Using Color to Communicate, Not to Decorate

LeRoy Bessler PhD, Bessler Consulting and Research
Mequon, Wisconsin, USA – Le_Roy_Bessler@wi.rr.com

Abstract

The principles of effective use of color to communicate, rather than decorate, are software-independent, but this tutorial, for any data visualization or reporting situation, includes information specific to SAS, Excel, or the web. ODS Graphics in SAS Version 9.3 and its predecessors have a built-in color communication problem, which you need to know how to avoid. Color does more than merely add visual excitement to your output. Among the commonest visual communication design mistakes are those made with color. When it comes to the pitfalls of, and best practices for, using color, you don't know what you don't know. This tutorial can help.

Introduction

This paper is based on decades of working with color for communication, starting when color devices for computer output were uncommon, were not cheap, and did not deliver very good results. This is the latest update to my work on color in Reference 2. Since *color does not improve bad design*, see also Reference 3, which is my broader discussion of communication-effective data visualization presented at the Wisconsin Illinois SAS Users Conference on 4 November 2013. My thanks to Robert Allison for a link to where he demonstrates the utility of transparent colors.

Provided here are guidelines for color communication, a discussion of related topics, and code samples and macro tools that you might find useful. Color swatches can be generated with just a few SAS statements, and sample charts for evaluating text-background color combinations are not much more difficult. For high-volume samples, macro-based color design tools are also provided. Also included are:

(a) code to show you how to color every possible part of a table created with PROC PRINT; and
(b) code for an example of communication-effective color coding of a data display, as an antidote to the regrettably popular "traffic-lighting" documented and demonstrated so often elsewhere .

I regret that I cannot provide bibliographic citations for the research study results cited immediately below, nor for the remarks attributed later to some experts. Such information is drawn from notes taken from reading and listening about twenty years ago. Work by those people, as well as research reports on color, can be found via the web. For web search, also use the British spelling "colour".

Among the benefits of color reported in studies are:

(1) increased readership;
(2) increased reading speed and comprehension;
(3) faster learning
(4) reduced error rates; and
(5) improved recognition, recall, and response.

The intended audience for this paper is any SAS user, and users of any software that outputs color.

NOTE: The code in Appendix C, developed in 2004, has NOT been recently tested.

Access To Color Excess:
All of the Ways to Color a PROC PRINT Table

How to “achieve” the above is explained later in the paper.

Bessler's Ten Guidelines for Color Communication

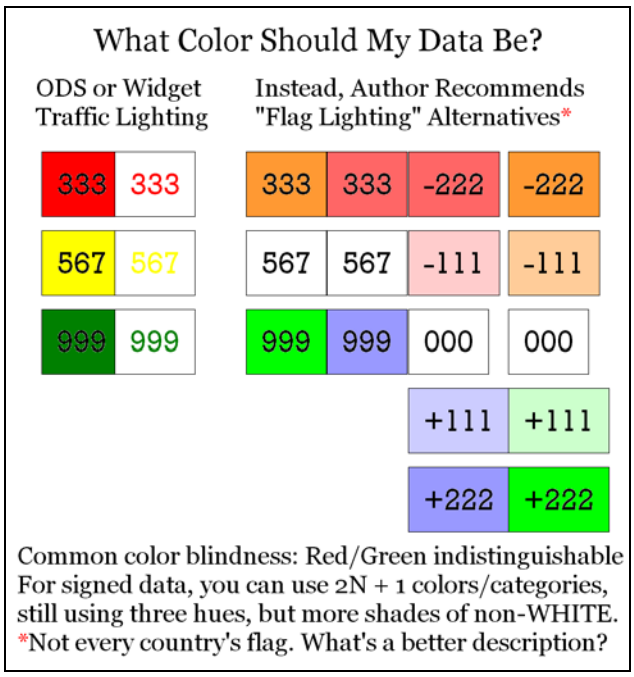
1. For Those Who Can't See a Color Difference, There Is None
2. Use Color To Communicate, Not To Decorate
3. Use of Color Can Confuse, Rather Than Communicate
4. Maximize Color Contrast between Text and Background
5. Make Colored Text and Lines Thicker,
   Colored Plot Point Symbols and Colored Legend Elements Bigger
6. A Light Color Might Be the Right Color
7. UnColor Might Be the Right Color
8. "Transparent Color" Might Be the Right Color
9. Beware Of Color Names
10. Color Differs on Different Media

Let me discuss in detail each of these, before moving on to other color communication information.

1. For Those Who Can't See a Color Difference, There Is None

The commonest color blindness cannot distinguish between red and green.

Despite this fact, first reported to SAS users in Reference 1, there are many and continuing well-intended, but misguided, examples of papers and web postings explaining how to do "traffic lighting".



Prof. Jay Neitz of the Eye Institute of the Medical College of Wisconsin reported that 8 to 10 percent of American males have some form of color blindness, but, due to genetic differences, only about one-half percent of American females. I first learned this in a report in a local Milwaukee newspaper back in 1995. I cannot provide the bibliographic citation for this information. This fact has been repeatedly mentioned elsewhere since. I am confident that you can verify it via a web search, and maybe the incidence reported will be somewhat different. I myself have had a boss and a client who suffer from this regrettably underappreciated problem.

Figure 1 in Appendix B uses "Irish Flag Lighting", but with gray substituted for white.

A sample program to color code a table with an alternative to traffic lighting is provided later.

2. Use Color To Communicate, Not To Decorate

There are two necessary uses of color to communicate:

- a legend of color squares used to match area fills of pie slices, bars, or geographic unit areas
- a legend of colored line segments, colored markers, or a combination of the two used to match plot lines/points

NOTE: If using colored markers for a plot, use solid-filled markers so that the color is more massive—i.e., visible. In ODS Graphics, all of those have a symbol name that ends with the suffix "Filled", as in CircleFilled, SquareFilled, etc.

There is an optional use of color to communicate:

- Color-coding of data entries in a table, BUT NOT TRAFFIC LIGHTING (as noted in Guideline 1)

The color coding can be applied to the foreground text, i.e., the data entry characters themselves (digits and/or alphabetic characters), or instead to the table cell background. Since the color of text is sometimes hard to discern (if insufficiently large and/or not thickly drawn enough), it is better to color-code the background, but background and text must always be high contrast—see Guideline 4.

NOTE: Color gradient backgrounds for a graph or web page are always anti-readable, and unnecessary.

When and How To Use Color

1. If your visual communication has no need to distinguish response levels or categories, use Black and White, or some other color pair for foreground and background.

2. If you have a few levels or categories, gray shades may suffice.

3. If you have many levels or categories, color is necessary.

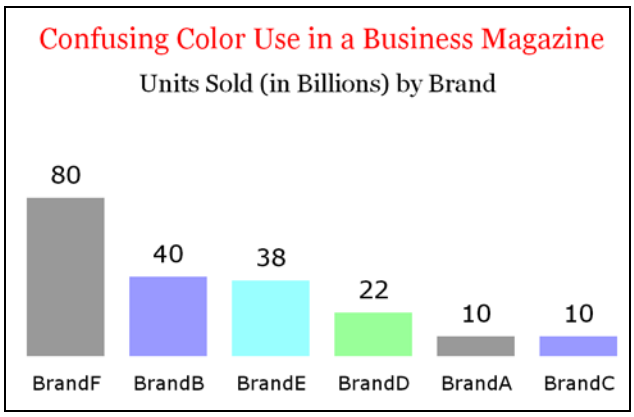
NOTE: In Case 3 above, be careful when trying to limit your palette to a range of shades (lightness/darkness) levels of a single hue. When trying to color code the values of a continuous variable, this seems like a natural and appropriate solution. Unfortunately, *it is impossible to reliably distinguish more than five shades of a single hue*, and any legend with color samples for too numerous values or ranges is actually useless—the graph or map looks very nice, but it is impossible to reliably interpret with precision. Of course, you may be able to expand your palette with Black and/or White, depending on the application and the background color.

3. Use of Color Can Confuse, Rather Than Communicate

Viewers attribute significance to your use of color, even when none is intended, proving the wisdom of Guideline 2. Be careful what you do whenever you use color. Using color without a design objective can disorient, confuse, or even mislead the viewer. Failed communication is always the fault of the transmitter, not the receiver.

The content of the example below is different from a magazine illustration that I saw, but its misuse of color is exactly parallel. There is NO relationship between BrandF and BrandA, and none between BrandB and BrandC. So, what does this use of color mean? Absolutely nothing!

Actually, after looking at other color graphs in that magazine article, I realized what had happened. All of the illustrations were limited (why I do not know) to a palette of only four colors. However, this bar chart would have been communication-effective, rather than confusing, if rendered in just ONE color.



4. Maximize Color Contrast between Text and Background

Contrast between foreground text and its background is essential to communication. ODS opened the door to "enhancing" tables with color. Besides the unfortunately popular Traffic Lighting, there are problems using Yellow with White, or Black (or other dark) text on dark or intense background colors.

It is no accident that books, newspapers, magazines, and scholarly journals are published with black text on white paper. The second most readable color combination is black and yellow.

Tip: A cheap, easy way to communicate with color is with black text on high-contrast color paper. If you want a way to enhance anyone's ability to quickly find your hardcopy handout, this is it.

One of the worst color combinations is yellow and white. I have seen traffic-lit tables with (extremely hard to read) yellow numbers in a white box. I have also seen black text on blue backgrounds. Yes, there are worse choices that come to mind: white on white, black on black, etc. ☺

Evaluate the contrast samples below.

gray black
yellow orange red magenta
green
cyan blue medium blue dark blue

white gray
yellow orange red magenta
green
cyan blue medium blue dark blue

See also the contrast demonstration charts in Figures 6 & 7 of Appendix B. A macro tool to create your own samples of text-background color combinations is provided in Appendix C.

NOTE: If designing any graph or table that might be printed, remember that adequate contrast for online display, which is bright and shines at you, does not guarantee the same for hardcopy, which is not brightly backlit. This is just one example of Guideline 10.

5. Make Colored Text and Lines Thicker, Colored Plot Point Symbols and Colored Legend Elements Bigger

The color of thin text, thin lines, small plot markers, and small legend elements is difficult to distinguish.

Text thickening can be achieved with bold or with use of font faces that are inherently thicker.

Both ODS Graphics and SAS/GRAPH provide controls for line thickness, and for marker size as well marker choice.

However, Excel pie chart legends suffer from this color distinguishability problems. The size of the color samples in the legend is small, and there is no way to increase it.

