MWSUG 2018 - Paper SP-078

Improving Plots Using XAXISTABLE and YAXISTABLE

Jacob Keeley, Henry Ford Health System, Detroit, MI
Carl Nord, Eli Lilly and Company, Indianapolis, IN

ABSTRACT

New to the SGPLOT procedure for SAS 9.4TM, the XAXISTABLE and YAXISTABLE statements respectively create an X/Y axis aligned row/column of textual data placed at specific locations in relation to the primary plot within the given SGPLOT procedure. The XAXISTABLE and YAXISTABLE statements are applicable with any primary plot, aside from BAND, BLOCK, FRINGE, REG, LOESS, and PBSPLINE plots. Along with directing the X, Y coordinates of the supplementary data values, there are many options accompanying the XAXISTABLE/YAXISTABLE statements which allow the user to change the color, order, and position of the accompanying row(s) of data. The XAXISTABLE statement proves its worth when used in conjunction with survival analysis. When dealing with Kaplan Meier survival curves, the XAXISTABLE statement essentially allows the user to personalize their own at risk tables when the LIFETEST procedure lacks the functionality necessary for the request. In this framework, the XAXISTABLE improves greatly upon what would have previously been an arduous task. Overall, the XAXISTABLE and YAXISTABLE statements are a welcome addition to the SGPLOT syntax, as the user is given even more control over the appearance of a desired plot, making it less likely that the plot needs to be altered after it has been output.

INTRODUCTION

As the functionality of the SGPLOT procedure continues to expand, two new editions to SAS 9.4, the XAXISTABLE and YAXISTABLE statements, prove to be quite useful. Per the statement name, an X/Y axis aligned row of textual data is placed at specific locations in relation to a primary plot within a given SGPLOT procedure. When used in conjunction with other plot statements it is easy to see the value that the XAXISTABLE and YAXISTABLE statements bring. This is especially true when it is necessary to convey additional information that would otherwise be left off a plot, or perhaps haphazardly added afterwards. In an effort to create better plots, this paper aims to provide information on when and how to employ the use of the XAXISTABLE and YAXISTABLE statements.

USING XAXISTABLE/YAXISTABLE

The XAXISTABLE/YAXISTABLE statements can be used along with any primary plot aside from the following: BAND, BLOCK, FRINGE, REG, and LOESS. The statements are equipped to handle many different options in order to format and position the values as necessary, however only one argument, the variable, is required. An example of a simple XAXISTABLE statement is as follows:

```
proc sgplot data=dsname;
      scatter x=Type y=Size;
      xaxistable Score/ valueattrs=(color=red);
      xaxis values = (1 to 10 by 1) display=all;
run;
```

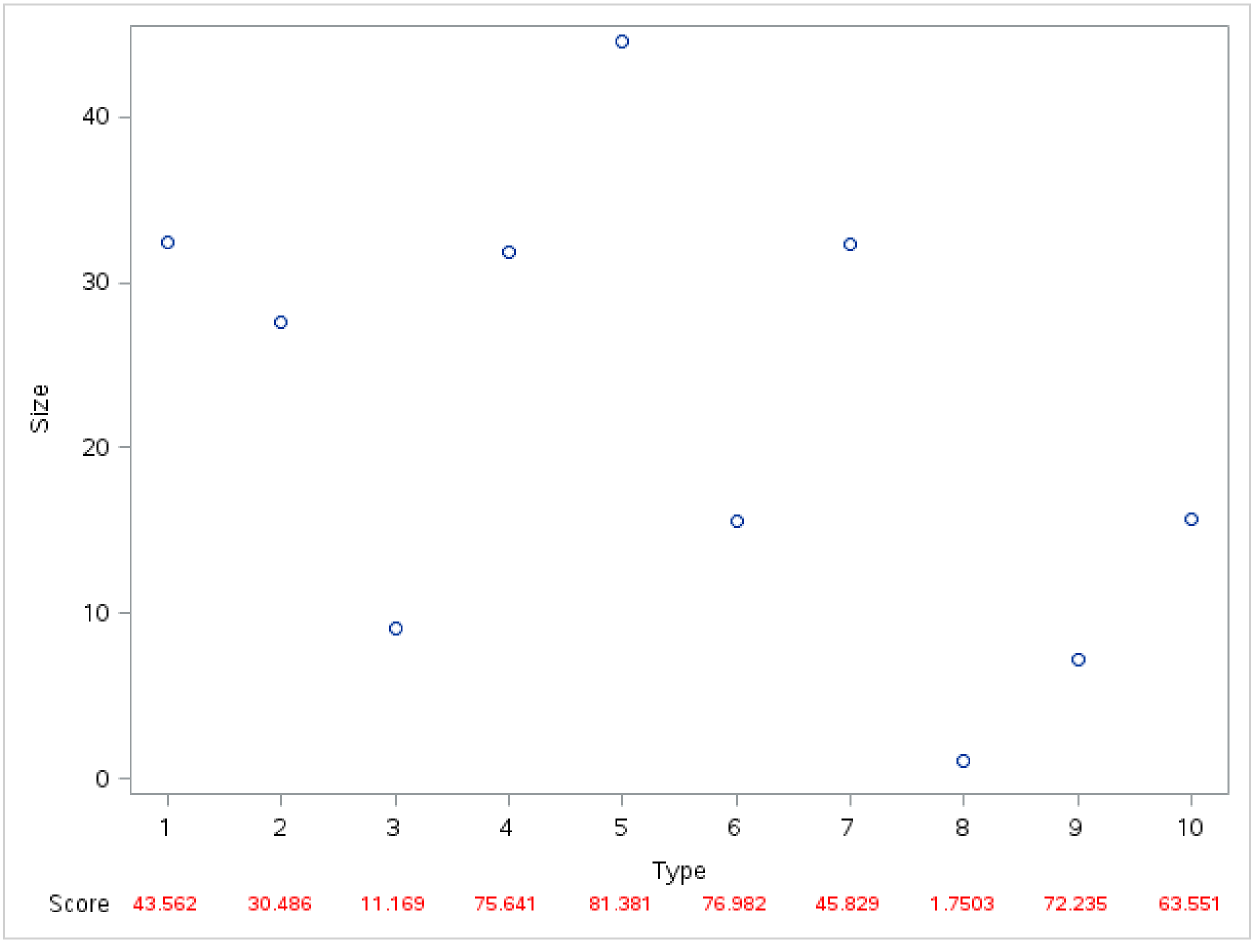

Figure 1. Displaying a Size by Type Plot with the associated score shown on the horizontal axis.

