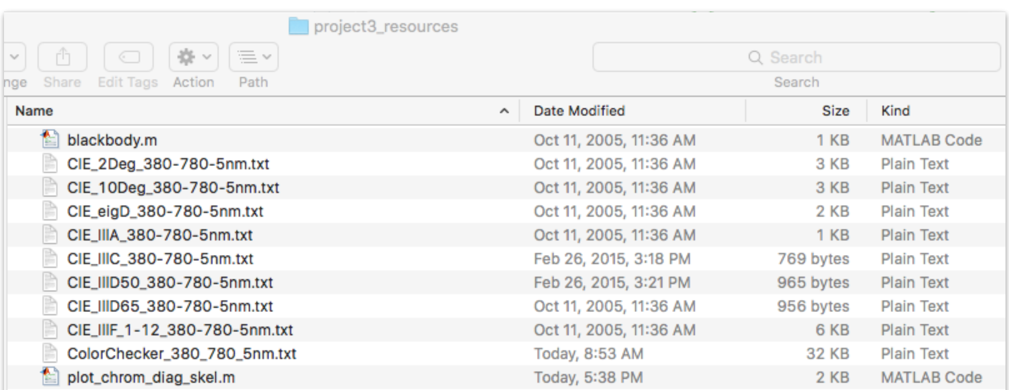
Project 3: Colorimetry

In this project you will develop MATLAB functions to load a structure of CIE observer and illuminant data, calculate CIE XYZ tristimulus values from reflectance spectra, and calculate x,y chromaticity coordinates from the XYZ values. You will test your functions by calculating the tristimulus values and chromaticity coordinates of the patches in the ColorChecker chart. You will then use your functions to calculate the tristimulus values and chromaticity coordinates of your real, imaged, and matching patches from the spectral measurements you made in Project 2, and compare the measured values and the calculated values.

1) Download and unpack the “project3_resources.zip” file provided in myCourses to your working directory.



2 a) Create a MATLAB <u>function</u> **cie = loadCIEdata** that returns a structure of CIE observer and illuminant data by performing the following steps.

a. Create a new function in the editor with the header **function [cie] = loadCIEdata**

b. Inside the function define a structure with the following fields (lambda, cmf2deg, cmf10deg, illA, illC, illD50, illD65, illE, illF, eigD).

c. Fill the structure by reading in the text files listed below and loading the data into the appropriate fields of the structure.

```
CIE_2Deg_380-780-5nm.txt
CIE_10Deg_380-780-5nm.txt
CIE_eigD_380-780-5nm.txt
CIE_IllA_380-780-5nm.txt
CIE_IllC_380-780-5nm.txt
CIE_IllD50_380-780-5nm.txt
CIE_IllD65_380-780-5nm.txt
CIE_IllF_1-12_380-780-5nm.txt
ColorChecker_380_780_5nm.txt
```

First store the <u>wavelength data</u> from the file CIE_2Deg_380-780-5nm.txt as a 1x81 row vector in the "lambda" field. Then store the remaining <u>color matching function data</u> as an 81x3 column array in the "cmf2deg" field. Next, remove the wavelength information from each of the remaining datasets by deleting the first column of each loaded matrix, and then store the remaining data in the corresponding field. (Note that that CIE_IllF_1-12_380-780-5nm.txt contains 12 illuminants)

d. Include the CIE equal energy illuminant E in your structure by creating a vector in the "illE" field with a length equal to the wavelength range and a constant value of 1.0.

e. When called as shown below, your function should produce the following output:

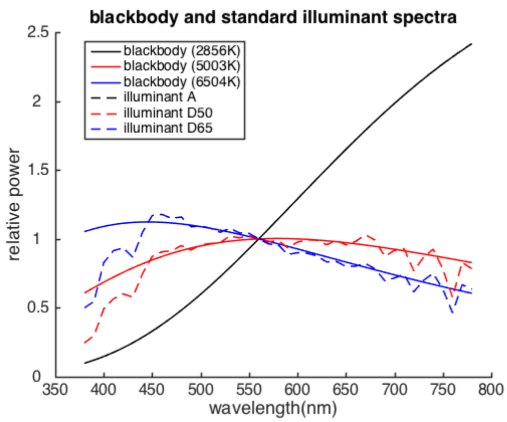
```
>> cie = loadCIEdata

cie =

      lambda: [1x81 double]
     cmf2deg: [81x3 double]
    cmf10deg: [81x3 double]
        illA: [81x1 double]
        illC: [81x1 double]
      illD50: [81x1 double]
      illD65: [81x1 double]
        illE: [81x1 double]
        illF: [81x12 double]
        eigD: [81x3 double]
```