Also all legends with color squares created with ODS Graphics or the SG procedures in SAS Version 9.3 and prior also suffer from this usability defect, as you can see in several examples of communication-INeffective color legends in Reference 3.

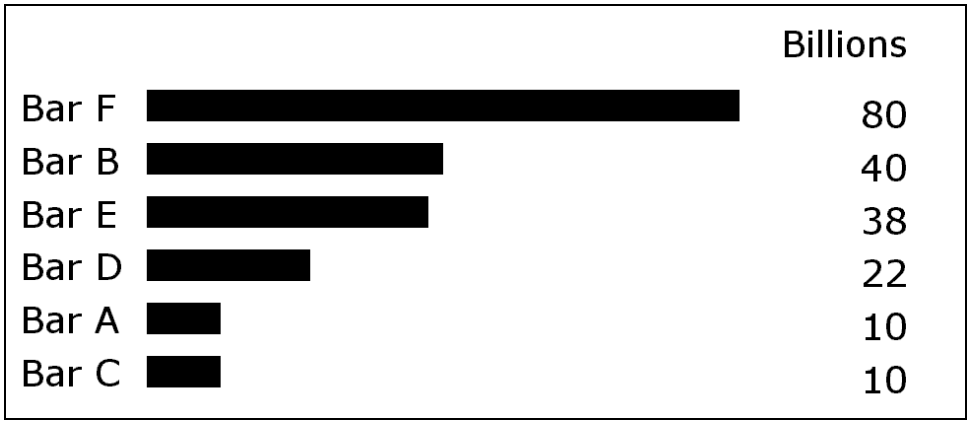
SAS/GRAPH provides absolute control of legend element size.

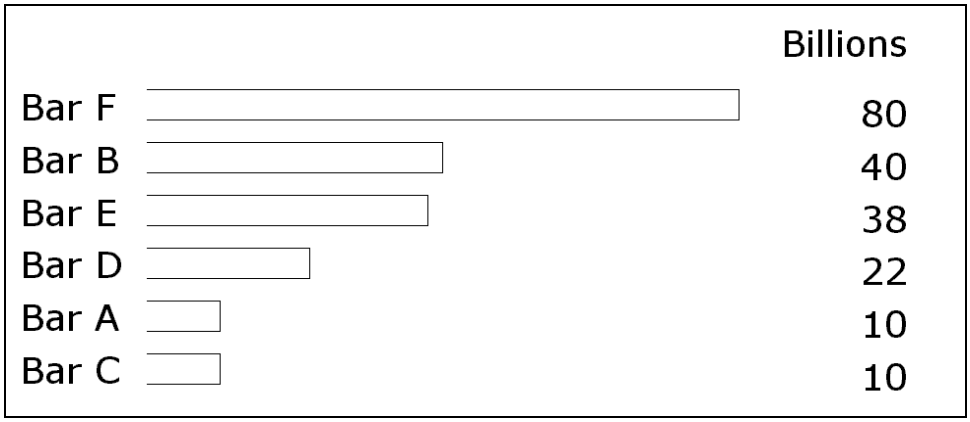
I reported the ODS Graphics bar chart legend color communication problem to SAS Technical Support, and it has been remedied in Version 9.4 with the AUTOITEMSIZE option for the KEYLEGEND statement.

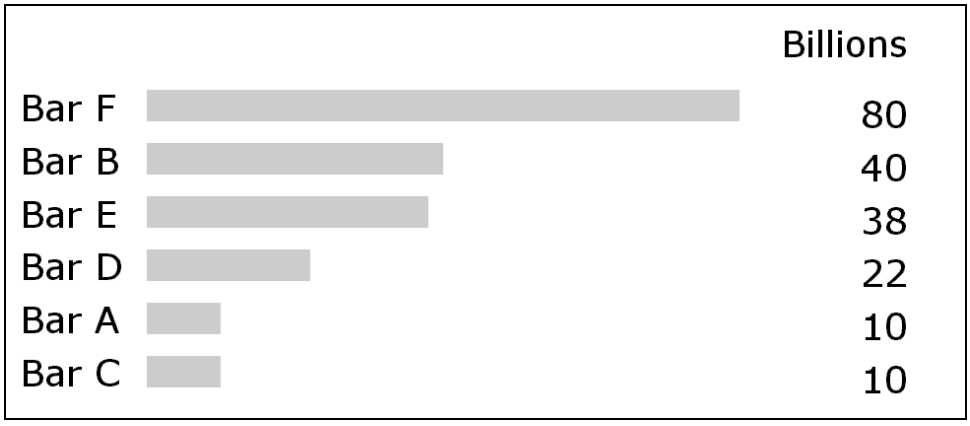
This feature is actually superior to the sizing option in SAS/GRAPH, in that AUTOITEMSIZE sizes the markers in the legend in proportion to the font size used for the legend value description. SAS/GRAPH requires you to experiment to get the marker size in the correct proportion to the text font size—if you care, which I do.

6. A Light Color Might Be the Right Color

This pertains to bar charts with area fill, especially ones with numerous bars. Filling the bars with a full-strength color causes the bars to dominate the image. Leaving them empty can cause visual confusion between bars and spaces. If the examples below had more bars, these visual effects would be more pronounced.







7. UnColor Might Be the Right Color

When Uncolor Suffices

- If you have no response levels/categories, use black & white.
- For a few levels or categories, gray shades may suffice.

Only if you have many levels or categories is color really necessary.

Benefits of Boring Black-and-White

Technology to print black and shades of gray is faster, cheaper, and more reliable.

Black, white, and shades of gray are easier to use. Not only is the equipment simpler, but also their use requires no agonizing over color selection.

Finally, such output is more copyable. Regardless of the proliferation of cheap color printers at work and at home, the copiers that you find in abundance in the workplace are still almost always black-and-white. Why does that matter? Well, good graphs, maps, and tables—if hardcopy—will get copied when people want to share them. If softcopy, users might decide to print them.

SAS Names for Grays

Light Gray, Medium Gray, and Dark Gray (e.g., CXCCCCCC, CX999999, and CX666666), even with White and Black, may not provide enough colors. If so, use color names of the form GRAY*ll*, where *ll* is a hexadecimal code with range 00-FF. FF (hex for decimal 255) is 0% gray, i.e., WHITE. 00 (hex for decimal 0) is 100% gray, i.e., BLACK. 80 (hex for decimal 128) is 50% gray.

Here are other correspondences for your possible use:

D5: 17%, CC: 20%, C0: 25%, AA: 33%, 99: 40%, 66: 60%, 55: 67%, 40: 75%, 33: 80%, 2B: 83%

Unfortunately, however, the very dark shades of gray tend to be unusable.

How to Choose/Use SAS Grays

Gray shades too close together are difficult or impossible to distinguish.

Here is a theoretical algorithm for creating a gray color palette. Decide how many grays, N, are needed for the chart, divide 256 by N - 1, and use the quotient (in hexadecimal) as the increment from 00 to FF for *ll* in GRAY*ll* assignments. Subsets of the values provided in the section above can produce equally spaced grays for sets of 3, 4, 5, 6, or 7 PATTERN statements. However, as noted earlier, the human eye cannot reliably distinguish more than five shades of gray (or of any other color), and dark grays are problematic. Hence, I characterize this algorithm as “theoretical”.

Sometimes gray shades do not photocopy well. And black text on a gray background can be a problem.

NOTE: Grays with names of the form GRAYll are not browser-safe, `if that is a concern`.

8. “Transparent Color” Might Be the Right Color

At most, I have used transparent colors only once. They did not become available until Version 9.3 of SAS software. The best documentation is in the SAS/GRAPH User’s Guide, and the information there also applies if you are using ODS Graphics and any of its SG Procedures.

Here is a SAS Institute documentation link to a discussion of Using Transparency:

http://support.sas.com/documentation/cdl/en/graphref/65389/HTML/default/viewer.htm#p06kdvbdiulixcn1traua7qrtn4b.htm

You can find a section on RGBA colors (i.e., transparent RGB colors) at this link on Color-Naming Schemes:

http://support.sas.com/documentation/cdl/en/graphref/65389/HTML/default/viewer.htm#p0ekhb3mdqahk3n15wzt55qtctr7.htm

For numerous examples of uses of transparent colors, go to this collection prepared by the world’s most prolific demonstrator of the power of SAS/GRAPH (and author of *SAS/GRAPH: Beyond the Basics*):

http://robslink.com/SAS/democd_transparency/aaaindex.htm

9. Beware Of Color Names

Verify What You Get from SAS CNS Color Names, HTML Color Names, or Any Other Color Names

Through Version 8 of SAS/GRAPH, there were 292 “SAS Predefined Color Names”. These were, in effect, a subset of what the SAS documentation now calls CNS Color Names. A problem that I reported many years ago with the predefined color names (i.e., that some light shades turned out to be darker than the “unlight” shades of the same hue) might have been solved. I recommend that you assume nothing.

However, now color variations that are theoretically possible in the CNS color naming scheme are not necessarily supported. When not supported, the color substituted is (always or sometimes) a shade of gray. Furthermore, though when you try to use it to color a graphic object in SAS/GRAPH, you are notified of the substitution in the SAS log, if you instead (unsuccessfully try to) use it, e.g., to produce colored text for the title of table in a web page, you are NOT notified. Since there is a mind-boggling array of theoretically possible color names in the CNS system, to prepare an automated exercise to test them all, and then document the results, would be a big effort.

Also, there is another list of 151 SAS color names, with their RGB codes, in the SAS Color Registry. Of these color names, only 147 are HTML color names. Those 147 colors were originally developed for the Unix X Window system, and were later adopted as HTML color names.

Four of the 151 colors in the SAS registry are SAS abbreviations for HTML Brown, HTML Green, HTML Orange, and HTML Purple. The registry also includes the four corresponding unabbreviated HTML color names. (Why there are abbreviations for these particular four colors, and only these four colors, might be known only to the developer who put them into the registry.)

As with the previously discussed SAS predefined color names, for these HTML color names, too, assume nothing. Make yourself a sample chart. The HTML names are not necessarily reasonable descriptions of the colors. E.g., NavajoWhite is not at all close to White.

Working with the SAS Color Registry

To get a listing of the registry colors in the SAS log, use this:

```
proc registry startat='HKEY_SYSTEM_ROOT\COLORNAMES' list;
run; quit;
```

To export the listing to a .txt file, use this:

```
proc registry startat='HKEY_SYSTEM_ROOT\COLORNAMES'
export='C:\YourFolderName\YourFileName.txt';
run; quit;
```

Using the .txt file created with the code above as a template, you could create your own color list and import it back into the SAS Color Registry with a different color list name. That would enable you to use SAS software with your own custom palette, with RGB assignments that you like (e.g., browser-safe ones if you like, or really anything), and with your own names for them (presumably ones that you regard as reliably descriptive).

