There are many built-in color tables you can use with false color images. Also, you can create your own color table waves - see **Color Table Waves** on page II-399.

The **CTabList** returns a list of all built-in color table names. You can create a color index wave or a color table wave from any built-in color table using **ColorTab2Wave**.

The ColorsMarkersLinesPatterns example Igor experiment, in "Igor Pro Folder:Examples:Feature Demos 2", demonstrates all built-in color tables. These color tables are summarized in the section **Color Table Details** on page II-395.

Image Color Table Ranges

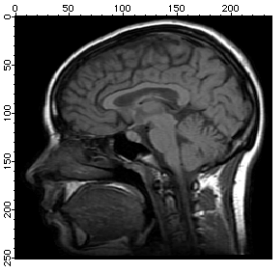
The range of data values that maps into the range of colors in the table can be set either manually or automatically using the Modify Image Appearance dialog.

When you choose to autoscale the first or last color, Igor examines the data in your image wave and uses the minimum or maximum data value found.

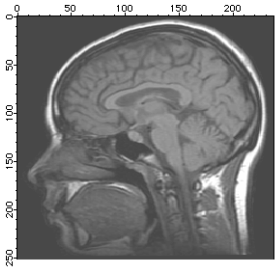
By changing the "First Color at Z=" and "Last Color at Z=" values you can examine subtle features in your data.

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.

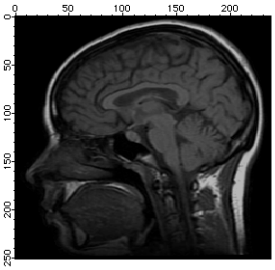
You can adjust these settings interactively by choosing Image→Image Range Adjustment.



ctab = {0,255,Grays}



ctab = {-100,255,Grays}



ctab = {0,355,Grays}

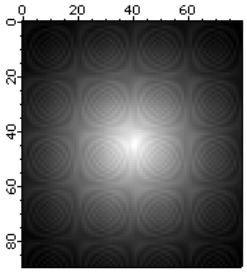
Data values greater than the range maximum are given the last color in the color table, or they can all be assigned to a single color or made transparent. Similarly, data values less than the range minimum are given the first color in the color table, or they can all be assigned to a single color (possibly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

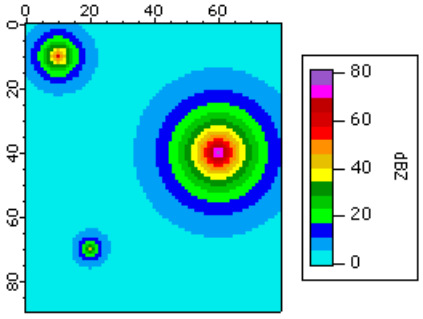


Then we create some "weather" radar data ranging from about 0 to 80 dBZ:

```
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
```

We append the overlayWeather wave 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
```



We 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:

We 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
```