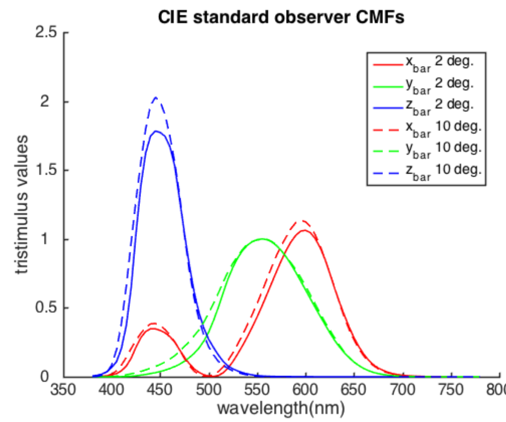
f. Confirm that the returned structure contains the correct data in the correct formats.

2 b) Include a listing of your loadCIEdata function in your published report by including the code below in your project script.

```
%%
% include a listing of the indicated function in a published report
% (note that the "include" markup syntax must be part of a
% contiguous comment block that starts with a section marker %%)
%
% <include>loadCIEdata.m</include>
```

3) a) Test your loadCIEdata function by reproducing the graphs shown below (including all text and styles). The function **blackbody.m** has been provided to produce the spectral data for the blackbody curves. Read the function header to understand how to use the function. The illuminant data is normalized to 100 at 560nm. You will need to re-normalize the this data to 1.0 so you can plot the illuminant and blackbody spectra on the same scale. b) Include these graphs in your report.



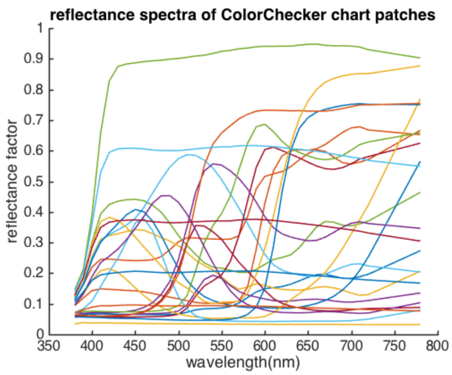


4) The equations for calculating XYZ tristimulus values from surface reflectance R(λ), color matching function $\bar{x}(\lambda)$,$\bar{y}(\lambda)$,$\bar{z}(\lambda)$, and illumination S(λ) data are shown below. a) Create a function **XYZ = ref2XYZ(refs, cmfs, illum)** that takes as input: ‘refs’ an (nx1) vector of reflectance factor data; ‘cmfs’ an (nx3) set of CIE color matching functions; and ‘illum’ an (nx1) spectral power distribution of a light source, and returns ‘XYZ’ a (3x1) vector of CIE XYZ tristimulus values. b) Include a listing of your function in your report.

$$X = k\sum_{\lambda}\bar{x}(\lambda)S(\lambda)R(\lambda)\Delta\lambda,$$
$$Y = k\sum_{\lambda}\bar{y}(\lambda)S(\lambda)R(\lambda)\Delta\lambda, \qquad k = \frac{100}{\sum_{\lambda}\bar{y}(\lambda)S(\lambda)\Delta\lambda.}$$
$$Z = k\sum_{\lambda}\bar{z}(\lambda)S(\lambda)R(\lambda)\Delta\lambda$$