10. Color Differs on Different Media: Do You See What I See?

My wife and I disagree on whether certain colors are green (what I see) or brown (what she sees). But there are more than mere differences in human visual perception. In addition to what can happen when using browser-unsafe colors on the web, and what are called "gamma differences" between PC, Mac, and Unix, there are other technology-related sources of variation. Here are some of them:

- CRT monitor color and LED flat panel color differ.
- On an LED panel, very light colors wash out to near-White.
- LED projector color differs from color on the presenter's PC or laptop that feeds the projector.
- CRT or LED color differs from printer color.
- Hardcopy color varies from printer to printer.

Among my experiences in color communication was to see an LED projector convert blue and red text on my PowerPoint slides into violet and orange. Fortunately, the presentation was not about communication-effective use of color. More disappointing was when a presentation that I was doing included my discussion above about the problem with SAS shades of pink, but was delivered using a projector that converted my shades of pink into shades of gray.

These experiences involved more than the color difference phenomena mentioned above. The LED projector is probably the riskiest color communication tool. Tuning a shared projector to suit one's own laptop is time-consuming, and might impair the usefulness of the projector for some other presenter.

Black, white, and gray shades do not suffer the above risks.

NOTE: You might have noticed that colors on a laptop vary as you vary the slope of screen. WHAT is the true color on a laptop? I never noticed this phenomenon on CRT monitors.

Guidance from Other Experts

Jan White on Color Communication

- If everybody screams, all you get is noise. The less color is used, the more effective it is.
- Color consistency provides recognition.
- Use color to sort and/or link information.
- Make large areas pale, small areas bright.
- Don't waste color on titles—for emphasis, use large or bold print instead.

Michael Turton on Color Communication

- Color works better with space around it.
- Color prioritizes information, whether it is meant to or not.

Aaron Marcus on Color Communication

- Use blue for large areas, not text or lines. Blue-sensitive color receptors are the least numerous in the retina's central focusing area.
- Use red or green in the center of the visual field. The edges of the retina are not very sensitive to these colors.

An Easy To Understand and Easy To Use Color System

It is easy to vary lightness with constant hue by using the HLS color system. When your target is hardcopy, HLS colors are an excellent choice, also providing easy tunability of transition in hue and "saturation". HLS color names are of the form H*hhhllss*. Here is how they work:

- *hhh* is the hexadecimal code for Hue
- *ll* is the hexadecimal code for Lightness (also called "Luminance")
- *ss* is the hexadecimal code for Saturation
- *hhh*, *ll*, *ss* ranges are 000-168, 00-FF, 00-FF
- *hhh* = 000 - 168 defines a "wheel of hues", 0 - 360 degrees
- ll = 00 (0%) always produces black, regardless of hue or saturation
- ll = FF (100%) always produces white, regardless of hue or saturation
- ss = FF (100%) always produces the fully saturated hue
- ss = 00 (0%) always produces a gray, regardless of selected hue
- llss = 80FF is what I call the "true color"

There are six/seven special hues (primary colors and their combinations) in the HLS color wheel.

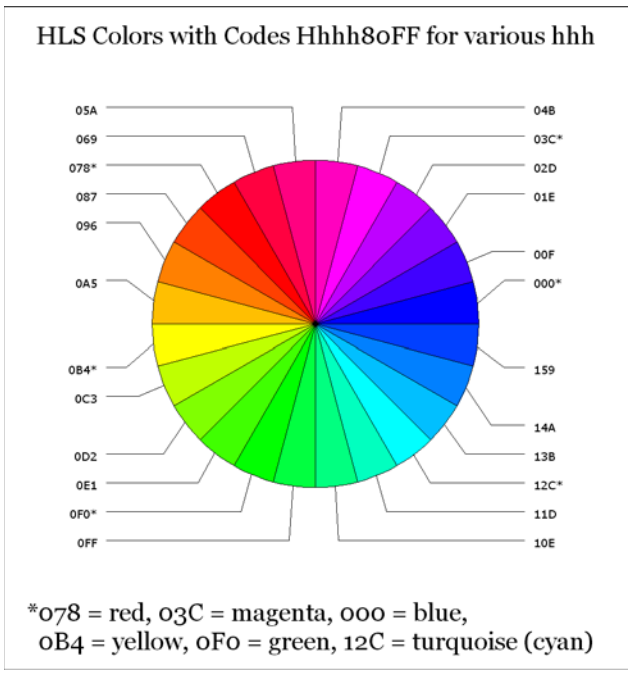
hhh	**color**	**position**
000	Blue	0 degrees
03C	Magenta	60 degrees
078	Red	120 degrees
0B4	Yellow	180 degrees
0F0	Green	240 degrees
12C	Cyan (Turquoise)	300 degrees
168	Blue	360 degrees

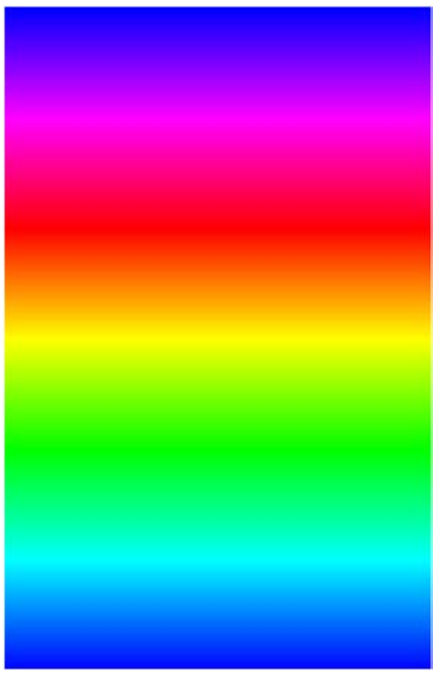
In this scheme, Violet lies between Blue and Magenta, Orange between Red and Yellow, Yellow-Green between Yellow and Green, etc. To get to these other colors, and to adjust their precise hue, you have to "turn the dial" between the successive relevant pairs of *hhh* values listed above. Below is some code to convert color wheel degrees into their hexadecimal codes for HLS hues, and to create color samples:

```
data _null_;
length HLScode $8.;
degrees = 45;
HLScode = 'H'||put(degrees,hex3.)||'80FF';
call symput('MyViolet',HLScode);
degrees = 150;
HLScode = 'H'||put(degrees,hex3.)||'80FF';
call symput('MyOrange',HLScode);
run;
goptions reset=all;
goptions device=PNG gsfname=anyname border;
goptions vpos=06 vsize=0.90 IN ymax=0.90 IN ypixels=270;
goptions hpos=34 hsize=3.25 IN xmax=3.25 IN xpixels=975;
filename anyname "YourDrive:\YourFile.png";
proc gslide;
title; footnote; note h=1 ' ';
note j=C               f='Courier New/Bold' h=1 c=H0000000  "C=&MyViolet"
     move=(+0.5,-1.25) f='Monotype Sorts'   h=4 c=&MyViolet '6E'X
     move=(+2,+1.25)   f='Courier New/Bold' h=1 c=H0000000  "C=&MyOrange"
     move=(+0.5,-1.25) f='Monotype Sorts'   h=4 c=&MyOrange '6E'X;
run; quit;
filename anyname clear;
```

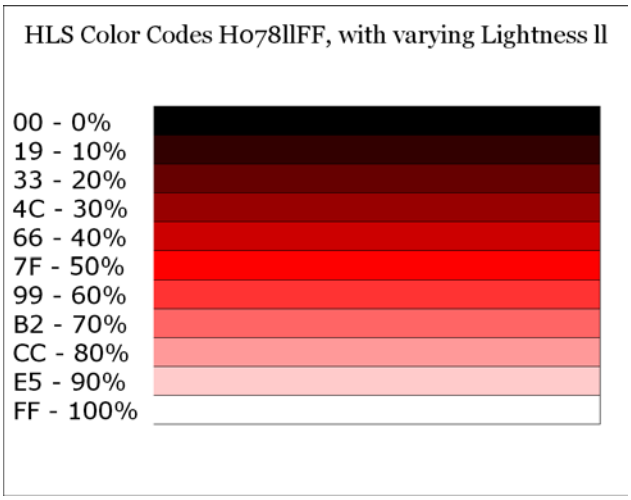
Here are the color samples for “My Violet” and “My Orange”:

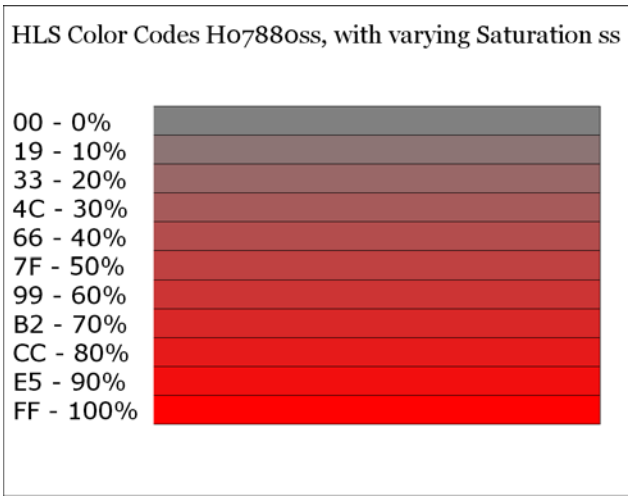
Here is what an HLS Color Wheel and a much denser, linearized version of it look like:





Varying the lightness or the saturation, while holding other parameters constant, is demonstrated below.





