A LON publication

Colorimetry v1.33 User Manual

Written by J.Romaya W.D.C.N.

12th August 2015

Terms and conditions

Cogent Graphics is provided to you free of charge under the following conditions:-

1) Acknowledgement

Acknowledgement from users helps us justify the time we are spending further developing and maintaining this free software. Therefore we request that, when you use Cogent Graphics for your experiments, you include the following statement in your publication:

"This experiment was realised using Cogent Graphics developed by John Romaya at the LON at the Wellcome Department of Imaging Neuroscience."

2) Copying

This package may be distributed freely as long as it is distributed in its original form. It may not be sold without permission.

3) Liability

The package is distributed as it is, without any warranty implied. No liability is accepted for any damage or loss resulting from the use of these routines.

Changes v1.32 to v1.33

1/ Function xyz2rgb has been updated. It is now much faster when dealing with out-of-gamut conditions. When dealing with error 2 (high Y values) there are now two correction options, either to reduce the brightness while keeping the brightness constant (the previous default) or to desaturate the colour and attempt to maintain the brightness. There is also an option to return uncorrected linear RGB values instead of correcting them.

Changes v1.30 to v1.32

1/ Function dspcalib now defaults to the PR-670 photometer rather than the previous PR-650. The photometer model can be specified as an additional argument. When using the PR-670 the aperture size can be specified.

2/ Function dspcalib has been modified so that it now also saves the spectra of the principal colours of the display. Function readdcf has been modified accordingly and the dspalyze function can now display these radiant spectra.

3/ The dspcheck and xyzcheck functions have been updated.

Changes v1.29 to v1.30

1/ Function xyz2rgb now works correctly with array arguments which entail a gamut repositioning towards the white point.

Changes v1.29

1/ Functions xyz2rgb and rgb2xyz now work much more efficiently with arrays

2/ Script dspcalib has been modified. If CalPoints is negative we use cgphotometer('XYZB') to average over several measurements.

Changes v1.27 to v1.28

There are no changes between v1.27 and v1.28

Changes v1.25 to v1.27

1/ Bug fix – xyz2rgb will now work with a monitor calibration file which has repeated RGB data points.

Changes v1.24 to v1.25

1/ Bug fix – dspalyze now shows the monitor phosphor CIE XYZ and xy values correctly on the CIE plot.

2/ New functions added rgb2xyz and xyz2rgb. These functions allow you to convert between RGB and CIE (1931) XYZ values.

3/ dspcalib has now been changed so that you can specify monitor resolution, bit depth, refresh rate and monitor number.

4/ dspalyze now displays the monitor number on the notes page.

5/ New utilities added – dspcheck and XYZDemo

Changes v1.17 to v1.24

There are no changes between v1.17 and v1.24.

Changes v1.16 to v1.17

The dspcalib function has been modified so that it works with the new v1.17 time functions which return a time in seconds as well as the pre-v1.17 functions which returned a time in microseconds.

Changes v1.08 to v1.16

There are no changes between v1.08 and v1.16. The version number has simply been incremented to keep in step with the other graphics libraries.

Changes v1.07 to v1.08

1/ Global variables have been tidied up and are cleared when the function exits.

The cover shows the false clown anemonefish Amphipron Ocellaris. Though brightly coloured and highly visible they are safe within the stinging cells of the anemone.

Table of Contents

Introduction

This manual describes a number of colorimetric tools provided to supplement the Cogent Graphics package. These may be downloaded from the Cogent Graphics website. The tools provided are:-

cieplot Plot a CIE diagram.

dspalyze Analyze a display calibration file (dcf).

dspcalib Calibrate a display for colorimetry and save a display calibration file (dcf).

readdcfRead a display calibration file into a structure in the Matlab workspace.

rgb2xyz Convert RGB values into CIE (1931) XYZ values.

spcplot Plot a radiant spectrum.

spc2xyz Convert a radiant spectrum into CIE (1931) XYZ values.

xyy2xyz Convert CIE (1931) xyY values into CIE (1931) XYZ values.

xyz2rgb Convert CIE (1931) values into RGB triplets.

xyz2xyy Convert CIE (1931) XYZ values into CIE (1931) xyY values.

A knowledge of colorimetry is assumed in this manual and the reader is referred for further information to:-

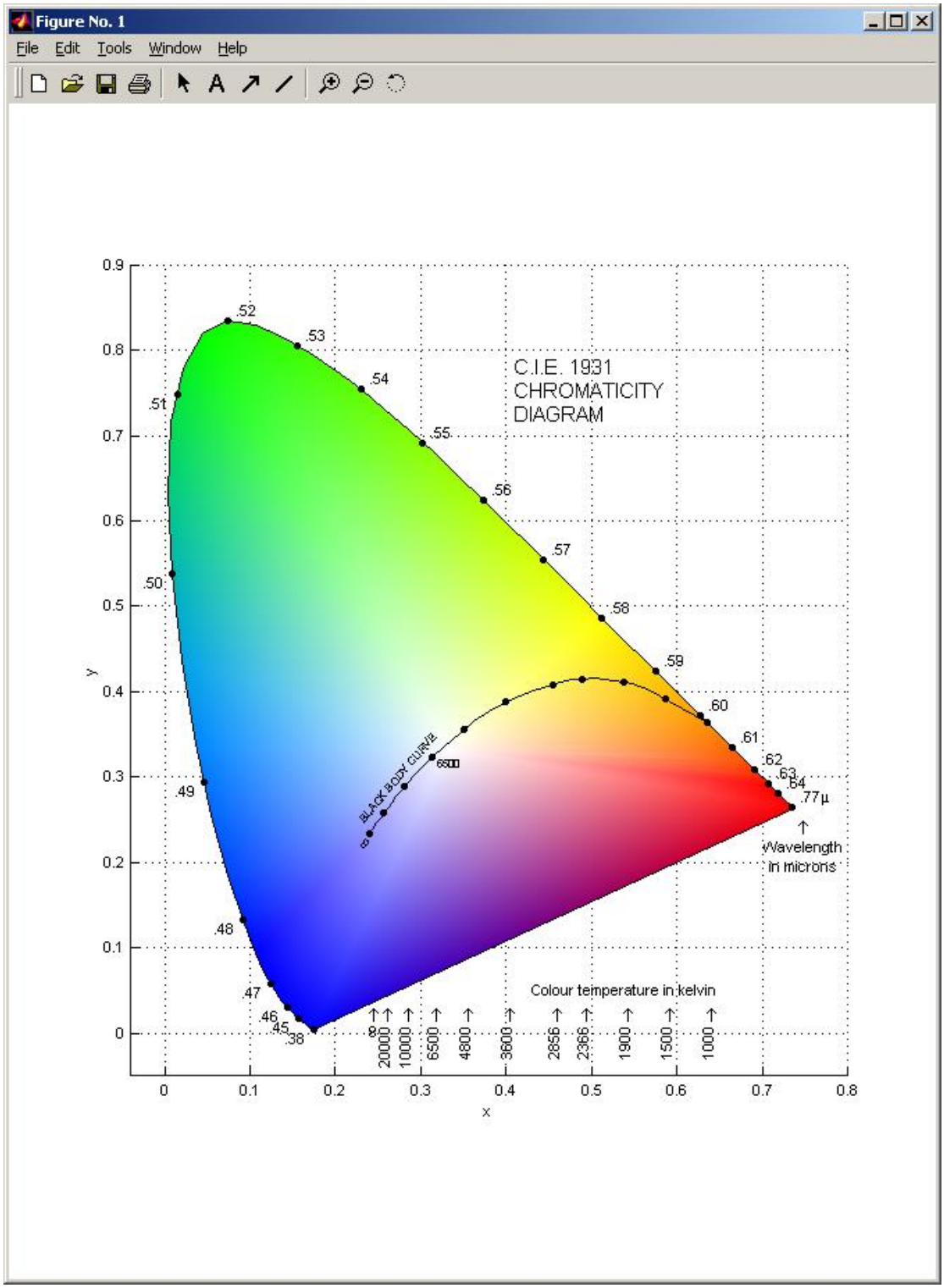
Color Science: Concepts and Methods, Quantitative Data and Formulae Second Edition
Günter Wyszecki & W.S.Stiles 1982 John Wiley & Sons

The display calibration file referred to in this manual is an ascii file and can be read as text but it should be edited with great caution as the precise format of the file is very strictly defined.

Two sample data files; ‘spc.mat’ and ‘test.dcf.txt’ can be downloaded from the Cogent Graphics website for use in this manual.

cieplot

When you type the command cieplot you get the figure below; the CIE 1931 Chromaticity diagram:-

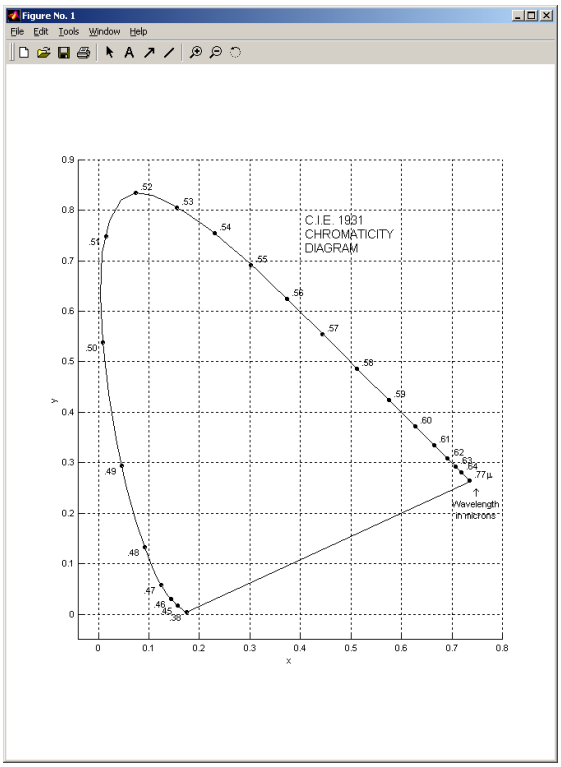


Initially the plot will appear rather small and in that case you should enlarge the figure window to whatever size you require and then type cieplot again.