5) a) Test your ref2XYZ function by calculating XYZ values for the patches in the ColorChecker chart. The spectral data is provided in the file **ColorChecker_380_780_5nm.txt** (see plot). Use the CMFs for the 2 degree observer and illuminant D65 in your calculations. The Matlab code for the test is given below. Confirm that your values match the values shown to within rounding error. b) Include a listing of the results in your report.

```
CC_spectra = importdata('ColorChecker_380_780_5nm.txt');
for patch_num = 2:25
    CC_XYZs(:,patch_num-1) = ref2XYZ(CC_spectra(:,patch_num),cie.cmf2deg,cie.illD65);
end
CC_XYZs
```




```
CC_XYZs =

  Columns 1 through 7

   11.5145   39.1346   18.3488   11.1492   25.8437   31.7110   37.1457
   10.3819   36.5981   19.6332   13.8551   24.3868   43.8600   29.5592
    7.1502   27.0564   35.6470    7.4267   45.6142   44.8778    6.5006

  Columns 8 through 14

   13.8627   29.1328    8.5889   33.9174   46.1864    8.9183   15.0353
   12.3179   19.8475    6.4569   44.1533   42.4957    6.4177   24.1079
   39.3093   14.9941   15.4745   11.4297    8.6771   32.2736    9.6379

  Columns 15 through 21

   19.3447   55.8457   29.6768   14.4138   87.8402   57.9621   35.2286
   11.3576   58.9726   19.3515   19.9750   92.3781   61.0426   37.0414
    5.5526    9.6411   32.2626   39.0008   95.6125   65.4909   40.2256

  Columns 22 through 24

   19.3492    8.7646    3.2111
   20.4708    9.2915    3.3763
   22.1545   10.3188    3.9312
```

6) The equations for calculating x,y chromaticity coordinates from XYZ tristimulus values are shown below. a) Create a function **xyY = XYZ2xyY(XYZ)** that takes as input 'XYZ' a (3xn) vector of XYZ tristimulus values and returns 'xyY' a (3xn) vector of chromaticity coordinates (x,y) and luminance factor (Y). b) Include a listing of your function in your report.

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

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

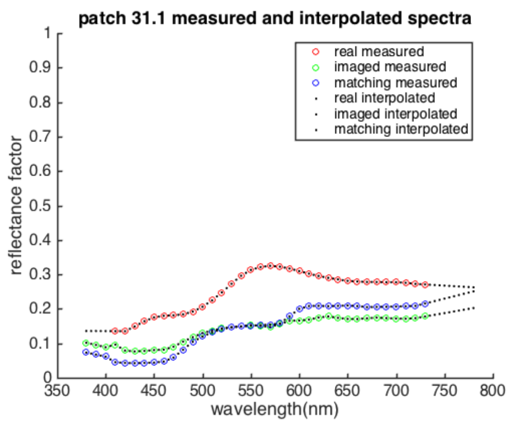
7) a) Test your **XYZ2xyY** function by calculating xyY values for the ColorChecker chart from the XYZ values you calculated in step 5. The Matlab code for the test is given below. Confirm that your values match the values shown to within rounding error. b) Include a listing of the results in your report.

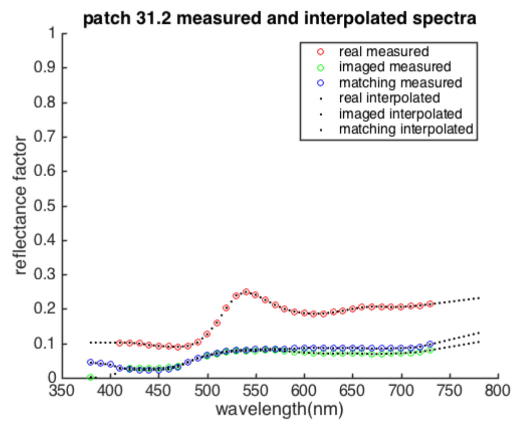
```
CC_xyYs = XYZ2xyY(CC_XYZs)

CC_xyYs =

  Columns 1 through 7

    0.3964    0.3807    0.2492    0.3438    0.2696    0.2633    0.5074
    0.3574    0.3561    0.2667    0.4272    0.2544    0.3641    0.4038
   10.3819   36.5981   19.6332   13.8551   24.3868   43.8600   29.5592

  Columns 8 through 14

    0.2117    0.4554    0.2814    0.3790    0.4744    0.1873    0.3082
    0.1881    0.3102    0.2116    0.4933    0.4365    0.1348    0.4942
   12.3179   19.8475    6.4569   44.1533   42.4957    6.4177   24.1079

  Columns 15 through 21

    0.5336    0.4487    0.3651    0.1964    0.3185    0.3142    0.3132
    0.3133    0.4738    0.2381    0.2722    0.3349    0.3309    0.3293
   11.3576   58.9726   19.3515   19.9750   92.3781   61.0426   37.0414

  Columns 22 through 24

    0.3122    0.3089    0.3053
    0.3303    0.3275    0.3210
   20.4708    9.2915    3.3763
```