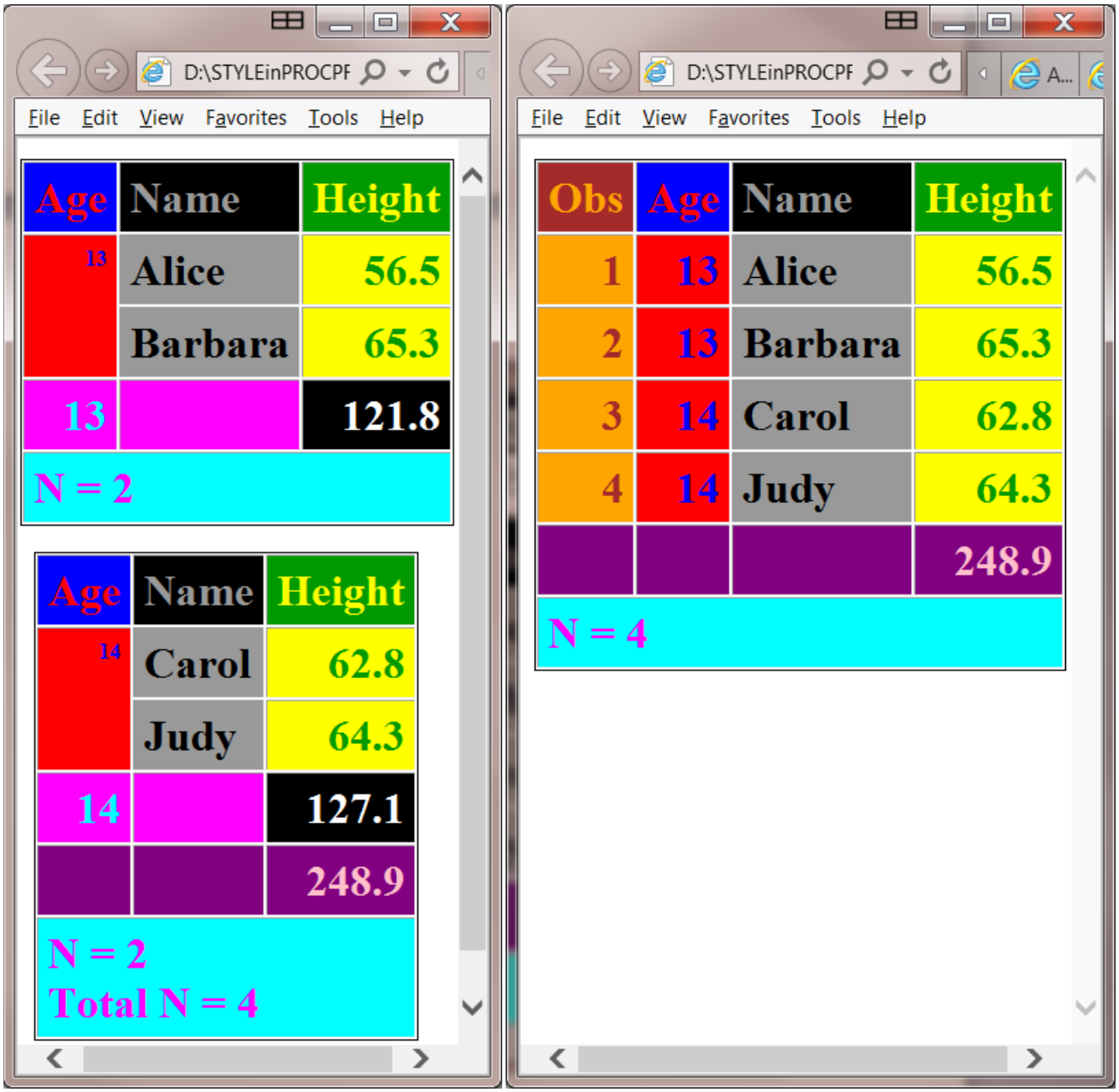
Tools for Generating Large Numbers of Sample Color Combinations or Sample Colors

In Appendix C is laborsaving code that can be used to:
(a) evaluate a large number of text (foreground) / background color combinations for readability; or
(b) create a large number of color samples.
See Figures 6-9 in Appendix B for examples generated with these tools.

Except for the macro that is used to generate all 216 browser-safe colors, these tools (despite the fact that the macro parameters used to specify colors have the suffix "RGBcolor") can actually be used with any other color that the SAS System recognizes: HLS colors, the SAS Predefined Color Names (such as TAN, CREAM, etc.), and any of the long HTML color names supported by SAS ODS.

The examples do use a custom ODS style preferred by the author. You could substitute any ODS style that you prefer, but your web page background should be white, so as to not affect the visual perception of the colors being evaluated/sampled.

Access To Color Excess:
All of the Places To Apply ODS STYLE Parameters in PROC PRINT



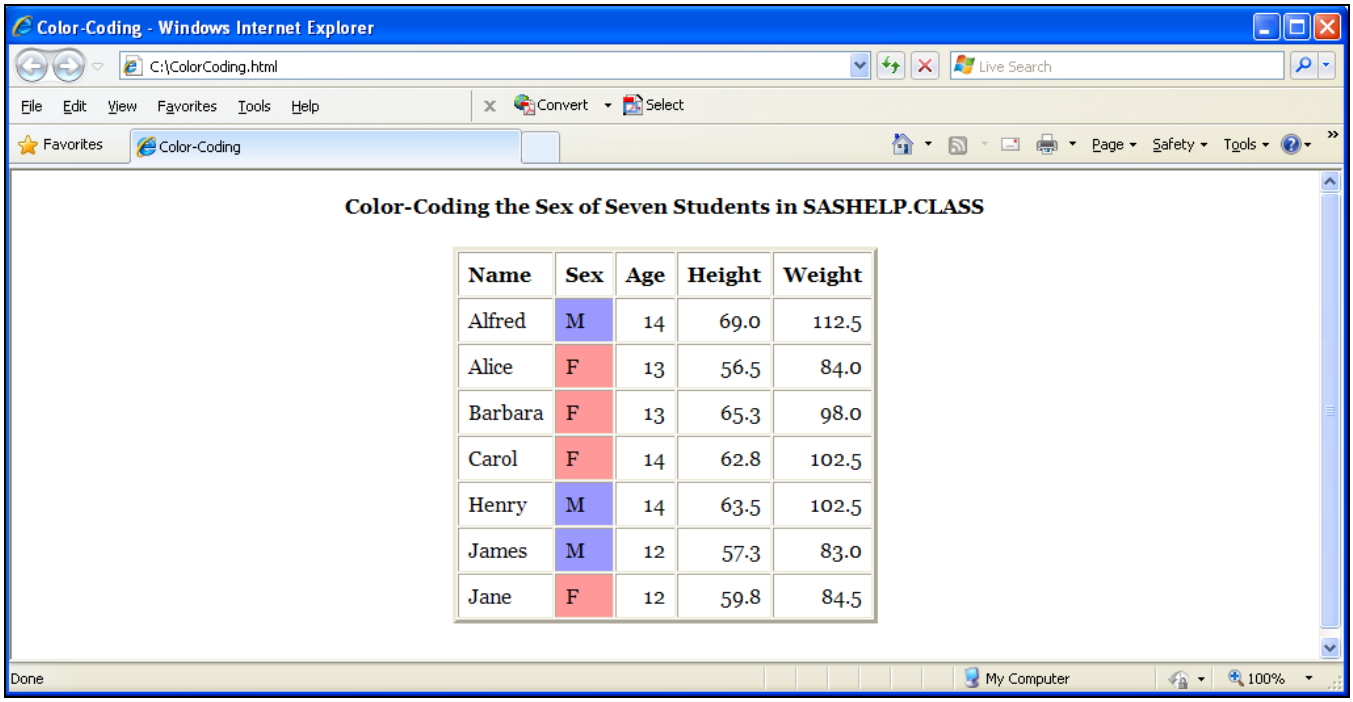
One can control the color and other display/format aspects of a table by creating a custom ODS style, but for maximum flexibility one can apply the controls directly inside the reporting procedure code. Listed below is the code used to create the above demonstration tables with PROC PRINT. Similar controls are available for PROC TABULATE and PROC REPORT.

```
proc sort data=sashelp.class out=ToPrint;
where name in ('Alice' 'Barbara' 'Carol' 'Judy');
by Age Name; run;

%let FontFormatting = %str(font_weight=Bold font_size=6);
```

```
ods noresults;
ods listing close;

ods html path='D:\' (url=none)
  body='STYLEinPROCPRINToutput_WithBYandIDvar.html'
  (title='All The Places To Apply ODS STYLE= BY and ID variable')
  style=styles.Minimal;
title;
proc print data=ToPrint label N
  style(header)  = [&FontFormatting]
  style(data)    = [&FontFormatting]
  style(bylabel) = [&FontFormatting background=magenta foreground=cyan]
  style(total)   = [&FontFormatting background=magenta foreground=cyan]
  style(grandtotal) = [&FontFormatting background=purple foreground=pink]
  style(N)       = [&FontFormatting background=cyan foreground=magenta just=left];
by age;
id age /
  style(header)  = [background=blue foreground=red]
  style(data)    = [background=red foreground=blue];
var Name /
  style(header)  = [background=black foreground=CX999999]
  style(data)    = [background=CX999999 foreground=black];
var Height;
sum Height /
  style(header)  = [background=CX009900 foreground=yellow]
  style(data)    = [background=yellow foreground=CX009900]
  style(total)   = [background=black foreground=white];
sumby age;
run;
ods html close;

ods html path='D:\' (url=none)
  body='STYLEinPROCPRINToutput_WithOBSvar.html'
  (title='All The Places To Apply ODS STYLE= If No BY and No ID variables')
  style=styles.Minimal;
title;
proc print data=ToPrint label N
  style(header) = [&FontFormatting]
  style(data)   = [&FontFormatting]
  style(obsheader) = [&FontFormatting background=brown foreground=orange]
  style(obs)    = [&FontFormatting background=orange foreground=brown]
  style(total)  = [&FontFormatting background=magenta foreground=cyan]
  style(grandtotal) = [&FontFormatting background=purple foreground=pink]
  style(N)      = [&FontFormatting background=cyan foreground=magenta just=left];
var age /
  style(header) = [background=blue foreground=red]
  style(data)   = [background=red foreground=blue];
var Name /
  style(header) = [background=black foreground=CX999999]
  style(data)   = [background=CX999999 foreground=black];
var Height;
sum Height /
  style(header) = [background=CX009900 foreground=yellow]
  style(data)   = [background=yellow foreground=CX009900]
  style(total)  = [background=black foreground=white];
run;
ods html close;

ods listing;
```

Color-Coding Your Data (NOT “Traffic-Lighting”)

As previously mentioned the regrettably popular fascination with traffic-lighting of data is non-communicative for color-blind viewers. Here the colors used for coding are light red and light blue for female and male, respectively.



