INF1761 - T1 - João Victor Galindo

In [1]:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

In [2]:

```
path = "data_files/"
cie_file = path + "all_1nm_data.xls"
macbeth_file = path + "ColorChecker_RGB_and_spectra.xls"
planilha = "spectral_data"
print(cie_file, "\n", macbeth_file)
```

```
data_files/all_1nm_data.xls
 data_files/ColorChecker_RGB_and_spectra.xls
```

In [3]:

```
cie = pd.read_excel(cie_file, header = 3)
cie
```

Out[3]:

	nm	CIE A	CIE D65	VM(l)	V'(l)	x bar	y bar	z bar	x bar.1	y bar.1
0	300	0.930483	0.03410	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	301	0.967643	0.36014	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	302	1.005970	0.68618	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	303	1.045490	1.01222	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	304	1.086230	1.33826	NaN	NaN	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...
526	826	260.217000	59.16370	NaN	NaN	0.000002	5.980895e-07	0.0	0.000002	7.946400e-07
527	827	260.567000	59.45090	NaN	NaN	0.000002	5.575746e-07	0.0	0.000002	7.497800e-07
528	828	260.914000	59.73810	NaN	NaN	0.000001	5.198080e-07	0.0	0.000002	7.074400e-07
529	829	261.259000	60.02530	NaN	NaN	0.000001	4.846123e-07	0.0	0.000002	6.674800e-07
530	830	261.602000	60.31250	NaN	NaN	0.000001	4.518100e-07	0.0	0.000002	6.297000e-07

531 rows × 11 columns

In [4]:

```
# passando do "dat frame" para numpy array

nm = cie["nm"].to_numpy(dtype=np.int)
Aw = cie["CIE A"].to_numpy(dtype=np.float32)
D65 = cie["CIE D65"].to_numpy(dtype=np.float32)
V = cie["VM(l)"].to_numpy(dtype=np.float32)
xbar = cie["x bar"].to_numpy(dtype=np.float32)
ybar = cie["y bar"].to_numpy(dtype=np.float32)
zbar = cie["z bar"].to_numpy(dtype=np.float32)
```

In [5]:

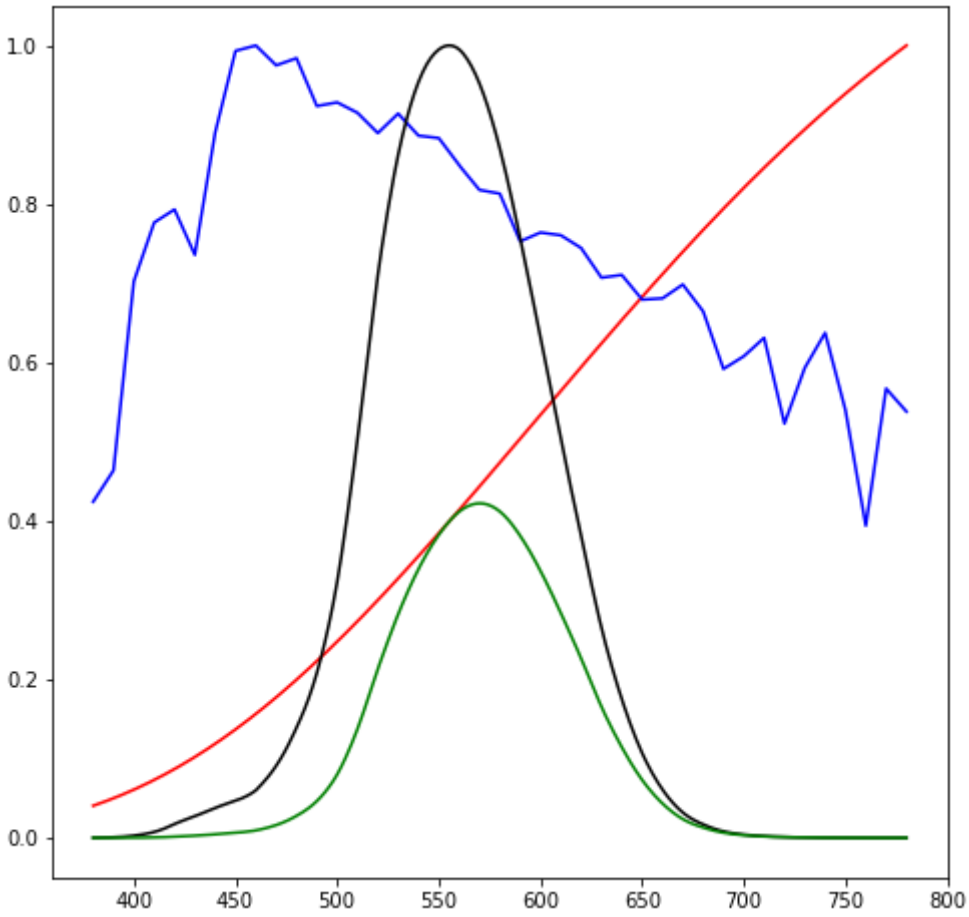
```
# selecionar a parte que interessa (lambda de 380 a 780)

Aw = Aw[(nm > 379) & (nm < 781)]
D65 = D65[(nm > 379) & (nm < 781)]
V = V[(nm > 379) & (nm < 781)]
xbar = xbar[(nm > 379) & (nm < 781)]
ybar = ybar[(nm > 379) & (nm < 781)]
zbar = zbar[(nm > 379) & (nm < 781)]

nm = nm[(nm > 379) & (nm < 781)]
```

