Paper 150-2007

Animating SAS/GRAPH® Output—Telling a Story That Changes over Time

Stuart A. Nisbet, SAS Institute Inc., Cary, NC

ABSTRACT

With the rapid advancements in computer processing speeds and high-end graphics displays, we've seen a proliferation of eye-catching animations, multimedia, and special effects in many common applications. It's easy to forget that one of the most traditional methods of display can be used without mandatory software or hardware upgrades to deliver compelling content. Additionally, the use of these techniques doesn't have to be just for glitz and glamour; their use can actually depict meaning and context in a business application. This video poster describes an age-old technology—animated GIFS—that can be created and deployed from within SAS/GRAPH® software. Animated GIFs can provide additional meaning and information without requiring additional screen space or an investment in faster CPUs, graphics display cards, or software. This poster shows how animated GIFS can be built from simple modifications to the SAS® jobs that you run every day and deployed using the built-in support for this file format in all the major HTML browsers (Internet Explorer, Firefox, Safari, and so on). So, if you thought that GIF animation was only about spinning logos, take a look at what SAS/GRAPH can do for you!

INTRODUCTION

This paper is intended to accompany a video poster submission of the same title for SAS Global Forum 2007. A topic such as computer animation certainly benefits from an interactive display much more than a printed paper with static images. Nevertheless, this paper provides

1) an explanation of the benefits of animating business data
2) an overview of the technical concepts required
3) perhaps most importantly, a source of code snippets to show the syntax for generating animated GIF files from SAS/GRAPH

This paper follows on the excellent work done by Mike Zdeb, Robert Allison, and Barbara Schneider, among others, who have previously shown compelling examples of GIF animations using SAS/GRAPH.

THE BUSINESS CASE FOR GIF ANIMATION

There are many different kinds of maps, plots, and charts that display information, including information that changes over time, in a very clear and concise way. Stock plots showing high/low/close prices over a period of time have been used very effectively, for example. Grouped or subgrouped bar charts are equally effective at showing comparison information among categorized values.

So, why and when does it make sense to introduce animation into a data presentation or analysis project? There are many scenarios where animation can be used to enhance the users' interpretation of information, and this paper presents three:

- **A 3D surface or scatter plot contains data that can be obscured from certain viewpoints**. Rotating or tilting the graph can show additional features, but it can also obscure other features at the same time. Using animation to rotate around the entire plot affords a 360-degree view of the data.

- **In many plots generated over time, some or all of the data points might exhibit nonsequential movement which is difficult to track without seeing the data animated**. In this case, a series of separate graphs would be difficult to interpret because they require the reader to look back and forth between images to gather context.

- **When the volume of data is large, it is very difficult to interpret subtle trends in data movement over time without the added visual enhancement of animation**. Even slight changes in a plotted point's location are much easier to see in an animated graph than comparing images side-by-side.

HOW GIF ANIMATION WORKS

Without going into the specifics of the GIF file format, a single-image GIF file contains a list of colors to be used in the image and a sequence of pixels which refer back to the color map to represent the image. In the case of an animated GIF, there are multiple, distinct color map and pixel pairs stored sequentially in the file which represent the series of frames in the animation loop. Additionally, there is header information that specifies attributes such as the number of times the loop is to iterate, the delay time between frame display, and whether or not to refresh or restore

the background when subsequent frames are played. At the end of a properly generated GIF file, there is a termination character which signals the end of file. There is a special SAS/GRAPH driver which generates animated GIF files and additional GOPTIONS which control these attributes.

You can control the GIFANIM device driver with these GOPTIONS settings:

ITERATION=iteration-count
specifies the number of times to repeat the animation loop, or whether to loop infinitely. An iteration of 0 specifies an infinite loop.

GSFMODE=REPLACE | APPEND (REPLACE is the default.)
specifies whether the graphics output should replace the contents of an existing file or be appended to it. In addition, the GIFANIM driver uses the value of GSFMODE to determine when to write the GIF header. For the first frame in an animated GIF file, the GSFMODE should be set to REPLACE. For subsequent graphs that you would like added to the animation sequence, you should set the GSFMODE to APPEND. If all graphs in the animation are generated with a single PROC invocation by group processing (for example), the GSFMODE should be set to REPLACE. All of the subsequent graphs generated from the same PLOT/VBAR/etc. statement will be added to the same file in sequence.

DELAY=delay-time
controls the amount of time between graphs in the animation sequence. A delay time of 1 specifies a delay of .01 seconds.