Color-Coding the Sex of Seven Students in SASHELP.CLASS

Name	Sex	Age	Height	Weight
Alfred	M	14	69.0	112.5
Alice	F	13	56.5	84.0
Barbara	F	13	65.3	98.0
Carol	F	14	62.8	102.5
Henry	M	14	63.5	102.5
James	M	12	57.3	83.0
Jane	F	12	59.8	84.5

Here is the code used:

```
proc format;
value $SexCol
      'F'='CXFF9999'
      'M'='CX9999FF';
run;

ods noresults;
ods listing close;
ods html path='C:\' (url=none)
    body='ColorCoding.html'
    (title='Color-Coding')
style=styles.Minimal;
title font='Georgia/Bold'
      'Color-Coding the Sex of Seven Students in SASHELP.CLASS';
options obs=7;
proc print data=sashelp.class label noobs
     style(header) = [font_face='Georgia']
     style(data) = [font_face='Georgia'];
var Name;
var Sex / style(data) = [background=$SexCol.];
var Age Height Weight;
run;
options obs=max;
ods html close;
ods listing;
```

Conclusion

Color is something that we take for granted. However, without getting into details about the physiology of color perception, optical illusions due to color perception, and other arcane subtleties, this paper demonstrates that color selection and use requires care if you want to get beyond mere decoration and into effective communication, and has provided guidelines as well as color-use code examples and color evaluation tools.

References

1. Bessler, LeRoy (1995), "Communicate Effectively in Color with SAS/GRAPH Software", *Proceedings of the Twentieth Annual SAS Users Group Conference*, Cary, NC, USA: SAS Institute Inc., 1995. This was the first SAS users group conference paper on communication-effective use of color.

2. Bessler, LeRoy (2004), "Communication-Effective Use of Color for Web Pages, Graphs, Tables, Maps, Text, and Print", *Proceedings of the Twenty-Ninth Annual SAS Users Group Conference*, Cary, NC, USA: SAS Institute Inc., 2004. Find it on the web at http://www2.sas.com/proceedings/sugi29/176-29.pdf.

3. Bessler, LeRoy (2013), "Communication-Effective Data Visualization When Using SAS or Other Graphic Tools", *Proceedings of the Fall 2013 Wisconsin Illinois SAS Users Conference*, Chicago, IL, USA: Software User Services, Inc., 2013.

Author Information

Your questions, comments, and ideas about communicating with color are always welcome.

LeRoy Bessler PhD
Bessler Consulting and Research
Converting Complexity Into Clarity™
Strong Smart Systems™
1-262-512-1729
Le_Roy_Bessler@wi.rr.com

Dr. LeRoy Bessler has presented at software user conferences in the US, Canada, and Europe, on effective visual communication (using graphs, tables, web pages, or color), highly formatted Excel reporting from SAS, custom-developed tools to assist SAS server administrators, users, and managers, and Software-Intelligent Application Development methods to maximize Reliability, Reusability, Maintainability, Extendibility, and Flexibility. His SAS experience includes application development and supporting users, servers, software, and data.

Appendix A. It Once Was More Important to Use "Browser-Safe" or "Web-Safe" Colors

When and Why

At one time, many web users still had displays or video cards limited to 256 colors. Even when the display and the video card had a higher capability, the video card might have been set to display only 256 colors. When I was working as a consultant prior to May 2003, I did encounter sites where PC's were limited to 256 colors. Today, your users of older technology mobile devices might be limited to 256 colors. **This Appendix takes up this now diminishing problem and its circumvention.**
To check or change the setting of your video card on a Windows computer, click
Start > Settings > Control Panel > Display > Settings > Colors.
To deal with equipment diversity, web browsers determine the currently set limits of the display unit's video card, and, if needed, will remap unsupported colors. (Compare Figures 8 and 9 in Appendix B.)

Video displays produce colors as combinations of Red, Green, and Blue, the RGB color system. All web browsers agree on a universal common subset of 216 browser-safe RGB colors. They are RGB colors with names, in SAS, of the form CXrrggbb. The web-safe RGB colors restrict rr, gg, and bb to the six values 00, 33, 66, 99, CC, FF, which correspond to 0%, 20%, 40%, 60%, 80%, 100% of red, green, and blue. (216 = 6 X 6 X 6.)

If a web browser detects a color outside this set on a web page to be shown on a 256-color display, it remaps the color to a browser-safe one. Then, Web Designer Color does not equal Web Viewer Color. There are 16,777,216 RGB colors, but only 216 are browser-safe.

All of the SAS CNS Color Names (see above) and all of the HTML color names (see below) have RGB equivalents, but only seven of each are browser-safe. SAS GREEN is not browser-safe—even though Green is one of the three RGB primaries. The new SAS name, and the HTML name, for browser-safe green is "LIME". I agree that browser-safe green is perhaps not exactly what most people consider to be a "typical" green (but "typical" being vague, imprecise, and inherently subjective). Browser-safe color CX009900 can serve well as a typical green.

See Figure 2 in Appendix B for 81 samples of browser-safe colors. The basic colors are Red (CXFF0000), Yellow (CXFFFF00), Green (CX00FF00), Cyan or Turquoise (CX00FFFF), Blue (CX0000FF), Magenta (CXFF00FF), Black (CX000000), and White (CXFFFFFF). The upper chart shows the only way for RGB colors to vary in lightness with constant hue.

If you study the full set of 216 browser-safe colors in Figure 5, you may conclude, as I have, that from the browser-safe palette it is difficult to select subsets of "related" colors, other than those in Figure 2. For how to add gray to each of the browser-safe primaries and secondaries, see Figure 3. Another selection of small sets of related browser-safe colors is presented in Figure 4.

NOTE: This seeming limitation can actually be a benefit. A palette of "only" 216 colors does reduce the opportunity for needless agonizing about which colors to use. Presentation of computer-sourced information or charts does not have the same palette requirements as painting a portrait or a landscape.

How To See the Effect of Browser-Unsafe Colors (Compare Figures 8 and 9 in Appendix B)

You need a display unit and video card that can display more than 256 colors. As explained in the prior section, use the Control Panel to verify that your video card is set to display more than 256 colors.

Either by using the code in Appendix C, or with any other means that you like, create a web page that includes easily visible patches of SAS Predefined Colors BLUE, TAN, and CREAM. BLUE is one of the only seven web-safe SAS CNS color names.

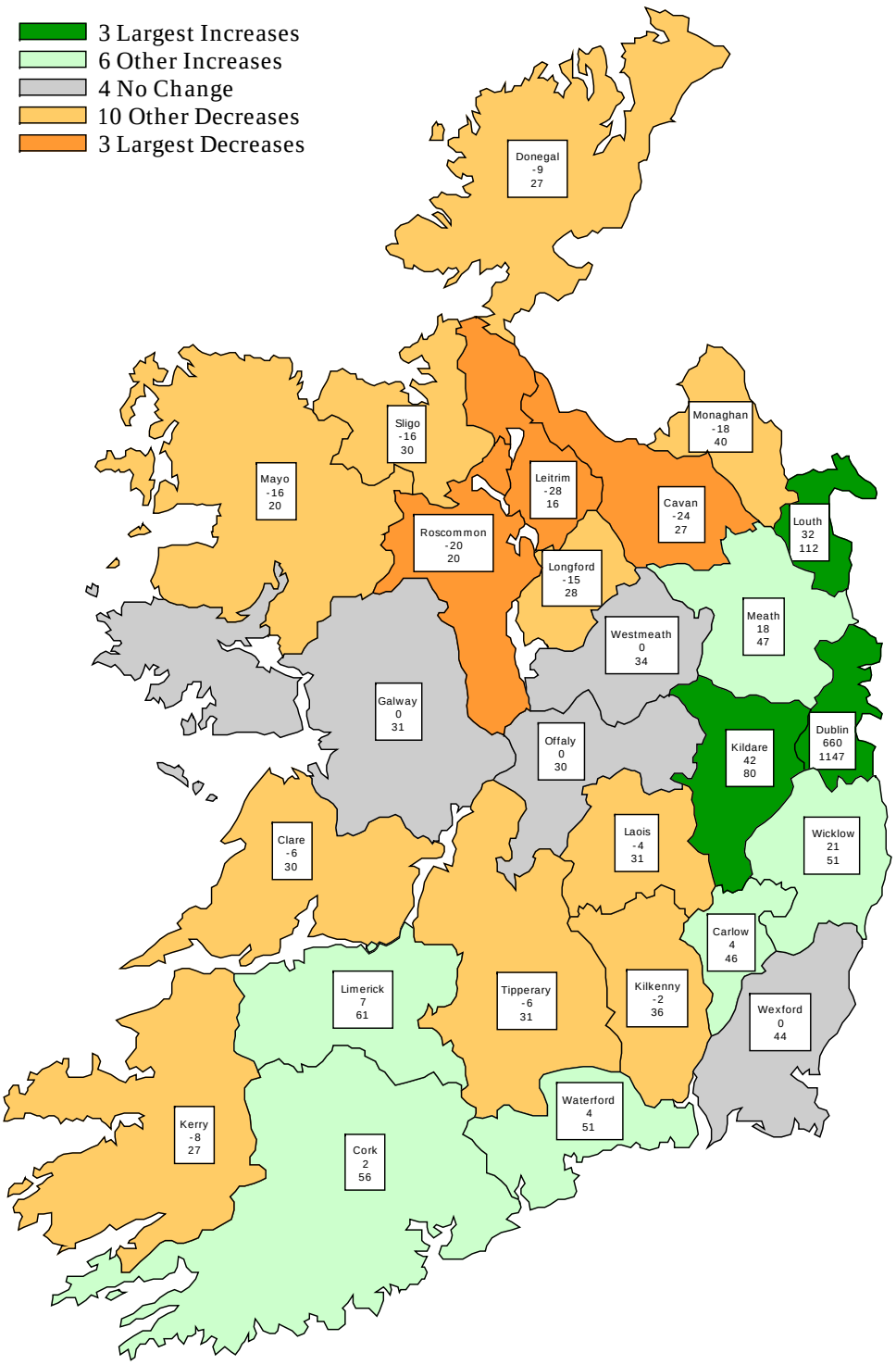
Open the web page with your web browser. The colors will look OK. Close your web browser. Use the Windows Control Panel to change your video card to display only 256 colors. Now re-open the web page with your web browser. It will detect the video card's color impoverishment. You will see that the browser has remapped TAN and CREAM, with the browser-safe color subset.

Be sure to reset your video card back to its normal setting.

Appendix B.

Figure 1. An Alternative to Traffic Lighting: augmented with light shades of the signal colors.

