Chapter

II-15

Image Plots

Overview

You can display matrix data as an image plot in a graph window. Each matrix data value defines a colored rectangle in the image plot. The size and position of these rectangles is controlled by the range of the graph axes, the graph width and height modes, and the X and Y coordinates of the image rectangles.

Note: The terms "matrix data value", "pixel", and "Z value" are used interchangeably in this chapter.

Image data can be false color, indexed color or direct color.

False Color Images

In false color images the data values in the 2D matrix are linearly mapped onto a color table. This is a very powerful way to view matrix data and is often more effective than either surface plots or contour plots. For best results you can superimpose a contour plot on top of a false color image of the same data.

Igor has several built-in color tables (see **Color Tables** on page II-349). You can provide color index waves that define custom color tables (described in **Indexed Color Details** on page II-356).

Indexed Color Images

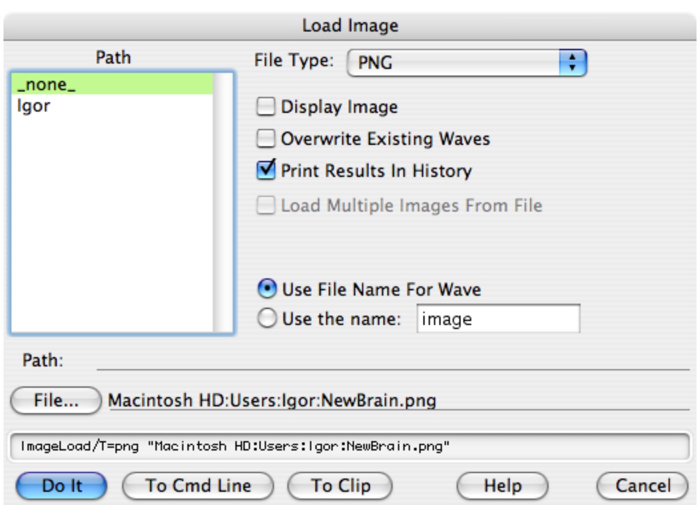
Indexed color images use the data values stored in your 2D matrix as indices (usually integers) into a three-column wave of color values that you supply. "True color" images such as those that come from video cameras or scanners generally use indexed color. Indexed color images are more common than direct color because they consume less memory. See **Indexed Color Details** on page II-356.

Direct Color Images

Direct color images contain the actual red, green and blue values at each point in the image matrix. Direct color images use a 3D wave with 3 color planes containing absolute values for red, green and blue, providing 24-bit color. While an indexed color image contains at most the number of colors in the color index wave, a direct color image can have a unique color for every pixel. See **Direct Color Details** on page II-358.

Loading an Image

To create an image plot, first create or load your matrix wave. You can load TIFF, PICT, JPEG, PNG, GIF, BMP, PhotoShop, Targa, Silicon Graphics, and Sun Raster image files into matrix waves using the **ImageLoad** operation (see page V-269) or the Load Image dialog under the Data menu. Depending on the type of image that you would like to read, this operation may require that you have Apple's QuickTime software (version 4 or newer) installed on your computer.

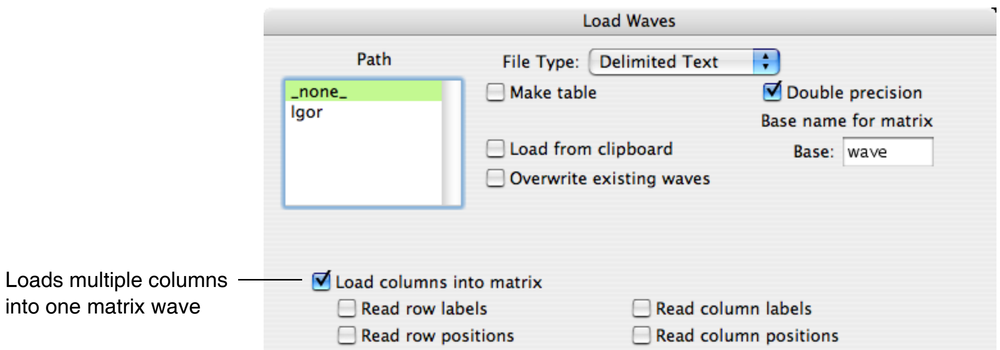


Except for TIFF and Sun Raster images, all images are loaded onto a 3D Igor wave (unsigned byte with plane 0 containing the red channel, plane 1 the green and plane 2 the blue channel). You can convert the 3D wave into other forms using the **ImageTransform** operation (see page V-290).

When loading TIFF files into Igor, a number of additional options are available. For stacked TIFF files that contain multiple images, you can read all of the images into a single 3D wave (where each image occupies a sequential plane), into individual 2D waves, or you can specify a particular image, or range of images, that you would like to read. Reading a TIFF image stack into a single 3D wave is applicable only for images that are 8 or 16 bits/pixel deep.

If you want to load an image file but you are not sure about it's file format, use the "any" or "*.*" specification for the file type. Most of the supported file types can be uniquely recognized by the ImageLoad operation.

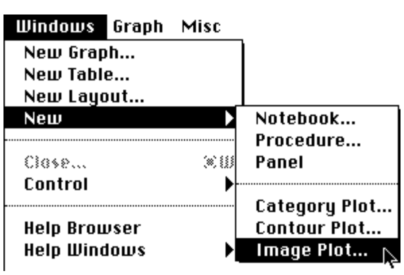
You can also load numeric text and binary files using the Load Waves dialog in the Data menu. Select the "Load columns into matrix" checkbox:



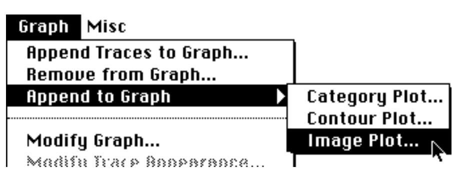
You can also obtain images by acquisition of video using appropriate frame grabber's XOPs (e.g., QTGrabber) or load images using specific file loader XOPs (e.g., HDF5 Loader).

Creating an Image Plot

You can create an image plot in a new graph window with the New Image Plot dialog. This dialog creates a blank graph to which the plot is appended. You can add an image plot to an existing graph with the Append Image Plot dialog.



Creating a new image plot



Appending an image plot to an existing graph

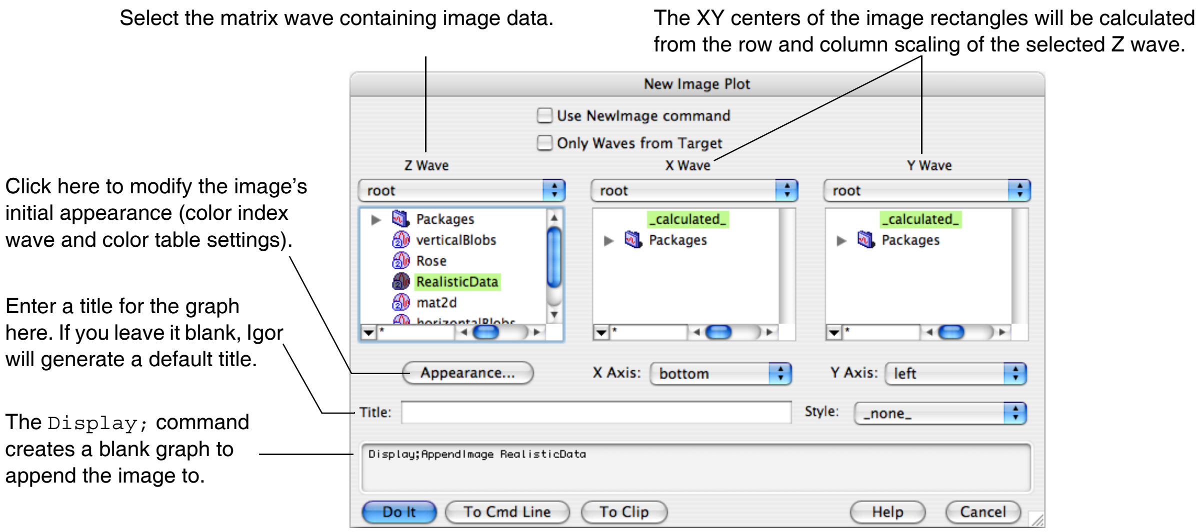
You can also use the **NewImage** operation (see page V-436) or the **AppendImage** operation (see page V-23). Also see **How Images Are Displayed** on page II-346.

Image plots are displayed in an ordinary graph window. All the features of graphs apply to image plots: axes, line styles, drawing tools, controls, etc. See Chapter II-12, **Graphs**.

You can show lines of constant image value by appending a contour plot to a graph containing an image. Igor will draw the contour above all the image plots. See **Creating a Contour Plot** on page II-322 for an example of combining contour plots and images in a graph.