You can control what elements appear on the plot by passing a 'Flags' argument to the function. The Flags value is the arithmetic sum of the following components:-

1 Plot the figure as white on a black background
2 Plot the wavelength outline on the figure
4 Plot the central colour guide
8 Plot the blackbody radiation curve

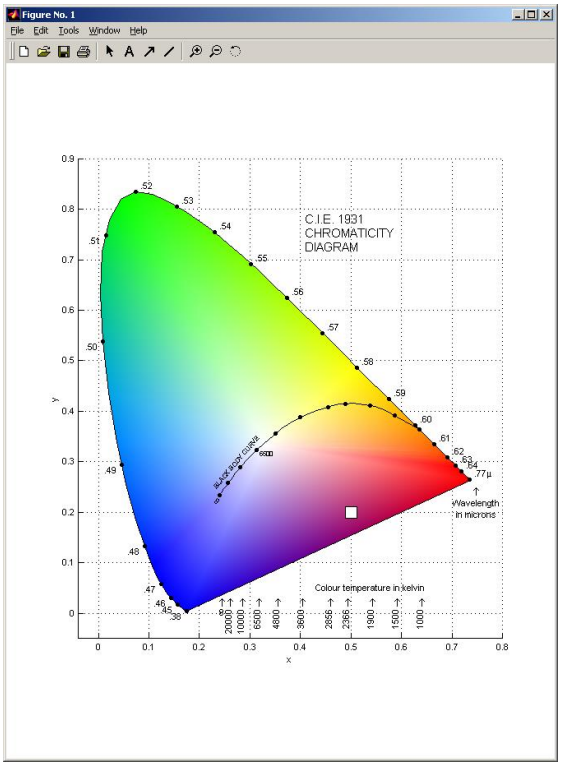
For example, if you just want to plot the wavelength outline alone, use cieplot(2):-



Or, to plot the central colour guide on a black background use (1 + 4); cieplot(5):-

Once you have your plot you can add points to it in the following way:-

EDU» **cieplot**
EDU» **hold on**
EDU» **plot(.5,.2,'s','MarkerEdgeColor','k','MarkerFaceColor','w','MarkerSize',12)**
EDU» **hold off**



The matlab **plot** command has been used to plot a point on the graph.

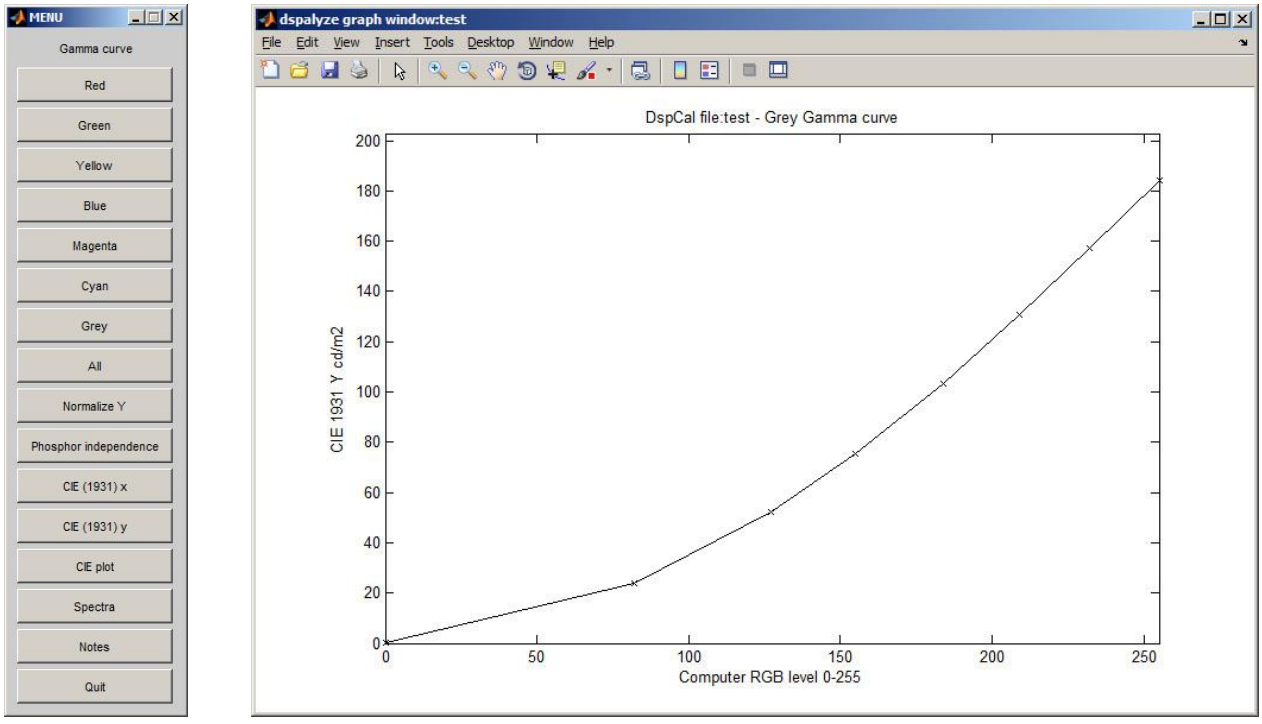
dspalyze

This function is used to analyze the data in a display calibration file. A sample file can be downloaded from the website, named 'test.dcf.txt'.

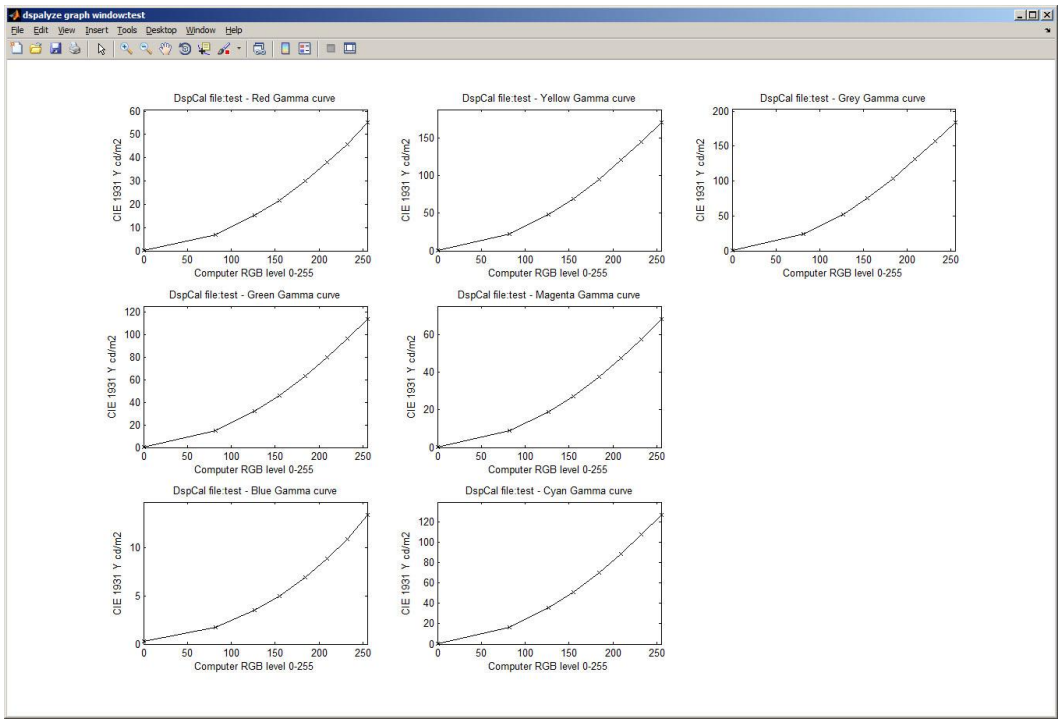
A number of analyses can be made including:-

Gamma curve	A plot of the monitor gamma curve for the colours red, green, blue, yellow, magenta, cyan and grey. Strongly curved plots are undesirable.
Phosphor independence	A plot of the independence of the phosphors for different colours. The independence curve for grey for example shows the ratio of the intensity of a given grey level to the sum of the individual red, green and blue intensities at the same level. Curves can be plotted for yellow, magenta, cyan and grey. A flat line close to unity is desirable.
CIE (1931) x & y	A plot of how particular colours change with varying intensity. A horizontal line is desirable.
CIE plot	A plot of the monitor red, green and blue phosphors on the CIE (1931) chromaticity diagram. A large triangle is desirable.
Spectra	The spectra for the display's peak red, green, blue, yellow, magenta, cyan, white and black. Only available for calibrations made using dspcalib v1.32 or later.
Notes	Ancilliary information made during the calibration.