8) Use the functions you've just created to calculate the tristimulus values and chromaticity coordinates of your real, imaged, and matching color patches from the spectral data you measured in Project 2. To do this, first adapt the script below to load your spectral data.

```
% load the CIE observer and illuminant data

% define ColorMunki/Argyll/spotread measurement wavelengths
cm_lams = 380:10:730;

% define header offsets for reading the .sp files
cm_h_offset = 19;

% load and normalize the measured spectral data for the patch #1
data = importdata('31.1_real.sp', ' ', cm_h_offset);
real_311 = data.data/100;

data = importdata('31.1_imaged.sp', ' ', cm_h_offset);
imaged_311 = data.data/100;

data = importdata('31.1_matching.sp', ' ', cm_h_offset);
matching_311 = data.data/100;

% repeat the section above for patch #2
```

9) Now extend the script to interpolate your data to the correct sampling interval and range for your ref2XYZ function. The ColorMunki measures spectra at 10nm intervals over a 380-730nm range. Your ref2XYZ function requires spectra defined at 5nm intervals over a 380-780nm range. To use ref2XYZ with the Colormunki data you need to interpolate/extrapolate the data. This can be done using the “interp1” function in Matlab. Use the vector 380:10:730 for the input range, the Colormunki-measured spectral data as the input values, cie.lambda(:) as the output range, ‘linear’ as the ‘method’, and ‘extrap’ as an option. a) Using this technique resample the Colormunki-measured spectral data for your real, imaged, and matching patches. b) Create graphs like the ones shown below that visualize and confirm the process. c) Include the graphs in your report.





10) a) Now use your ref2XYZ function to calculate XYZ tristimulus values for each of your patches from the interpolated spectra. Use the CMFs for the CIE 2º observer and illuminant D50 as input parameters to ref2XYZ. b) Use the “fprintf" function to create a table with the same formatting as the one shown below that summarizes the XYZ values you measured directly with the ColorMunki and the ones you calculated using your ref2XYZ function. (the code to do this should be very similar to the code your wrote in project 2, step 9)). c) The XYZ values for your patches will be different than the ones shown below, but you should confirm that your measured and calculated values are nominally the same. d) Include a listing of the table in your report.

Measured and calculated tristimulus values

patch 31.1

	measured			calculated		
	X	Y	Z	X	Y	Z
real	27.6317	28.9911	14.4655	27.6207	28.9776	14.4681
imaged	14.6797	15.1442	7.3526	14.6796	15.1416	7.3660
matching	15.9330	15.8489	4.8137	15.9317	15.8463	4.8306

patch 31.2

	measured			calculated		
	X	Y	Z	X	Y	Z
real	17.9585	20.3373	8.1219	17.9607	20.3217	8.1352
imaged	6.5926	7.3943	2.8582	6.5927	7.3908	2.8639
matching	7.4234	8.0171	2.5965	7.4239	8.0139	2.6067

11) a) Now use your XYZ2xyY function to calculate chromaticity coordinates from the measured and calculated XYZ values you tabulated in the previous step. b) Use the fprintf function to created a table with the same formatting as the one shown below the summarizes the measured and calculated x,y,Y values. c) The XYZ values for your patches will be different than the ones shown below, but you should confirm that your measured and calculated values are nominally the same. d) Include a listing of the table in your report.