Image Plot Dialogs

Choose New Image Plot from the Windows menu to bring up the New Image Plot dialog:



The Title, Style, X Axis and Y Axis items work the same as in the New Graph dialog (See **Creating Graphs** on page II-237). You can specify a new title for the graph and select or create the axes used in the image plot. Use the Style pop-up menu to apply a style macro to the newly created graph window.

This dialog normally generates two commands — a Display command to make a blank graph window, and an AppendImage command to append a image plot to that graph window. Selecting the "Use NewImage command" checkbox replaces Display and AppendImage with the NewImage command, which changes what options you have for creating an image plot in this dialog (see **Use NewImage Command** on page II-344).

The Append Image Plot dialog is similar, except that the Only Waves from Target, Use NewImage command, Title, and Style items are missing.

X, Y, and Z Wave Lists

The X Wave, Y Wave, and Z Wave lists show the available waves that will be accepted by the AppendImage operation. Select the Only Waves from Target checkbox to show only the waves in the target window (most useful when the target window is a table).

You should select the matrix wave containing your image data in the Z Wave list. This will update the X Wave and Y Wave lists to show only those waves, if any, that may be combined with the image data matrix wave.

Choosing _calculated_ from the X Wave list uses the row scaling (X scaling) of the matrix selected in the Z Wave list to provide the X coordinates of the image rectangle centers.

Choosing _calculated_ from the Y Wave list uses the column scaling (Y scaling) of the matrix to provide Y coordinates of the image rectangle centers.

You can also select a 1D wave to provide the X or Y values for a matrix of Z values; only those waves with the proper length for the selected Z Wave are shown in the X Wave and Y Wave lists. See **Image X and Y Coordinates** on page II-346 and the **AppendImage** operation on page V-23.

Use NewImage Command

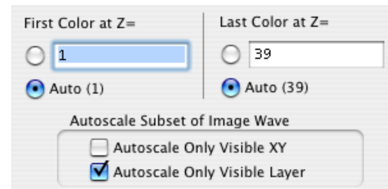
The NewImage command nicely sets the window margins and axes to maximize the image in the window, presets the window size to match the number of pixels in the image, and automatically reverses the left (vertical) axes so that pictures aren't displayed upside down.

However NewImage does not support window titles or 1D X and Y waves, axes other than left and top, or graph style macros.

A useful option of NewImage is the "Do not treat this three-layer image as direct (rgb) color" checkbox that appears only for three-layer Image Waves.

Selecting that option displays one layer from a three-layer wave as a false-color or indexed-color image (the data values in one layer are linearly mapped onto a color table), instead of using all three layers to display a direct color image (see **Direct Color Details** on page II-358).

Click the Appearance button to chose which layer is displayed.



The other two layers are not visible and are ignored except for possibly autoscaling the color table Z limits. If Autoscale Only Visible Layer is checked in the Modify Image Appearance dialog, the other layers are then completely ignored.

Modifying an Image Plot

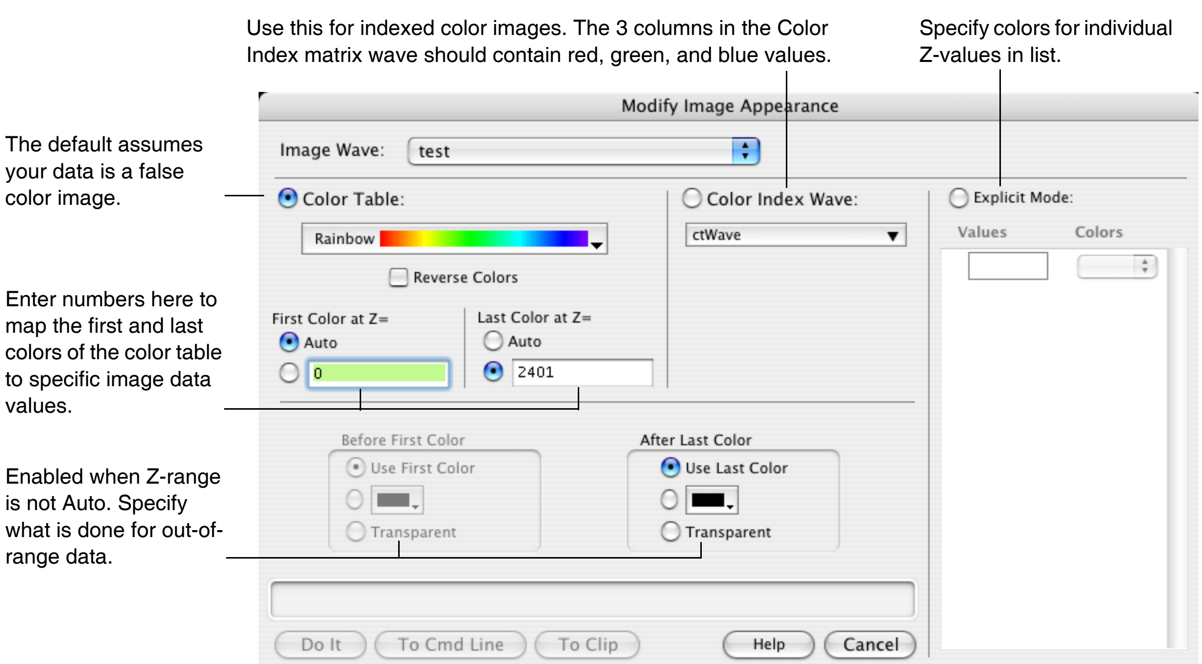
You can change the appearance of the image plot using the Modify Image Appearance dialog. This dialog is also available as a subdialog of the New Image Plot dialog.

Tip #1: You can open the Modify Image dialog quickly by Control-clicking (*Macintosh*) or right-clicking (*Windows*) and choosing Modify Image from the pop-up menu. The graph must contain an image plot for this to work.

Tip #2: Use the preferences to change the default image appearance, so you won't be making the same changes over and over. See **Image Preferences** on page II-361.

The Modify Image Appearance Dialog

The Modify Image Appearance dialog applies to false color and indexed color images, but not direct color images. See **Direct Color Details** on page II-358.



If your image is indexed color, choose the matching color index wave from the Color Index Wave pop-up menu. For Color Index Wave details, see **Indexed Color Details** on page II-356.

If your image is false color, choose a built-in color table from the pop-up menu. Autoscaled color mapping assigns the first color in a color table to the minimum value of the image data and the last color to the maximum value. (The dialog uses “Z” to refer to the values in the image wave.) For more information, see **Color Tables** on page II-349.

Indexed and color table colors are distributed between the minimum and maximum Z values either linearly or logarithmically, based on the `ModifyImage log` parameter, which is set by the Log Colors checkbox.

Only in rare cases will you want to choose a Lookup Wave; "_none_" is the usual choice. See **Color Table Ranges** on page II-349 for a reason to use a Lookup Wave.

You can achieve special effects using a Color Index Wave to select colors for your image; see **Indexed Color Details** on page II-356. The built-in color tables are easier to use, however.

Use Explicit Mode to select specific colors for specific Z values in the image. If an image element is exactly equal to the number entered in the dialog, it will be given the assigned color. This is not very useful for images made with floating-point data; it is intended for integer data. It is almost impossible to enter exact matches for floating-point data.

When you select Explicit Mode for the first time, two entries are made for you assigning white to 0 and black to 255. A third blank line is added for you to enter a new value. If you put something into the blank line, another blank line is added.

To remove an entry, click in the blank areas of a line in the list to select it and press Delete (*Macintosh*) or Backspace (*Windows*).

If you make an image using a three-layer wave containing direct RGB values, you cannot alter it with this dialog (unless it has multiple chunks, in which case you can modify only the displayed chunk).

How Images Are Displayed

Igor displays images by drawing one “image rectangle” for each data value in the image matrix wave. The size and position of these rectangles is controlled by the range of the graph axes, the graph width and height modes, and the X and Y coordinates of the image rectangles.

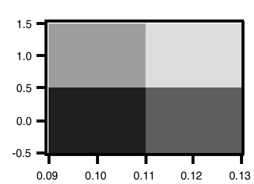
Images are displayed behind all other objects in a graph except the background color.

If your image data is a floating point type, you can use NaN to represent missing data. This allows the graph background color to show through. This feature is particularly useful when you want to create overlays.

Image X and Y Coordinates

Images are displayed versus axes just like XY plots. Individual pixels fill a rectangle defined by adjacent X and Y coordinate values and each rectangle is centered on the X and Y coordinate for the pixel. Here is an example that uses a 2x2 matrix to exaggerate the effect:

```
Make/O small={{0,1},{2,3}}
SetScale/I x 0.1,0.12,"", small
Display
AppendImage small    // _calculated_ X & Y
ModifyImage small ctab={-0.5,3.5,Grays}
```

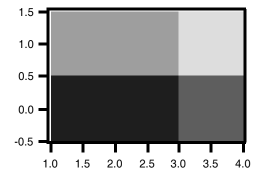


Note that on the X axis the rectangles are centered on 0.1 and 0.12, the row X indices of the matrix as defined by its X scaling. On the Y axis they are centered on 0 and 1 since the Y scaling was left at the default (point scaling).