The initial plot shows the gamma curve for grey as well as a menu which allows you to select other plots:-

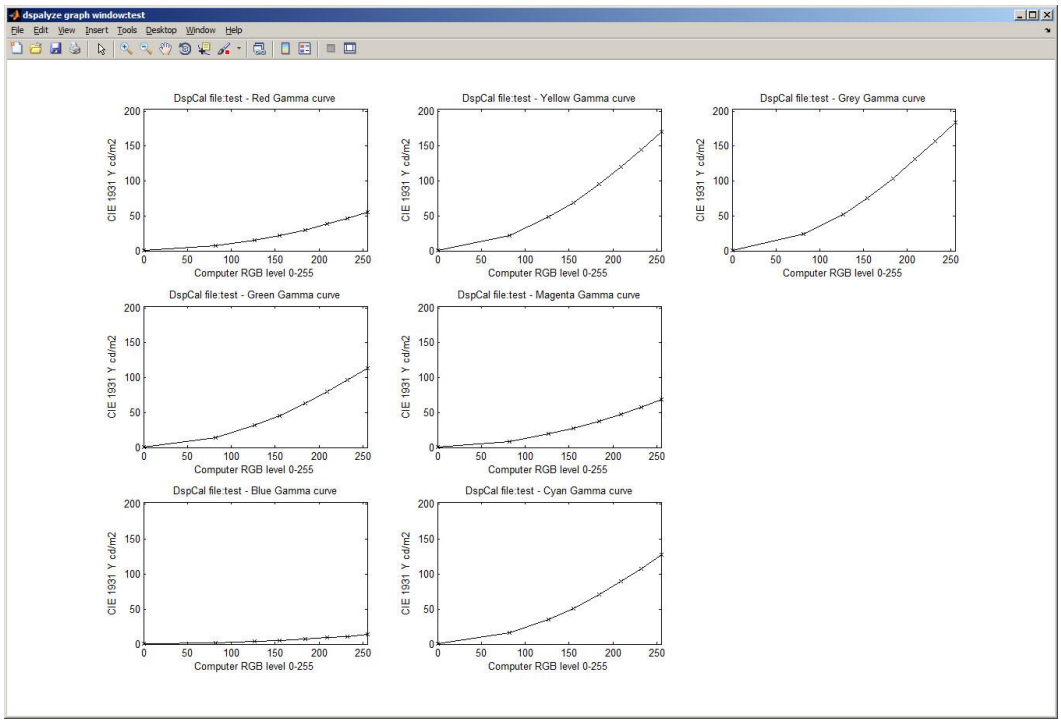


If you select 'All' a multiple plot of the gamma curves of all the colours is shown:-

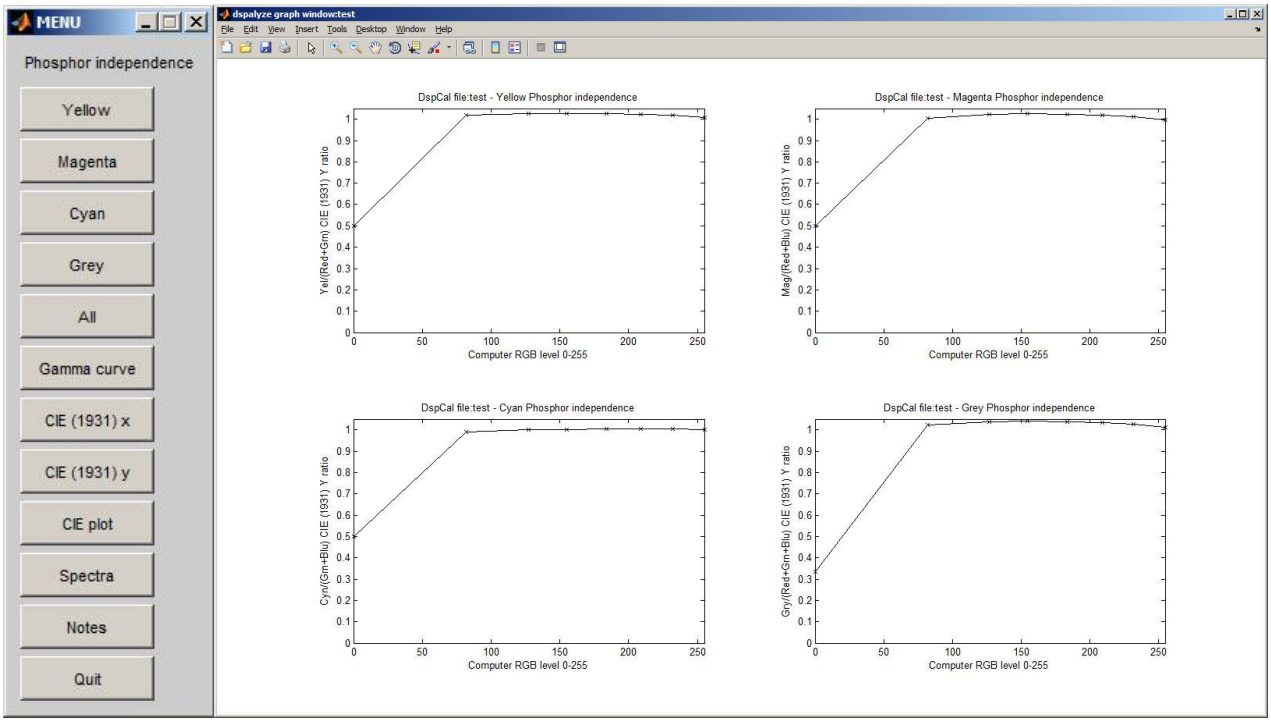


The positions of the graphs for the different colours has been changed from earlier versions of the dspalyze function. The first column now shows red, green and blue, the second column shows yellow, magenta and cyan, and grey is shown in the upper right.

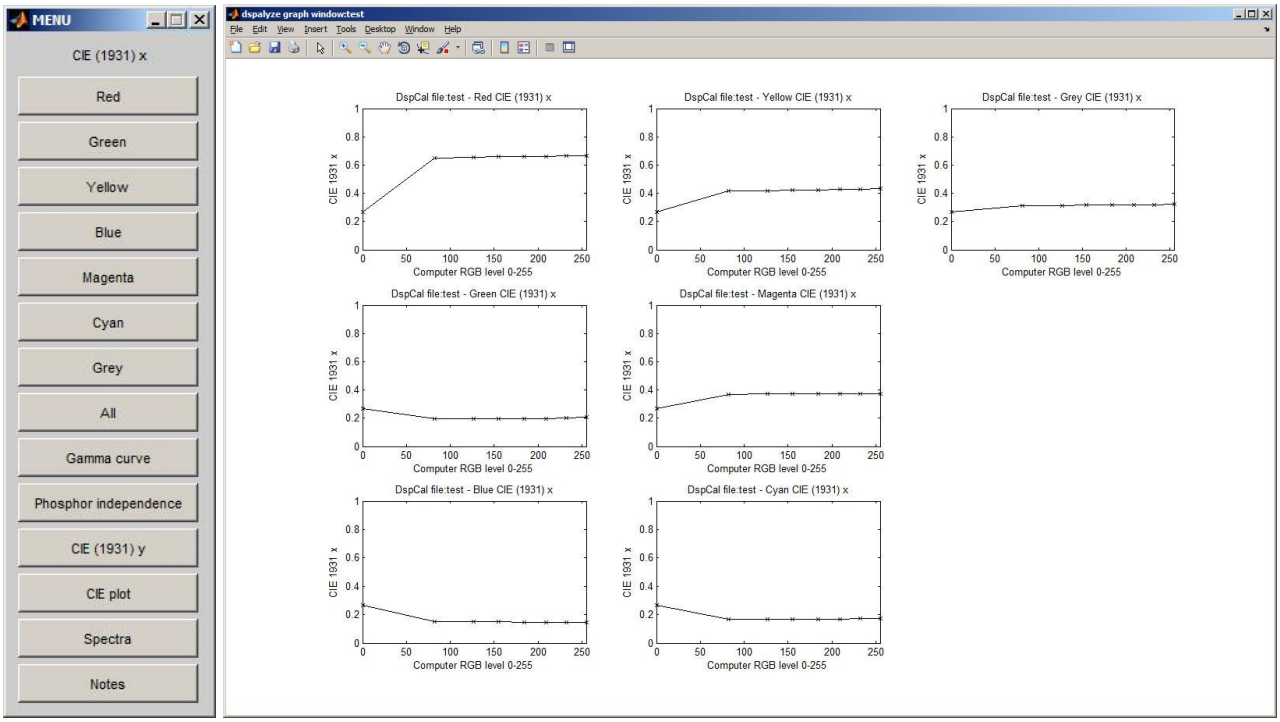
The default setting scales the y-axes of all the gamma curves so that the maximum value occupies most of the graphs as shown above. However, if you select the "Normalize" button the graphs are redrawn so that the y-axes all cover exactly the same range and you can see how the maximum intensities vary.