Measured and calculated chromaticity coordinates

patch 31.1

	measured			calculated		
	x	y	Y	x	y	Y
real	0.3887	0.4078	28.9911	0.3887	0.4078	28.9776
imaged	0.3949	0.4074	15.1442	0.3947	0.4072	15.1416
matching	0.4354	0.4331	15.8489	0.4352	0.4329	15.8463

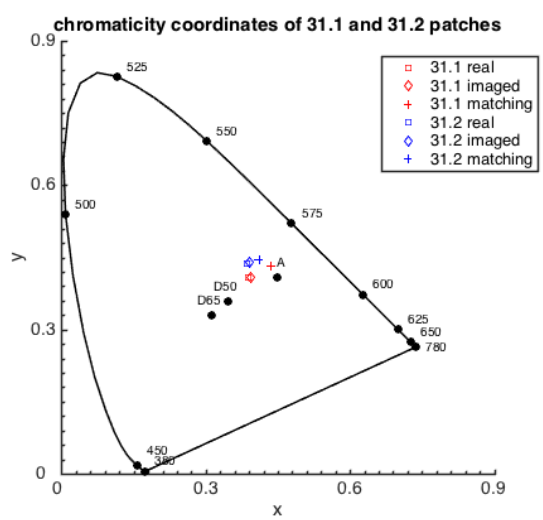
patch 31.2

	measured			calculated		
	x	y	Y	x	y	Y
real	0.3869	0.4381	20.3373	0.3869	0.4378	20.3217
imaged	0.3914	0.4390	7.3943	0.3913	0.4387	7.3908
matching	0.4116	0.4445	8.0171	0.4114	0.4441	8.0139

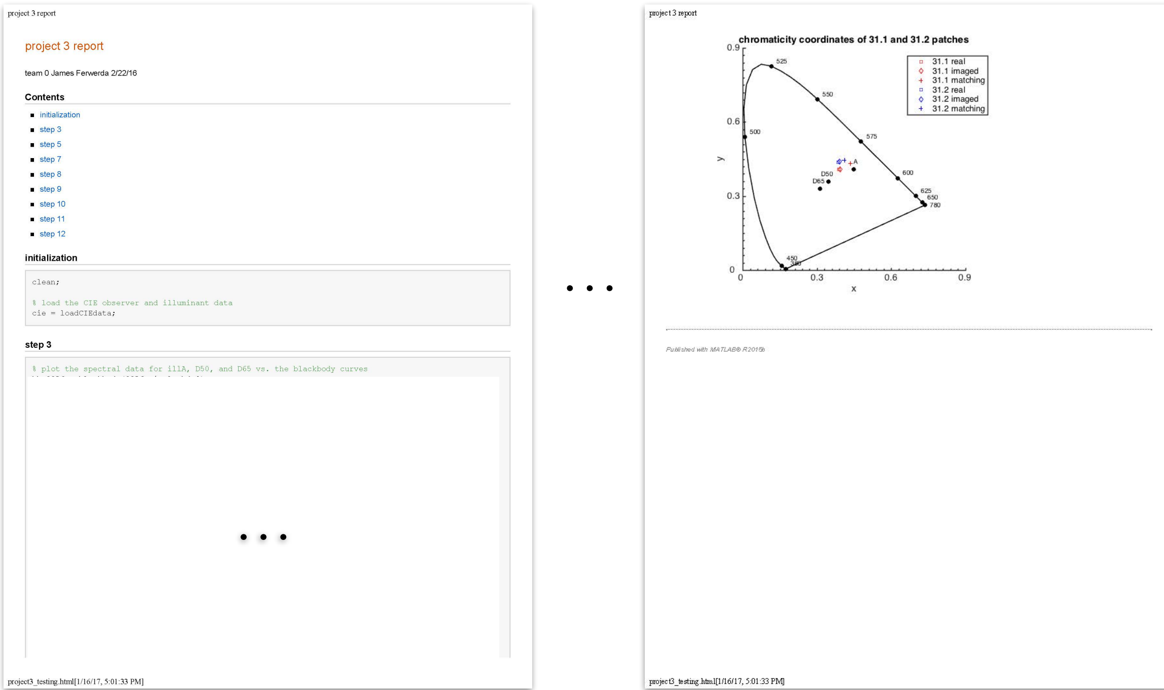
12) a) Visualize the calculated patch chromaticities by plotting their values on a clearly formatted chromaticity diagram as like the one shown below. The function **plot_chrom_diag_skel.m** has been provided to plot the skeleton of the diagram. You should use the function by just including the line

```
plot_chrom_diag_skel;
```

in your project script, and then adding code to plot your data and add a legend on top of the figure skeleton. Make sure that the glyphs for the data are distinct and that the legend does not obscure the graph. b) Include the graph in your report.



