ABSTRACT
You can create and deploy informative and influencing graphs and tables for the web, and link them to each other, with SAS/GRAPH®, SAS®, and the Output Delivery System (ODS), even without knowledge of HTML.

There are a variety of web deployment and linking options and tools, for which trade-offs between advantages and disadvantages need to be understood. And there are design concerns unique to presentation on the web. This paper covers these matters, and the poster will include a live demonstration of working examples.

You will learn: how to link and/or provide drill-down for trend lines, bar charts, pie charts, maps, text slides, and tables via clickable points, areas, legends, labels, images, and navigational hot spots; how to partition a web page to display in a single frame both a table of contents and any one of the graphs or tables listed therein; how to use the SAS/GRAPH WEBFRAME device driver; how to provide "flyover" pop-up labels as an alternative to permanent annotation; and how to use animation.

You will also learn how to use colors, fonts, PROC TEMPLATE and ODS styles, and the options and features of SAS and SAS/GRAPH, to create elegant and effective web pages.

This paper can help you to deliver communication-effective design, and to deploy graphs and tables on the web in a manner best suited to your needs and situation. It assumes at least novice knowledge of SAS/GRAPH, uses SAS Release 8.2, and is not intended to be specific to an operating system (any system dependencies or differences known to the authors will be discussed).

INTRODUCTION
The aim of this paper is to show you as much as possible about important SAS software capabilities for publishing graphs, tables, and slides on the Web.

Design-guided techniques have been used to override SAS defaults, to show drillable and linked graphs and tables in a web-page package in a communication-effective way. Whether it is in the body of the paper, or in comments in code, or in code itself, there are: (a) best practices for dealing with titles, footnotes, legends, symbols, axes, tick marks, labels, colors, fonts, dimension, backgrounds, drill-down, crosslinks, navigational hot spots; and (b) technical advice for using ODS, SAS, and SAS/GRAPH.

The objective is the creation of packages of web pages that are:
- likely to look as intended—and as seen during development by the original creator—with any other viewer’s web browser;
- easily and quickly navigated; and
- easy to read and interpret.

Strengths and limitations are identified for these facilities:
- WEBFRAME driver
- Table of Contents
- GIF driver
- Output Delivery System

Suggestions for their most effective use are provided below.

DESIGN AND IMPLEMENTATION
The particular operating system used was Windows 98, but no platform dependencies were intended.

The linked and drillable graphs, tables, and slides created can be viewed with either Internet Explorer 5.0 or Netscape Navigator 4.0 on a monitor with a diagonal as small as 14 inches (35.6 centimeters) and (except for the CrossLink Welcome page) a resolution as low as 800x600, which is the commonest resolution setting for people browsing the web.

The graph images have been sized to leave enough space for the browser’s frame, and for any links that are required outside the graphic area, without causing a requirement for scrolling. Only the GIF, GIFANIM, and WEBFRAME drivers are used to create graphs. Use of Java and ActiveX is avoided in order to deliver web pages that can be easily and readily browsed without requiring any extra features or facilities that might not be available for the viewer’s web browser. Furthermore, well-designed graphs for data presentation should require no viewer/recipient modification for successful interpretation. Delivery of complex, multi-dimensional data, which may require the recipient to attempt to look at it in different ways, is not the commonest data presentation need.

Browser-Safe Colors have been used. There are only 216 browser-safe colors. Despite its huge range of possible color choices, SAS/GRAPH cannot work with a palette greater than 256 colors at any one time. It is not an unreasonable limitation of design/construction freedom to commit to using only colors that any browser can be guaranteed to display as you intended.

Browser-Safe colors are RGB colors (mixtures of red, green, and blue) with SAS names of the form CRXRRGBB, where RR, GG, and BB are restricted to the six possible hexadecimal values 00, 33, 66, 99, CC, and FF. This palette yields 6x6x6=216 colors. All web browsers are guaranteed to be able to display these colors. It is a fact that many, if not most, people browsing the web have their displays set to view a maximum of 256 colors, regardless of the higher capabilities of their monitors and video cards. If non-browser-safe colors are used, such web page viewers will not see the same colors as the web page creator.

Combinations of background and foreground colors that maximize readability are used as much as possible. Avoided are those combinations that severely impair readability, such as black text on a dark background, or yellow text or lines on a white background.

TrueType fonts have been used. On a Unix machine, we would use the quasi-equivalent Unix system fonts. Both these choices are industry standards, which will be well rendered and familiar to a web page audience—unlike SAS/GRAPH-specific “software fonts”. Moreover, software fonts require extra processing for creation, and presumably yield larger web page files to transmit and display. Whenever possible, the TrueType fonts used have been those developed by Matthew Carter for Microsoft. He designed his Georgia (serif) and Verdana (sans serif) fonts expressly for readability on screen and on the web.