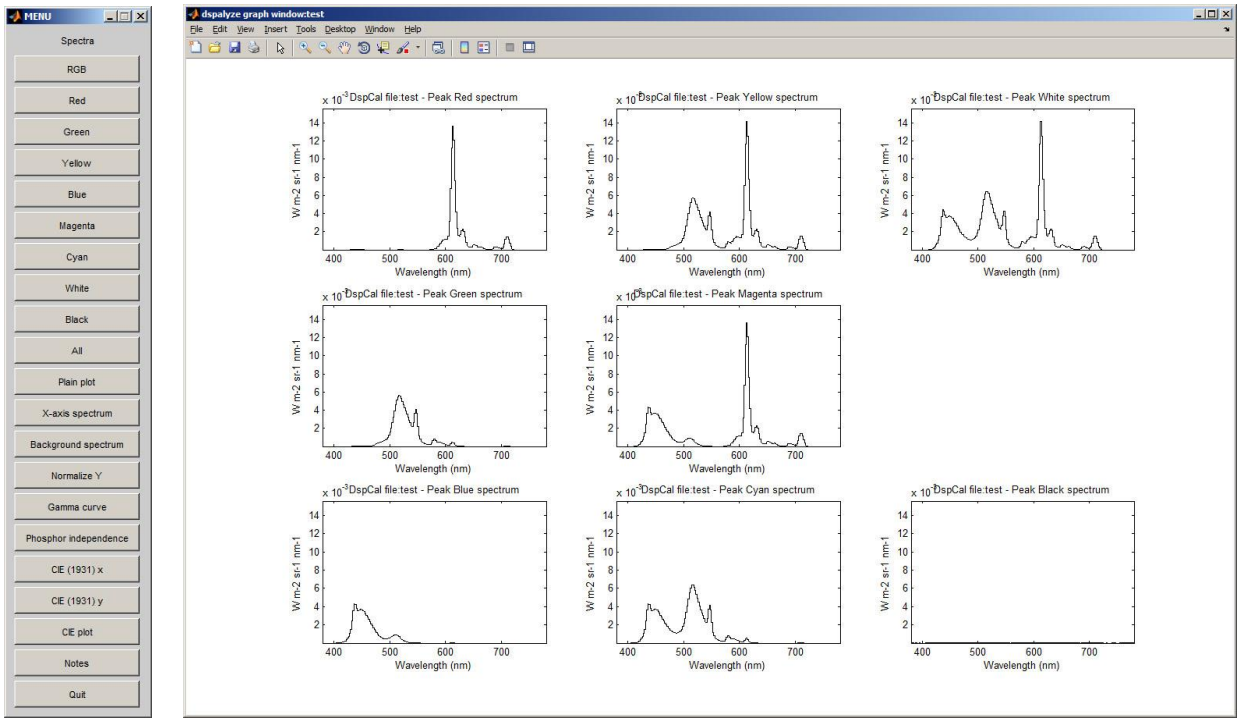
If you now select ‘Phosphor Independence’ you get a plot of the phosphor independence for all colours:-



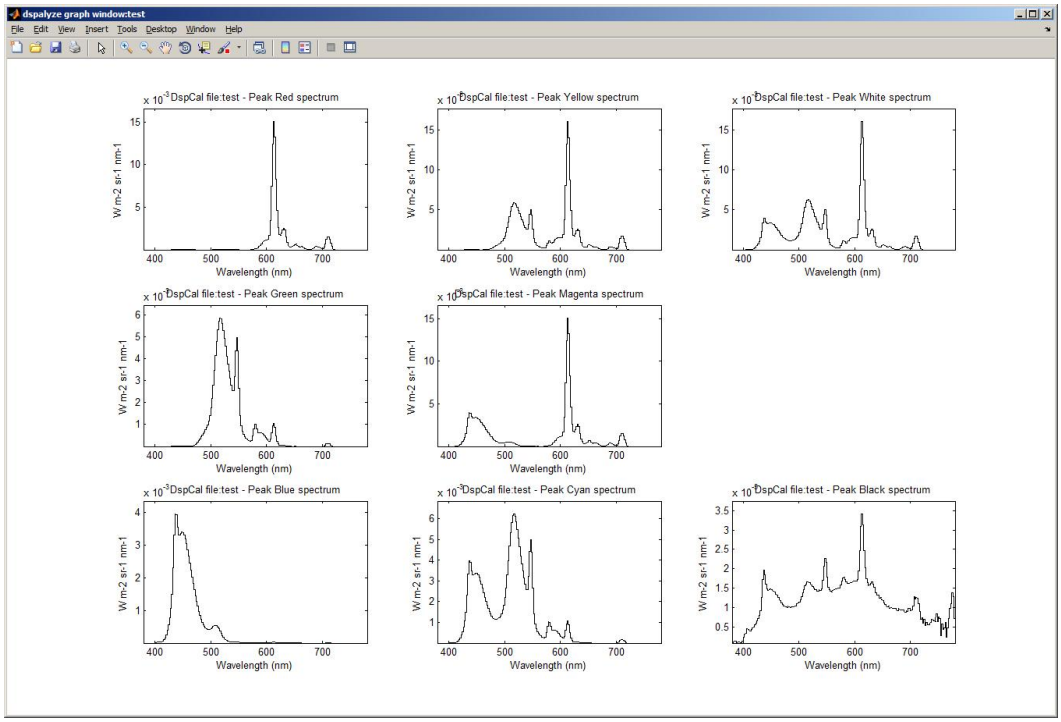
You can obtain a plot of CIE (1931) x and y co-ordinates by hitting the appropriate buttons…



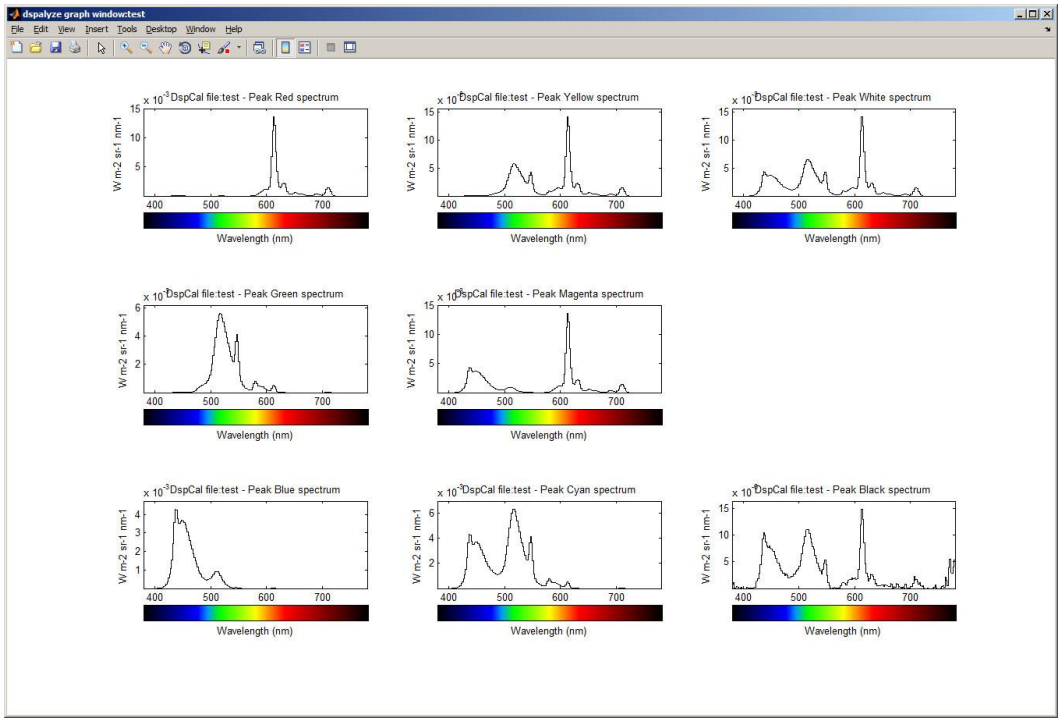
The "spectra" button displays the spectra for the peak intensity colours of the display; red, green, blue, yellow, magenta, cyan, white and black:-



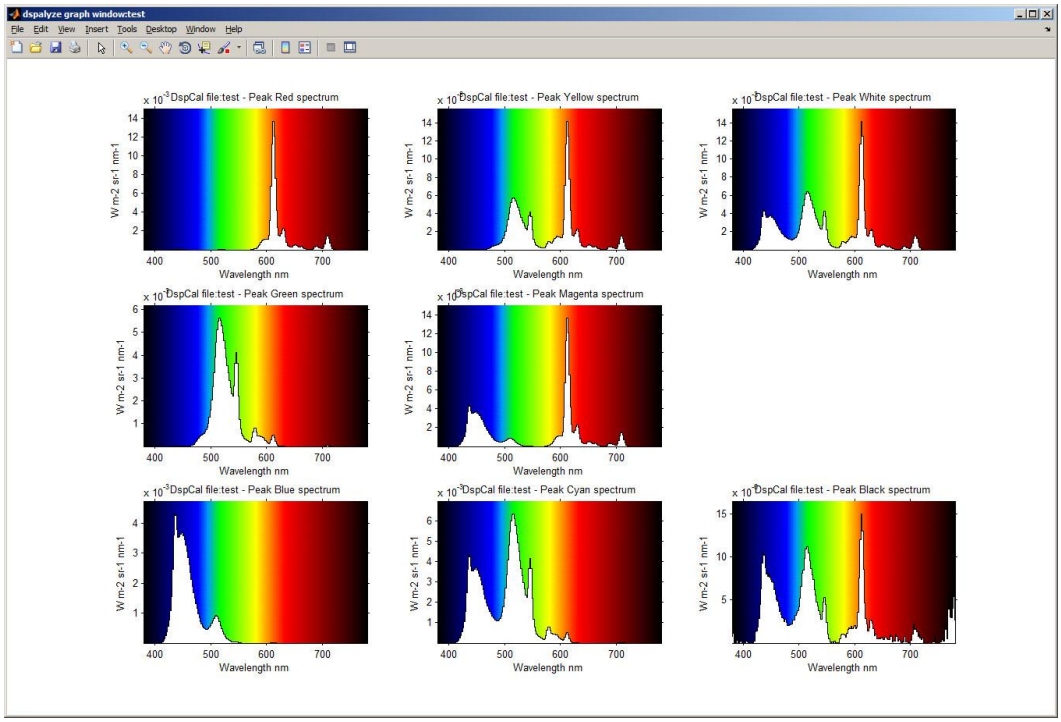
This time the default is to draw all the y-axes to cover the same range so that you can compare the intensities of the different spectra. If you select the "Normalize" button the y-axes are redrawn so that the peak value of each graph is drawn at the same height:-