When appending an image to a graph, you can supply optional X and Y waves to define the coordinates *of the rectangle edges*. These waves need to contain one more data point than the X (row) or Y (column) dimension of the matrix to define the end of the last rectangle.

Here is an example, using the matrix from above, that should make this point clear:

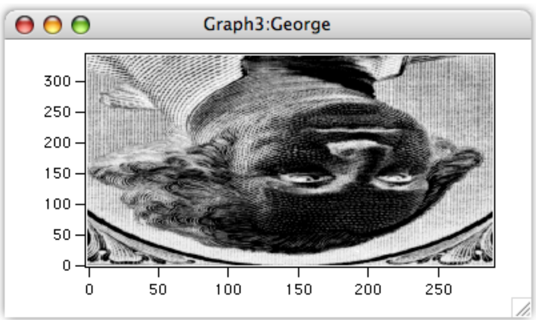
```
Make smallx={1,3,4}
Display
// define X edges with smallx
AppendImage small vs {smallx,*}
ModifyImage small ctab={-0.5,3.5,Grays,0}
```



In this case the Y is unchanged because we did not supply a Y coordinate wave. For the X axis, the X coordinate wave (smallx) now controls the start and end of each rectangle. The rectangles are not centered because they can not be: if we tried to center the rectangles horizontally on 1 and 3 then there would be a gap between them.

Image Orientation

By default, the AppendImage operation draws increasing Y values (matrix column indices) upward, and increasing X (matrix row indices) to the right. Most image formats expect Y to increase downward:

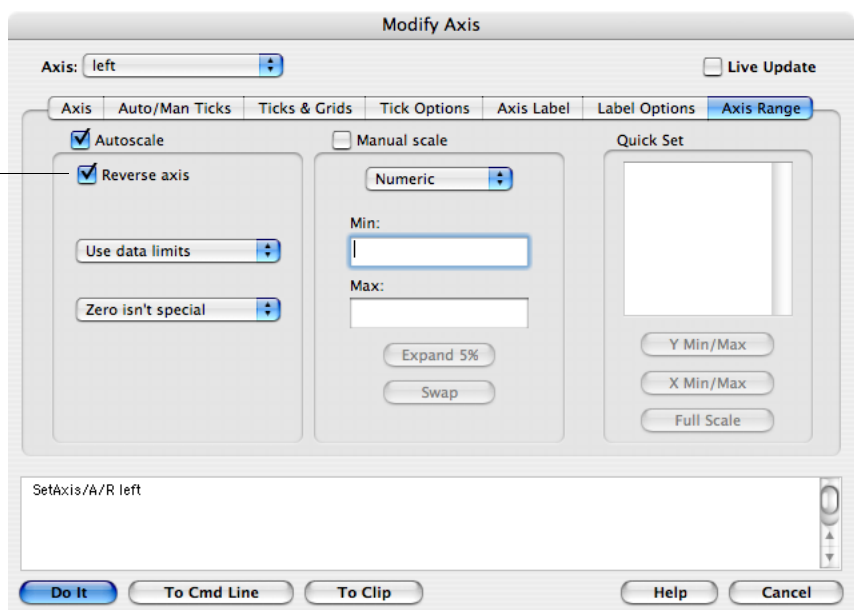


TIFF file displayed with default AppendImage settings

Alternatively you can select the “Use NewImage command” checkbox in the New Image Plot dialog. NewImage automatically reverses the left axes (see **Use NewImage Command** on page II-344).

You can flip an image vertically by reversing the Y axis, and horizontally by reversing the X axis, using the Axis Range tab in the Modify Axes dialog:

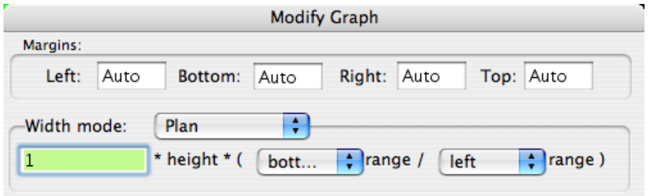
Reverse autoscaled vertical axes with this checkbox to make images appear right-side-up.



You can also flip the image vertically by reversing the Y scaling of the matrix wave. Note that if you use the **NewImage** operation (see page V-436), there is no need to flip the image.

Image Rectangle Aspect Ratio

By default, Igor does not make the image rectangles square. Use the Modify Graph dialog (in the Graph menu) to correct this by setting Plan width mode:



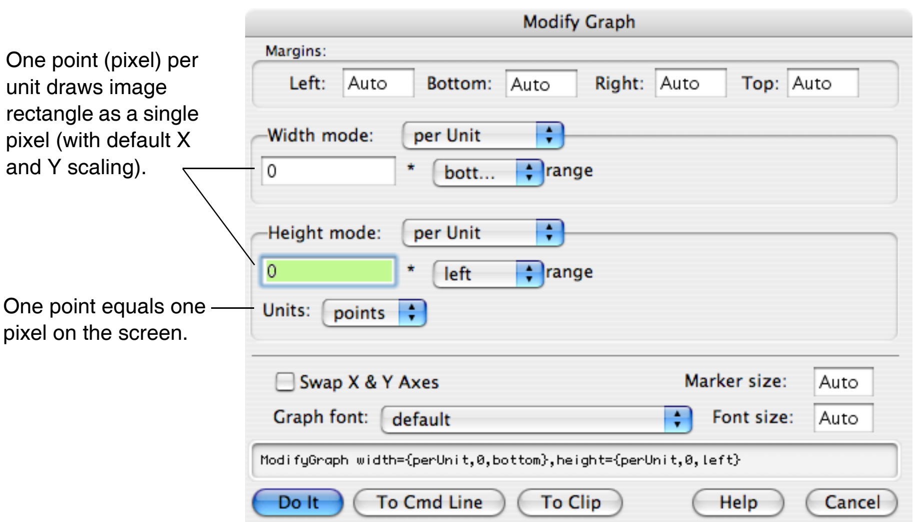
You can use the Plan height mode to accomplish the same result.

If DimDelta(*matrixWave*,0) ¦ DimDelta(*matrixWave*,1), you will need to enter the ratio (or inverse ratio) of these two values in the Plan height (or width):

```
SetScale/P x 0,3,"", mat2dImage
SetScale/P y 0,1,"", mat2dImage
ModifyGraph width=0, height={Plan,3,left,bottom}
// or
ModifyGraph height=0, width={Plan,1/3,bottom,left}
```

Do not use the Aspect width or height modes; they make the entire image plot square even if it shouldn't be.

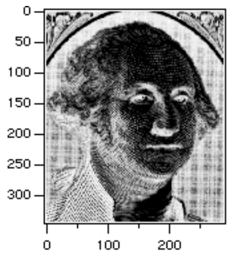
Plan mode ensures the image rectangles are square, but it allows them to be of any size. If you want each image rectangle to be a single (square) pixel, use the per Unit width and per Unit height modes. With point X and Y scaling of an image matrix, use one point (not one inch) per unit:



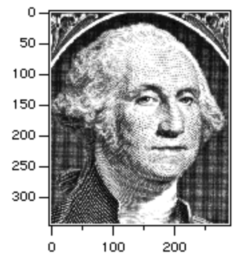
You can also flip an image along its diagonal by setting the Swap XY checkbox.

Image Polarity

Sometimes the image's pixel values are inverted, too. False color images can be inverted by reversing the color table. Select the Reverse Colors checkbox in the Modify Image Appearance dialog. See **Color Tables** on page II-349. To reverse the colors in an index color plot is harder: the rows of the color index wave must be reversed.



After SetAxis/A/R left
ModifyGraph width={Plan,1,bottom,left}



After reversing
the Grays color table

Color Tables

The data values contained in your matrix are normally linearly mapped into a table of colors containing a sweep of colors that lets the viewer easily identify the data values. The data values can be logarithmically mapped by using the `ModifyImage log=1` option, which is useful when the data values span multiple orders of magnitude.

There are 56 built-in color tables you can use with false color images.