The three-dimensional feature is not used to render images that do not show the relationship between three variables. It is a fact that, despite their popularity, 3D pie charts are deceptive. 3D bar charts can be more difficult to interpret, and 3D maps sometimes have the response for one geographic area hiding the response for another.

The presentation of the web pages is content-focused and message-focused. For instance, the web page is limited to one graph or table, so that the viewer need not scroll up and down to find the image of interest. By default, the more common HY- processing, done for related images, usually spills all of them out into a web page that requires scrolling. Multi-image scrollable web pages have two problems. First, the individual images are unable to capture focused attention of the viewer, except on a very small screen. Second, there is a situation that can cause viewer confusion. Suppose the viewer has been routed to one of the images via a hyperlink. If that image is at the bottom of the scrollable page, but the vertical space on the screen is large...
enough to display two successive images, then the viewer will naturally look at the image at the top of the page and think that something is wrong.

For presentation of shares of the whole when there would be too many pie chart slices to be readable, a customized ranked horizontal bar chart is provided, designed to also show shares of the whole, not just the usual absolute response measures. The web pages produced by the programs in this paper deliver images that are as sparse as possible, but which deliver all the required information. This focuses viewer attention on the meaning and message in the data, and may decrease web page transmit and display times. The defaults for bar charts and trend plots are rooted in the hand-drawn, graph-paper- and pen-based tradition of axis lines, tick marks, and tick mark value labeling. However, the design and programs in this paper deliver web pages and images that tell the viewer only what, and everything that, she/he really needs to know, either statically, or dynamically upon request.

For trend lines, there are permanent labels displayed for the start and end values, but the intermediate values are available via web-enabled “flyover” pop-up labels, which are transient. The examples include a few elegant customized uses of the SAS/GRAPH mapping capabilities and mapping support data sets.

Use of OPTOPTIONS NOGTITLE and/or NOGFOOTNOTE has been avoided. With those overrides, it is possible to specify links in the SAS/GRAPH TITLE and FOOTNOTE statements, but have their display diverted to the “ODS area” outside the graph area. However, the vertical space that the title(s) and/or footnote(s) would require if displayed in the graph area is reserved, even though the links are displayed outside the graph area. The result is that the graph is artificially compressed. The empty white space may look strange, and the compression can make it difficult to read the graph, especially if an annotated map. Fortunately, circumvention is possible. You can execute an otherwise unnecessary PROC PRINT after the SAS/GRAPH PROC, but within the same HTML body file. The desired links are supplied via a TITLE statement(s) for the PROC PRINT. The data set that it “prints” contains one observation with one variable, which is a blank. The custom ODS style built for this PROC PRINT is defined to display everything except the links in white, which is invisible on the white background that is used. The design assures that the links in the title fit in the web browser window without any requirement for scrolling.

A macro-language solution has been chosen to handle groups of graphs that differ only by the value of a classification variable, instead of using ODS HTML ANCHOR= and BY processing. This solution, at least as coded, automatically assures that each graph or table is a separate web page, the design benefits of which have been explained above. The macro-based solution may be somewhat difficult for the novice to initially understand, but it is a powerful and reliable tool. Once its use is mastered, it eliminates error-vulnerable situation-specific hand coding.

The programs (that use ODS) supply a custom frame title in the web browser banner line at the top of the screen. For the real tables output by PROC PRINT (which could be any other PROC-generated tables as well), the custom ODS style built with PROC TEMPLATE makes the background for all the data cells and row and column labels white. Also, it makes invisible all the grid lines that ODS “adds” to the table if you use STYLES.DEFAULT. There may be some situation in which these extras add communication value. If so, PROC TEMPLATE can be used to build a style to suit your design preferences. In The CrossLink Example, by default, there would be a horizontal rule between the graph output and the PROC PRINT that is used to provide the non-drill-down links (i.e., “crosslinks”) to other graphs or tables. The unneeded pseudo-pagebreak has been suppressed within the body file, using the custom style built with PROC TEMPLATE.

There are several customizations specifically for the Table of Contents example. A custom title has been defined to replace the default “Table of Contents”, and the width available to the Table of Contents has been adjusted to prevent line breaks in the entries. For the Table of Contents entries, the PROCLABEL line has been blanked out, since it would be a no-added-value extra, and the second line of each Table of Contents entry with the not-useful default link label text supplied by ODS (such as “plot of y*x” or “print of lib.data”) has been replaced. The custom link labels are defined with contents in table-generating PROCs and with default in graph-generating PROCs. The bullets that by default are to the left of the links, and to the left of the blanked-out PROC labels, have been suppressed. When the PROCLABEL line is blank, it still occupies useless white space, which causes a requirement for scrolling if the Table of Contents is long. That white space has been eliminated.

