CIELAB

Github Link: https://github.com/pratheepkumar1/Principles-of-Color-Science/tree/main/Assignment%206%20CIELAB

1. Started by reading the given dataset (ColorChecker spectra, standard illuminant data, standard observers, and the CIE 1931 chromaticity coordinates)
2. Interpolated the essential dataset to wavelength (380 to 780 with 5nm difference)
3. Calculated the d_lamda (change in wavelength

```
% Reading the Dataset
warning('off','all');
xyz_std_obs_two_deg_dataset = readtable("StdObsFuncs.xlsx",Sheet="2-degree");
xyz_std_obs_ten_deg_dataset = readtable("StdObsFuncs.xlsx",Sheet="10-degree");
source_dataset = readtable("Illuminant Data.xlsx");
patch_dataset = readtable("MacbethColorChecker.xlsx");
cc_spectral_locus_dataset = readtable("TwoDegChromaticity.xlsx");
% cc_spectral_locus_cc = interpolateData(cc_spectral_locus_dataset{:,1},cc_spectral_lo


% Create a interpolated data for the datasets
intp_wavelength_info = struct('min',380,'max',780,'range',5);

xyz_std_obs_two_deg = interpolateData(xyz_std_obs_two_deg_dataset{:,1},xyz_std_obs_two_
xyz_std_obs_ten_deg = interpolateData(xyz_std_obs_ten_deg_dataset{:,1},xyz_std_obs_ten_
patches = patch_dataset{2:end,2:25};

%Calculate change in Wavlength
wavelength = patch_dataset{2:end,1};
d_lambda = mean(diff(wavelength));
```

Question 1:

Calculate CIE XYZ tristimulus values and chromaticity coordinates for each ColorChecker patch using the CIE standard D50 illuminant and each of the 2-deg and 10-deg standard observers. Please put these in a table, one row per patch.

Steps:

- Interpolated the illuminant D50.
- Calculated tristimulus using the below equation (written as function at the end)

$$X = k\int_\lambda S_\lambda R_\lambda \bar{x}_\lambda \, d\lambda$$
$$Y = k\int_\lambda S_\lambda R_\lambda \bar{y}_\lambda \, d\lambda$$
$$Z = k\int_\lambda S_\lambda R_\lambda \bar{z}_\lambda \, d\lambda$$
$$k = {100}\Big/{\int_\lambda S_\lambda \bar{y}_\lambda \, d\lambda}$$

- Calculated chromaticity coordinates for all patches under D50 light source with 2 and 10 degree viewing angle using the below equation (written as function at the end)

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

```
%Get spectral radiance of Source D50
sources_D50 = interpolateData(source_dataset{:,1},source_dataset.D50,intp_wavelength_i

% Tristimulus value for all patches in D50 Source for 2 and 10 degree observer
tristimulus_XYZ_D50_two_deg = calcTristimulus(xyz_std_obs_two_deg,sources_D50,patches,
tristimulus_XYZ_D50_ten_deg = calcTristimulus(xyz_std_obs_ten_deg,sources_D50,patches,

% chromacity_coordinates for all patches in D50 Source for 2 and 10 degree observer
cc_D50_two_deg = calChromacityCoordinates(tristimulus_XYZ_D50_two_deg);
cc_D50_ten_deg = calChromacityCoordinates(tristimulus_XYZ_D50_ten_deg);
```

Question 1: Answer

```
table_cc_D50 = createTablexyz(cc_D50_two_deg)
```

table_cc_D50 = 24×3 table

	x	y	z
1	0.4451	0.3803	0.1745
2	0.4181	0.3759	0.2060
3	0.2770	0.3024	0.4206
4	0.3659	0.4538	0.1803
5	0.3037	0.2886	0.4077

	x	y	z
6	0.2885	0.3904	0.3211
7	0.5290	0.4109	0.0602
8	0.2380	0.2201	0.5419
9	0.5019	0.3300	0.1681
10	0.3312	0.2529	0.4159
11	0.3988	0.4998	0.1014
12	0.4933	0.4420	0.0647
13	0.2042	0.1647	0.6310
14	0.3246	0.5111	0.1643
15	0.5741	0.3262	0.0997
16	0.4679	0.4726	0.0595
17	0.4211	0.2729	0.3060
18	0.2128	0.3024	0.4848
19	0.3465	0.3597	0.2938
20	0.3461	0.3586	0.2953
21	0.3455	0.3583	0.2962
22	0.3455	0.3583	0.2963
23	0.3432	0.3573	0.2995
24	0.3405	0.3542	0.3053

Question 2

Calculate CIE XYZ tristimulus values and chromaticity coordinates for each ColorChecker patch using the CIE 2-deg standard observer and the A and D65 illuminants. Please put these in a table, one row per patch.

Steps:

- Same as the previous question except the illuminants are change

```
%Get spectral radiance of Source A and Source D65
sources_A = interpolateData(source_dataset{:,1},source_dataset.A,intp_wavelength_info)
sources_D65 = interpolateData(source_dataset{:,1},source_dataset.D65,intp_wavelength_in


% Tristimulus value for all patches in Source A and Source D65 for 2 degree observer
tristimulus_XYZ_A_two_deg = calcTristimulus(xyz_std_obs_two_deg,sources_A,patches,d_lam
tristimulus_XYZ_D65_two_deg = calcTristimulus(xyz_std_obs_two_deg,sources_D65,patches,d

% chromacity_coordinates for all patches in Source A and Source D65 for 2 degree obser
cc_A_two_deg = calChromacityCoordinates(tristimulus_XYZ_A_two_deg);
```

```
cc_D65_two_deg = calChromacityCoordinates(tristimulus_XYZ_D65_two_deg);
```