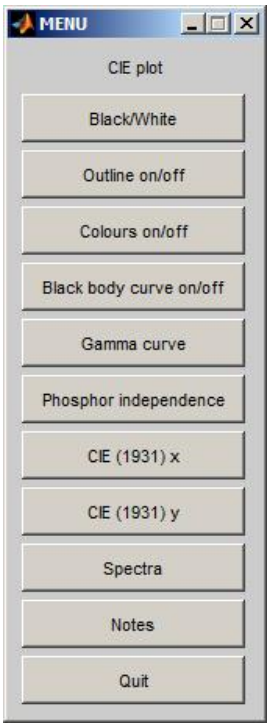
The simplest spectra plot draws the data as a plain histogram, but if you select the "X-axis spectrum" button the graphs are redrawn with a representation of the visible spectrum just below the x-axis:-

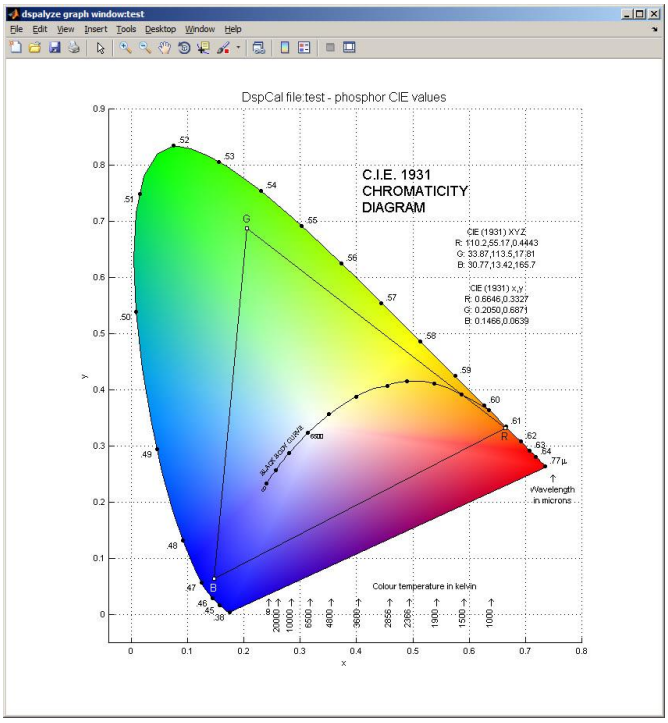


Alternatively, if you select the "Background spectrum" button the graphs appear with a background representing the visible spectrum:-



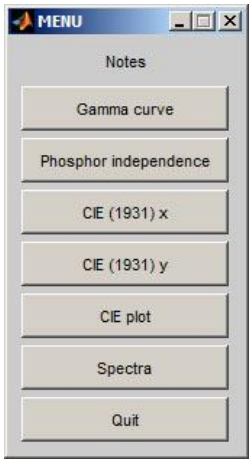
The CIE plot button shows the monitor phosphors on the CIE chromaticity diagram:-

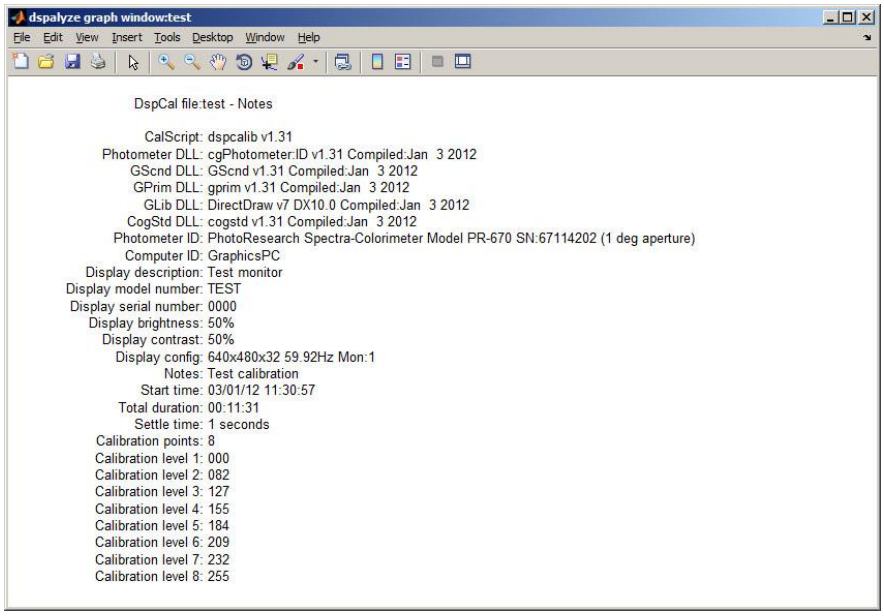




You can modify the appearance of the display by toggle-selecting the "Black/White", "Outline on/off", "Colours on/off" and "Black body curve on/off" buttons.

Finally, the Notes page shows all the other ancilliary information that is stored in the file (including the aperture size used by the PR-670, which appears in the "Photometer ID" line:-





dspcalib

This function is used to calibrate a display and create a display calibration file. Various colours are displayed on the screen and the CIE XYZ values are measured with the photometer. The sample script "dspcheck" described elsewhere in this manual can be used to check the accuracy of the calibration and of the associated conversion functions xyz2rgb and rgb2xyz.

You can use this function with either the PR-670 photometer or with the earlier PR-650 model. The default is to assume the PR-670 is being used. If you are using the PR-650 then you should include the 'PhotoModel' argument in the command, specifying 'PR650'.

This function has been modified in v1.32 (and later versions) so that it now records the spectra for the display's peak red, green, yellow, blue, magenta, cyan, white and black.

Usage: dspcalib(Port,CalPoints,SettleTime,LeaveTime,Filename) or
dspcalib(Port,CalPoints,SettleTime,LeaveTime,Filename,Res,BPP,Ref,Mon) or
dspcalib(PhotoModel,Port,CalPoints,SettleTime,LeaveTime,Filename) or
dspcalib(PhotoModel,Port,CalPoints,SettleTime,LeaveTime,Filename,Res,BPP,Ref,Mon)

PhotoModel	Photometer model ('PR650 or PR670). Defaults to PR670 if omitted.
Port	serial or COM port (1-8)
CalPoints	Number of calibration points (2-256) or (-2 to -256)
SettleTime	seconds to allow display to settle (0-60)
LeaveTime	seconds to leave the room (0-60)
Filename	name for display calibration data file
Res	Display resolution (1-6) [1] or 2 element array [HorRes VerRes]
BPP	Bits per pixel (0/8/16/24/32) [0]
Ref	Refresh rate in Hz (>= 0) [0]
Mon	Monitor number (>= 0) [1]

The port number should identify the serial (COM) port to which you connect the photometer. Normally COM1 should be used (Port = 1).

You may select anything from 2 to 256 calibration points. As a general guide I would recommend either 16 or 32 for this value. This gives a good compromise between accuracy and calibration time. If you select a negative value for CalPoints then the light measurements made during the calibration are averaged over several values, giving a more stable and accurate reading.

The settle time allows the display to settle before each colour is measured. A value of 5 seconds should be ample. The leave time gives you some time to leave the room if you want to get on with something during the calibration. It is best to calibrate in a blacked out room so this gives you time to open the door and get out without affecting the calibration.

When using the PR-670 you select an aperture size for the measurements; 1, 0.5, 0.25 or 0.125 degrees. Usually the 1 degree aperture is used for measurements but for bright light sources this may result in very short measurement times. In some cases (such as computer displays) these bright light sources have fast fluctuations which result in poor repeatability when short measurement times are used. In such cases a smaller aperture can be used which will result in a longer measurement time and a more reliable reading. The aperture size is recorded in the display calibration file at the end of the "Photometer ID" string and can be seen there on the "Notes" page when viewing the file using the **dspalyze** utility.

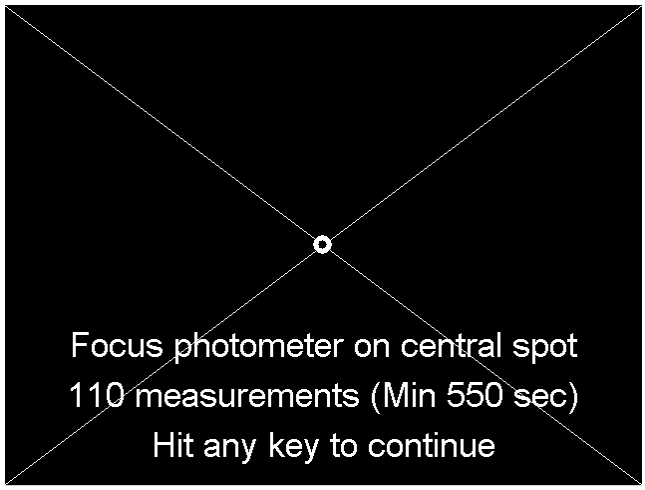
When you start the calibration you are asked a number of questions as shown below:-

```
EDU» dspcalib(4,16,5,60,'test')

      The PR670 can use 4 aperture sizes:-
         A) 1 degree (default)
         B) 1/2 degree
         C) 1/4 degree
         D) 1/8 degree (for very bright light sources)

    Select an aperture A/B/C/D: A
     Enter Display Description: Computer room main monitor
    Enter Display Model Number: Sony Multiscan20seII
   Enter Display Serial Number: 6303081
      Enter Display Brightness: 50%
        Enter Display Contrast: 100%
         Enter a line of notes: sample calibration

Connect PR670 directly to serial port COM4
Switch in CTRL position
Hit a key when ready
```

You should enter all the values as accurately as possible, especially the display brightness and contrast settings. If you are in doubt about some of these values, please ask one of the technicians what you should enter in response to the questions.