DISPOSAL = NONE | BACKGROUND | PREVIOUS | UNSPECIFIED (NONE is the default.)
specifies what happens to the previous frame before the subsequent frame is displayed. In most cases, you'll want to start with a "blank slate" by setting the disposal method to BACKGROUND. This sets the entire background to a single color and ensures each previous frame is cleared before the next frame is displayed. In the case where you want each subsequent frame to be played on top of the underlying frame (without erasing), set DISPOSAL to NONE and enable TRANSPARENCY (below). Quite frankly, all of the browsers and image processing programs I've found treat PREVIOUS and BACKGROUND exactly the same. Both result in the background being cleared before the next graph is drawn. The UNSPECIFIED option is documented in the GIF89a specification, but each GIF viewer and Web browser I tested defaulted to the same behavior as the BACKGROUND option would have produced.

TRANSPARENCY | **NOTRANSPARENCY** (NOTRANSPARENCY is the default.)
specifies whether the background of the image should appear to be transparent when the image is displayed in the browser. This is very useful if you want to display the GIF file in an environment where the background is intended to be shown behind the graph. NOTRANSPARENCY clears a rectangular region of the background color upon which the graph will be drawn. Also, if your intention is to append each subsequent graph on top of the preceding ones, you'll want to turn on TRANSPARENCY as well.

GEPILOG
sends a string to a device or file after all graphics commands are sent. An animated GIF file remains open to allow subsequent frames to be appended to the end of the file. By specifying the GEPILOG option as "GOPTIONS GEPILOG='3b'x; after the GIF file is created, the terminating hex character '3b' is appended to the end of the file. An example is also provided to show how to manually append this GIF file termination character in the case where multiple PROC invocations are used to generate a single animated GIF file.

GIF ANIMATION IN PRACTICE

Here are a few examples of different practical uses for animated GIF files along with the code used to generate them.

FIRST EXAMPLE: ROTATING 3D SURFACE PLOT

In some cases, animated GIF files can be used to change the viewing angle to allow for obscured data points to be visible. An example is a surface plot using the SAS/GRAPH G3D procedure. While these types of plots can be individually rotated and tilted at various angles to reveal data that is obscured by other sections, there is often not one single rotation or tilt angle that affords a completely unobstructed view of the data. Using an animated GIF file, several different plots can be replayed in sequence to view the surface plot from multiple viewing angles. While the appropriate choice of rotation and tilt angles are very much data-dependent, the result is a single GIF file which shows the data from all chosen views (Figure 1).

```
/* Designate a GIF file for the G3D output. */
filename anim 'pond.gif';

/**  Set the GOPTIONs necessary for the      **/
/**  animation.                              **/
goption reset  dev=gifanim  gsfmode=replace border
   gsfname=anim  xpixels=640  ypixels=480
   iteration=0  delay=5
   gepilog='3B'x  /* add a termination char to the end of the GIF file */
   disposal=background;

title height=3 'Contour of Mill Pond Depths';
footnote font=swissl height=3 justify=left ' Animated GIF to show obscured
features';

/* … data and G3GRID code omitted for space */
proc g3d data=pondgrid;
   plot vdist*hdist=height / noaxes rotate=0 to 350 by 10;
   run;

quit;
```