WAYS TO WEB PUBLISH GRAPHS AND TABLES
Following sections present three ways to produce a communication-effective web application:

- the SAS/GRAPH WEBFRAME driver;
- the Output Delivery System with Table of Contents and the GIF device driver; and
- the authors’ custom alternative, called “The CrossLink Example”

These methods are discussed starting with the simplest. Illustrations and code are presented for each example.

WEBFRAME DEVICE DRIVER
This method is the easiest to implement, but applies only to graphs. The web frame provides a column of thumbnail images with clickable 8-character labels/links at the left side of the page. The serious limitation of this deployment is that it does not allow drill-down links or any other hyperlinks except from the thumbnail index column. Also, on a small screen, the thumbnails do occupy space, which may impair the viewability of your real images by requiring you to make them small enough to fit in the restricted real viewing area.

The WebFrame Example

```sas
%let PATH = c:\YourFolder\YourSubFolder1; /* Output goes into the above named folder. After running this program, go there, and click on the index.html file */
libname DB 'c:\YourFolderForData'; /* empty the graphics catalog to avoid entry-name conflicts */
proc ggrepplot igout=work.gseg nofs; delete all; run; quit;
filename webfrout "GPATH";
goptions device=webframe xpixels=540 ypixels=405 nodisplay
gsname=webfrout gsfilemode=replace;
/* Now, insert all the code to invoke various SAS/GRAPH PROCs, and use name options here adequate. It will appear as the label of the thumbnail image. Here is just one example. */
title h=14 pt f='Georgia'
"Map of Three States and Chicago-Gary-Kenosha";
```

Illustrations and code are presented for each example.

WEBFRAME DEVICE DRIVER
This method is the easiest to implement, but applies only to graphs. The web frame provides a column of thumbnail images with clickable 8-character labels/links at the left side of the page. The serious limitation of this deployment is that it does not allow drill-down links or any other hyperlinks except from the thumbnail index column. Also, on a small screen, the thumbnails do occupy space, which may impair the viewability of your real images by requiring you to make them small enough to fit in the restricted real viewing area.

The WebFrame Example

```sas
%let PATH = c:\YourFolder\YourSubFolder1; /* Output goes into the above named folder. After running this program, go there, and click on the index.html file */
libname DB 'c:\YourFolderForData'; /* empty the graphics catalog to avoid entry-name conflicts */
proc ggrepplot igout=work.gseg nofs; delete all; run; quit;
filename webfrout "GPATH";
goptions device=webframe xpixels=540 ypixels=405 nodisplay
gsname=webfrout gsfilemode=replace;
/* Now, insert all the code to invoke various SAS/GRAPH PROCs, and use name options here adequate. It will appear as the label of the thumbnail image. Here is just one example. */
title h=14 pt f='Georgia'
"Map of Three States and Chicago-Gary-Kenosha";
```

Illustrations and code are presented for each example.

WEBFRAME DEVICE DRIVER
This method is the easiest to implement, but applies only to graphs. The web frame provides a column of thumbnail images with clickable 8-character labels/links at the left side of the page. The serious limitation of this deployment is that it does not allow drill-down links or any other hyperlinks except from the thumbnail index column. Also, on a small screen, the thumbnails do occupy space, which may impair the viewability of your real images by requiring you to make them small enough to fit in the restricted real viewing area.

The WebFrame Example

```sas
%let PATH = c:\YourFolder\YourSubFolder1; /* Output goes into the above named folder. After running this program, go there, and click on the index.html file */
libname DB 'c:\YourFolderForData'; /* empty the graphics catalog to avoid entry-name conflicts */
proc ggrepplot igout=work.gseg nofs; delete all; run; quit;
filename webfrout "GPATH";
goptions device=webframe xpixels=540 ypixels=405 nodisplay
gsname=webfrout gsfilemode=replace;
/* Now, insert all the code to invoke various SAS/GRAPH PROCs, and use name options here adequate. It will appear as the label of the thumbnail image. Here is just one example. */
title h=14 pt f='Georgia'
"Map of Three States and Chicago-Gary-Kenosha";
```

Illustrations and code are presented for each example.
ODS WITH TABLE OF CONTENTS AND GIF DEVICE DRIVER
This is easy to implement. The program shows how to customize the Table of Contents column. Like the WEBFRAME solution above, this deployment does take up space on web browser screen. However, it allows the use of drill-down links.