Question 2: Answer

```
table_cc_A = createTablexyz(cc_A_two_deg)
```

table_cc_A = 24×3 table

	x	y	z
1	0.5328	0.3955	0.0717
2	0.5139	0.3963	0.0898
3	0.3776	0.3822	0.2402
4	0.4471	0.4674	0.0854
5	0.4212	0.3599	0.2189
6	0.3769	0.4458	0.1773
7	0.5768	0.4007	0.0225
8	0.3404	0.3103	0.3493
9	0.5876	0.3494	0.0629
10	0.4658	0.3263	0.2080
11	0.4652	0.4860	0.0488
12	0.5470	0.4268	0.0262
13	0.2876	0.2513	0.4611
14	0.3907	0.5233	0.0860
15	0.6364	0.3302	0.0334
16	0.5256	0.4474	0.0270
17	0.5482	0.3231	0.1287
18	0.2899	0.3871	0.3231
19	0.4482	0.4079	0.1440
20	0.4478	0.4075	0.1447
21	0.4471	0.4076	0.1452
22	0.4471	0.4076	0.1453
23	0.4449	0.4076	0.1475
24	0.4431	0.4058	0.1511

```
table_cc_D65 = createTablexyz(cc_D65_two_deg)
```

table_cc_D65 = 24×3 table

	x	y	z
1	0.4105	0.3630	0.2265
2	0.3826	0.3571	0.2603

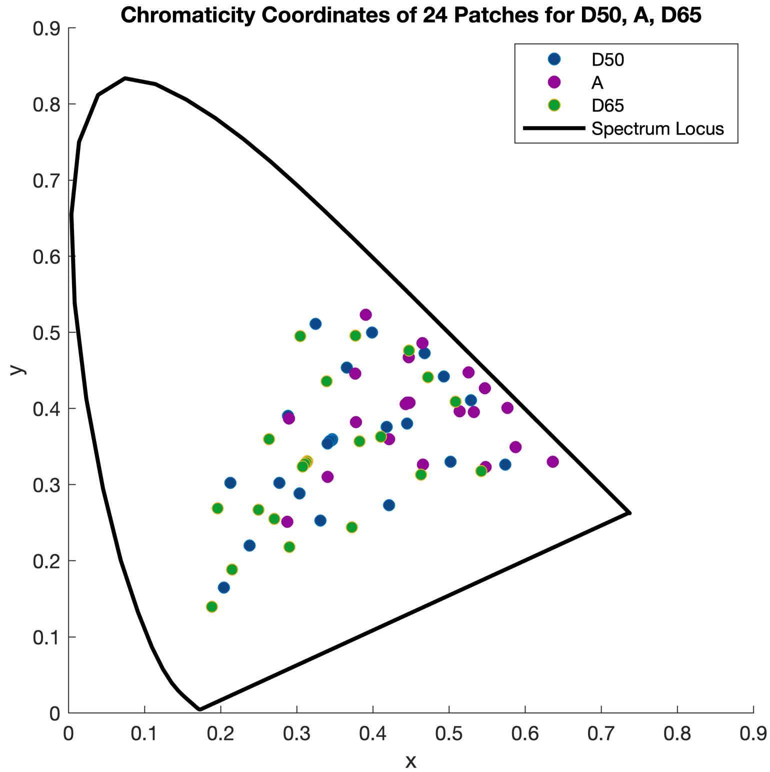
	x	y	z
3	0.2498	0.2669	0.4832
4	0.3394	0.4359	0.2247
5	0.2704	0.2552	0.4745
6	0.2635	0.3599	0.3766
7	0.5086	0.4090	0.0824
8	0.2150	0.1884	0.5966
9	0.4632	0.3129	0.2238
10	0.2904	0.2182	0.4914
11	0.3770	0.4958	0.1271
12	0.4725	0.4413	0.0862
13	0.1884	0.1395	0.6721
14	0.3047	0.4953	0.2000
15	0.5423	0.3179	0.1398
16	0.4475	0.4762	0.0763
17	0.3725	0.2440	0.3835
18	0.1960	0.2689	0.5351
19	0.3136	0.3307	0.3557
20	0.3132	0.3292	0.3576
21	0.3126	0.3288	0.3585
22	0.3126	0.3287	0.3587
23	0.3104	0.3274	0.3621
24	0.3077	0.3239	0.3684

Question 3

Plot all of your 2-deg (x,y) results (for Illum A, D50, and D65) on a chromaticity diagram, with a legend to make it clear which points go with which illuminant. You can label the points (1-24) using the **text** function in Matlab, and you can plot the spectrum locus using the (x,y) coordinates provided. In chromaticity diagrams, it is important to scale the x-axis and y-axis equally (*think about why*). In Matlab please use the command **axis equal**.

```
% Working
h = figure(1);
hold on
scatter(table_cc_D50{:,1},table_cc_D50{:,2},32,'MarkerFaceColor','#0F4786');
scatter(table_cc_A{:,1},table_cc_A{:,2},32,'MarkerEdgeColor','#930796','MarkerFaceColo
scatter(table_cc_D65{:,1},table_cc_D65{:,2},32,'MarkerFaceColor','#089E32');
plot(cc_spectral_locus_dataset{:,2},cc_spectral_locus_dataset{:,3},'-','LineWidth',2,'
```

```
axis equal
xlim([0 0.9]);
ylim([0 0.9]);
xlabel('x');
ylabel('y');
title('Chromaticity Coordinates of 24 Patches for D50, A, D65');
legend('D50','A','D65','Spectrum Locus');
dcm_obj = datacursormode(h);
set(dcm_obj,'UpdateFcn',{@custom_label})
hold off
```



Note: The the patch number is added as text which will appear when the cursor is clicked on a plot value. This is done to minimize text noise in visualisation.

Question 4

What do you observe about the relative chromaticity coordinates for the different light sources?