Population Density (per sq. km.) Change in the Counties of the Republic of Ireland, 1901-1996
A Highlighted "Spatial Population Table" of County Name, Density Change, & 1996 Density



Shown at the Eighteenth Annual SAS European Users Group International Conference, Dublin, 2000

Figure 2. Samples of Browser-Safe SAS/GRAPH Colors, with Their RGB Codes

Figure 3. Adding Gray to Browser-Safe Primaries and Secondaries

Figure 4. Some Other Sets of “Related” Browser-Safe Colors

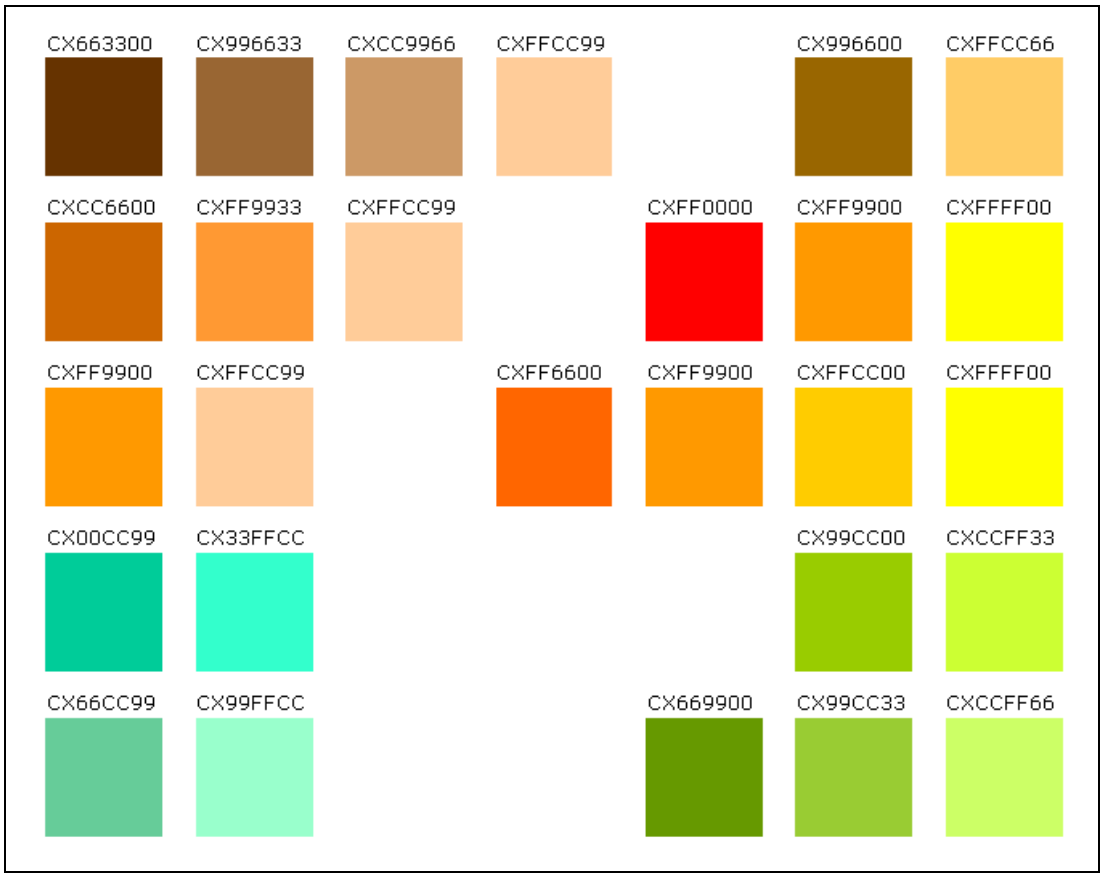


Figure 5. The 216 Browser-Safe Colors with Their RGB Codes

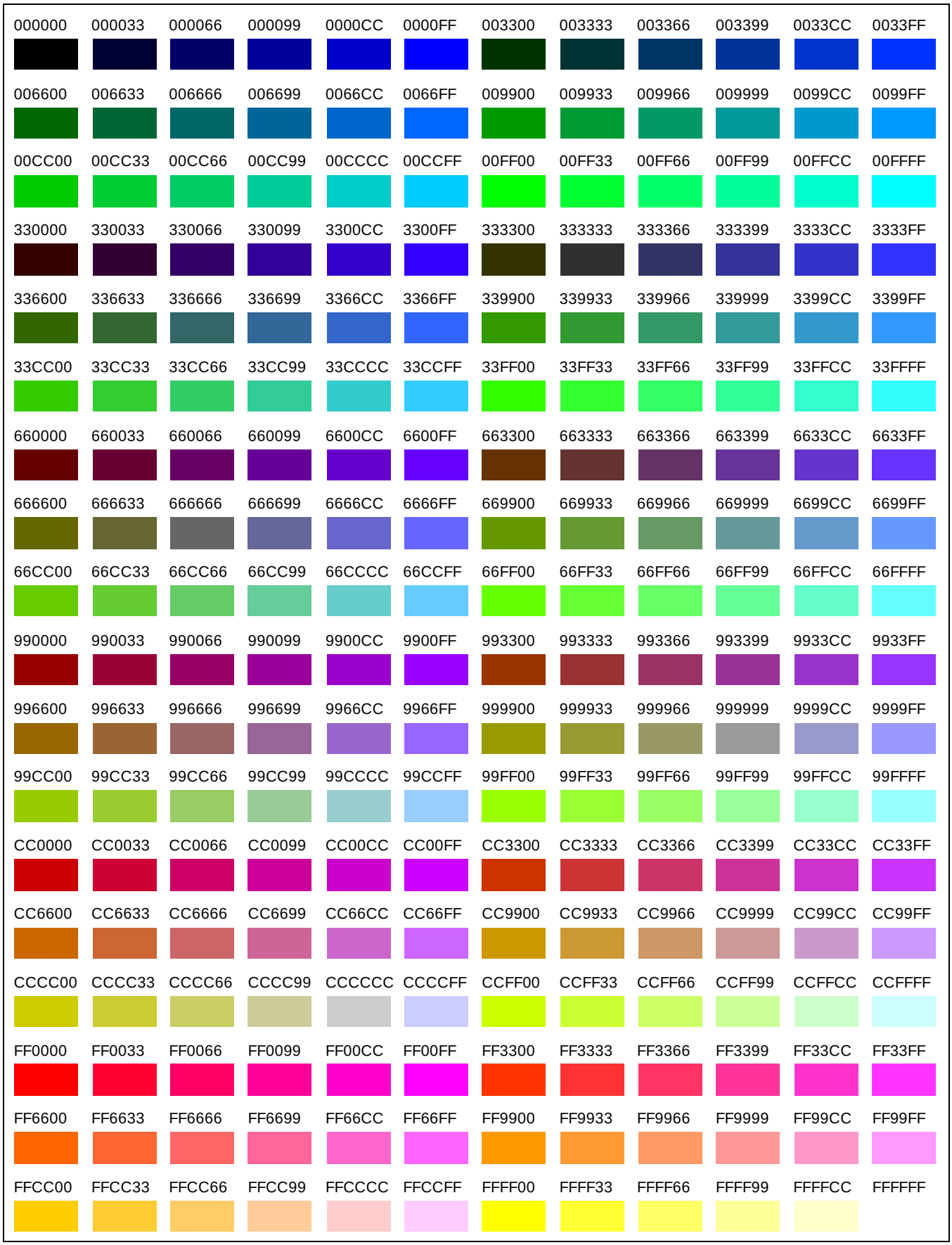