In [6]:

```
Aw = Aw/np.amax(Aw)
D65 = D65/np.amax(D65)
```

In [7]:

```
fig = plt.figure(figsize = (8, 8))

plt.plot(nm, Aw, 'red')
plt.plot(nm, D65, 'blue')
plt.plot(nm, V, 'black')
plt.plot(nm, V * Aw, 'green')
plt.show()
```



In [8]:

```
1 macbeth = pd.read_excel(macbeth_file, planilha, skiprows = 1, nrows = 24)
2 macbeth
```

Out[8]:

	No.	Color name	380	390	400	410	420	430	440	
0	1	dark skin	0.054928	0.058196	0.060952	0.062206	0.062053	0.061716	0.061316	0.060
1	2	light skin	0.121190	0.148141	0.180052	0.196914	0.201313	0.203969	0.208183	0.215
2	3	blue sky	0.140803	0.184342	0.253856	0.306988	0.324564	0.331075	0.334410	0.333
3	4	foliage	0.050885	0.053654	0.055276	0.056408	0.057415	0.058832	0.060468	0.061
4	5	blue flower	0.158202	0.208656	0.300129	0.379623	0.412229	0.424513	0.429127	0.428
5	6	bluish green	0.145329	0.185461	0.249941	0.299415	0.323054	0.339699	0.357762	0.383
6	7	orange	0.053094	0.052865	0.052598	0.053118	0.053421	0.053938	0.054314	0.054
7	8	purplish blue	0.131629	0.171464	0.232881	0.289817	0.329289	0.361718	0.387262	0.398
8	9	moderate red	0.098325	0.116366	0.131251	0.135560	0.134019	0.132384	0.131113	0.128
9	10	purple	0.095340	0.118847	0.148021	0.172260	0.182266	0.176078	0.159757	0.139
10	11	yellow green	0.060108	0.060886	0.062169	0.063440	0.064455	0.066481	0.069792	0.075
11	12	orange yellow	0.062336	0.062329	0.063459	0.063565	0.063688	0.064533	0.065603	0.066
12	13	blue	0.064458	0.073910	0.093564	0.137984	0.192346	0.238826	0.280255	0.311
13	14	green	0.051452	0.052675	0.054062	0.055432	0.056888	0.058733	0.061542	0.065
14	15	red	0.048997	0.048123	0.046975	0.046706	0.046643	0.046921	0.046996	0.046
15	16	yellow	0.055893	0.052558	0.051514	0.051585	0.052325	0.053849	0.056014	0.059
16	17	magenta	0.154660	0.202328	0.284373	0.345673	0.361699	0.354580	0.333798	0.305
17	18	cyan	0.113870	0.144635	0.192188	0.234629	0.259178	0.284157	0.316040	0.352
18	19	white 9.5 (.05 D)	0.198909	0.259195	0.420628	0.659732	0.811365	0.863100	0.877124	0.883
19	20	neutral 8 (.23 D)	0.182197	0.240142	0.367207	0.506212	0.566116	0.581326	0.585635	0.587
20	21	neutral 6.5 (.44 D)	0.151760	0.196731	0.271888	0.329589	0.349304	0.355931	0.359727	0.361
21	22	neutral 5 (.70 D)	0.108667	0.133327	0.163127	0.180763	0.186802	0.190631	0.193816	0.194
22	23	neutral 3.5 (1.05 D)	0.068051	0.076379	0.082990	0.086445	0.088084	0.089786	0.091230	0.091
23	24	black 2 (1.5 D)	0.031118	0.031818	0.032134	0.032363	0.032572	0.032659	0.032684	0.032