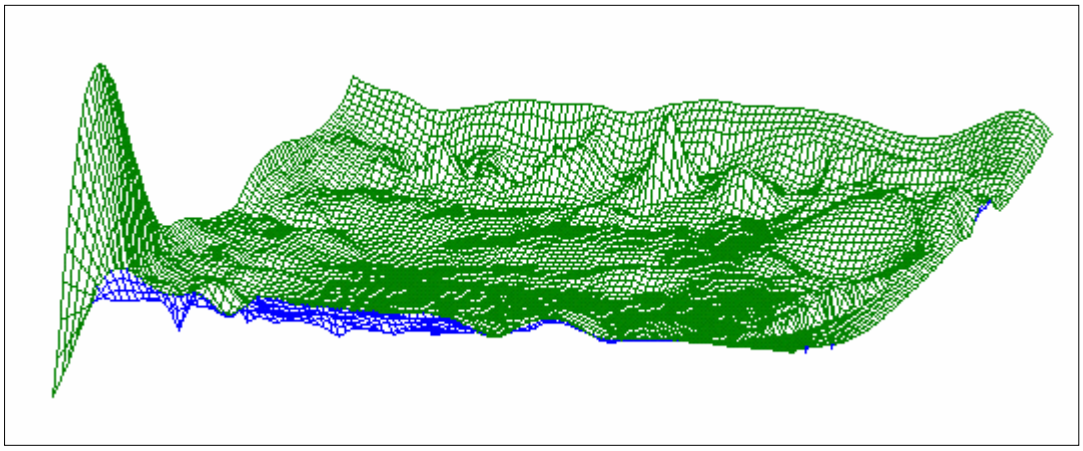

Figure 1. Contour of Mill Pond Depths—Animated GIF to Show Obscured Features

SECOND EXAMPLE: SCATTER PLOT OF FERTILITY RATE VS. LIFESPAN OVER TIME

In this example, the data contains the expected lifespan and fertility rate for over 150 nations from 1955 to 2005. The plotted points are colored based on the nation's continent. From the static image below (Figure 2), you can see the tightly clustered green points in the upper left of the graph representing Europe (relative small families and long lifespan). You can also see the cluster of purple points in the lower right representing African countries (large families and relatively short lifespan). In the final animated GIF, you can also see many nations like China, Viet Nam, and India moving from the lower right to the upper left of the graph. Because of the large number of points, only a few of the countries of particular interest are labeled.

```
/* Designate a GIF file for the GPLOT output. */
filename anim 'c:/population.gif';

/**  Set the GOPTIONs necessary for the animation       **/
/* iteration=0 causes the animation to loop continuously */
/* delay=200 causes a 2 second pause between frames      */
goption dev=gifanim  gsfmode=replace cback=cxE6C970
        gsfname=anim  xpixels=640  ypixels=480
        iteration=0  delay=200 gepilog='3B'x
        disposal=background;  /* clear the entire image */
```

```
proc sort data=final; by year; run;

data temp; set final; labelval = country;
if (labelval ^= "China") and
(labelval ^= "United States") and
(labelval ^= "India") and
(labelval ^= "Viet Nam") and
(labelval ^= "Cambodia") and
(labelval ^= "Rwanda")
then labelval = ""; run;

symbol1 v=dot c=cx8B3F89 pointlabel=("#labelval");
symbol2 v=dot c=cxFABC46 pointlabel=("#labelval");
symbol3 v=dot c=cx61A835 pointlabel=("#labelval");
symbol4 v=dot c=cx3F769A pointlabel=("#labelval");
symbol5 v=dot c=cxCC0067 pointlabel=("#labelval");
symbol6 v=dot c=cxD77100 pointlabel=("#labelval");

legend1 shape=symbol(0.2, 1) frame cframe=cxFFFEBD;   /* make the legend smaller so
only one DOT appears */

proc gplot data=temp;
plot span * rate = cont_code / vaxis = 20 to 90 by 10 haxis = 1 to 9 legend=legend1
cframe=cxFFFEBD;
by year;
run;
quit;
```

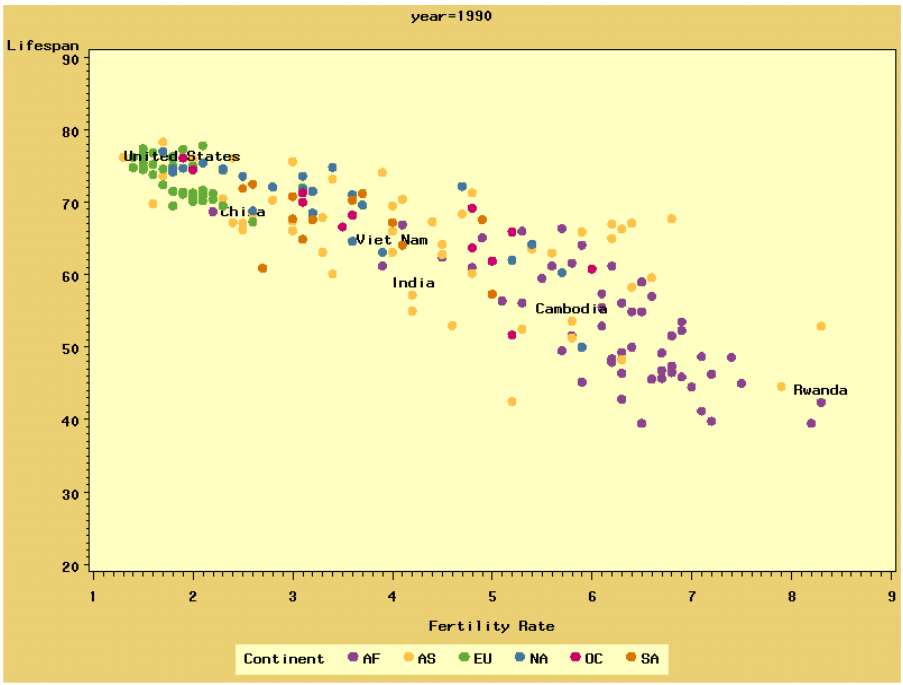

Figure 2. Scatter Plot of Fertility Rates by Nation—Animated GIF to Show Rate Over Time