Answer:

On looking at the chromaticity coordinates, the plot of grayscale color patches indicates the color impact of each light sources. The graph is plotted separately for each illuminants to have extra explanation. The illuminant D65 has neutral color impact in its white light which is indicated in image: 1 (marked as red arrows), whereas A illuminant has a yellowishness impact in its white light image: 2. To further strengthens this claim, I plotted the white point values in the graph and it validates my assumption image: 3). The values of patches are shift from blue region to yellow region for D65, D50 and A. So illuminant has greater intereference in the perception of color.

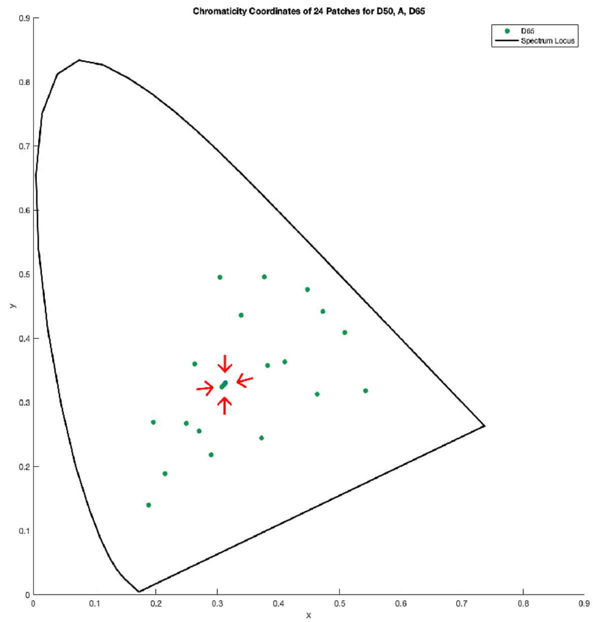


Image 1

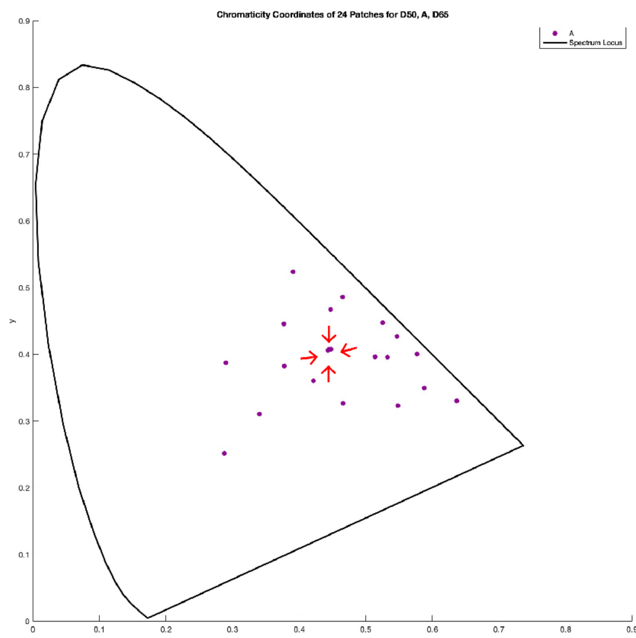


Image 2

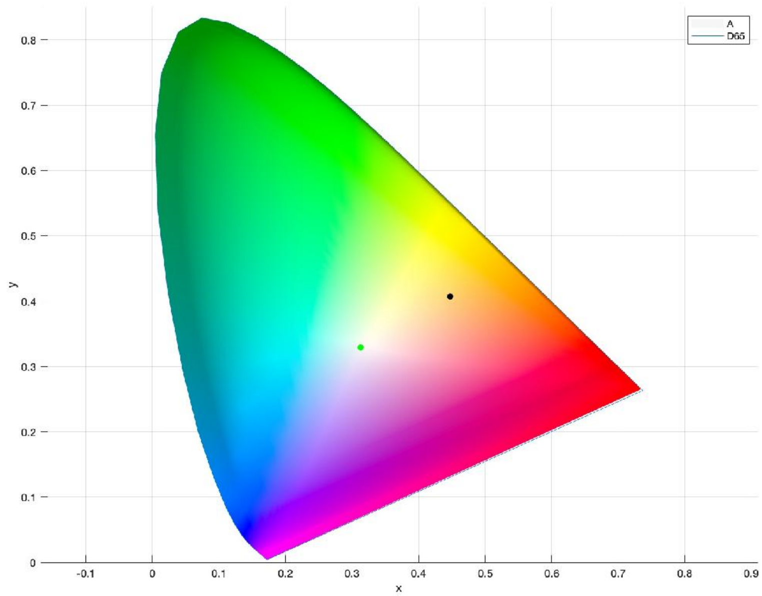


Image 3 (Incorrect legend in the image. Correct legend: Green - D65 illuminant; Black - A illuminant)

Question 5

Why did I not ask you to plot the 10-deg (x,y) values with the rest?

Answer:

Plotting viewing angle will have impact on xyY 3d plot because viewing angle has significantly different value in z axis (lightness) due to changes in illuminant strength for 10 degree to tackle rod instrusion. . So representing different degrees in 2d plot unit plane is insufficient and it should be projected obliquely. Also, If we plot the 10 deg values along with 2-deg values in xy graph, the value will be overlapped in the plot since the value difference between 2-degree and 10-degree is less significant when compared to illuminant value difference.