The Table of Contents Example
/* specify text for top left corner of web browser frame */
%let FRAMETTL = Pierri-Bessler Demo of Web Table of Contents;
/* Output goes into the following named folder. After running this program, go there, and click on the welcome.html file. */
%let PATH = c:\YourFolder\YourSubFolder2;
/* stop output to the Results window */
ods listing close;
ods noresults;
libname DB 'c:\YourFolderForData';
/* Build ODS style for Table of Contents using as the parent the same styles.our_style as built for the CrossLink program. */
proc template;
edit styles.our_style as
styles.our_toc_style;
style Frame from Document /contentsize=24%;
/* remove extra space between TOC entries */
style Contents from Document /
pagebreakhtml_= undef_;
/* suppress bullet in front of PROC names */
style ContentProcName from IndexProcName /
bullet=none;
/* suppress bullet in front of TOC entries */
style ContentItem / bullet=none
font=('Georgia',1)
PreHtml_= undef_PostHtml_= undef_;
/* specify font name and font size */
style IndexTitle from Index /
font=('Georgia',3);
/* define black TOC title and white TOC background */
style colors / 'contitlefg'=CX000000
'contentbg'=CXFFFFFF;
/* customize the title of the TOC */
style text /
'Describe Title'='Trends & Tables';
end; run; quit;
/* Need all the same code as used in the CrossLink program to prepare for definition and use of %stplots and %yrtabls macros. It will not be listed here. Also, the macro definitions below are incomplete listings. See CrossLink for complete macro code. Essential difference is use of contents= option and des= option to specify text for Table of Contents entries. */
/* define the macro which creates a plot for each state */
%macro stplots;
%do i=1 %to &stcount;
ods html body="st&&state&i..Plt.html"
style=styles.our_toc_style;
ods proclabel ' ';
proc gplot data=DB.stathist
(where=(statenam eq "&&state&i");
plot pop*year / des="Trend&stabbrev";
run; quit;
%end;
%mend stplots;
/* empty the graphics catalog to avoid entry-name conflicts */
proc greplay igout=work.gseg nofs;
delete _all_; run; quit;
/* Create all the web pages now. Each slide, table, or graph is a separate page. */
ods html path="&PATH" (url=none) gtitle
gfootnote contents='contents.html'
frame='welcome.html' (title="&FRAMETTL")
body='welcbody.html'
style=styles.our_toc_style;
ods proclabel ' ';
goptions reset=all;
goptions device-gif xpixels=440 ypixels=330;
proc gslide name='welcslid' des='Welcome';
/* add title and note statements for the Welcome page */
run; quit;
goptions reset=all;
%stplots; run;
goptions reset=all;
%yrtabls; run;
ods html close;
THE CROSSLINK EXAMPLE
This is the most powerful solution, and the one that makes best use of available screen space. Here, navigation and drill-down through the graphs and tables are absolutely customizable to suit your needs and preferences. The authors call it "The CrossLink Example", because, with the drill-down options, the crosslinks, and the flyover text, one can explore one's way through the information by a diversity of routes. A sufficiently curious or information-needful user can easily find all the available information, or only the information that is of interest to her/him. The crosslinks require some vertical space, but not very much. If vertical space is scarce, and the image is a graph, it is possible to
produce clickable hot spots at any open white space on the graph image, by use of the Annotate facility.

The CrossLink Example Code

/* specify text for top left corner of web browser frame */
%let BODYTTL = Pierri-Bessler Demo of Graphs & Tables on the Web;

/* Output goes into the following named folder. After running this program, go there, and click on the welcome.html file */
%let PATH = c:\YourFolder\YourSubFolder1;

/* stop output to the Results window */
ods listing close;
ods noresults;
libname DB 'c:\YourFolderForData';

/* In the examples that follow, many of the web pages are drillable and/or drill-down “destinations”. However, every web page has one or more clickable links at the bottom to take you to other pages. When the web page is a table, those links are defined with link= in a SAS FOOTNOTE statement. When the web page is a graph, the graph image is immediately followed by a PROC PRINT of the “data” in DB.lnksonly with a SAS TITLE statement that uses link=. The reason for this was explained earlier in the paper. */
data DB.lnksonly;
  length heading $ 1;
  heading = ' ';
run;

/* PROC TEMPLATE Code for Our Style */
proc template;
  edit styles.default as styles.our_style;
  style fonts /
    'TitleFont'=('Georgia',10 pt)
    /* table titles and footnotes */
    'headingFont'=('Georgia',10 pt)
    /* table row and column labels */
    'docfont'=('Georgia',10 pt);
    /* table data */
  style color_list /
    'fgA2'=CX000000 /* black row and column labels */
    'fgA1'=CX000000 /* black for data, divider between labels and data, outer frame for table */
    'fgA'=CX000000 /* black titles & footnotes for PROC PRINT, if no C= override */
    'bgA3'=CXFFFFFF /* white background for table data */
    'bgA2'=CXFFFFFF /* white background for row and column labels */
    'bgA'=CXFFFFFF /* white frame for titles, white frame and background for footnotes */
  style systemtitle /
    font_size=3;
  end;
  run;
end;