24 rows × 38 columns

In [9]:

```
reflectance = macbeth.loc[:, macbeth.columns[2:]].to_numpy(dtype = np.float32)
colors = macbeth['Color name']
print(colors)
```

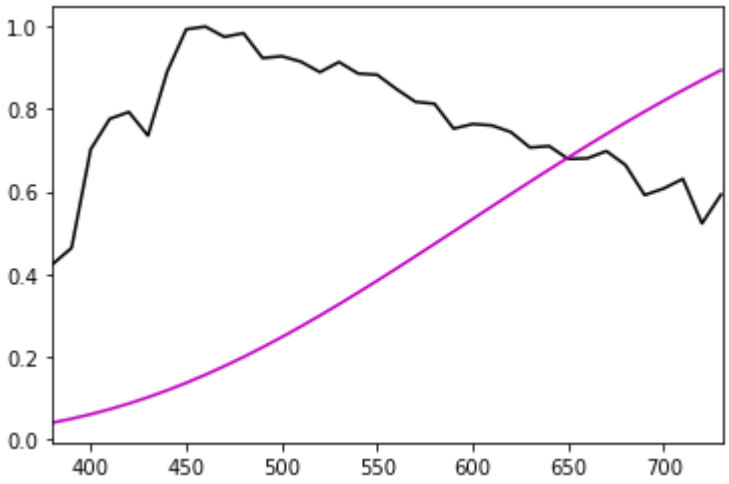
```
0               dark skin
1              light skin
2                blue sky
3                 foliage
4             blue flower
5            bluish green
6                  orange
7           purplish blue
8            moderate red
9                  purple
10           yellow green
11          orange yellow
12                   blue
13                  green
14                    red
15                 yellow
16                magenta
17                   cyan
18       white 9.5 (.05 D)
19       neutral 8 (.23 D)
20     neutral 6.5 (.44 D)
21       neutral 5 (.70 D)
22    neutral 3.5 (1.05 D)
23         black 2 (1.5 D)
Name: Color name, dtype: object
```

In [10]:

```
beta_green = reflectance[13,:]
beta_cyan = reflectance[17,:]

mask = (nm < 731) & (nm%10==0)
```

In [11]:

```
plt.plot(nm[mask], D65[mask], 'k')
plt.plot(nm[mask], Aw[mask], 'm')
plt.xlim(380, 731)
plt.show()
```



In [12]:

```
areaD65 = np.sum(D65)
areaAw = np.sum(Aw)
print(areaD65, areaAw)
```

```
299.09442 195.73885
```

In [13]:

```
lumD65 = np.sum(D65 * V)
lumAw = np.sum(Aw * V)
print(lumD65, lumAw)

ratioD65 = lumD65 / areaD65
ratioAw = lumAw / areaAw
print(ratioD65, ratioAw)
```

```
90.210594 44.70941
0.30161244 0.22841358
```