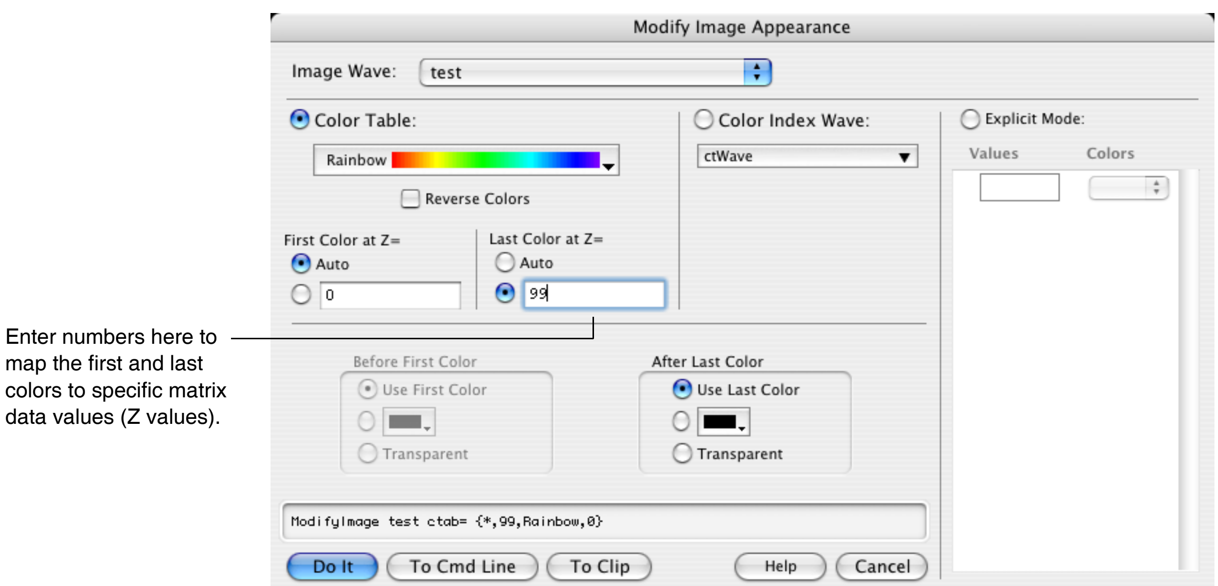
Note: You can not create your own color tables, but you can use a color index wave to create a custom range of colors. See **Indexed Color Details** on page II-356.

The **CTabList** function (see page V-81) returns a list of all color table names. You can create a color index wave from any built-in color table with the **ColorTab2Wave** operation (see page V-57).

The ColorsMarkersLinesPatterns example Igor experiment (In the Testing & Misc folder) implements a Color Table Visual Guide. These various color tables are summarized in the section **Color Table Details** on page II-353.

Color Table Ranges

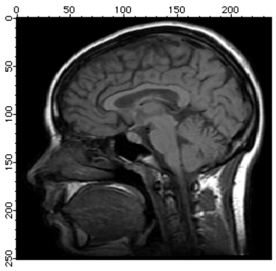
The range of data values that maps into the range of colors in the table can be set either manually or automatically using the Modify Image Appearance dialog.



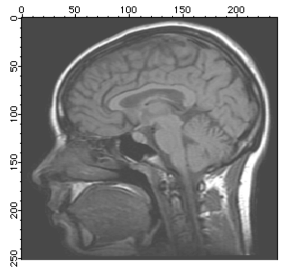
When you choose to autoscale the first or last color, your image is scaled using, respectively, the minimum or maximum data values.

By changing the “First Color at Z=” and “Last Color at Z=” values you can examine subtle features in your data.

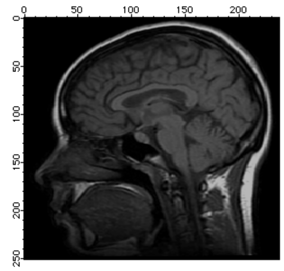
For example, when using the Grays color table, you can lighten the image by assigning the First Color (which is black) to a number lower than the image minimum value. This maps a lighter color to the minimum image value. To darken the maximum image values, assign the Last Color to a number higher than the image maximum value, mapping a darker color to the maximum image value.



ctab = {0,255,Grays}



ctab = {-100,255,Grays}



ctab = {0,355,Grays}

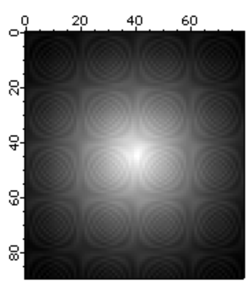
Data values greater than the range maximum are given the last color in the color table, or they can all be assigned to a single color or made transparent. Similarly, data values less than the range minimum are given the first color in the color table, or they can all be assigned to a single color (possibly different from the max color), or made transparent.

Example: Overlaying Data on a Background Image

By setting the image range to render small values transparent, you can see the underlying image in those locations, which helps visualize where the nontransparent values are located with reference to a background image. Here’s a fake weather radar example.

First, we create some “land” to serve as a background image:

```
Make/O/N=(80,90) landWave
landWave = 1-sqrt((x-40)*(x-40)+(y-45)*(y-45))/sqrt(40*40+45*45)
landWave = 7000*landWave*landWave
landWave += 200*sin((x-60)*(y-60)*pi/10)
landWave += 40*(sin((x-60)*pi/5)+sin((y-60)*pi/5))
NewImage landWave
```



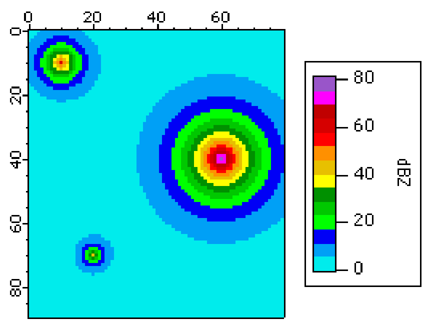
Then we create some “weather” radar data ranging from about 0 to 80 dBZ:

```
Duplicate/O landWave overlayWeather                    // "weather" radar values
overlayWeather=60*exp(-(sqrt((x-10)*(x-10)+(y-10)*(y-10))/5))      // storm 1
overlayWeather+=80*exp(-(sqrt((x-60)*(x-60)+(y-40)*(y-40)))/10)    // storm 2
overlayWeather+=40*exp(-(sqrt((x-20)*(x-20)+(y-70)*(y-70)))/3)     // storm 3
SetScale d, 0, 0, "dBZ", overlayWeather
```

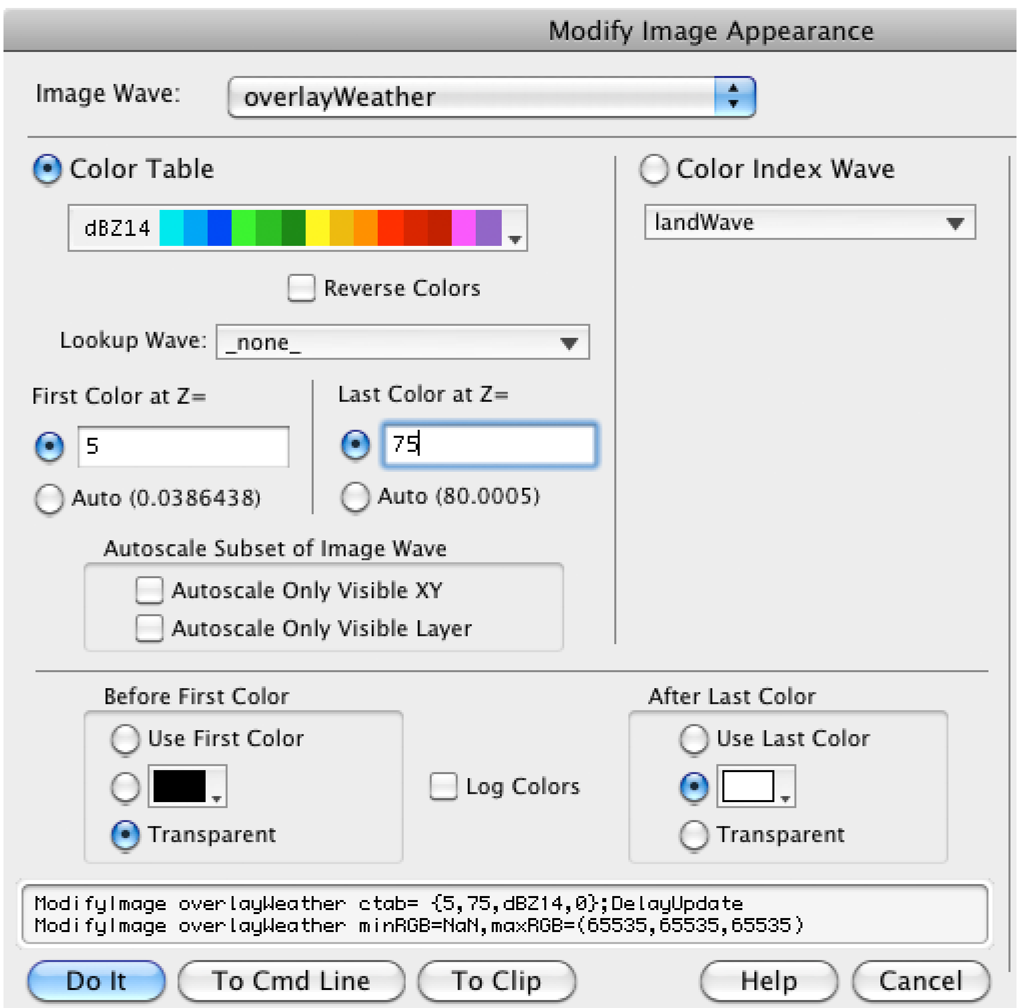
And append it using the same axes as the landWave to overlay the images. With the default color table range, the landWave is totally obscured:

```
AppendImage/T overlayWeather
ModifyImage overlayWeather ctab= {*,*,dBZ14,0}
// Show the image's data range with a ColorScale
```

```
ModifyGraph width={Plan,1,top,left}, margin(right)=100
ColorScale/N=text0/X=107.50/Y=0.00 image=overlayWeather
```