13) a) Use the “publish” menu/function in Matlab to document all the code, results, and figures you generated in steps 2-12. b) Include your names and team number at the beginning of the report. c) Include a feedback section at the end of the report that briefly discusses i) who did what parts of the project, ii) any problems you had with the project, iii) any parts of the project you thought were valuable, and iv) any improvements you’d like to see. d) Submit this document as a single PDF named “teamX_project3_report.pdf” to the dropbox on myCourses.



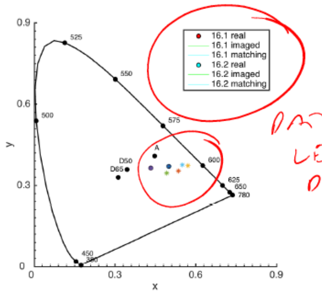
project grading rubric

- completeness (do all tasks)
- accuracy (do all tasks correctly)
- presentation quality (clarity, formatting)
- code quality (structuring, use of language features, commenting)

Project 3 post-mortem

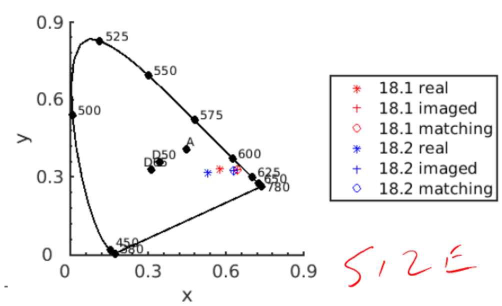
a.k.a. mistakes to avoid

Presentation quality



DATA & LEGEND DON'T MATCH

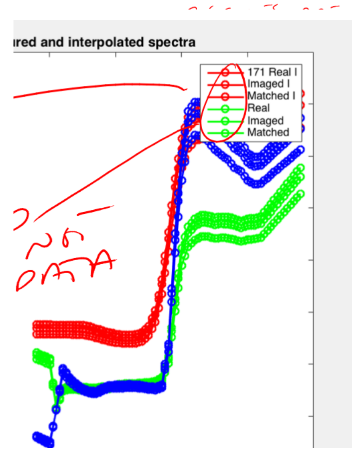
−0.5



SIZE ?

Step 10 Table:

Tristimulus Values	Patch # 1 (16.1)						Patch # 2 (16.2)					
	Measured			Calculated			Measured			Calculated		
	X	Y	Z	X	Y	Z	X	Y	Z	X	Y	Z
Real	12.671	9.367	3.273	12.661	9.374	3.273	6.704	5.610	3.127	6.699	5.612	3.123
Imaged	27.885	18.360	5.827	27.865	18.384	5.840	7.878	5.511	2.580	7.873	5.517	2.584
Matched	9.108	5.936	0.878	9.100	5.943	0.879	2.648	1.806	0.354	2.644	1.808	0.348



NOT DATA

Code quality

```
plot(0.31271,0.32902,'ko','MarkerFac

% D50
text(0.34567-0.03,0.35850+0.03,'D50'
plot(0.34567,0.35850,'ko','MarkerFac

% A
text(0.44757,0.40745+0.03,'A');
plot(0.44757,0.40745,'ko','MarkerFac


% 29.1 real
%text(   ,    , 'square');
data1 = plot( 0.3596,
 0.4255, 'Marker', 'square','MarkerF

% 29.1 imaged
%text(   ,    , 'diamond');
data2 = plot(  0.3079,
 0.3418, 'Marker','diamond','MarkerF

% 29.1 matching
%text(   ,    , '+');
data3 = plot( 0.354,0.4565 , 'Marker

% 29.2 real
%text(   ,    , 'square');
data4 = plot( 0.3375,
 0.3966 , 'Marker','square','MarkerF

% 29.2 imaged
%text(   ,    , 'diamond');
data5 = plot(  0.2643,
 0.2988, 'Marker','diamond','MarkerF

% 29.2 matching
%text(   ,    , '+');
data6 = plot(  0.3606,  0.4549,'Mark

legend([data1, data2, data3, data4, 
 imaged', '29.1 matching', ...
    '29.2 real', '29.2 imaged', '29.
    'Location', 'northeast', 'FontSi
title('chromaticity coordinates of 2
```