In [14]:

```
def beta2XYZ(Lw, beta, x_, y_, z_):
    k = 1/np.sum(Lw * y_)
    x = k * np.sum(Lw * beta * x_)
    y = k * np.sum(Lw * beta * y_)
    z = k * np.sum(Lw * beta * z_)

    return np.array([x, y, z])

XYZ_green = beta2XYZ(D65[mask], beta_green, xbar[mask], ybar[mask], zbar[mask])
print(XYZ_green)
```

```
[0.14531483 0.23340117 0.09790131]
```

In [15]:

```
def gamma_sRGB(x):
    t = x if x > 0 else -x
    if t > 0.0031308:
        gamma = 1.055 * pow(t ,1.0 / 2.4) - 0.055
    else:
        gamma = 12.92 * t
    return gamma if x > 0 else -gamma

def XYZTosRGB(XYZ):
    M1 = np.array([[ 3.2404542, -1.5371385, -0.4985314],
                   [-0.9692660,  1.8760108,  0.0415560],
                   [ 0.0556434, -0.2040256,  1.0572252]])
    sRGB = np.dot(M1, XYZ)
    sRGB[0] = gamma_sRGB(sRGB[0])
    sRGB[1] = gamma_sRGB(sRGB[1])
    sRGB[2] = gamma_sRGB(sRGB[2])
    return sRGB
```

In [16]:

```
import skimage.color as sc
```

In [17]:

```
Lab_green = sc.xyz2lab(XYZ_green, illuminant = 'D65', observer = '2')
print(Lab_green)
```

```
[ 55.42095702 -40.4905476   33.53996665]
```

In [18]:

```
#sRGB_green = sc.xyz2rgb(XYZ_green)
#sRGB_green = sRGB_green * 255

sRGB_green = XYZTosRGB(XYZ_green)
print(sRGB_green)
```

```
[0.2790902 0.5847912 0.2805381]
```

In [19]:

```
beta = np.zeros(D65[mask].shape, dtype = np.float)
beta[0] = 1

XYZ = beta2XYZ(D65[mask], beta, xbar[mask], ybar[mask], zbar[mask])
print(XYZ)
```

```
[6.46936203e-05 1.84433553e-06 3.05024787e-04]
```

In [20]:

```
XYZ_cyan = beta2XYZ(D65[mask], beta_cyan, xbar[mask], ybar[mask], zbar[mask])
Lab_cyan = sc.xyz2lab(XYZ_cyan, illuminant = 'D65', observer = '2')
sRGB_cyan = XYZTosRGB(XYZ_cyan)
print(Lab_cyan, sRGB_cyan)
```