Question 6 (Bonus)

```
%Get spectral radiance of Source A and Source D65
sources_A = interpolateData(source_dataset{:,1},source_dataset.A,intp_wavelength_info)
sources_D65 = interpolateData(source_dataset{:,1},source_dataset.D65,intp_wavelength_i

% Tristimulus value of Source A and Source D65 for 2 degree observer
wp_A_two_deg_source = calcTristimulusSource(xyz_std_obs_two_deg,sources_A,d_lambda);
wp_D65_two_deg_source = calcTristimulusSource(xyz_std_obs_two_deg,sources_D65,d_lambda
```

```
% Chromaticity Coordinates of Source A and Source D65 for 2 degree observer
cc_wp_A_two_deg = calChromacityCoordinates(wp_A_two_deg_source');
cc_wp_D65_two_deg = calChromacityCoordinates(wp_D65_two_deg_source');
```

Question 7

Using the 2-deg XYZ values for both D65 and A illuminants, calculate CIELAB values (L*, a*, b*, and C*). Please put these in a table, one row per patch. Note that you will need to determine what you will use as your white XnYnZn for each calculation, and please list the values in your answer.

- Calculated L*, a*, b* and C* using the below equation (written as function in the end)

$$L^* = 116 f\left({}^{Y}\!/_{Y_n}\right) - 16$$

$$a^* = 500\left[f\left({}^{X}\!/_{X_n}\right) - f\left({}^{Y}\!/_{Y_n}\right)\right]$$

$$b^* = 200\left[f\left({}^{Y}\!/_{Y_n}\right) - f\left({}^{Z}\!/_{Z_n}\right)\right]$$

where

$$f(x) = \begin{Bmatrix} x^{1/3} & x > \left({}^{24}\!/_{116}\right)^3 \\ \left({}^{841}\!/_{108}\right)x + {}^{16}\!/_{116} & x \le \left({}^{24}\!/_{116}\right)^3 \end{Bmatrix}$$

$$C^*_{ab} = \sqrt{(a^*)^2 + (b^*)^2}$$

```
%Calculate L_star, a_star, b_star, C_star for all patches in A and D65 source
[L_A, a_A, b_A, C_A] = calcXYZtoCIELAB(tristimulus_XYZ_A_two_deg,wp_A_two_deg_source);
[L_D65, a_D65, b_D65, C_D65] = calcXYZtoCIELAB(tristimulus_XYZ_D65_two_deg,wp_D65_two_
```

Question 7: Answer

```
Table_CIELAB_A = array2table([L_A; a_A; b_A; C_A]');
Table_CIELAB_A.Properties.VariableNames(1:4) = {'L*','a*','b*','C*'};
Table_CIELAB_A.Properties.Description = 'CIELAB of 24 patched for Illuminant A';
Table_CIELAB_A
```

Table_CIELAB_A = 24×4 table

	L*	a*	b*	C*
1	39.5437	16.8365	19.2800	25.5966
2	68.5780	20.7315	20.3537	29.0528
3	48.9852	-9.7193	-23.3837	25.3231
4	42.3058	-11.3227	20.0167	22.9972
5	55.8668	6.6138	-24.2490	25.1348
6	68.4021	-30.4098	-5.5113	30.9052
7	66.4956	33.5201	65.4149	73.5031

	L*	a*	b*	C*
8	38.5617	-0.1023	-44.0261	44.0263
9	56.5829	47.7269	25.4067	54.0680
10	31.7440	18.7837	-17.6562	25.7792
11	71.8561	-17.0053	52.1382	54.8414
12	75.7633	20.8776	70.1710	73.2109
13	27.0997	2.5465	-54.0655	54.1254
14	53.4963	-36.1874	27.1895	45.2636
15	47.8516	56.7308	37.6887	68.1089
16	84.4112	9.8062	77.2706	77.8904
17	55.5554	48.0971	-4.7397	48.3300
18	46.8937	-32.5010	-35.6449	48.2377
19	95.4764	0.0418	0.5119	0.5136
20	80.9755	0.0534	0.1227	0.1338
21	66.3801	-0.1641	-0.0671	0.1772
22	52.1803	-0.1298	-0.0717	0.1483
23	36.4288	-0.4857	-0.5099	0.7042
24	21.3480	-0.3158	-0.9764	1.0262

```
Table_CIELAB_D65 = array2table([L_D65; a_D65; b_D65; C_D65]');
Table_CIELAB_D65.Properties.VariableNames(1:4) = {'L*','a*','b*','C*'};
Table_CIELAB_D65.Properties.Description = 'CIELAB of 24 patched for Illuminant A';
Table_CIELAB_D65
```

Table_CIELAB_D65 = 24×4 table

	L*	a*	b*	C*
1	37.3036	13.6919	15.5636	20.7291
2	66.2002	14.4669	17.7397	22.8907
3	50.7810	-1.4728	-21.2662	21.3172
4	42.7404	-16.2982	22.3439	27.6565
5	56.4675	11.5176	-24.3994	26.9812
6	71.3712	-31.3929	1.9816	31.4554
7	61.0686	31.1258	57.1632	65.0880
8	40.8280	15.3969	-41.8876	44.6278
9	50.9517	45.9207	15.0859	48.3352
10	30.6956	23.9006	-22.0728	32.5337
11	72.0005	-27.1827	58.0333	64.0840