don't hand-code data

```
%%
% step 12

% plot the chromaticity diagram skeleton
plot_chrom_diag_skel;

line_weight = 1.5


% plot the data for 31.1
x = xyYcalc.real_311(1);
y = xyYcalc.real_311(2);
h(1) = plot(x,y,'rs','LineWidth', line_weight);

x = xyYcalc.imaged_311(1);
y = xyYcalc.imaged_311(2);
h(2) = plot(x,y,'rd','LineWidth', line_weight);

x = xyYcalc.matching_311(1);
y = xyYcalc.matching_311(2);
h(3) = plot(x,y,'r+','LineWidth', line_weight);

% plot the data for 31.2
x = xyYcalc.real_312(1);
y = xyYcalc.real_312(2);
h(4) = plot(x,y,'bs','LineWidth', line_weight);

x = xyYcalc.imaged_312(1);
y = xyYcalc.imaged_312(2);
h(5) = plot(x,y,'bd','LineWidth', line_weight);

x = xyYcalc.matching_312(1);
y = xyYcalc.matching_312(2);
h(6) = plot(x,y,'b+','LineWidth', line_weight);

legend(h, '31.1 real','31.1 imaged', '31.1 matching',...
    '31.2 real','31.2 imaged', '31.2 matching','Location','best');
title('chromaticity coordinates of 31.1 and 31.2 patches');
```

don't edit supplied functions

```
%Load CIE Information from a series of provided text files.
%Loads into a struct titled cie with 10 defined properties

%Separate functions are defined to load either the first column
%of a textfile (importLambda), the last three columns (importCMF),
%the second column(importCData), or the final 12 columns (importIllF)

%Authors: Team 14

function[cie] = loadCIEData()

cie.lambda = importLambda('CIE_2Deg_380-780-5nm.txt')';

cie.cmf2deg = importCMF('CIE_2Deg_380-780-5nm.txt');
cie.cmf10deg = importCMF('CIE_10Deg_380-780-5nm.txt');
cie.eigD = importCMF('CIE_eigD_380-780-5nm.txt');

cie.illA = importCData('CIE_IllA_380-780-5nm.txt');
cie.illC = importCData('CIE_IllC_380-780-5nm.txt');
cie.illD50 = importCData('CIE_IllD50_380-780-5nm.txt');
cie.illD65 = importCData('CIE_IllD65_380-780-5nm.txt');

cie.illF = importIllF('CIE_IllF_1-12_380-780-5nm.txt');

cie.illE = linspace(100,100,81)';

%Graph blackbody plot as noted in step 3
cla

plot(cie.lambda,blackbody([2856,5003,6504],cie.lambda), cie.lambda, ...
    (cie.illA ./100), '--', cie.lambda, (cie.illD50 ./ 100), '--',...
    cie.lambda,(cie.illD65 ./ 100), '--')
title('blackbody and standard illuminant spectra')
axis([350 800 0 2.5])
xlabel('wavelength(nm)');
ylabel('relative power');
legend('blackbody 2856k', 'blackbody 5003k', 'blackbody 6504k',...
    'Illuminant A', 'Illuminant D50', 'Illuminant D65')
```

write a main project script that calls required external functions

```
%making sure illE is all 100s
h = [100, 100, 100, 100, 100, 100, 100, 100, 100, 100, 100, 100, 100,
 100, 100, 100 , 100, 100, 100, 100, 100, 100, 100, 100 , 100, 100,
 100, 100, 100, 100, 100, 100 , 100, 100, 100, 100, 100, 100, 100,
 100 , 100, 100, 100, 100, 100, 100, 100, 100 ,100, 100, 100, 100,
 100, 100, 100, 100 ,100, 100, 100, 100, 100, 100, 100, 100, 100 100,
```

vs.

```
cie.illE = ones(
```

```
sumx = 0;
sumy = 0;
sumz = 0;

%does summation for calculating XYZ values
for m = 1:n
    a = x(m) * illum(m) * ref(m) * dL;
    sumx = sumx + a;
    b = y(m) * illum(m) * ref(m) * dL;
```

NO NEED TO INIT

take advantage of MATLAB's capabilities