At this point you should connect the photometer to the computer and switch the photometer on by pressing the red '0/1' button. The power light on the photometer should glow. Then hit a key on the keyboard to go on to the next stage…

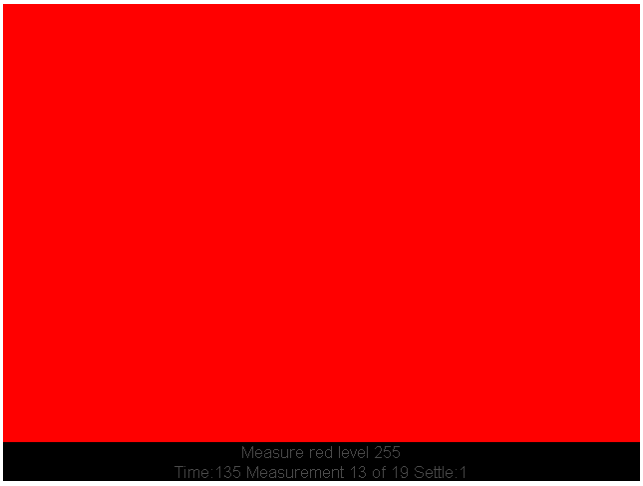
The photometer display should illuminate to indicate that communications have been established and a target should appear on the display you are going to calibrate:-



Focus and aim the photometer target spot at the central circle and then make the room you are in as dark as possible. Then press any key to start the calibration. If you wish to leave the room you should have set the 'LeaveTime' appropriately and you should then have time to leave the room and lock the door behind you. A countdown screen appears while this is going on:-

The calibration now begins. First of all four grey screens of varying brightness are displayed to get a rough idea of how the monitor brightness varies over the full range. These screens are measured and used to estimate a linear distribution of calibration points across the monitor output range. Then, for each of the calibration levels the screen XYZ is measured for red, green, yellow, blue, magenta, cyan and grey in turn.

During this phase the display typically appears a uniform colour or as shown below during the settle time:-



The thin dark band at the bottom contains text which gives some indication of how the calibration is progressing; typically:-

Measure Red Level 255
Time:456 Measurement 145 of 149 Settle:5

Finally, when the calibration has finished the screen returns to Windows and you should see a message on the matlab window similar to:-

```
*****************************************
* Calibration completed successfully    *
* Calibration file "test" saved         *
* Now disconnect and pack the photometer*
*****************************************
```

The first form of the function selects default values for the display configuration:-

Res = 1	(640 x 480 pixels)
BPP = 0	(Maximum bitdepth – usually 24 or 32)
Ref = 0	(Default refresh rate)
Mon = 1	(Primary display)

However, if you know that you will be using a particular display configuration you can specify it at the command line:-

dspcalib(Port,CalPoints,SettleTime,LeaveTime,Filename,**Res**,**BPP**,**Ref**,**Mon**)

This will give you the most accurate calibration possible for your display.

readdcf

This function can be used to read a display calibration file into a structure in the matlab workspace:-

```
EDU» dcf=readdcf('test')

dcf =

        CalScript: 'dspcalib v1.07'
    PhotometerDLL: 'GScnd:cgPhotometer v1.32 Compiled:Jan 25 2012'
         GScndDLL: 'GScnd v1.32 Compiled:Jan 25 2012'
         GPrimDLL: 'GPrim v1.32 Compiled:Jan 25 2012'
          GLibDLL: 'DirectDraw v7 Compiled:Jan 25 2012'
        CogStdDLL: 'CogStd v1.32 Compiled:Jan 25 2012'
     PhotometerID: [1x58 char]
       ComputerID: 'mustard00 128.40.206.2'
           DspDsc: 'John Romaya's PC monitor, rm 507, LON'
         DspModNo: 'Sony Multiscan20seII'
         DspSerNo: '6303081'
           DspBrt: '50%'
           DspCnt: '100%'
           DspCnf: [1x1 struct]
            Notes: {2x1 cell}
        StartTime: [1x1 struct]
    TotalDuration: [1x1 struct]
       SettleTime: 1
      CalibPoints: 4
       CalibLevel: [0 159 212 255]
              XYZ: [7x4x3 double]
           DspSpc: (See below)
```

The display configuration structure has the following members:-

```
EDU» dcf.DspCnf

ans =

     Width: 640
    Height: 480
      Bits: 32
        Hz: 59.6700
       Mon: 1
```

The start time structure has the following members:-

```
EDU» dcf.StartTime

ans =

     yr: 1
    mon: 2
    day: 23
     hr: 14
    min: 47
    sec: 41
```

The DspSpc member takes three forms, depending on whether the PR-670 photometer was used or the PR-650 or whether spectral data was not collected (Cogent v.130 or earlier):-

1) [201 x 2 double] array of wavelengths (in nm) and energy (in W m-2 sr-1 nm-1) for PR-670
2) [101 x 2 double] array of wavelengths (in nm) and energy (in W m-2 sr-1 nm-1) for PR-650
3) [] (empty) if there is no spectral data present

Spectral data is recorded for peak red, green, yellow, blue, magenta, cyan white and black.

The total duration structure has the following members:-

```
EDU» dcf.TotalDuration

ans =

     hr: 0
    min: 4
    sec: 15
```

The CalibPoints array gives the RGB level points for each of the requested calibration points and the XYZ array gives the XYZ values for each of the 7 colours at each calibration point.

rgb2xyz

This function converts an RGB triplet into CIE (1931) XYZ values.

Usage:- GAMXYZ = rgb2xyz(filename) or
[XYZ,Err] = rgb2xyz(RGB) or
[XYZ,Err] = rgb2xyz(R,G,B)

filename = display calibration file name
GAMXYZ = 3 x 3 matrix of monitor gamut XYZ values
RGB = (n x 3) matrix of RGB values (0 to 1)
R,G,B = individual RGB arrays of equal size
XYZ = (n x 3) matrix of XYZ values
Err = (n x 1) matrix of error values:-
Err(i)=1 – RGB was reset to range 0-1

You must first set up the monitor calibration file you want to use:-

```
EDU» GAMXYZ = rgb2xyz('test')

GAMXYZ =
      26.5500   14.5800    1.3810
      22.4200   46.9200    8.0660
      14.3500    6.5090   74.8400
```

The optional returned array `GAMXYZ` gives the monitor gamut. This is shown as a triangle on the dspalyze CIE plot where each vertex of the triangle corresponds to one of the monitor phosphors; Red, Green or Blue. When using this monitor you can only produce colours within this triangle. You may alternatively use the function **`xyz2rgb('test')`** for this initialization step.

Thereafter you may obtain conversions as follows:-

```
EDU» [xyz,err] = rgb2xyz(0.5,0,0)

xyz =

      4.3063    2.3761    0.2369

err =

      0
```

Instead of specifying individual R, G and B values as shown above, you may alternatively supply an array of the form (n x 3):-

```
EDU» [xyz,err] = rgb2xyz([0 0.75 0])

xyz =

      11.3646   23.7735    4.0929

err =

      0
```

If you supply an RGB value which lies outside the valid range of 0 to 1, that value will be reset to lie within the valid range and the returned "Err" value will equal unity:-

```
EDU» xyz = rgb2xyz([0 0.75 1.2])

xyz =

      25.6926   30.2569   78.9163

err =

      1
```

The function will also accept several values at a time, if they are supplied as an (n x 3) array:-

```
EDU» [xyz,err] = rgb2xyz([0.5 0 0;0 0.75 0])

xyz =

       4.3063    2.3761    0.2369
      11.3646   23.7735    4.0929

err =

      0
      0
```

You can check the accuracy of a display calibration and the associated conversion functions rgb2xyz and xyz2rgb using the dspcheck sample script described elsewhere in this manual. Or you can use another sample script; xyzcheck to display specific XYZ values using those functions.