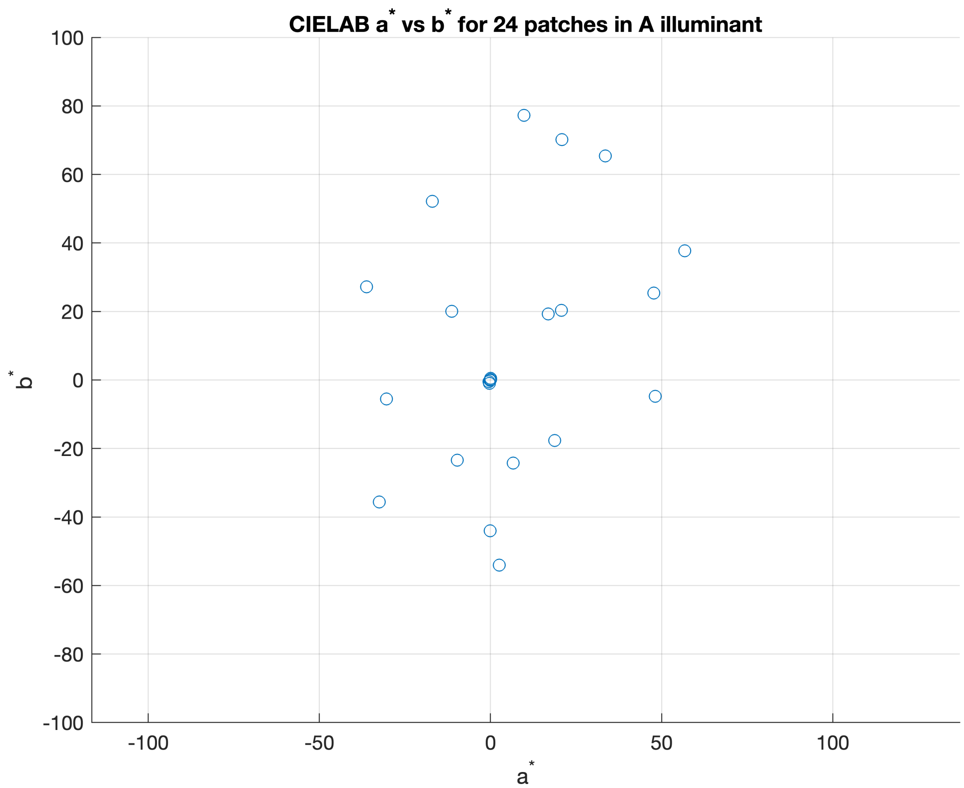
	L*	a*	b*	C*
12	71.6424	15.3238	65.8839	67.6425
13	29.9861	24.6090	-50.8654	56.5057
14	55.6552	-41.6824	34.7747	54.2835
15	40.9374	52.8479	25.6074	58.7251
16	81.6408	-1.5754	79.4743	79.4899
17	51.0001	49.4247	-15.0391	51.6621
18	51.6863	-24.7271	-25.9823	35.8679
19	95.4648	-0.3571	0.7780	0.8561
20	80.9525	0.1417	0.1331	0.1944
21	66.3800	0.0467	-0.0714	0.0853
22	52.1807	0.0580	-0.0855	0.1033
23	36.4781	-0.1904	-0.4746	0.5114
24	21.4126	-0.0341	-0.9470	0.9476

Question 8:

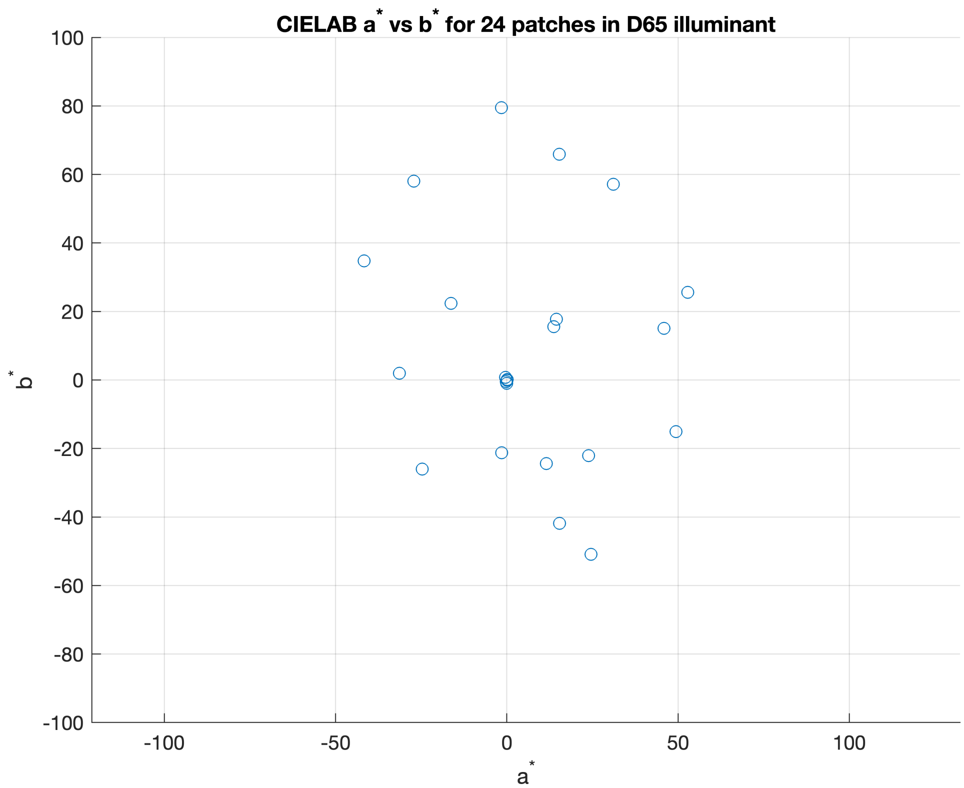
Plot the L*, C* and a*, b* values of each patch on the a*b* and L*C* planes in CIELAB space (see the examples in the book for proper orientation, for example L* is typically the vertical axis). Make two sets of plots: one for D65 and one for A. Please label the points with patch number and/or color them appropriately. In these plots of a uniform color space, it is important to scale the x-axis and y-axis equally (*think about why*). In Matlab please use the command **axis equal**.

Note: The patch number is added as text which will appear when the cursor is clicked on a plot value. This is done to minimize text noise in visualisation.