Figure 6. Bad Examples of Text-Background Color Combinations

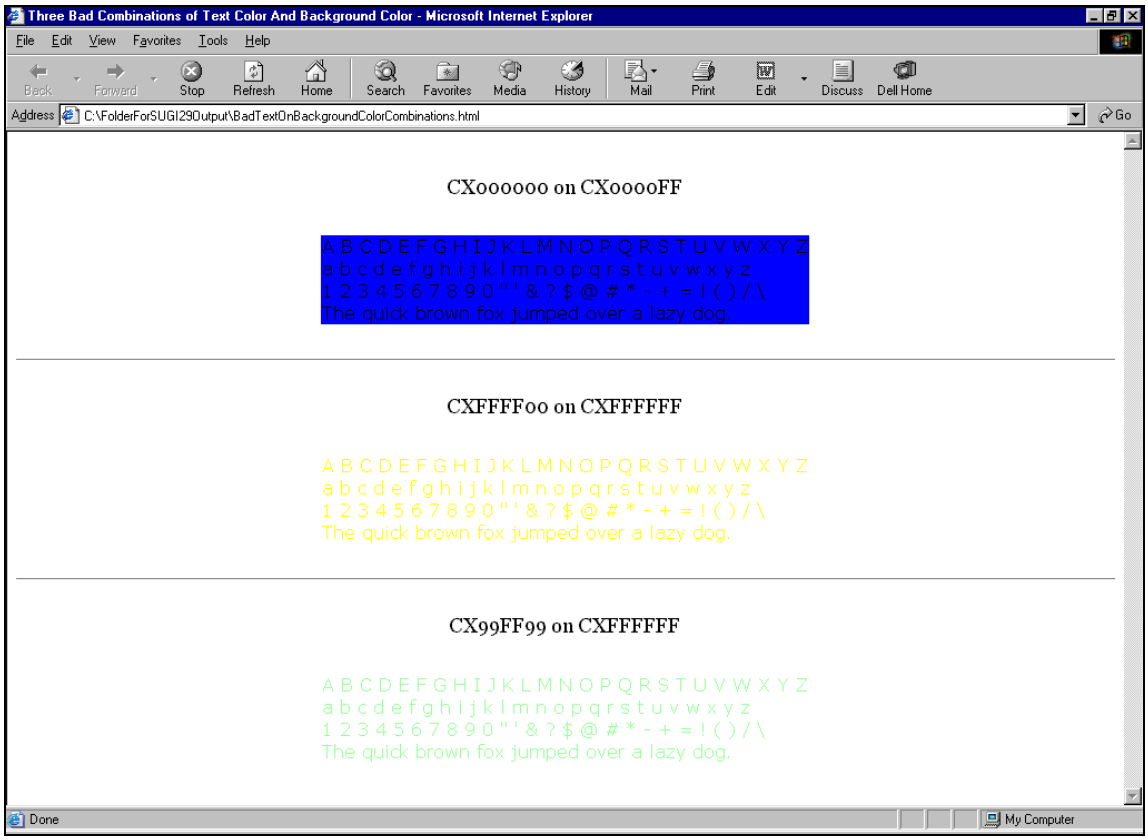


Figure 7. Good Examples of Text-Background Color Combinations

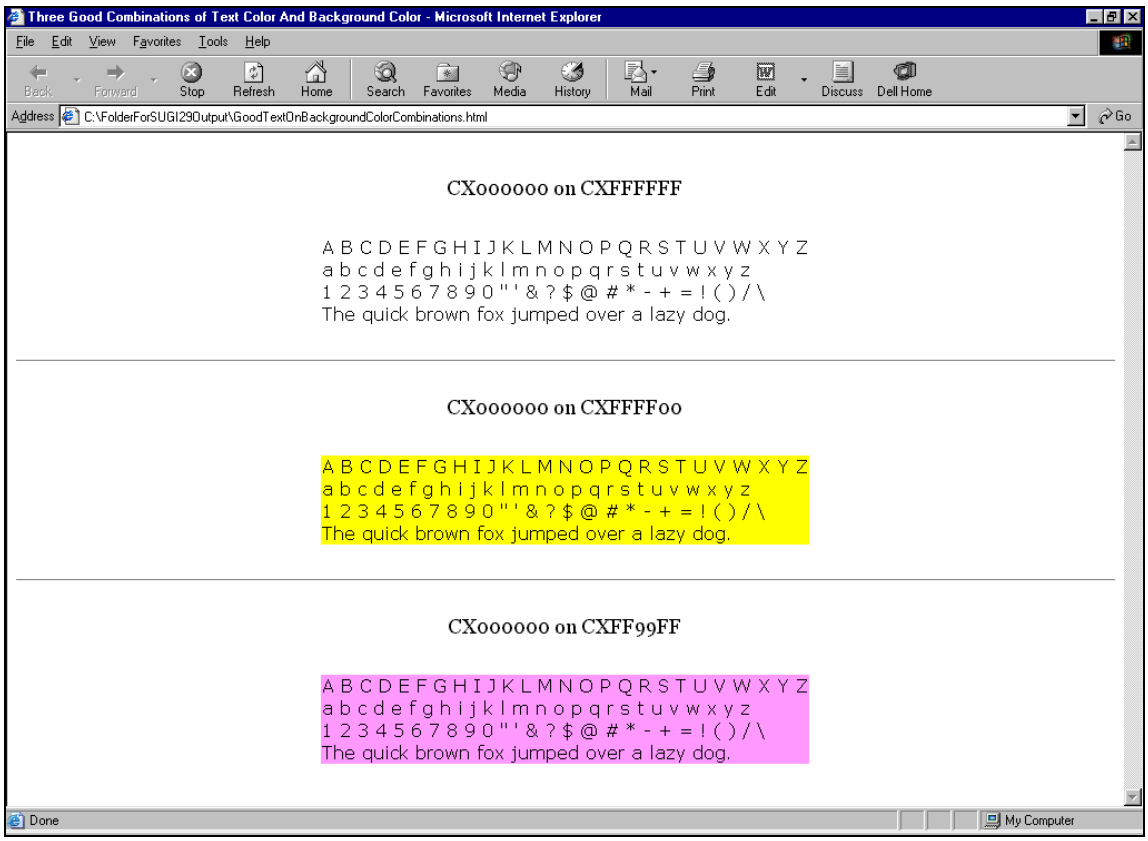


Figure 8. SAS Colors Blue, Tan, & Cream, with My Monitor Set to 32-bit Color Mode

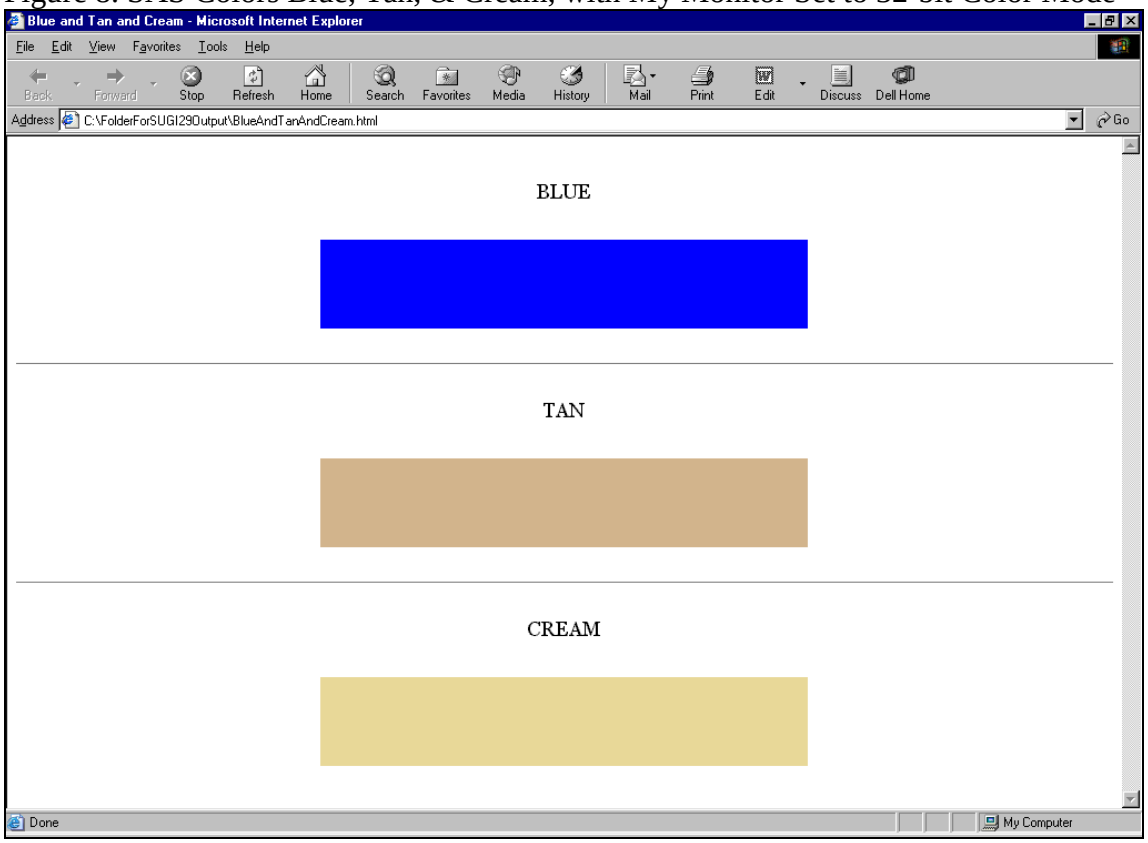
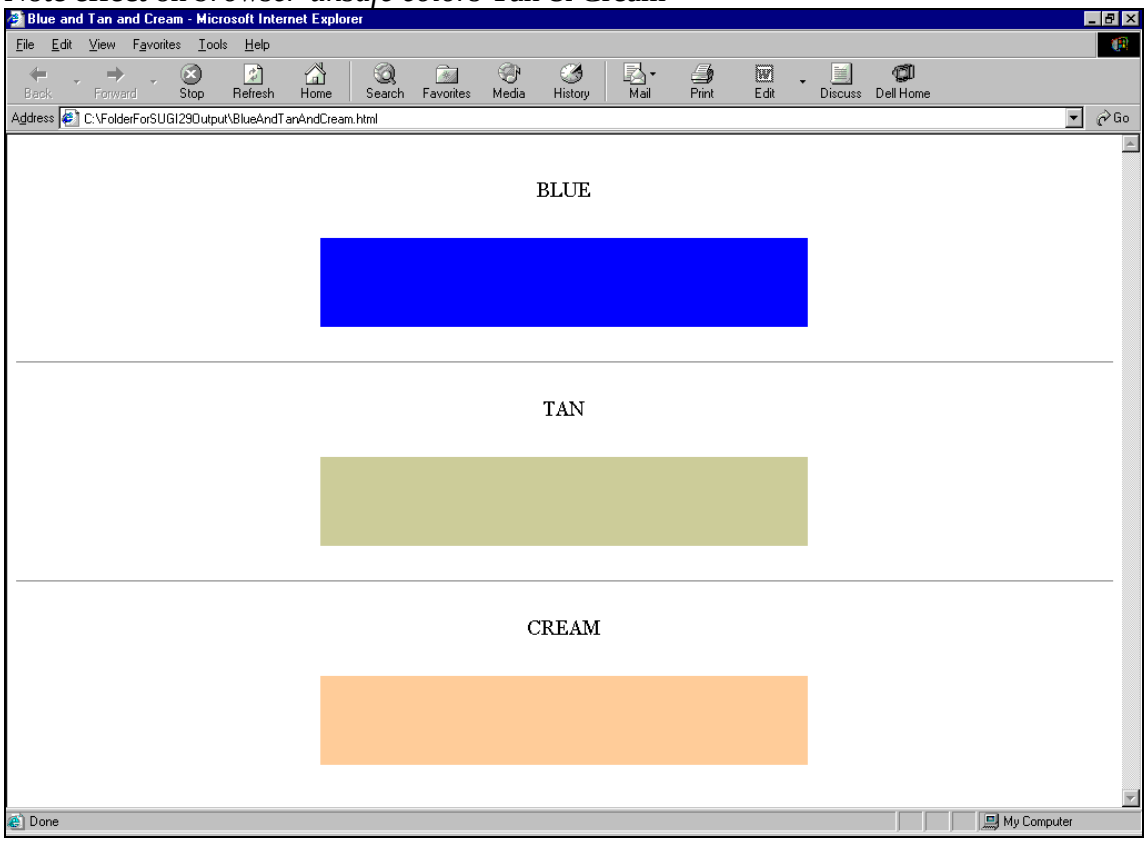


Figure 9. SAS Colors Blue, Tan, & Cream, with My Monitor Set to 256-Color Mode
Note effect on *browser-unsafe colors* Tan & Cream



Appendix C. Macro Tools and Sample Programs for Evaluating Colors and Combinations