THIRD EXAMPLE: USING ANIMATION, TRANSPARENCY, AND OVERLAY TO SHOW HISTORICAL TRENDS

Using the same data as the example above, but subsetting a few countries of interest, this example shows another interesting capability of animated GIF files. In this case, TRANSPARENCY is turned on, and the DISPOSAL mode is set to NONE. This causes each subsequent frame to be written directly over the top of the previous one, but the underlying image is not cleared. Since TRANSPARENCY is enabled, only the plotted points and text are written. The background is transparent, so the previously plotted points are still visible. This results in a "trail" of points being plotted to show the historical values. Since the X and Y axes don't change between graphs, they appear static even though they are plotted each time. The year does change between frames, so the "year=19##" title at the top

becomes overwritten and illegible very quickly. To fix this situation, a very simple annotated rectangle is drawn to clear this area between frames. The color of this rectangle is set to almost white to match the background, but it can't be set to pure white. This is because the GIF file specification has a provision for specifying the transparent color. Any pixels in the GIF file which match this color exactly will not be drawn at all. For this reason, if the rectangle color was set to WHITE (or CXFFFFFF), the rendered GIF would ignore these pixels altogether. It is much easier to see in the actual animated GIF file, but the green plotted points representing India show a definite trend from the lower right to the upper left. Similarly, the trend of China from large families and relatively short life spans to smaller families and longer lives is very marked and obvious in the animated GIF (Figure 3).

```
/* Designate a GIF file for the GPLOT output. */
filename anim 'c:/population_trailer.gif';

/**  Set the GOPTIONs necessary for the animation.     **/
/**  In particular, set transparency so each new frame **/
/**  is written over top of the preceding one without  **/
/**  erasing the background first                      **/
goption dev=gifanim  gsfmode=replace
        gsfname=anim  xpixels=640  ypixels=480
        transparency  iteration=0  delay=100
           gepilog='3B'x
        disposal=none;  /* none, don't erase the background */

data few; set final;
if (country = "United States") or
(country = "China") or
(country = "Viet Nam") or
(country = "India") or
(country = "Cambodia") or
(country = "Italy");
run;

proc sort data=few; by year; run;

symbol1 v=dot c=cx8B3F89;
symbol2 v=dot c=cxFABC46;
symbol3 v=dot c=cx61A835;
symbol4 v=dot c=cx3F769A;
symbol5 v=dot c=cxCC0067;
symbol6 v=dot c=cxD77100;

legend1 shape=symbol(0.2, 1) frame;  /* make the legend smaller so only one DOT
appears */

/* Build a simple annotate data set to blank out the region behind the "year=" title
*/
data anno;
length function $ 8;
length color $ 8;
length xsys $ 1;
length ysys $ 1;
input function $ x y color $ style $  xsys $ ysys $;
cards;
poly 40 100 cxfefefe solid 3 3
polycont 60 100 cxfefefe solid  3 3
polycont 60 95 cxfefefe solid  3 3
polycont 40 95 cxfefefe solid 3 3
;
run;

proc gplot data=few anno=anno;
plot span * rate = country / vaxis = 30 to 90 by 10 haxis = 1 to 9 legend=legend1;
by year;
run;
quit;
```