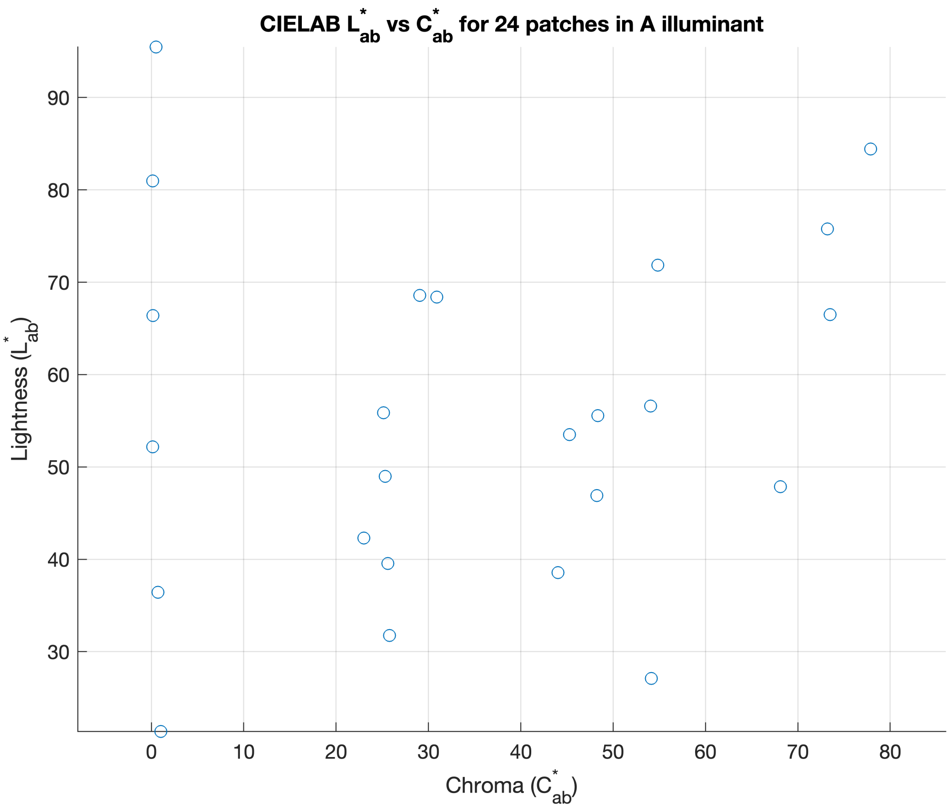
```
%Plot a against b under source A
h = figure(3);
hold on
scatter(a_A',b_A');
xlabel('a^{*}');
ylabel('b^{*}');
title('CIELAB a^{*} vs b^{*} for 24 patches in A illuminant');
xlim([-100 100]);
ylim([-100 100]);
axis equal
grid on
dcm_obj = datacursormode(h);
set(dcm_obj,'UpdateFcn',{@custom_label})
hold off
```



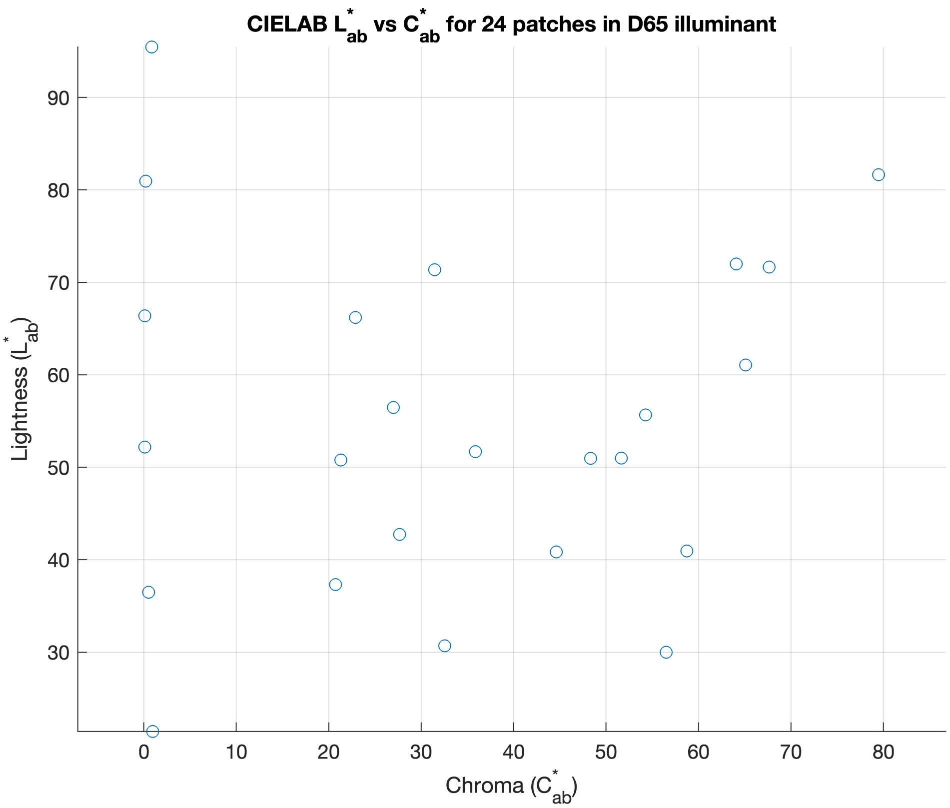

```
h = figure(4);
hold on
scatter(a_D65',b_D65');
xlabel('a^{*}');
ylabel('b^{*}');
title('CIELAB a^{*} vs b^{*} for 24 patches in D65 illuminant');
xlim([-100 100]);
ylim([-100 100]);
axis equal
grid on
dcm_obj = datacursormode(h);
set(dcm_obj,'UpdateFcn',{@custom_label})
hold off
```




```
h = figure(5);
hold on
scatter(C_A',L_A');
xlabel('Chroma (C^{*}_{ab})');
ylabel('Lightness (L^{*}_{ab})');
title('CIELAB L^{*}_{ab} vs C^{*}_{ab} for 24 patches in A illuminant');
axis equal
grid on
dcm_obj = datacursormode(h);
set(dcm_obj,'UpdateFcn',{@custom_label})
hold off
```




```
h = figure(6);
hold on
scatter(C_D65',L_D65');
xlabel('Chroma (C^{*}_{ab})');
ylabel('Lightness (L^{*}_{ab})');
title('CIELAB L^{*}_{ab} vs C^{*}_{ab} for 24 patches in D65 illuminant');
axis equal
grid on
dcm_obj = datacursormode(h);
set(dcm_obj,'UpdateFcn',{@custom_label})
hold off
```



Question 9:

What do you observe about the locations of Color Checker patches on these plots for the different illuminants? How does the use of XnYnZn relate to what we did previously with von Kries chromatic adaptation?