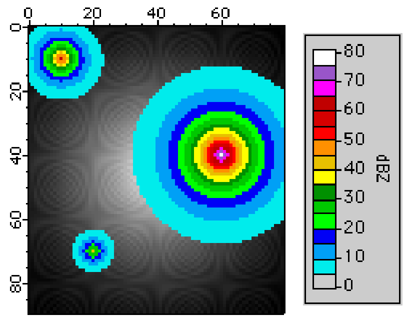


Calibrate the image plot colors to National Weather Service values for precipitation mode by selecting the dBZ14color table for data values ranging from 5 to 75, where values below 5 are transparent and values above 75 are white:



Modify the ColorScale to show a range larger than the color table values (0-80):

```
ColorScale/C/N=text0 colorBoxesFrame=1,heightPct=90,nticks=10
ColorScale/C/N=text0/B=(52428,52428,52428) axisRange={0,80},tickLen=3.00
```



Color Table Ranges - Lookup Table (Gamma)

Normally the range of data values and the range of colors are linearly related or logarithmically related if the `ModifyImage log` parameter is set to 1. You can also cause the mapping to be nonlinear by specifying a lookup (or "gamma") wave, as described in this next example.

Example: Using a Lookup for Advanced Color/Contrast Effects

The **ModifyImage** operation (see page V-415) with the lookup parameter specifies a 1D wave that modifies the mapping of scaled Z values into the current color table. Values in the lookup wave should range from 0.0 to 1.0. A linear ramp from 0 to 1 would have no effect while a ramp from 1 to 0 would reverse the colormap. Used to apply gamma correction to grayscale images or for special effects.

```
Make luWave=0.5*(1+sin(x/30))
Make /n=(50,50) simpleImage=x*y
NewImage simpleImage
ModifyImage simpleImage ctab= {*,*,Rainbow,0}

// After inspecting the simple image, apply the lookup:
ModifyImage simpleImage lookup=luWave
```

This example is a simplified version of the Image Contrast Lookup panel available in the Image Processing package. Choose Analysis→Packages→Image Processing to load the package, then choose Image Contrast to open the panel.

Specialized Color Tables

Some of the color tables are designed for specific uses and specific numeric ranges.

The BlackBody color table shows the color of a heated "black body" (though not the brightness of that body) over the temperature range of 1,000 to 10,000 K.

The Spectrum color table is designed to show the color corresponding to the wavelength of visible light as measured in nanometers over the range of 380 to 780 nm.

The SpectrumBlack color table does the same thing, but over the range of 355 to 830 nm. The fading to black is an attempt to indicate that the human eye loses the ability to perceive colors at the range extremities.

The GreenMagenta16, EOSOrangeBlue11, EOSSpectral11, dBZ14, and dBZ21 tables are designed to represent discrete levels in weather-related images, such as radar reflectivity measures of precipitation and wind velocity and discrete levels for geophysics applications.

The LandAndSea, Relief, PastelsMap, and SeaLandAndFire color tables all have a sharp color transition which is intended to denote sea level. The LandAndSea and Relief tables have this transition at 50% of the range. You can put this transition at a value of 0 by setting the minimum value to the negative of the maximum value:

```
ModifyImage imageName, ctab={-1000,1000,LandAndSea,0}     // image plot
ColorScale/C/N=scale0 ctab={-1000,1000,LandAndSea,0}      // colorscale
```

The PastelsMap table has this transition at 2/3 of the range. You can put this transition at a value of 0 by setting the minimum value to twice the negative of the maximum value:

```
ModifyImage imageName, ctab={-2000,1000,PastelsMap,0}     // image plot
ColorScale/C/N=scale0  ctab={-2000,1000,PastelsMap,0}     // colorscale
```

This principle can be extended to the other color tables to position a specific color to a desired value. Some trial-and-error is to be expected.

Note: The BlackBody, Spectrum, and SpectrumBlack color tables are based on algorithms from the Color Science web site:

`<http://www.physics.sfasu.edu/astro/color.html>`.

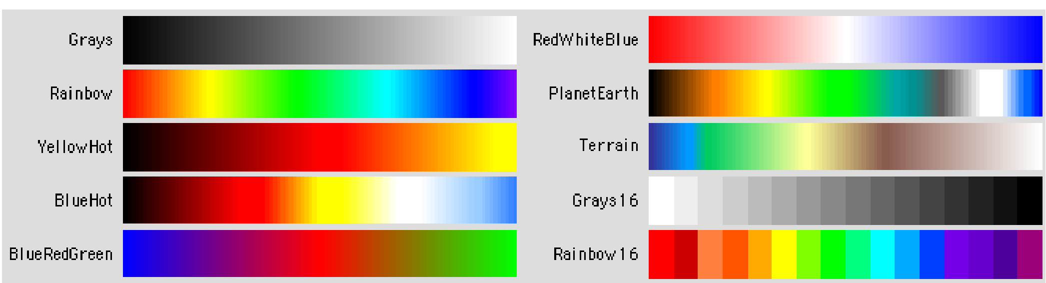
Also see *Color Science* by Wyszecki and Stiles.

Color Table Details

The built-in color tables can be grouped into several categories. For most purposes you may only need to use the “compatible” color tables; this is especially true if you wish to use your experiments with earlier versions of Igor Pro.

Igor Pro 4-Compatible Color Tables

Igor Pro 4 supports10 built-in color tables: Grays, Rainbow, YellowHot, BlueHot, BlueRedGreen, RedWhiteBlue, PlanetEarth, Terrain, Grays16, and Rainbow16. These color tables have 100 color levels except for Grays16 and Rainbow16, which only have 16 levels.



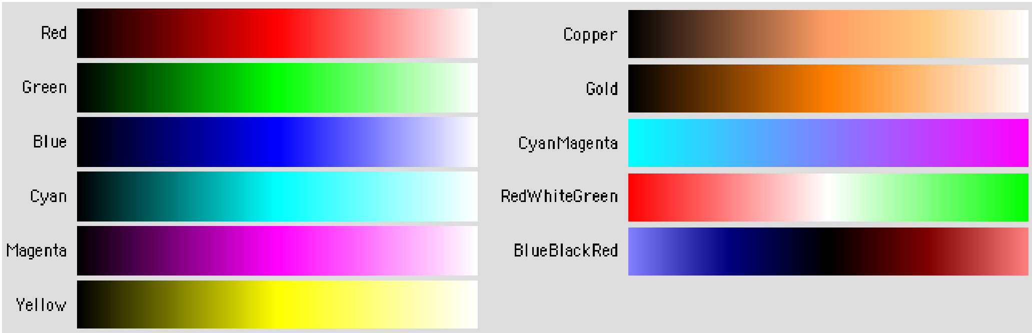
Igor Pro 5-Compatible Color Tables

Igor Pro 5 added 256-color versions of the eight 100-level color tables in Igor Pro 4 (Grays256, Rainbow256, etc.), new gradient color tables, and new special-purpose color tables.

Gradient Color Tables

These are 256-color transitions between two or three colors.

