Subject: Re: color table
Posted by J.D. Smith on Wed, 26 Mar 1997 08:00:00 GMT
View Forum Message <> Reply to Message

Ye Hong wrote:
>
> Does anyone know why the system variable !d.table_size is changing ?
> Is !d.table_size machine-dependent?

!D.Table_Size represents the number of colors in IDL's internal color table. It depends on your machine color bit depth (e.g. 8-bit or 24-bit).

> I have my own color table which includes 235 colors. When I need color
> image, I load it in. It has been working fine until yesterday when I
> found only first 90 color were loaded. It took me a while to figure out
> that !d.table_size was changed from 256 to 90. After I exited and
> re-ran IDL several time, I found !d.table_size was changing although it
> was 256 in most cases.

!D.Table_Size isn't set until you open an IDL window for the first time. Until then, it defaults to 256 (which *doesn't* mean you have 256 colors). I open and delete a small window upon startup (via the idl startup file) to ensure the system variable actually reflects the number of available colors. Loadct does this for you when loading a built-in color table. If you don't want to do this on startup, you could always add code like:

```
 if !d.name eq 'X' and !d.window eq -1 then begin ;Uninitialized?
    ;;If so, make a dummy window to determine the # of colors available.
    window,/free,/pixmap,xs=4, ys=4
    wdelete, !d.window
 endif
```

The more colors being used by other applications (e.g. Netscape), the fewer IDL can take for itself. Likely, when you only got 90, you were running some other color hungry app.

To implement your color table, either create it on the fly to fit into !D.Table_Size (which is fine if it's simple -- see <http://www.dfanning.com/tips/create_colortable.html>), or store the r,g,b vectors in a file, read them in, and down-sample by interpolation to fit. This is essentially what loadct does with the built-in color tables with a set of commands like:

```
   if nc ne 256 then begin      ;Interpolate
      p = (lindgen(nc) * 255) / (nc-1)
      r = r(p)
```

```
    g = g(p)
    b = b(p)
  endif
```

where nc is the number of colors (defaults to !D.Table_Size) and the r,g,b vectors are read in from colors1.tbl. You could be more careful with your interpolation, if, for instance, you had a few colors that you require to be in your map (e.g. some plotting colors at the bottom end). You might then set these required colors (n of them,say), and then interpolate the rest of your colormap onto the remaining !D.Table_Size-n spots in the palette.

Good Luck,

JD