Observation in a* vs b* plot for illumination D65 and A

In plotting the a* and b* graph, we could notice proper grouping of opponent colors - red against green in x axis representing a* value and yellow and blue in y axis representing b* value.

- Patches 8,10,13 represents the bluish purple shades in the patches. Since it is opponent to yellow in the repesentation, the illuminant A (which has yellowish impact) minimise the spread of these 3 patches in the plot when compared to illuminant D65.

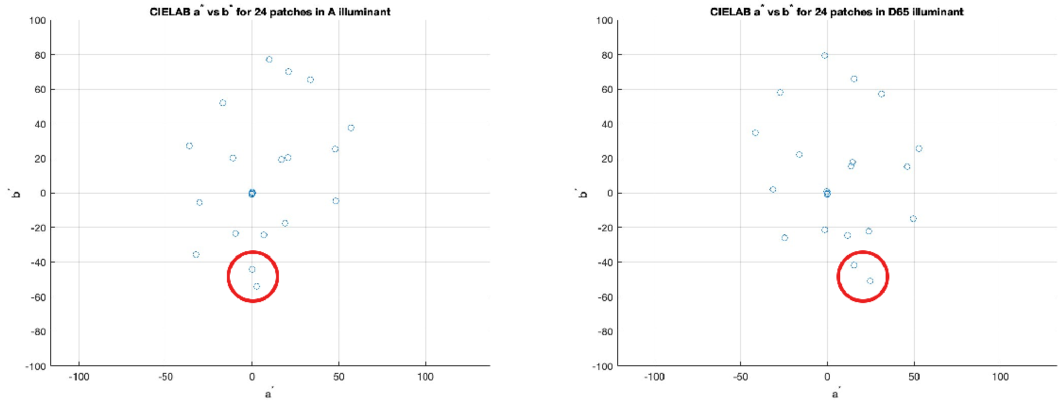


Image 1

- The greyscale patches of different lightness (19-24) are crowded at centre 0,0 in the plot. We couldn't notice the distinction in plotting due to graph's inability to represent lightness along an axis (Image 2).

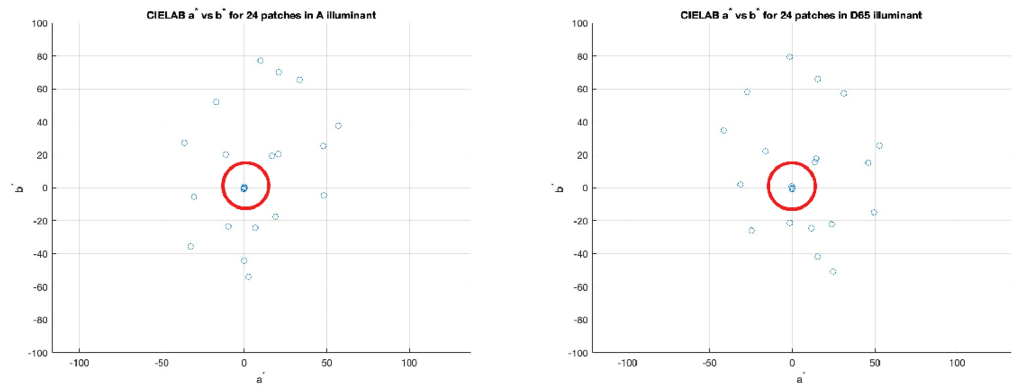


Image 2

Observation in Lightness vs Chroma plot for illumination D65 and A

- D65 have better distinction in chroma for the color patches when compared to illuminant A. This is very evident in image 3 where the patches located between chroma values 20 -30.

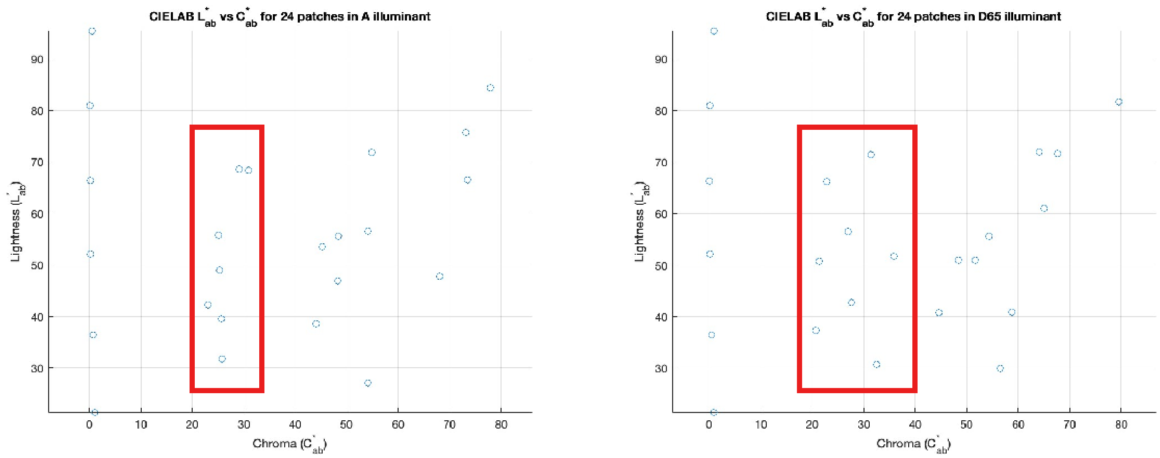


Image 3

- There is noticeable enhancement in chroma for yellow patches 7, 12, 16 in illuminant A when compated to illuminant D65. This is due to the yellowish nature of the illuminant A.