/* Styles for CrossLinks */
proc template;
  edit styles.our_style as
    styles.lnksonlysmall;
    style color_list /
      'fgA2'=CX000000 /* WHITE labels */
      'fgA1'=CX000000 /* WHITE data, separator between & outer frame of labels & data */
      'fgA'=CX000000 /* WHITE titles & footnotes for PROC PRINT, if no C= override */
    style systemtitle /
      font_size=3; /* equivalent to 10 pt, if browser text size set to Medium */
  end;
  run;
end;

/* Let SAVE=c:\YourGraphFolder; */
create annotate data set with images that were previously created and saved /*
data wellcano;
  length function $8;
  function='move';
  xsys='3'; ysys='3'; x=0; y=0;
  output;
  function='bar'; x=100; y=88.89;
  style='solid'; color='white';
  html='href="FotoText.html"';
  output;
  function='move';
  xsys='3'; ysys='3'; x=62.5; y=0;
  output;
  function='image'; x=62.5; y=88.89;
  imgpath="&SAVE.USAmap.gif";
  style='fit';
  html='href="FotoText.html"';
  output;
end;
run;
quit;

/* Welcome Page. You can click on the map image for information about the authors, or click the link to start the demo */
%let SAVE=c:\YourGraphFolder;
/* create annotate data set with images that were previously created and saved */
data welcano;
/* pagebreakhtml=_undef_ */
/* suppress rule between successive proc outputs */
background=CX000000 /* white page background */
style systemtitle /
  background=CX000000 /* white title background */
end;
run;
quit;
*/

*/
/* Welcome Page. You can click on the map image for information about the authors, or click the link to start the demo */
/* create annotate data set with images that were previously created and saved */
data welcano;
/* pagebreakhtml=_undef_ */
/* suppress rule between successive proc outputs */
background=CX000000 /* white page background */
style systemtitle /
  background=CX000000 /* white title background */
end;
run;
quit;
*/

SUGI 27
Posters
function='image'; x=100; y=88.89;
imgpath="&SAVE.ITmap.gif";
stylist='fit';
output; run;
goptions reset=all;
ods html path="&PATH" (url=none)
style=styles.our_style gtitle gfootnote
body="Welcome.html" (title="&BODYTTL"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
/* The macro %makelink dynamically creates a format, providing a link for every distinct value of the variable in the data set. */
proc sort data=&data out=toprep nodupkey;
by &var;
run;
data toformat;
  length bodyname $ 40;
  set toprep;
  retain fmtname "&fmtname"; value=&var;
  rename &var=start;
  bodnamen="&varprefix||value||"&varsuffix..html&htmlsuffix";
  label='&varprefix'||trim(&var)||'&varsuffix..html&htmlsuffix";
run;
proc format cntlin=toformat;
run;
%mend makelink;
%makelink(data=DB.statepop,var=statenam,
  varprefix=st,varsuffix=Plt,htmlsuffix=,
  fmtname=$lnkplt)
run;

/* provide crosslinks to other web pages */
ods html path="&PATH" (url=none)
  body='st3tbl.html' (title="&BODYTTL")
  style=styles.lnksfootnotesmall;
  title1 h=10 pt f='Georgia'
  "History of State Populations (in millions)");
  footnote1 h=10 pt f='Georgia' c=CXFF0000
  'Click on any State name for its Population Trend Plot';
  footnote2 f='Georgia' link="st3pie.html" 'Tri-State Population Shares'
  link="tristate.html" 'Tri-State Map';
  proc print data=DB.statepop split="_'
  id statenam;
  var stpop97 stpop90 stpop80 stpop70
  stpop60 stpop50;
/* the format makes the state name label a drill-down link */
  format statenam $lnkplt.;
run;
ods html close;