spc2xyz

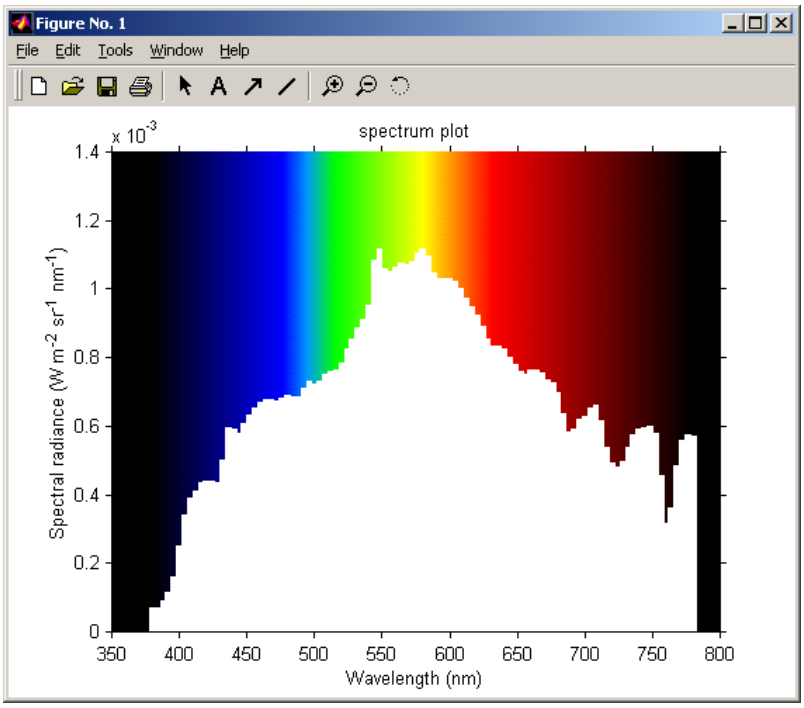
This function converts a spectrum into CIE (1931) XYZ values. A sample spectrum file ‘spc.mat’ can be downloaded from the website.

```
EDU» load spc
EDU» XYZ=spc2xyz(spc)

XYZ =

   66.9183   69.6944   44.4374
```

spcplot

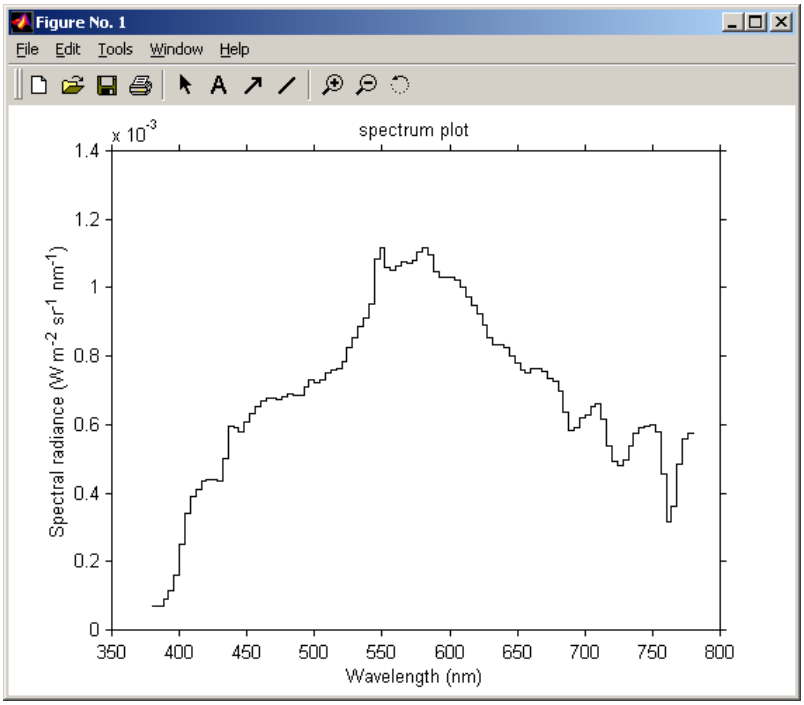
This function plots a spectrum in graphical form. You may download a sample spectrum data file 'spc.mat' from the website:-

```
EDU» load spc
EDU» spcplot(spc)
```

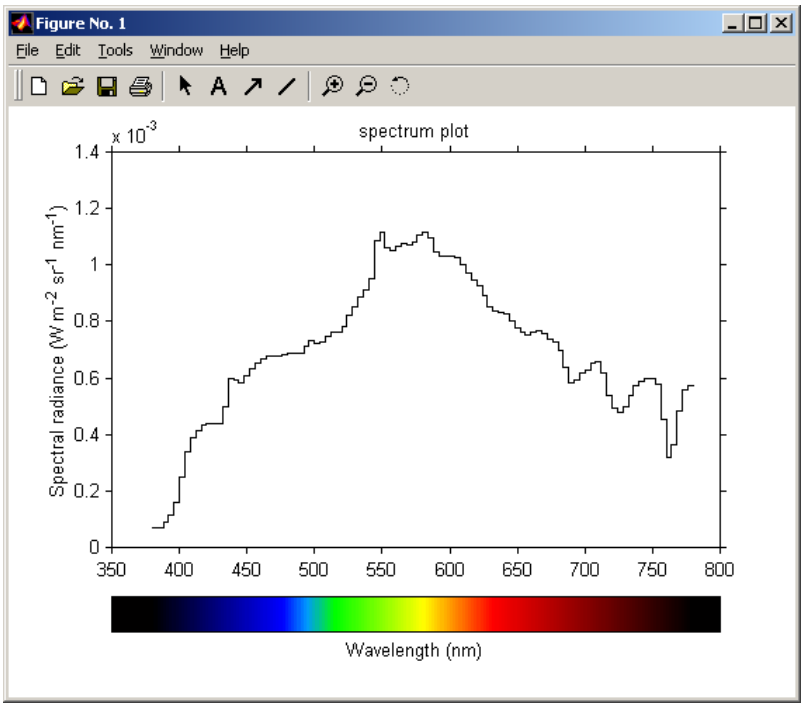


The spectrum is plotted out as shown above. You can also use the command spcplot(spc,3) to obtain the figure shown above.

You may plot a simple black and white plot with the command spcplot(spc,1):-



Or you can plot a simple plot with a horizontal colour bar using spcplot(spc,2):-



xyy2xyz

This function converts an array of CIE (1931) xyY values to XYZ values:-

```
EDU» xyY = xyz2xyy([20 50 30;30 40 30])

xyY =

    0.2000    0.5000   50.0000
    0.3000    0.4000   40.0000
```

xyz2rgb

This function converts CIE (1931) XYZ values into RGB values.

Usage:- GAMXYZ = xyz2rgb(filename) or
[RGB,Err] = xyz2rgb(X,Y,Z<,Options>) or
[RGB,Err] = xyz2rgb(XYZ<,Options>)

filename = display calibration file name
GAMXYZ = 3 x 3 matrix of monitor gamut XYZ values
X,Y,Z = individual XYZ arrays of equal size
XYZ = (n x 3) matrix of XYZ values
Options = 0 (keep hue) or 1 (keep int) when Err = 2
or 2 to return uncorrected linear RGB
RGB = (n x 3) matrix of RGB values (0 to 1)
Err = (n x 1) array of error values (see below)

You must first set up the monitor calibration file you want to use:-

```
EDU» GAMXYZ = xyz2rgb('test')

GAMXYZ =

        110.2000   55.1700    0.4443
         33.8700  113.5000   17.8100
         30.7700   13.4200  165.7000
```

The optional returned array `GAMXYZ` gives the monitor gamut. This is shown as a triangle on the dspalyze CIE plot where each vertex of the triangle corresponds to one of the monitor phosphors; Red, Green or Blue. When using this monitor you can only produce colours within this triangle. You may alternatively use the function **`rgb2xyz('test')`** for this initialization step.

Thereafter you may obtain conversions as follows:-

```
EDU»  rgb=xyz2rgb(33.8700, 113.5000, 17.8100)

rgb =

     0     1     0
```

The optional `err` value gives warnings when you request an xyz value which is outside the monitor gamut. The returned value is the sum of the following:-

err	Meaning
0	No errors
1	One of the XYZ values was less than zero. Zero has been substituted instead for the conversion.
2	Requested XYZ "Y" value was too bright. The same colour has been achieved but at a lower brightness.
4	You requested an XYZ value outside the monitor phosphor triangle on the CIE plot. The colour has been desaturated enough to lie within monitor gamut with the brightness held constant. Desaturation is carried out along the line connecting (X,Y,Z) with (Y,Y,Y) in CIE space.

You may also specify the XYZ values as an (n x 3) array:-

```
EDU» [rgb,err] = rgb=xyz2rgb([33.87 113.5 17.81])

rgb =

      0         1         0

err =

      0
```

And you may also specify multiple values:-

```
EDU» [rgb,err] = xyz2rgb([110.2 55.17 0.4443; 33.87 113.5 17.81])

rgb =

      1         0         0
      0         1         0

err =

      0
      0
```

You can check the accuracy of a display calibration and the associated conversion functions rgb2xyz and xyz2rgb using the dspcheck sample script described elsewhere in this manual. Or you can use another sample script; xyzcheck to display specific XYZ values using those functions.

Now consider the following case:-

```
EDU» [rgb,err] = xyz2rgb([50 150 50])

      rgb = 0.0549    1.0000    0.3059
      err = 2

EDU» xyz = rgb2xyz(rgb)

      xyz = 38.6061  115.6023   38.6589

EDU» xyz*150/max(xyz)

      ans = 50.0934  150.0000   50.1620
```

In this case, it has not been possible to create an rgb triplet which reproduces XYZ = [50 150 50] because the Y value (150) is too large. The best achievable value creates XYZ = [38.6 115.6 38.6]. Notice that the ratio between X, Y and Z has been maintained. In other words, the requested colour has been created, but at a lower brightness. Now consider the following sequence, where the Options = 1 argument has been used:-

```
EDU» [rgb,err] = xyz2rgb([50 150 50],1)

      rgb = 0.7098    1.0000    0.7216
      err = 2

EDU» xyz = rgb2xyz(rgb)

      xyz = 109.0759  149.8487  113.6681
```

This time the requested brightness (Y=150) has been achieved, but the colour has become less saturated.