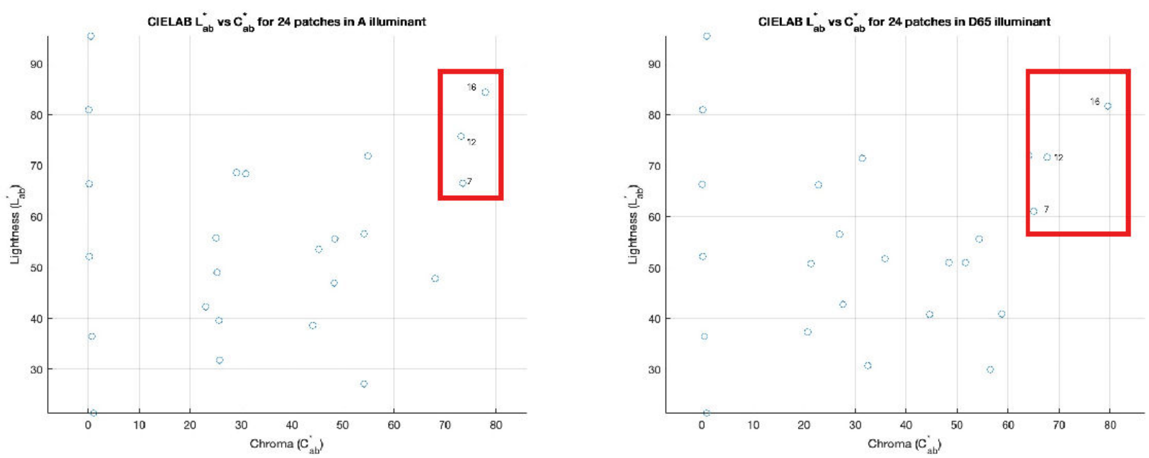


Image 4

Image 5: 24 patches used for calculation

Von Kries model helps to predict the corresponding light under two 2 different illumination condition. The similar concept of normalising the calculated LMS value of material by LMS value of light source/white point is relatable in the Tristimulus calculation. Here, we are dividing the calculated Tristimulus value of the material by the tristimulus value of the source/white point (XnYnZn) to caluculate the XYZ primes.

Functions

```
%Function to interpolate the values to a consistent wavelength
```

```
function i = interpolateData(wavelengths,values,intp_wavelength_data)
    %Create the interpolant by passing wavelength and corresponding values to griddedI
    GI = griddedInterpolant(wavelengths,values);

    % Create a vector of query points with 5nm spacing for 380 to 780 wavelength.
    wl = intp_wavelength_data.min:intp_wavelength_data.range:intp_wavelength_data.max;

    % Evaluate the interpolant at the each wavelength for each value set
    i = GI(wl);
end
```

%Function to calculate Tristimulus values for materials

```
function t = calcTristimulus(xyz_value,source,material,d_lambda)

    %Calculate normalizing constant
    k = (100/(source * xyz_value(:,2) * d_lambda))/100;

    s_lambda = diag(source);

    %Calculating tristimulus
    t = k.*((s_lambda*xyz_value)'*material)*d_lambda;

%   t = custom_normalization(t);
end


%Function to calculate Chromacity Coordinate
function cc = calChromacityCoordinates(tristimulus_XYZ)
    sum_XYZ = sum(tristimulus_XYZ,1);
    cc = tristimulus_XYZ./sum_XYZ;
end
```

```
%Function to calculate Tristimulus values for sources (Whitepoint)
function ts = calcTristimulusSource(xyz_value,source,d_lambda)

    %Calculate normalizing constant
    k = (100/(source * xyz_value(:,2) * d_lambda))/100;

    %Calculating tristimulus
    ts = k.*((source*xyz_value))*d_lambda;

%   t = custom_normalization(t);
end


%Function to calculate CIELAB from XYZ
function [L_star,a_star,b_star,C_star] = calcXYZtoCIELAB(XYZ,whitepoint);
    XYZ_Prime = calcXYZPrime(XYZ,whitepoint);
    L_star = (116 * calConstants(XYZ_Prime(2,:))) - 16;
    a_star= 500*(calConstants(XYZ_Prime(1,:)) - calConstants(XYZ_Prime(2,:)));
    b_star = 200*(calConstants(XYZ_Prime(2,:)) - calConstants(XYZ_Prime(3,:)));
    C_star = ((a_star).^2 + (b_star).^2).^(1/2);
end


% Calculate X Prime, Y Prime and Z Prime
function XYZ_Prime = calcXYZPrime(XYZ,whitepoint)
    XYZ_Prime = XYZ./whitepoint';
end


%Function to calculate constants in a and b
function k = calConstants(x)
    if (x > (24/116)^3)
        k = (x).^(1/3);
    else
        k = ((841/108).*x) + (16/116);
    end
end


%Function to create table and assign x,y,z
function a2t = createTablexyz(values)
    a2t = array2table(values');
    a2t.Properties.VariableNames(1:3) = {'x','y','z'};
end


% Function to customize text of data tips in the plot
function txt = custom_label(~,event_obj)
pos = get(event_obj,'Position');
I = get(event_obj, 'DataIndex');
```

```
txt = {['X: ',num2str(pos(1))],...
       ['Y: ',num2str(pos(2))],...
       ['Patch: ',num2str(I)]};
end

% Function to normalize a matrix with its peak value (not using default normalize func
function n = custom_normalization(x)
    max_value = max(x, [], 'all');
    n = x/max_value;
end
```