/* Draggable Pie Chart of Population by Sliced by State with slices linked to trend plots of population for the states. */
*/
/* define drill-down links */
data DB.citylink;
set DB.annocity;
length html $ 40;
/* Annotate text variable is only one character if it is the marker rather than the city name. */
if length(text) eq 1 then html= 'href="city'|| trim(left(city))||'Tbl.html";
/* Draggable Map of Cities of the MSA with city markers linked to tables of information about each city. */
*/
*/
else html=' ';
run;
goptions reset=all;
ods html path="&PATH" (url=none)
  body='msacitymap.html' (title="&BODYTTL")
  style=styles.our_style
gtitle gfootnote;
goptions device=gif
  xpixels=440
  ypixels=330;
  pattern v=msolid c=CXFFFFCC;
  /* lightest yellow */
title1 h=14 pt f='Georgia'
  'Major Cities and Special Cities in the MSA';
footnote1 h=14 pt f='Georgia'
  c=CXFF0000
  'Click on any City marker for More Information';
proc gmap map=DB.msamapnc data=DB.msamapnc
  anno=DB.citylink;
id state;
choro fakevar / nolegend
  coutline=black
  name='citymap';
run; quit;
/* provide crosslinks to other web pages */
ods html style=styles.lnksonlysmall;
titlen f='Georgia'
  link="msacntytopmap.html" 'MSA Counties Map'
  link="tristate.html"      'Tri-State Map';
footnotel;
proc print data=DB.lnksonly noobs; run;
ods html close;
/* Prepare for Map and Bar Chart of County Population */
proc format;
  value rangtxt 0-99999 = 'Under 100,000'
    100000-499999 = '100,000-499,999'
    500000-999999 = '500,000-999,999'
    1000000-HIGH  = 'At Least 1M';
run;
data toformat(keep=fmtname start label);
  length label $ 20;
  set DB.countynm;
  retain fmtname '$stcntnm';
  rename stcounty=start;
  if countynm ne 'Lake'
    then label=countynm;
  else label=trim(left(countynm)) || ' (' ||
     trim(left(fipstate(state))) || ');'
run;
proc format cntlin=toformat; run;
proc sort data=DB.coupop out=coupop;
  by descending couJUL97; run;
data msapop;
  set coupop end=last;
  if _N_ eq 1 then call
    symput('maxJUL97',couJUL97);
  msapop + couJUL97;
  if last;
run;
proc sort data=DB.coupop out=sorted;
  by state county; run;
data DB.stcoupop;
  length barlabel $ 14;
  if _N_ eq 1 then set msapop;
  set sorted;
  coudrill='href="stcnty'||put(state,z2.)
    ||'/'||put(county,z3.)||'Tbl.html"';
  barlabel=put(couJUL97,fmtname.);
  substr(barlabel,10,5)=
    put(((couJUL97/msapop) * 100),z4.1) || '%';
  if substr(barlabel,10,1) eq '0'
     then substr(barlabel,10,1) = ' ';
run;
/* Drillable Map of Populations in Counties of the MSA with areas linked to tables of information about each county. */
goptions reset=all;
ods html gtitle gfootnote
  path="&PATH" (url=none)
  body='msacntytopmap.html' (title="&BODYTTL")
  style=styles.our_style
gtitle gfootnote;
goptions device=gif
  xpixels=440
  ypixels=330;
  legend1 across=1
    label=none
    mode=protect
    shape=bar(6 pct, 2.8 pct)
    value=(f='Georgia'c=CX000000 h=12 pt)
    position=(top right inside);
  pattern1 v=msolid c=CXCC99CC;
    /* light purple */
  pattern2 v=msolid c=CXCCCCFF;
    /* lightest blue   */
  pattern3 v=msolid c=CXCCFFCC;
    /* lightest green */
  pattern4 v=msolid c=CXFFFFCC;
    /* lightest yellow */
title1 h=14 pt f='Georgia'
  'County Populations in the MSA As Of 1997';
footnote1 h=14 pt f='Georgia'
  c=CXFF0000
  'Click on any County area for More Information';
proc gmap map=DB.msamapwc
  data=DB.stcoupop
  anno=DB.annocnty;
id state county;
choro couJUL97 / discrete
  coutline=black
  name='cnpopmap'
  html=coudrill;
  format couJUL97 rangtxt.;
run; quit;
/* provide crosslinks to other web pages */
ods html style=styles.lnksonlysmall;
titlen f='Georgia'
  link="msacntytoprank.html" 'County Population Shares'
  link="msacitymap.html" 'MSA Cities Map'
  link="tristate.html"   'Tri-State Map';
footnotel;
proc print data=DB.lnksonly noobs;
Drillable, Ranked, Maximally Informative, Horizontal Bar Chart with bars linked to tables of information about each county

*/*

goptions reset=all;
ods html close;

/* Drillable, Ranked, Maximally Informative, Horizontal Bar Chart with bars linked to tables of information about each county */

goptions device=gif
xpixels=440 ypixels=330
ctext=CX000000 htext=14 pt
ftext='Courier';

axis1 style=0 label=none;
axis2 style=0 label=none
major=none minor=none
value=none order=(0 &maxJUL97);
title1 h=14 pt f='Georgia'
'County Shares and Counts of Population in the MSA';
footnote1 h=14 pt f='Georgia' c=CXFF0000
'Click on any County bar for More Information';