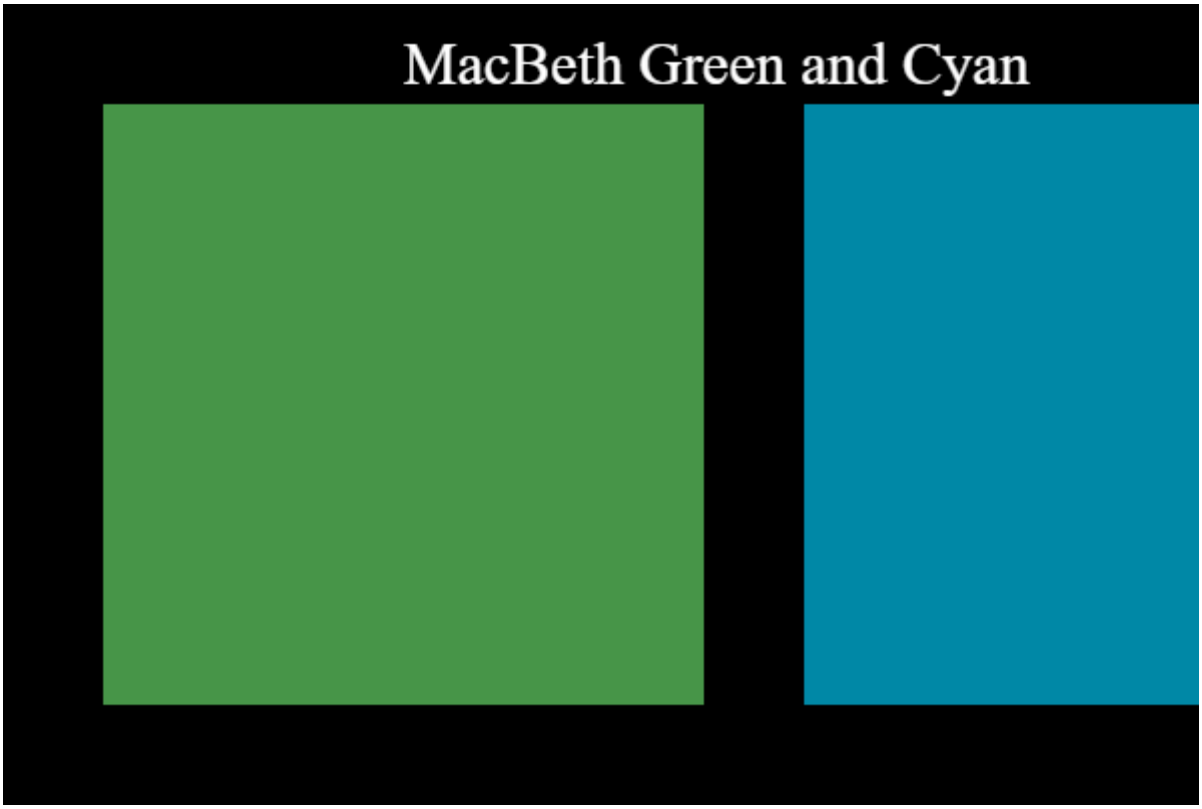
```
[ 51.6183847  -24.10756678 -25.70953849] [-0.1866621   0.53416846  0.6516803
8]
```

In [21]:

```python
def clip(x):
    if x < 0:
        x = 0
    elif x > 255:
        x = 1
    return x

def rgbString(sRGB):
    r = round(255 * sRGB[0])
    g = round(255 * sRGB[1])
    b = round(255 * sRGB[2])

    r = clip(r)
    g = clip(g)
    b = clip(b)

    cor = f'rgb({r:.0f}, {g:.0f}, {b:.0f})'
    #print(cor)
    return cor
```

In [22]:

```
from ipycanvas import Canvas

border = 50
size = 300
canvas_width = 3 * border + 2 * size
canvas_height = 2 * border + size

canvas = Canvas(width = canvas_width, height = canvas_height, sync_image_data = True)
canvas.fill_style = 'black'
canvas.fill_rect(0, 0, canvas_width, canvas_height)
canvas.fill_style = rgbString(sRGB_green)
canvas.fill_rect(border, border, size, size)
canvas.fill_style = rgbString(sRGB_cyan)
canvas.fill_rect(2 * border + size, border, size, size)
canvas.fill_style = 'white'
canvas.font = '30px serif'
canvas.fill_text("MacBeth Green and Cyan", 200, 40)

canvas
```



In [33]:

```
beta_spec = np.zeros(D65.shape, dtype = np.float)
beta_spec[0] = 1

XYZ_spec = np.zeros(shape = (D65.shape[0], 3), dtype = np.float)
sRGB_spec = np.zeros(shape = (D65.shape[0], 3), dtype = np.float)
for i in range(D65.shape[0]):
    beta = np.roll(beta_spec, i)
    XYZ_spec[i,:] = beta2XYZ(D65, beta, xbar, ybar, zbar)

    if (XYZ_spec[i,0] + XYZ_spec[i,1] + XYZ_spec[i,2]) > 0 :
        x = XYZ_spec[i,0] / (XYZ_spec[i,0] + XYZ_spec[i,1] + XYZ_spec[i,2])
        y = XYZ_spec[i,1] / (XYZ_spec[i,0] + XYZ_spec[i,1] + XYZ_spec[i,2])
        Y = 1
        X = x * (Y / y)
        Z = (1 - x - y) * (Y / y)

        sRGB_spec[i,:] = sc.xyz2rgb(np.array([X, Y, Z]))
    #print(sRGB_spec[i,:])