Color Table Name	Colors	Notes
Red	256	Black → red → white.
Green	256	Black → green → white.
Blue	256	Black → blue → white.
Cyan	256	Black → cyan → white.
Magenta	256	Black → magenta → white.
Yellow	256	Black → yellow → white.
Copper	256	Black → copper → white.
Gold	256	Black → gold → white.
CyanMagenta	256	
RedWhiteGreen	256	
BlueBlackRed	256	

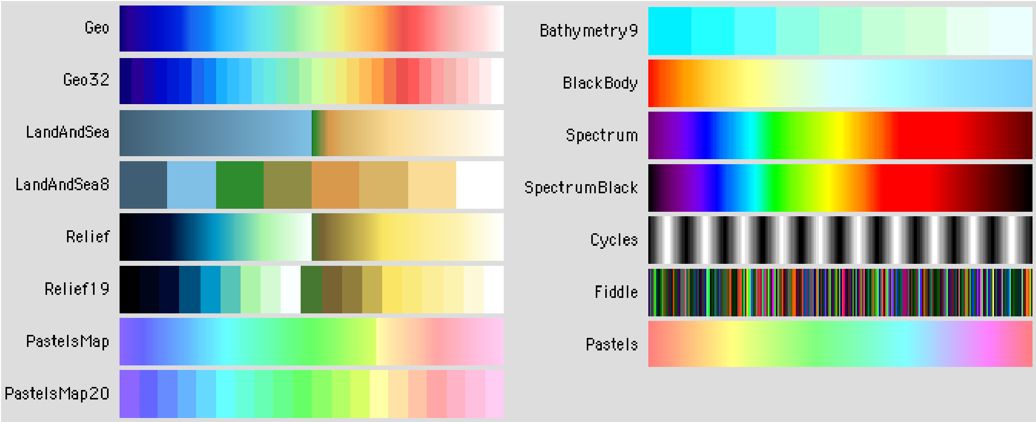


Special-Purpose Color Tables

The special purpose color tables are ones that will find use for particular needs, such as coloring a digital elevation model (DEM) of topography or for spectroscopy. These color tables can have any number of color entries.

The following table summarizes the various special-purpose color tables.

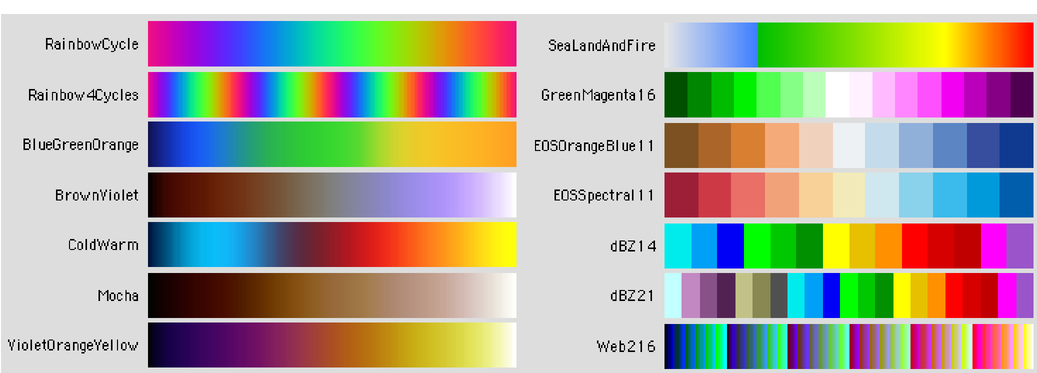
Color Table Name	Colors	Notes
Geo	256	Popular mapping color table for elevations. Sea level is around 50%.
Geo32	32	Quantized to classify elevations. Sea level is around 50%.
LandAndSea	255	Rapid color changes above sea level, which is at exactly 50%. Ocean depths are blue-gray.
LandAndSea8	8	Quantized, sea level is at about 22%.
Relief	255	Slower color changes above sea level, which is at exactly 50%. Ocean depths are black.
Relief19	19	Quantized, sea level is at about 47.5%.
PastelsMap	301	Desaturated rainbow-like colors, having a sharp green→yellow color change at sea level, which is around 66.67%. Ocean depths are faded purple.
PastelsMap20	20	Quantized. Sea level is at about 66.67%.
Bathymetry9	9	Colors for ocean depths. Sea level is at 100%.
BlackBody	181	Red → Yellow → Blue colors calibrated to black body radiation colors (neglecting intensity). The color table range is from 1,000 K to 10,000 K. Each color table entry represents a 50 K interval.
Spectrum	201	Rainbow-like colors calibrated to the visible spectrum when the color table range is set from 380 to 780 nm (wavelength). Each color table entry represents 2nm. Colors do not completely fade to black at the ends of the color table.
SpectrumBlack	476	Rainbow-like colors calibrated to the visible spectrum when the color table range is set from 355 to 830 nm (wavelength). Each color table entry represents 1 nm. Colors fade to black at the ends of the color table.
Cycles	201	Ten grayscale cycles from 0 to 100% to 0%.
Fiddle	254	Some randomized colors for “fiddling” with an image to detect faint details in the image.
Pastels	256	Desaturated Rainbow.



Igor Pro 6-Compatible Color Tables

Igor Pro 6 added 14 new color tables.

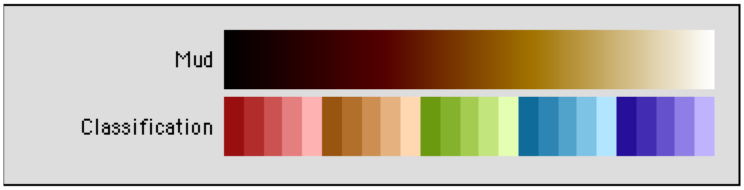
Color Table Name	Colors	Notes
RainbowCycle	360	Red, green, blue vary sinusoidally, each 120 degrees (120 values) out of phase. The first and last colors are identical.
Rainbow4Cycles	360	4 cycles with one quarter of the angular resolution.
BlueGreenOrange	300	Three-color gradient.
BrownViolet	300	Two-color gradient.
ColdWarm	300	Multicolor gradient for temperature.
Mocha	300	Two-color gradient.
VioletOrangeYellow	300	Multicolor gradient for temperature.
SeaLandAndFire	256	Another topographic table. Sea level is at 25%.
GreenMagenta16	16	Similar to the 14-color National Weather Service Motion color tables (base velocity or storm relative values), but friendly to red-green colorblind people.
EOSOrangeBlue11	11	Colors for diverging data (friendly to red-green colorblind people).
EOSSpectral11	11	Modified spectral colors (friendly to red-green colorblind people).
dBZ14	14	National Weather Service Reflectivity (radar) colors for Clear Air (-28 to +24 dBZ) or Precipitation (5 to 70 dBZ) mode.
dBZ21	21	National Weather Service Reflectivity (radar) colors for combined Clear Air and Precipitation mode (-30 to 70 dBZ).
Web216	216	The 216 “web-safe” colors, provides a wide selection of standard colors in a single color table. Intended for trace f(z) coloring using the ModifyGraph zColor parameter.



Igor Pro 6.2-Compatible Color Tables

Igor Pro 6.2 added 2 new color tables:

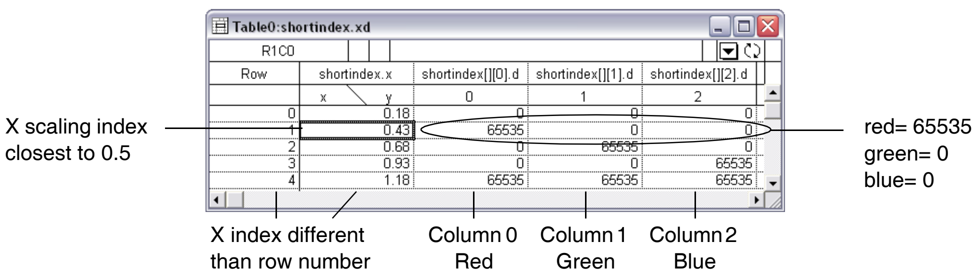
Color Table Name	Colors	Notes
Mud	256	Dark brown to white, without the pink cast of the Mocha color table. For Veeco atomic force microscopes.
Classification	25	5 hues for classification, 5 saturations for variations within each class.



Indexed Color Details

The data values contained in your matrix are used to select a color from a color index wave that you supply.

The color index wave must be a 2D wave with three columns containing red, green, and blue values that range from 0 (zero intensity) to 65535 (full intensity), and a row for each color.



Choosing a color from an X-scaled color index wave for image matrix Z value = 0.5

In the normal linear case, Igor finds the color for a particular matrix data value by choosing the row in the color index wave whose X index most closely matches the image matrix data value.

To choose the row, Igor converts the matrix data value into a row number as if executing:

```
colorIndexWaveRow= x2pnt(colorIndexWave,zImageValue)
```

which rounds to the nearest row and limits the result to the rows in the color index wave.

By setting the X scaling of the color index wave (Change Wave Scaling dialog), you can control how Igor maps the image matrix data value to a color. This is similar to setting the First Color and Last Color values for a color table.