/* the html= option links each bar to its county info table */
proc gchart data=DB.stcoupop;
hbar barlabel / sumvar=couJUL97 descending
sumlabel='Population in 1997'
maxis=axis1 raxis=axis2
width=0.7 noframe
name='cnpopbar' space=0.5
html=coudrill;
format couJUL97 comma9.;
run;
quit;

/* if not reset to null, htext=14 pt assigned above would override
the font_size set in the style used below */
goptions htext=;

/* provide crosslinks to other web pages */
ods html style=styles.lnksonlysmall;
title1 f='Georgia'
'Population for Each State (in millions) in Year &&year&i';
footnote1 f='Georgia'
'Click on any State name for its Population Trend Plot';
footnote2 f='Georgia'
'Click on any State name for its Population Trend Plot';
proc print data=DB.stathist (where=(year eq "&year&i")) split='_';
id statenam;
var pop;
format statenam $lnkplt.;
/* $lnkplt. Must have been built with the %makelink macro that
was used for the Tristate Population History Table */
format pop 6.3;
run;
ods html close;
%mend yrtabls;
%yrtabls; run;
proc sort data=DB.stathist(keep=statenam)
out=sorted nodupkey; by statenam;
run;
data _null_;
set sorted end=lastone;
call symput('state'||trim(left(_N_)),
trim(left(statenam)));
if lastone;
call symput('stcount',trim(left(_N_)));
run;
/* empty the graphics catalog to avoid entry-name conflicts */
proc greplay igout=work.gseg nofs;
delete _all_; run; quit;
goptions reset=all;
/* the %stplots macro creates a plot for each state */
%macro stplots;
%do i=1 %to &stcount;
proc means noprint data=DB.stathist
(where=(statenam eq "&&state&i")) min max;
var pop;
output out=minmax min=minpop max=maxpop;
run;
data _null_;
set minmax;
call symput('realminy',minpop);
call symput('tickminy',trim(left(put(round
(minpop,0.001),6.3))));
call symput('realmaxy',maxpop);
call symput('tickmaxy',trim(left(put(round
(maxpop,0.001),6.3))));
call symput('range_y',maxpop - minpop); run;
ods html path="&PATH" (url=none)
body="st&&state&i..Plt.html"
(title="&BODYTTL")
style=styles.our_style
gtitle gfootnote;
%let stabbrev=%upcase(%substr(&&state&i,1,2));
goptions device=gif xpixels=440 ypixels=330;
/* the %plots macro creates a plot for each state */
%macro plots;
%do i=1 %to &stcount;
proc gplot data=DB.stathist
(where=(statenam eq "&&state&i"))
plot pop*year / noframe haxis=axis1
vaxis=axis2 name="st&stabbrev.plot"
html=yrdrill;
/* plot2 is needed to have the right-hand-side vertical axis */
plot2 pop*year / noframe haxis=axis1
vaxis=axis2 name="st&stabbrev.plot2"
html=yrdrill;
/* provide crosslinks to other web pages */
ods html style=styles.lnksonlysmall;
title1 f='Georgia' link="st3tbl.html"
'Tri-State Population History'
link="st3pie.html" 'Tri-State Population Shares'
link="tristate.html" 'Tri-State Map';
footnote1;
proc print data=DB.lnksonly noobs; run;
ods html close;
%mend plots;
%plots; run;
/* Animated Map of Population History by County */
/* circumvent a problem with GIFANIM driver */
libname gdevice0 'c:\FP_LeRB\fixgdev';run;
proc gdevice nofs cat=gdevice0.devices;
delete fixanim;
copy gifanim from=sashelp.devices
newname=fixanim;
modify fixanim hsize=0 vsize=0;
run; quit;
goptions reset=all;
proc greplay igout=work.gseg nofs;
delete _all_; run; quit;
goptions reset=all;
proc gplot data=DB.stathist
(where=(statenam eq "&&state&i"))
plot pop*year / noframe haxis=axis1
vaxis=axis2 name="st&stabbrev.plot"
html=yrdrill;
/* The trick used here to label endpoint of the plot works only
because the first point is the minimum response and the last point
is the maximum response. For general solutions for annotating a
trend line, see the authors' other SUGI 27 paper about their
%TREND macro. */
axis1 label=none minor=none
major=none style=0
value=(c=CX000000 h=14 pt f='Georgia');
axis2 label=none minor=none
major=none style=0
order=realmaxy by &range_y
value=(c=CX000000 h=14 pt f='Georgia' " &tickminy = ");
axis3 label=none minor=none
major=none style=0
order=realmaxy by &range_y
value=(c=CX000000 h=14 pt f='Georgia' " ");
SUGI 27 Posters