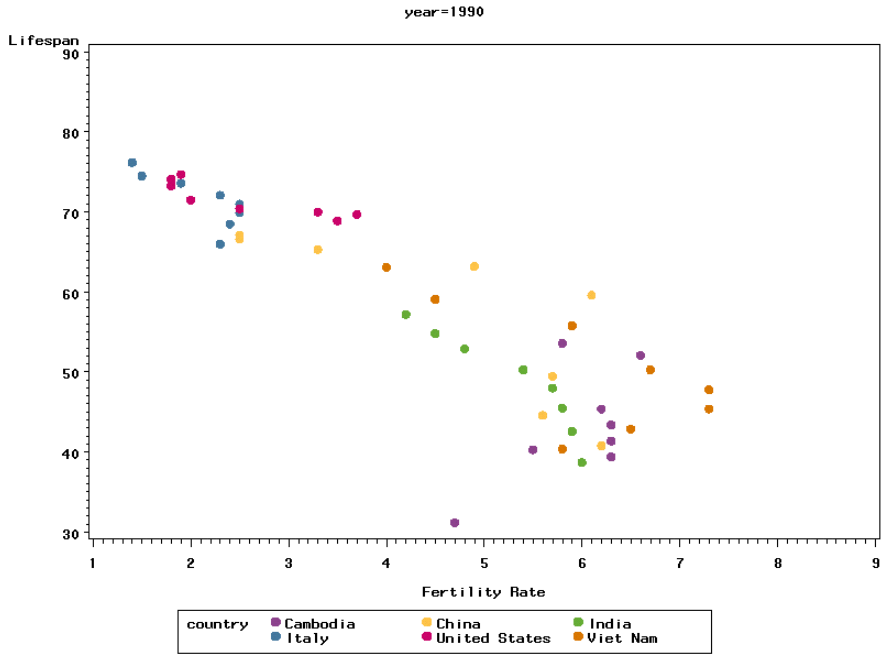

Figure 3. Scatter Plot of Fertility Rates by Nation—Animated GIF to Show Transparency and Overlay

QUIRKY WAY TO GET TRUE COLOR SUPPORT IN GIF FILES USING ANIMATION

A drawback to GIF files (animated or static) is the 256-color limit in the GIF file specification. Many other image formats allow 24-bit color support and have flourished on the Web as a result. Both PNG and JPG files support 24-bit colors, for example, but neither supports animation via the storage and playback of multiple frames. Interestingly, via clever use of the animation support feature of GIF files, 24-bit color support can actually be achieved. While this is not a typical use for animated GIF files, it is illustrative to understand how this is achieved. The key to this "workaround" is that each frame of an animated GIF can have its own 256-color map. As discussed above, animated GIF files can specify that the previous image is not erased before the next image is displayed. Putting these two concepts together, it is possible to store the first 256 colors and corresponding pixels in the first animated GIF frame, and the next 256 colors and pixels in the next frame. By continuing to store 256-color maps and pixels in subsequent frames until all of the original (24-bit) image is stored, a true-color image can be represented in a GIF file. When this animated GIF is displayed, each "layer" of the original image is displayed on top of the previous one(s). Once again, this isn't a common use for the animation feature, but it's worth noting that there are software packages available which will convert true-color (24-bit) images into these special animated GIF files.

ALTERNATIVES TO ANIMATED GIF FILES

While the focus of this paper and corresponding video poster is the use of animated GIF files for data presentation, it is worth mentioning that there are many alternative approaches to Web animation which offer significant additional functionality over GIF files. For example, both Macromedia Flash and the open file format SVG (Scalable Vector Graphics) provide animation capabilities which far surpass the simple frame playback of animated GIF files. The intent of this paper is not to enumerate the pros and cons of each, but if your needs dictate greater functionality than is afforded by the GIF specification, there are other options for dynamic graphics Web content. As always, the trade-off for this added functionality is the additional requirement of a plug-in to your browser, or the runtime download of Java applet code, or some other such mechanism.

CONCLUSION

There are many exciting new technologies being used for animated and interactive graphical displays on the Web, but one of the most broadly supported is the use of animated GIF files. All of the most popular Web browsers support animated GIF files without requiring third-party plug-ins or downloaded applets. SAS/GRAPH provides several options for specifying specific attributes of the animated GIF files to tailor the playback rate, looping, and transparency to achieve various animation effects. While all of these effects are targeted at an interactive environment versus printed output, they provide an added dimension to data presentation which static images cannot. While the most popular use of animated GIF files on the Web has traditionally been logos and eye-catching entertainment animations, using SAS/GRAPH to generate more meaningful data presentations via animated GIFs might warrant a fresh look.

RESOURCES

SAS Institute Inc. 2007. Data Visualization Web Site. SAS Graphing Components. Available at support.sas.com/rnd/datavisualization.

SAS Institute Inc. 2007. Documentation for SAS®9 Products Web Site. Available at v9doc.sas.com/sasdoc/.

SAS Institute Inc. 2007. SAS Customer Support Center Web Site. Samples. Available at support.sas.com/sassamples/index.html.

SAS Institute Inc. 2007. SAS/GRAPH Software Samples Web Site. Samples that Use the GIFANIM Device Driver. Available at support.sas.com/rnd/samples/graph/gifanimoverview.html.

SAS Institute Inc. 2007. SAS Technical Support Web Site. Available at support.sas.com/techsup/ftp/download.html.

SAS Institute Inc. 2007. Software Downloads Web Site. SAS/GRAPH Software. Available at support.sas.com/download.

SAS Institute Inc. 2007. Technologies/Business Intelligence Web Site. Visualization. Available at www.sas.com/technologies/bi/visualization/index.html.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author:

Stuart A. Nisbet
SAS Institute Inc.
SAS Campus Drive
Cary, NC 27513
Work Phone: (919) 677-8000
Fax: (919) 677-4444
E-mail: Stuart.Nisbet@sas.com
Web: www.sas.com