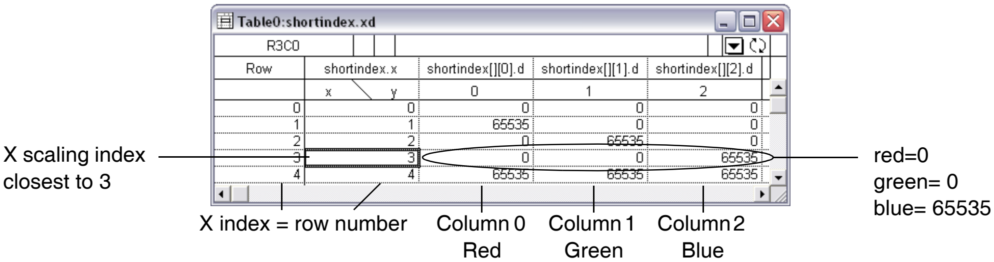
When the `ModifyImage log` parameter is set to 1, the colors are mapped using the log(x scaling) and log(image z) values this way:

```
colorIndexWaveRow=(nRows-1)*(log(zImageValue)-log(xMin))/(log(xmax)-log(xMin))
```

where,

```
nRows = DimSize(colorIndexWave,0)
xMin = DimOffset(colorIndexWave,0)
xMax = xMin + (nRows-1) * DimDelta(colorIndexWave,0)
```

The images are drawn fastest if the image matrix is an integer number type, the color index wave has point X scaling (the X index is equal to row number), and the normal linear color mapping is used. In this case, row 0 contains the color for image matrix values equal to 0, row 1 contains the color for image values equal to 1, etc.



Row	shortindex.x	shortindex[][0].d	shortindex[][1].d	shortindex[][2].d
	x / y	0	1	2
0	0	0	0	0
1	1	65535	0	0
2	2	0	65535	0
3	3	0	0	65535
4	4	65535	65535	65535

Choosing a color from a point-scaled color index wave for image matrix Z value = 3

Programming Note:

XOP (eXternal OPeration) code modules that read indexed image data from a file should create a color index wave to go with each image matrix.

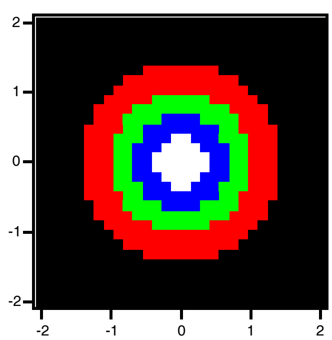
Example: Point-Scaled Color Index Wave

Here are the commands that created the point-scaled, unsigned 16-bit integer color index wave shown in the preceding table:

```
Make/O/W/U/N=(1,3) shortindex  // initially 1 row; more will be added
shortindex[0][]= {{0},{0},{0}}                  // black in first row
shortindex[1][]= {{65535},{0},{0}}              // red in new row
shortindex[2][]= {{0},{65535},{0}}              // green in new row
shortindex[3][]= {{0},{0},{65535}}              // blue in new row
shortindex[4][]= {{65535},{65535},{65535}}      // white in new row
```

These commands generate sample data and display it using the color index wave:

```
Make/O/N=(30,30)/B/U expmat   // /B/U makes unsigned byte image
SetScale/I x,-2,2,"" expmat
SetScale/I y,-2,2,"" expmat
expmat= 4*exp(-(x^2+y^2))     // test image ranges from 0 to 4
Display;AppendImage expmat
ModifyImage expmat cindex= shortindex
```



Programming Example: Adding Colors to a Built-In Color Table

A possible future extension to color tables would be to allow you to specify a special color to be used for out-of-range data. This example implements the feature with a color index wave generated by the **ColorTab2Wave** operation (see page V-57). The operation extracts colors from a built-in color table and places the red, green and blue values in a 3 column color index wave named M_colors.

We will insert special colors at the beginning and end of the color index wave, and then modify its X scaling so that the colors of the in-range data match those from the original color table.

First we create an image matrix containing data that ranges from -1 to 1 and display it over a range of -.5 to +.5 using the built-in Rainbow color table.

Tip: The commands in this example are in the Using Igor help file under the "Image Plots" topic. You can execute commands in a help file by selecting them and pressing Control or Ctrl plus Enter or Return.

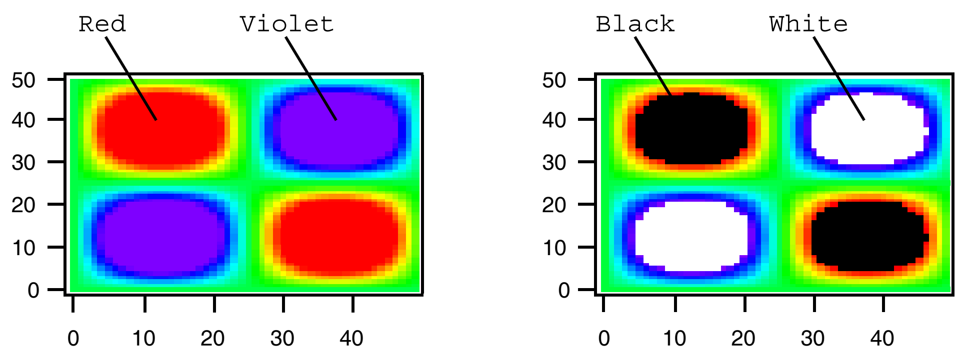
```
Make/O/N=(50,50) fpmat=sin(x/8)*sin(y/8)
Display;AppendImage fpmat;ModifyImage fpmat ctab= {-0.5,0.5,Rainbow}
```

All fpmat values below -0.5 are red, and all values above +0.5 are violet. See the Before picture below (you'll have to believe us about the colors).

We want under-range data (values below -0.5) to show as black and over-range data as white, so we execute the following lines:

```
ColorTab2Wave Rainbow          // Reads built-in color table into...
Rename M_colors,mycolors       // ...M_colors which we rename
InsertPoints 0,1,mycolors      // Insert a row of zero (which is black)
k0= DimSize(mycolors, 0)       // Number of rows we have now (101 currently)
InsertPoints k0,1,mycolors     // Append a row; k0 is now index of last row
mycolors[k0][]= {{65535},{65535},{65535}}     // last row is white

// Set X scaling of mycolors so that red is still at -0.5, and violet is still at 0.5
k1 = (0.5-(-0.5)) / (k0-1)     // color table increment
SetScale/I x,-0.5-k1,0.5+k1,"",mycolors
ModifyImage fpmat cindex= mycolors
```



Before adding special first and last colors to a Rainbow color index wave

After adding black first color and white last color

You can use this same method by substituting your value of "First Color at Z =" for -0.5 and your value of "Last Color at Z=" for 0.5 in the example commands. This method works with any built-in color table.

Direct Color Details

Direct color images use a 3D wave with 3 color planes containing absolute values for red, green and blue. Generally, direct color waves will be either unsigned 8 bit integers or unsigned 16 bit integers.

For 8-bit integer waves, 0 represents zero intensity and 255 represents full intensity. For all other number types, 0 represents zero intensity but 65535 represents full intensity. Out-of-range values are clipped to the limits.

Try the following example, executing each line one at a time. For best results, set your monitor to thousands or millions of colors.

```
Make/O/B/U/N=(40,40,3) matrgb;Display;Appendimage matrgb
matrgb[][][0]= 127*(1+sin(x/8)*sin(y/8))   // specify red,   0-255
matrgb[][][1]= 127*(1+sin(x/7)*sin(y/6))   // specify green, 0-255
matrgb[][][2]= 127*(1+sin(x/6)*sin(y/4))   // specify blue,  0-255
```

```
Redimension/S matrgb     // switch to floating point, image turns black
matrgb*=256              // floating point must be larger to display correctly
```

Because the appearance of a direct color image is completely determined by the image data, the Modify Image Appearance dialog has no effect on direct color images, and the dialog appears blank.

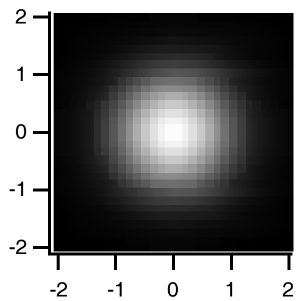
Creating Color Legends

You can create a color legend using a color scale annotation. You can find further details about creating legends using the Add Annotation dialog in Chapter III-2, **Annotations**, particularly in the **Legends** (see page III-52) and **Color Scales** (see page III-59) sections.