"create a placeholder, but the title will not display "*/
proc gslide name='mapanim';
title='animated gif placeholder';
run; quit;
ods html close;
goptions reset=all;
ods listing;
"* create the animated gif using the same file */
filename animout "&PATH\mapanim.gif";
goptions dev=fixanim xpixels=440 ypixels=330
gsname=animout iteration=1 delay=100;
pattern1 v=msolid c=CXCC99CC; /* light purple */
pattern2 v=msolid c=CXCCFFFF; /* lightest blue */
pattern3 v=msolid c=CXCCFFCC; /* lightest green */
pattern4 v=msolid c=CXFFFFFC; /* lightest yellow */
legend1 across=1 label NONE mode=protect
shape=bar (6 pct, 2.8 pct)
position=(top right inside)
order=('1' '2' '3' '4')
value=(c=CX000000 h=12 pt f='Georgia'
      'Under 100,000' '100,000-499,999'
      '500,000-999,999' 'At Least 1M')
proc sort data=DB.cntyhist(keep=year)
  order=('1' '2' '3' '4')
  on back arrow";
footnote1 h=14 pt f='Georgia'
   County Population, Year is &&cyear&i;
title1 h=14 pt f='Georgia' "Evolution of MSA
%end;
goptions gsfmode=append;
%else %do;
%end;
goptions gsfmode=replace;
%if &i eq 1 %then %do;
%do i=1 %to &cyrcount;
  %macro yrmaps;
  run;
  data _null_; set sorted end=lastone;
call symput('cyrcount',trim(left(_N_)));
  if lastone;
  call symput('cyear'||trim(left(_N_)),
    set sorted end=lastone);
  data _null_; run;
out=sorted nodupkey; by year;
run;
data _null_; set sorted end=lastone;
call symput('cyear'||trim(left(_N_)),
    set sorted end=lastone);
  if lastone;
  call symput('cyear'||trim(left(_N_)),
    set sorted end=lastone);
  data _null_; run;
out=sorted nodupkey; by year;
run;
data _null_; run;
filename animout recfm=n mod; put '3B'x;
run;
filename animout clear;
ods listing close;

CONCLUSION
After exercising, exploring, and comparing the capabilities of the SAS/GRAPH WEBFRAME driver, the customized ODS-supported Table of Contents, and the authors’ CrossLink Example, it is clear that the most flexible and powerful way to web-deploy and web-link data and image in tables and graphs that inform and influence is with the CrossLink method.

In any case, all of the illustrations used for this project employ a design that is intended to optimally use the software to communicate, not to decorate. When there eventually is increased presentation of tables and graphs via wireless communication and handheld devices, the minimal, but maximally informative and web-friendly, design presented here will be suitable for the small screen.

This paper compared tools currently available to SAS users, and demonstrated the authors’ suggested best practices for use of those tools. However, standards and features for web browsers, and capabilities of ODS and SAS software, will evolve.

The authors’ expected future technical initiatives will be to:
• see how other SAS web publishing features can be effectively used within the authors’ design framework;
• explore further the use of scanned images;
• develop simple macros to relieve the coding redundancy that occurs when assembling a package of web pages; and
• identify the simplest and most intuitive, but still adequate, way to use the complex PROC TEMPLATE facility to build the requisite styles.

Recommended software improvements are:
• support for the LINK= parameter (already available when using non-graphic PROCs) in the SAS/GRAPH TITLE, FOOTNOTE, and NOTE statements; and
• increased white space between the link labels produced with LINK=, to improve the readability of the link labels.

COMPATIBILITY NOTES
Netscape Navigator 4.0 does not support the flyover text as implemented with here ODS. Also, it shows grid lines for the tables, unlike Internet Explorer.

RELATED WORK BY THE AUTHORS
LeRoy Bessler, Francesca Pierri “%TREND: A Macro to Produce Maximally Informative Trend Charts with SAS/GRAPH, SAS, and ODS for the Web or Hardcopy”, SUGI 27 Proceedings.

CONTACT INFORMATION
Your comments and questions are valued and encouraged. Contact the authors at:
Dr. Francesca Pierri
Centro d’Ateneo per i Servizi Informatici
Università degli Studi di Perugia
Via G. Duranti 1/A, S. Lucia Canetola
06125 Perugia, Italy
Work Phone: 011 39 075 5853794
Fax: 011 39 075 5853615
Email: frc@unipg.it

Dr. LeRoy Bessler
Bessler Consulting & Research
PO Box 96
Milwaukee, WI 53201-0096, USA
Phone: 1 414 351 6748
Email: bessler@execpc.com

TRADEMARKS
SAS and SAS/GRAPH are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration. Other brand and product names are registered trademarks or trademarks of their respective companies.