Macro To Create a Custom ODS Style

NOTE: The macro parameters used to specify colors, despite the fact that they have the suffix "RGBcolor", can actually be assigned any color that the SAS System recognizes.

```
%macro CustomBaseStyleBuild(
StyleName=LeRBsugi29CustomStyle,
PROCoutputSeparators=NO, /* YES to put horizontal rule
  between successive PROC outputs in the same web page,
  but would have NO EFFECT if using NEWFILE = PROC. */
PROCoutputSepLineThickness=1, /* 2 is thicker,
                                 3 is Default */
WebPageBackgroundRGBcolor=CXFFFF99,
    /* CXFFFF99 is light Web-Safe yellow */
TitleFootnoteBackgroundRGBcolor=CXFFFF99,
TitleFootnoteBackgrdTransparency=NO, /* YES to let
  web page background show through */
TitleFootnoteRGBcolor=CX000000, /* Web-Safe black */
TitleFootnoteFont=Georgia,
TitleFootnoteSize=4,
TableBackgroundRGBcolor=CXFFFF99,
TableContentRGBcolor=CX000000, /* data and headings */
TableHeadingFont=Verdana,
TableHeadingSize=3,
TableDataFont=Verdana,
TableDataSize=3,
TableFrame=box, /* TableFrame=void to remove frame */
TableFrameRulesGridRGBcolor=CX9999FF,
                        /* light Web-Safe blue */
TableGrid=NO, /* YES for a grid between data cells */
TableSpacing=5); /* the SAS-shipped default is 7.
  This is space between cell data & cell boundaries. */

proc template;

edit styles.Default as styles.&StyleName;

  /* Create a modified style based
     on the ODS STYLES.DEFAULT.
     Anything not referenced or overridden here
     will be controlled by the ODS Default Style. */

style fonts /

  'TitleFont'   =
   ("&TitleFootnoteFont, Times New Roman, Times",
    &TitleFootnoteSize)
      /* "system" titles & footnotes */

  'HeadingFont' =
   ("&TableHeadingFont , Times New Roman, Times",
    &TableHeadingSize)
 /* column & row headings, and OBS, ID, & SUM values */

  'DataFont'    =
   ("&TableDataFont    , Arial          , Helvetica",
    &TableDataSize)
      /* table data. DataFont being added by LeRB.
         Not in ODS Default style. */

  'DocFont'     = ("Comic Sans MS, Courier",4);
      /* My default for unassigned fonts.
         Conspicuous font chosen to be obvious if used
         by ODS, so that a way can be found to assign a
         preferred font, instead of "my default". */

style color_list /

  'SafeRed'      = CXFF0000  /* Browser-Safe red */
  'SafeBlue'     = CX0000FF  /* Browser-Safe blue */
  'SafeMagenta'  = CXFF00FF  /* Browser-Safe magenta */

  'WebPageBackgroundColor'       =
   &WebPageBackgroundRGBcolor

  'TitleFootnoteBackgroundColor' =
   &TitleFootnoteBackgroundRGBcolor

  'TitleFootnoteColor'           =
   &TitleFootnoteRGBcolor

  'TableBackgroundColor'         =
   &TableBackgroundRGBcolor

  'TableContentColor'            =
   &TableContentRGBcolor

  'TableBoundariesColor'         =
   &TableFrameRulesGridRGBcolor;

style colors /
  'systitlefg'           =
   color_list('TitleFootnoteColor')
     /* "system" titles & footnotes */
  'systitlebg'           =
   color_list('TitleFootnoteBackgroundColor')
     /* background for "system" title/footnote areas
        However, if transparency is enabled,
        then this color is ignored. */
  'headerfg'             =
   color_list('TableContentColor')
     /* override fgA2, which is the
        ODS default for table row & column labels */
  'headerbg'             =
   color_list('TableBackgroundColor')
     /* background for table row & column labels */
  'datafg'               =
   color_list('TableContentColor')
     /* table cell data */
  'databg'               =
   color_list('TableBackgroundColor')
     /* background for table cell data */
  'docfg'                =
   color_list('SafeMagenta')
     /* My default for unassigned foreground colors.
        Conspicuous color chosen to be obvious if used
        by ODS, so that a way can be found to assign a
        preferred color, instead of "my default". */
  'docbg'                =
   color_list('WebPageBackgroundColor')
     /* background for web page and ??? */
  'tableborder'          =
   color_list('TableBoundariesColor')
     /* actually, for table frame AND table rules */
  'TableGrid'            =
   color_list('TableBoundariesColor')
     /* (TableGrid is an LeRB replacement for where
         tablebg is used by ODS default style)
         Color of table grid when cellspacing > 0 AND
         "style Table from Output"
         does not assign background. */
  'link2'                =
   color_list('SafeBlue') /* std for unvisited links */
  'link1'                =
   color_list('SafeRed'); /* std for visited links   */

style SysTitleAndFooterContainer from Container /
  cellpadding = 0 /* compact the title/footnote area */
  cellspacing = 0 /* no grid for title/footnote area */
%if %upcase(&TitleFootnoteBackgrdTransparency) = YES
%then %do;
  background = _undef_;
style systemtitle  / background = _undef_;
style systemfooter / background = _undef_;
     /* Three instances of background = _undef_ above
        make the title and footnote areas transparent.
        They let the web page background show through.
        When this option is selected,
        the systitlebg color is actually ignored. */
%end;
%else %do;
 ;   /* needed to end this STYLE statement */
%end;

style Output from Container /
     /* these statements control
        table grid and table border */
  rules = NONE /* NONE to override rules=GROUPS,
    preventing double line between table labels & data.
    ALL would create fixed-width thin line
    around all interior cells and
    at inner edges of all perimeter cells */
%if %upcase(&TableGrid) eq NO
```

```
%then %do;
  frame = &TableFrame /* BOX for on, VOID for off */
  cellspacing = 0 /* override 1. Space between cells.
    Also, space between outer cells and any frame. */
%end;
%else %do;
  frame = VOID /* but keeping default cellspacing=1 */
%end;
  cellpadding = &TableSpacing /* override default 7 */
  background = colors('TableGrid')
     /* color of grid (LeRB replacement for tablebg),
        if cellspacing > 0 AND
        "style Table from Output" does not override it.
   This is NOT the table background on the web page. */
  bordercolor = colors('tableborder')
     /* color of table frame and table rules */
  borderwidth = 1;
     /* thickness of table frame, same as default */
  /* NOTE: bordercolor affects more table parts
     than does borderwidth. There is no apparent
     way to thicken the rules, when present. */

style Data from Cell / font = fonts('DataFont');
     /* Added to override default use of DocFont */

%if %upcase(&PROCoutputSeparators) = NO %then %do;
style Body / pagebreakhtml = _undef_;
  /* suppress rule between successive PROC outputs */
%end;
%else %do;
style html / 'ThinLineAfterSpace' =
  " <hr size=&PROCoutputSepLineThickness>";
style Body / pagebreakhtml =
  html('ThinLineAfterSpace');
  /* one space and a thin line
     between successive PROC outputs */
%end;

end; run; quit;

%mend CustomBaseStyleBuild;
```

Creating the Custom ODS Style

```
%CustomBaseStyleBuild(
       StyleName=LeRBsugi29ColorDemo,
       TableHeadingSize=1,
       TableFrame=void,
       TableSpacing=1,
       WebPageBackgroundRGBcolor=CXFFFFFF,
                              /* Web-Safe white */
       TableBackgroundRGBcolor=CXFFFFFF,
       TitleFootnoteBackgrdTransparency=YES,
       PROCoutputSeparators=YES)
run;
```

Data for Color Text Displays

```
data FontCharacters; label FontCharacters='00'X;
infile cards; input @1 FontCharacters $51.;
cards;
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
a b c d e f g h i j k l m n o p q r s t u v w x y z
1 2 3 4 5 6 7 8 9 0 " ' & ? $ @ # * - + = ! ( ) / \
The quick brown fox jumped over a lazy dog.
;
run;
```

Macro to Display Text on a Background or to Create a Solid Color Sample

```
%macro ColorTable(TextColor=,BackgroundColor=);
title1 "&TextColor"
%if %upcase(&BackgroundColor) ne %upcase(&TextColor)
%then %do;
       " on &BackgroundColor"
%end;
                              ;
proc print data=FontCharacters noobs label
     style(data)=[foreground=&TextColor
                  background=&BackgroundColor];
run;
%mend  ColorTable;
```

Creating Text-Background Color Combinations and Solid Color Samples (Figures 6, 7, 8, & 9)

```
ods listing close; ods noresults; goptions reset=all;
ods html
   path    = "C:\FolderForSUGI29Output" (url=none)
   body    = "BadTextOnBackgroundColorCombinations.html"
   (title="A Few Bad Combinations of Text Color And
Background Color")
   style   = Styles.LeRBsugi29ColorDemo
   newfile = NONE; * one continuous scrollable
                     web page body file *;
%ColorTable(TextColor      =CX000000,
            BackgroundColor=CX0000FF)
run;
%ColorTable(TextColor      =CXFFFF00,
            BackgroundColor=CXFFFFFF)
run;
%ColorTable(TextColor      =CX99FF99,
            BackgroundColor=CXFFFFFF)
run;
ods html
   path    = "C:\FolderForSUGI29Output" (url=none)
   body  = "GoodTextOnBackgroundColorCombinations.html"
   (title="A Few Good Combinations of Text Color And
Background Color")
   style   = Styles.LeRBsugi29ColorDemo newfile = NONE;
%ColorTable(TextColor      =CX000000,
            BackgroundColor=CXFFFFFF)
run;
%ColorTable(TextColor      =CX000000,
            BackgroundColor=CXFFFF00)
run;
%ColorTable(TextColor      =CX000000,
            BackgroundColor=CXFF99FF)
run;
ods html
   path    = "C:\FolderForSUGI29Output" (url=none)
   body    = "BlueAndTanAndCream.html"
   (title="Blue and Tan and Cream")
   style   = Styles.LeRBsugi29ColorDemo newfile = NONE;
%ColorTable(TextColor      =BLUE,
            BackgroundColor=BLUE)
run;
%ColorTable(TextColor      =TAN,
            BackgroundColor=TAN)
run;
%ColorTable(TextColor      =CREAM,
            BackgroundColor=CREAM)
run;
ods html close; ods listing;
```

Macro and Program for Scrollable Display of Every Browser-Safe Color

```
%macro AllBrowserSafeColors;
%do RRdecimal = 0 %to 255 %by 51;
  %do GGdecimal = 0 %to 255 %by 51;
    %do BBdecimal = 0 %to 255 %by 51;
data _null_;
DecimalCode = &RRdecimal;
call symput('RR',put(DecimalCode,hex2.));
DecimalCode = &GGdecimal;
call symput('GG',put(DecimalCode,hex2.));
DecimalCode = &BBdecimal;
call symput('BB',put(DecimalCode,hex2.));
run;
%ColorTable(TextColor      =CX&RR&GG&BB,
            BackgroundColor=CX&RR&GG&BB)
run;
    %end;
  %end;
%end;
%mend  AllBrowserSafeColors;
ods listing close; ods noresults; goptions reset=all;
ods html
   path    = "C:\FolderForSUGI29Output" (url=none)
   body    =
   "AllOfTheBrowserSafeColors.html" (title=" . . . ")
   style   = Styles.LeRBsugi29ColorDemo newfile = NONE;
%AllBrowserSafeColors
run;
ods html close; ods listing;
```