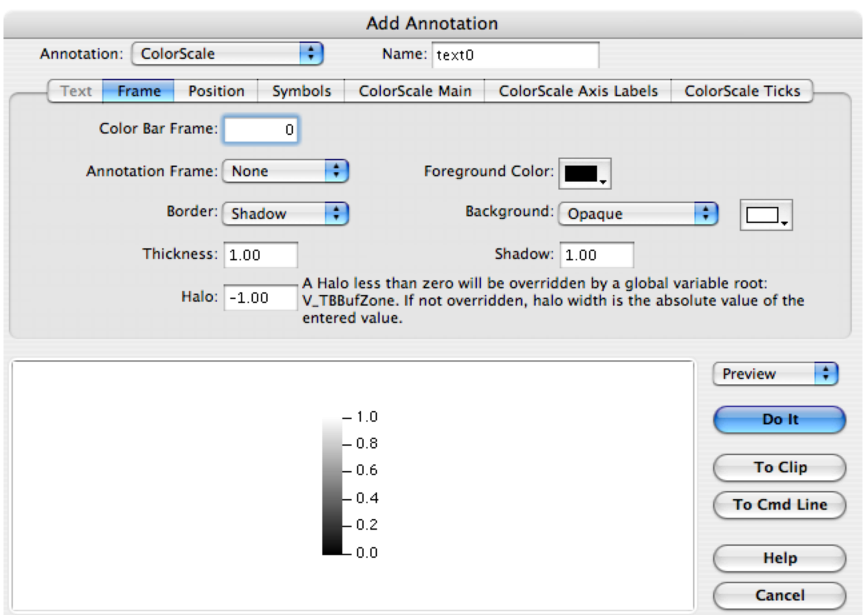
We will demonstrate with a simple image plot created by executing the following commands:

```
Make/O/N=(30,30) expmat
SetScale/I x,-2,2,"" expmat; SetScale/I y,-2,2,"" expmat
expmat= exp(-(x^2+y^2))          // data ranges from 0 to 1
Display;AppendImage expmat       // by default, left and bottom axes
ModifyGraph width={Plan,1,bottom,left},mirror=0
```

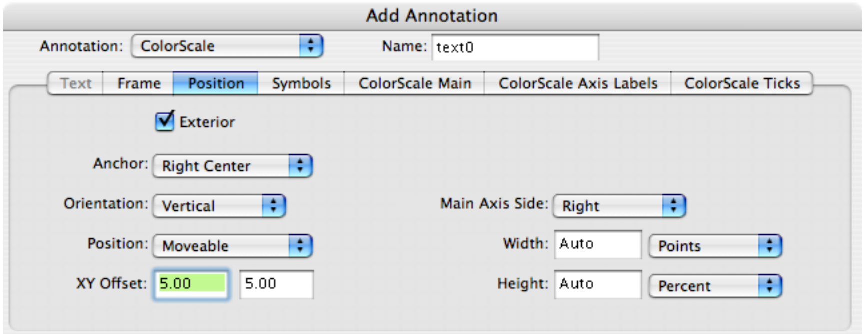
This creates the following image, using the autoscaled Grays color table:



Choose Add Annotation from the Graph menu. Choose “ColorScale” from the Annotation pop-up menu. Switch to the Frame tab, set the Color Bar Frame to 0 and the Annotation Frame to None.



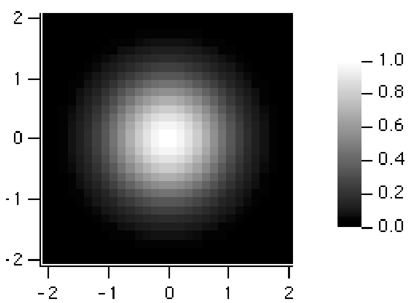
Switch to the Position tab, select Exterior and set the Anchor to Right Center:



Then click Do It. Igor will execute the following commands:

```
ColorScale/C/N=text0/F=0/A=RC/E image=expmat,frame=0.00
```

to generate the following image plot:



Double-click the color scale to edit it with the Modify Annotation dialog. Use the Position tab to change the color scale's orientation.

Image Instance Names

Igor identifies an image plot by the name of the wave providing Z values (the image matrix wave selected in the Z Wave list of the Image Plot dialogs). This "image instance name" is used in commands that modify the image plot.

In this example the image instance name is "zw":

```
Display; AppendImage zw             // new image plot
ModifyImage zw ctab={*,*,BlueHot}   // change color table
```

In the unusual case that a graph contains two image plots of the same data (to show different subranges of the data side-by-side, for example), an instance number must be appended to the name to modify the second plot:

```
Display; AppendImage zw; AppendImage/R/T zw  // two image plots
ModifyImage zw ctab={*,*,RedWhiteBlue}       // change first plot
ModifyImage zw#1 ctab={*,*,BlueHot}          // change second plot
```

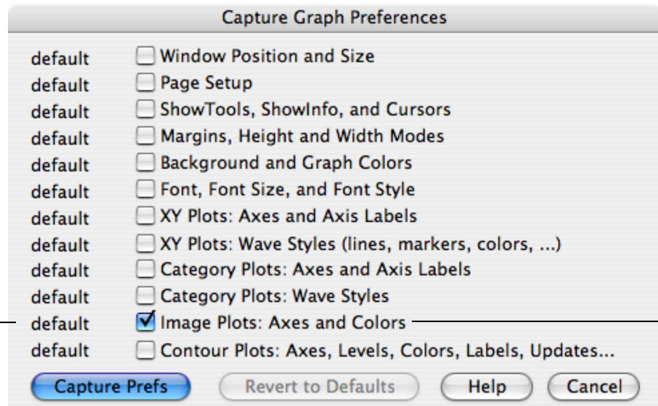
The Modify Image Appearance dialog generates the correct image instance name automatically. Image instance names work much the same way wave instance names for traces in a graph do. See **Instance Notation** on page IV-16.

Programming Note

The **ImageNameList** function (see page V-274) returns a string list of image instance names. Each name corresponds to one image plot in the graph. The **ImageInfo** function (see page V-262) returns information about a particular named image plot.

Image Preferences

You can change the default appearance of image plots by “capturing” preferences from a “prototype” graph containing image plots. Create a graph containing an image plot (or plots) having the settings you use most often. Then choose Capture Graph Prefs from the Graph menu. Select the Image Plots category, and click Capture Prefs.



The factory default settings for the Image Plot category are currently in effect.

The Image Appearance settings and the axis settings used for the image plot in the target graph will be captured as the new preferred settings.

Preferences are normally in effect only for *manual* operations, not for automatic operations from Igor procedures. Preferences are discussed in more detail in Chapter III-17, **Preferences**.

The Image Plots category includes both Image Appearance settings and axis settings.

Image Appearance Preferences

The captured Image Appearance settings are automatically applied to an image plot when it is first created (provided preferences are turned on). They are also used to preset the Modify Image Appearance subdialog of the New Image Plot dialog.

If you capture the Image Plot preferences from a graph with more than one image plot, the first image plot appended to a graph gets the settings from the image first appended to the prototype graph. The second image plot appended to a graph gets the settings from the second image plot appended to the prototype graph, etc. This is similar to the way XY plot wave styles work.

Image Axis Preferences

Only axes used by the image plot have their settings captured. Axes used solely for an XY, category, or contour plot are ignored.

The image axis preferences are applied only when axes having the same name as the captured axis are created by an AppendImage command. If the axes existed before AppendImage is executed, they will not be affected by the image axis preferences.

The names of captured image axes are listed in the X Axis and Y Axis pop-up menus of the New Image Plot and Append Image Plot dialogs. This is similar to the way XY plot axis preferences work.

For example, suppose you capture preferences for an image plot using axes named “myRightAxis” and “myTopAxis”. These names will appear in the X Axis and Y Axis pop-up menus in image plot dialogs.

- If you choose them in the New Image Plot dialog and click Do It, a graph will be created containing *newly-created* axes named “myRightAxis” and “myTopAxis” and having the axis settings you captured.
- If you have a graph which already uses axes named “myRightAxis” and “myTopAxis” and choose these axes in the Append Image Plot dialog, the image will be appended to those axes, as usual, but no captured axis settings will be applied to these *already-existing* axes.

You can capture image axis settings for the standard left and bottom axes, and Igor will save these separately from left and bottom axis preferences captured for XY, category, and contour plots. Igor will use the image axis settings for AppendImage commands only.

How to Use Image Preferences

Here is our recommended strategy for using image preferences:

1. Create a new graph containing a single image plot. Use the axes you will normally use, even if they are left and bottom. You can use other axes, too (select New Axis in the New Image Plot and Append Image Plot dialogs).
2. Use the Modify Image Appearance, Modify Graph, and Modify Axis dialogs to make the image plot appear as you prefer.
3. Choose Capture Graph Prefs from the Graph menu. Select the Image Plots category, and click Capture Prefs.

Image Plot Shortcuts

Since image plots are drawn in a normal graph, all of the **Graph Shortcuts** (see page II-306) apply. Here we list those which apply specifically to image plots.

Action	**Shortcut** (*Macintosh*)	**Shortcut** (*Windows*)
To modify the appearance of the image plot as a whole	Control-click in the plot area of the graph and choose Modify Image from the pop-up menu.	Right-click in the plot area of the graph and choose Modify Image from the pop-up menu.

References

Light, Adam, and Patrick J. Bartlein, The End of the Rainbow? Color Schemes for Improved Data Graphics, *Eos*, *85*, 385-391, 2004.

See also <`http://geography.uoregon.edu/datagraphics/color_scales.htm`>.

Wyszecki, Gunter, and W. S. Stiles, *Color Science: Concepts and Methods, Quantitative Data and Formula*, 628 pp., John Wiley & Sons, 1982.