```

In [34]:

```
def lamb2xy(x_, y_, z_):
    xy = np.zeros(shape = (401, 2), dtype = np.float)
    for i in range(401):
        soma = x_[i] + y_[i] + z_[i]
        xy[i, 0] = x_[i] / soma
        xy[i, 1] = y_[i] / soma
    return xy

XY_spec = lamb2xy(xbar, ybar, zbar)
```

In [35]:

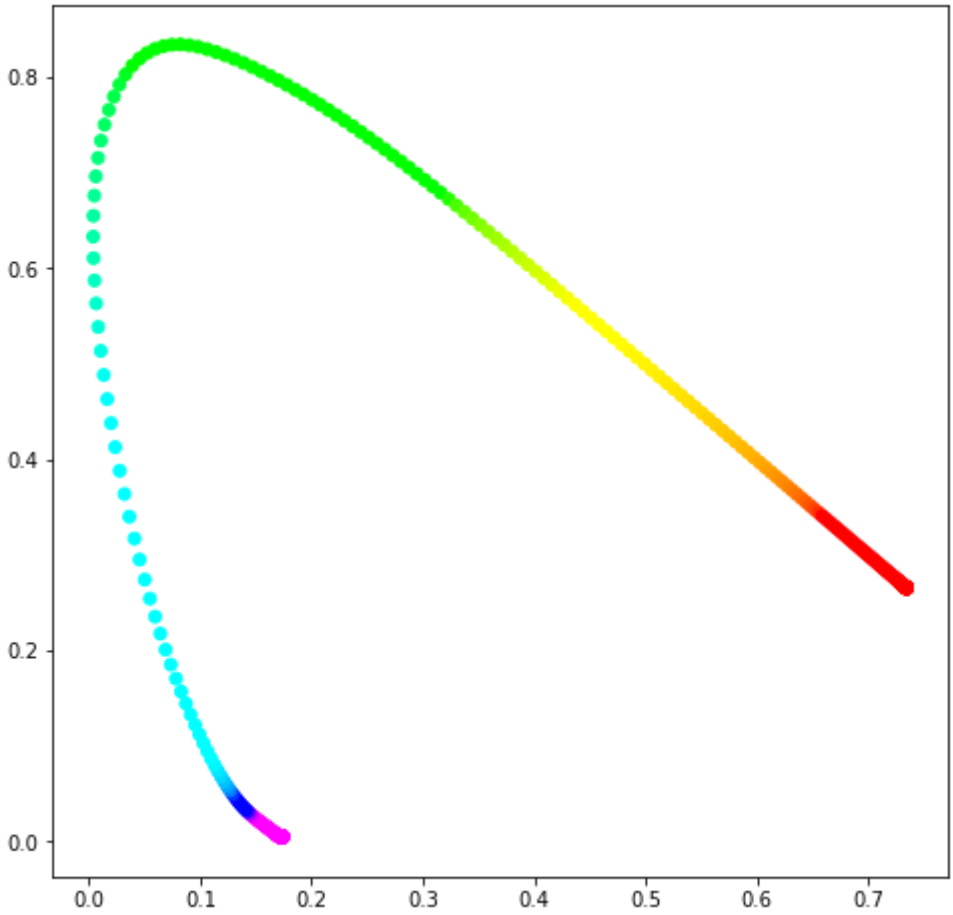
```
canvas2 = Canvas(width = 401, height = 100, sync_image_data = True)
canvas2.line_width = 1

for i in range(401):
    canvas2.stroke_style = rgbString(sRGB_spec[i,:])
    canvas2.stroke_line(i, 0, i, 100)

canvas2
```

In [32]:

```
canvas2.to_file('Joao_Victor_Galindo_spectral_range.png')
```

In [31]:

```
fig = plt.figure(figsize = (8, 8))
plt.scatter(XY_spec[:, 0], XY_spec[:, 1], color = sRGB_spec)
plt.show()
```



In [ ]:

```
1
```