Now consider the following case, using the Options = 2 argument:-

```
EDU» [rgb,err] = xyz2rgb([50 150 50],2)

      rgb = 0.0100    1.2985    0.1633
      err =  2
```

This time, the function returns uncorrected, linear RGB values. It can be seen that the maximum RGB value is 1.2985. This can be applied as a correction factor to the original XYZ values to bring them within the brightness range of the display:-

```
EDU» [rgb,err] = xyz2rgb([50 150 50]/1.2985)

      rgb =  0.0075    0.9996    0.1252
      err =  2

EDU»  rgb2xyz(rgb)

      ans = 38.5233  115.5610   38.6589
```

Note that the corresponding values for XYZ correlate very well with the conversion that is calculated with no options:-

```
EDU» [rgb,err] = xyz2rgb([50 150 50])

      rgb = 0.0549    1.0000    0.3059
      err = 2

EDU» xyz = rgb2xyz(rgb)

      xyz = 38.6061  115.6023   38.6589
```

xyz2xyy

This function converts an array of CIE (1931) XYZ values to xyY values:-

```
EDU» XYZ = xyy2xyz([.2 .5 50;.3 .4 40])

XYZ =

   20.0000   50.0000   30.0000
   30.0000   40.0000   30.0000
```

Utilities

The following utility scripts are included in the samples:-

dspcheck	This program checks the accuracy of a display calibration and the XYZ to RGB conversion functions using the photometer.
xyzcheck	This program accepts a set of XYZ values. It then converts them to RGB, displays each value on the screen and measures the resulting XYZ with a photometer. The resulting XYZ values are returned for comparison with the requested values.
XYZ Demo	This program allows you to select and display values from the CIE colourspace.

dspcheck

The script can be called in three ways:-

1/ dspcheck(DCFname,Levels,PhotometerID,PortNum) or
2/ dspcheck(DCFname,XYZ,PhotometerID,PortNum) or
3/ dspcheck(DCKname)

DCFname = Display calibration file name
Levels = Number of levels to check
XYZ = Requested XYZ values
PhotometerID = Which photometer to use (eg. 'PR670')
*PortNum = COM port for photometer
DCKname = dspcheck file name

*PortNum - if you select a negative PortNum (e.g. -2) , the script will use the positive value as the actual COM port number (i.e. 2 in this example) and photometer measurements will be made using the "XYZB" measurement mode. Otherwise, if PortNum is positive, the standard "XYZ" measurement mode is used.

1/ You pass the function the name of a display calibration file, a 'Levels' value and communication details for a photometer, and the script will calculate a number of XYZ values based on the display calibration file. It will then make the screen display each of those 'requested' XYZ values and measure it each time using the photometer to get a 'measured' XYZ. It then displays a graph showing the difference between 'requested' and 'measured' XYZ values. The check data is also saved in a file.

The number of XYZ values checked depends on the 'Levels' value; it is equal to (Levels x Levels x Levels) - 1. So be careful how you set 'Levels':-

Levels	XYZ
2	7
3	26
4	63

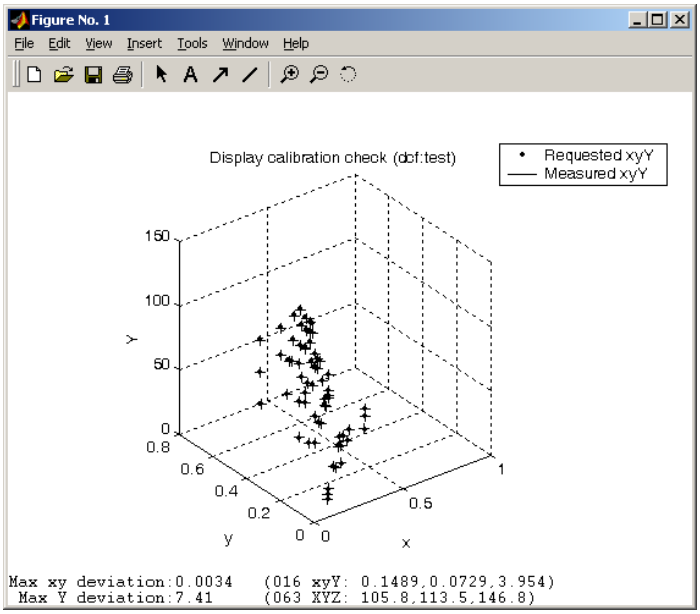
Levels	XYZ
5	124
6	215
7	342

Levels	XYZ
8	511
9	728
10	999

2/ Instead of 'Levels', you pass the function an array of (nx3) XYZ values which are then each displayed and measured with the photometer. It then displays a graph showing the difference between 'requested' and 'measured' XYZ values and the check data is also saved in a file in a similar way to option 1/ above.

3/ If you pass the script the name of the file created by option 1/ or option 2/ above, it displays the data.

The graph shows a three-dimensional plot of CIE xyY values. The 'requested' values are shown as '.' and the 'measured' values are shown as '+'. A line is also drawn connecting each 'requested' value to the corresponding 'measured' value. If the calibration and conversion functions are accurate then the lengths of these connecting lines should be very small. In addition, the maximum xy deviation and the maximum deviation in 'Y' are also written on the graph. You can rotate the three-dimensional graph using the standard matlab rotate function.



xyzcheck

This script checks CIE colour conversions.

You pass the script a set of XYZ values to generate, the name of a display calibration file to use and details for communicating with a photometer.

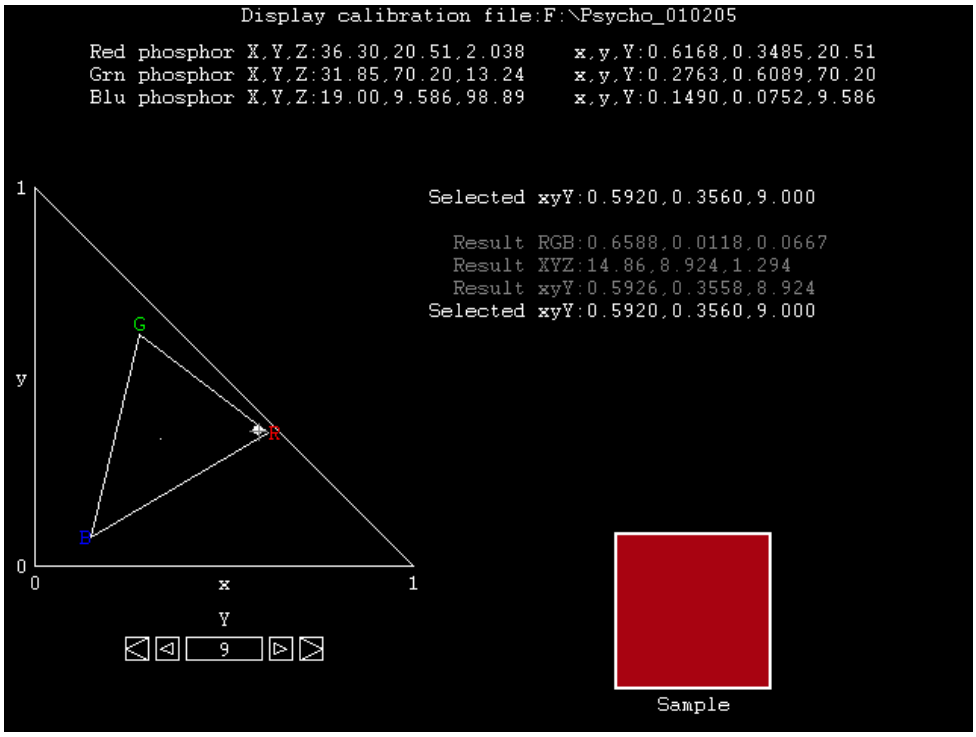
The script converts each XYZ value to RGB, displays the RGB value on screen and then measures the XYZ value of the display. It returns an array of the measured XYZ values for comparison with the requested values.

XYZOut = xyzcheck(XYZIn,DCFname,PhotometerID,PortNum)

XYZIn = Requested XYZ values
DCFname = Display calibration file name
PhotometerID = Which photometer to use (eg. 'PR670')
*PortNum = COM port for photometer
XYZOut = measured XYZ values

*PortNum - if you select a negative PortNum (e.g. -2) , the script will use the positive value as the actual COM port number (i.e. 2 in this example) and photometer measurements will be made using the "XYZB" measurement mode. Otherwise, if PortNum is positive, the standard "XYZ" measurement mode is used.

XYZDemo



Call this script in the following way:-

XYZDemo(Filename)

where Filename is a display calibration file name.

A triangular CIE x/y colourspace is displayed with the monitor gamut shown as another triangle within it. Click with the mouse within the monitor gamut triangle to select an xy value. You may also select Y using the boxes underneath the CIE colourspace triangle. In the example above, the following value has been selected:-

x = 0.5920
y = 0.3560
Y = 9.000

This is coverted into RGB values (0.6588, 0.0188, 0.0667 above) and this colour is displayed in the "Sample" rectangle. The resulting XYZ values (14.86, 8.924, 1.294 above) are displayed and these are reconverted back to xyY (0.5926, 0.3558, 8.924 above) for comparison with the selected xyY. You can measure the "Sample" rectangle with a spectrophotometer to check the accuracy of the conversions. You will get an error message if XYZDemo fails to open a graphics screen with the same configuration parameters as that recorded in the display calibration file.