65_wp_scaled = D_65_wp/D_65_wp[1]
    #scaling_coefs = np.dot(np.linalg.inv(RGB_709),D_65_wp_scaled)
    scaling_coefs = np.ones(3)
    M = np.dot(RGB_709.T, np.diag(scaling_coefs))
    M_inv = np.linalg.inv(M)
    rgb = np.zeros((N,N,3))
    for i in range(N):
        for j in range(N):
            rgb[i,j,:] = np.dot(M_inv,np.array([x[i,j],
                                                y[i,j],
                                                z[i,j]]))
            if(np.any(rgb[i,j,:]<0)): rgb[i,j,:] = 1

    for k in range(3):
        for i in range(N):
            for j in range(N):
                rgb[i,j,k] = np.exp((1.0/gamma)*np.log(rgb[i,j,k]))

    plt.figure()
    plt.imshow(rgb, extent=[0,1,0,1],origin='lower')
    plt.plot(x_data,y_data,'-ob',label='pure spectral source')
    plt.plot(RGB_709[:2,0], RGB_709[:2,1],'r', label='Rec 709')
    plt.plot(RGB_709[1:3,0], RGB_709[1:3,1],'r')
    plt.plot(RGB_709[[0,2],0], RGB_709[[0,2],1],'r')
    for i in range(3): plt.text(RGB_709[i,0], RGB_709[i,1], name[i])
    plt.xlabel('x')
    plt.ylabel('y')
    plt.legend()
    plt.savefig('chrom_plot.pdf')

if __name__=="__main__":
    step = 0.005
    gamma= 2.2
    data_path = sys.argv[1]
    data = np.load(data_path,allow_pickle=True)[()]
    X = data['x'][0]
    Y = data['y'][0]
    Z = data['z'][0]
    x = X/(X+Y+Z)
    y = Y/(X+Y+Z)
    main(step, gamma, x, y)
```

Listing 6: Python code for gamma correction

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
@author: rahul
course: ECE637-DIP-I
lab4- section 4.2 Gamma of monitor
"""
import argparse
import os
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
import matplotlib as mpl

```

```python
gamma_monitor=1.709511 # hard coded

def gamma_correct_from_linear(img,gamma_out):
    out_img = np.zeros_like(img)
    h,w,ch = img.shape
    for k in range(ch):
        for i in range(h):
            for j in range(w):
                #inverse of eq (5)
                out_img[i,j,k] = 255.0*np.exp((1.0/gamma_out)*np.log(img[i,j,k]/255.0))
    return out_img

def gamma_correct_from_gamma(img,gamma_out,gamma_in):
    out_img = np.zeros_like(img)
    h,w,ch = img.shape
    for k in range(ch):
        for i in range(h):
            for j in range(w):
                x = 255.0*(img[i,j,k]/255.0)**gamma_in
                out_img[i,j,k] = 255.0*np.exp((1.0/gamma_out)*np.log(x/255.0))
    return out_img

if __name__=="__main__":
    parser = argparse.ArgumentParser()
    parser.add_argument("image_file",type=str, help="path to input image file")
    parser.add_argument("-l","--linear", help="Linear scaled input flag",action="store_true")
    parser.add_argument("-gin","--gamma_input",type=np.float, help="gamma value of input")
    parser.add_argument("-gout","--gamma_output",type=np.float, help="gamma value of output")

    args = parser.parse_args()

    filename = args.image_file

    gamma_out = args.gamma_output if args.gamma_output else gamma_monitor
    if args.gamma_input: gamma_in = args.gamma_input

    basename = os.path.basename(filename).split('.')[0]
    im = Image.open(filename)
    img = np.array(im)

    #display input image
    plt.imshow(img,cmap=mpl.cm.gray)
    plt.savefig(basename+'.pdf')
    plt.close()

    if args.linear:
        #print('gamma_output: '+str(gamma_out))
        out_img = gamma_correct_from_linear(img,gamma_out)
    else:
        print("gamma_input: "+str(gamma_in))
        print("gamma_output: "+str(gamma_out))
        out_img = gamma_correct_from_gamma(img,gamma_out,gamma_in)

    #save output image
    plt.imshow(out_img,cmap=mpl.cm.gray)
    plt.savefig(basename+'_gamma_corrected.pdf')
    plt.close()
```

Listing 7: Bash code for running python code

```
#!/bin/bash

#ex2
python ex2.py ../../data.npy
mv ./*.pdf output/ex2
echo 'ex 2 done'

#ex3
python ex3.py ../../data.npy
mv ./*.pdf output/ex3
echo 'ex 3 done'

#ex4
name='d65'
python -W ignore ex4.py ../../data.npy ../../reflect.npy $name | tee "$name".log
#gamma correct rgb.tif
python -W ignore gamma_correction.py ./rgb_"$name".tif --linear -gout 2.2
mv ./*.tif output/ex4/
mv ./*.pdf output/ex4/
mv ./*.log output/ex4/
name='ee'
python -W ignore ex4.py ../../data.npy ../../reflect.npy $name | tee "$name".log
#gamma correct rgb.tif
python -W ignore gamma_correction.py ./rgb_"$name".tif --linear -gout 2.2
mv ./*.tif output/ex4/
mv ./*.pdf output/ex4/
mv ./*.log output/ex4/
echo 'ex 4 done'

#ex5
python -W ignore ex5.py ../../data.npy
mv ./*.pdf output/ex5/
echo 'ex5 done'
```