As seen in Figure 1, a typical Size by Type scatter plot is generated. However, the XAXISTABLE statement allows for additional information to be placed on the horizontal axis. This is a simple use case but a necessary first step of discussion.

The YAXISTABLE statement is used similarly to XAXISTABLE. However, as expected, the YAXISTABLE information will be produced on the vertical axis:

```
proc sgplot data=dsname;
      scatter x=Size y=Type;
      yaxistable Score/ valueattrs=(color=red);
      yaxis values = (1 to 10 by 1) display=all;
run;
```

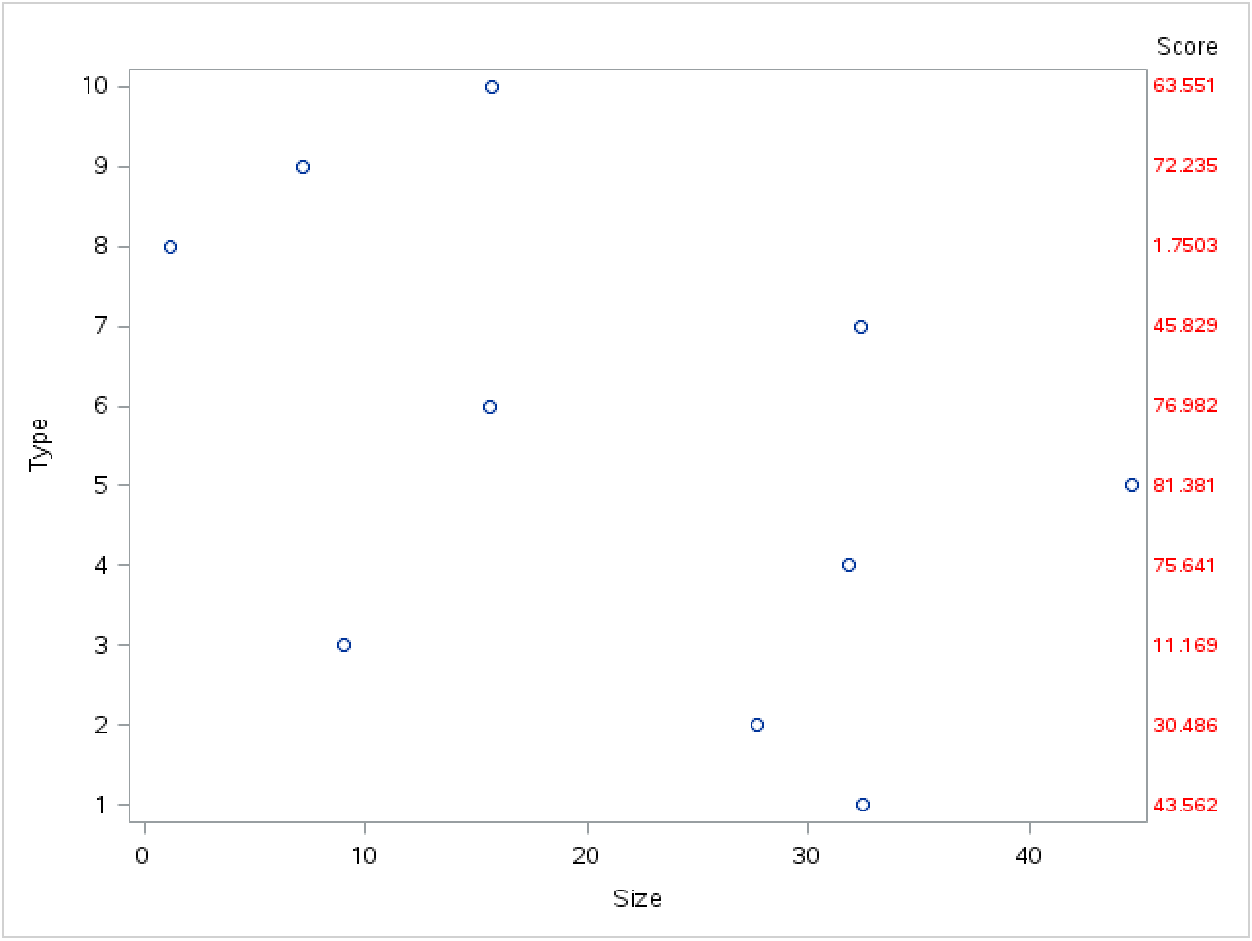

Figure 2. Displaying a Size by Type Plot with the associated score shown on the vertical axis.

While all options can be reviewed in the documentation, several options such as class, location and x seem to be utilized with relative frequency and are quite helpful. Per usual, the class option allows for differentiation between unique values of the specified variable. The location option specifies whether the axis table is placed inside or outside of the axis area. Lastly, the X option specifies the X variable to align the table values to the X axis. The next example shows how using options in SGPLOT can lead to different interpretations of the situation by an observer of the figure.

Let's say a company wants to see if two different production methods result in a different level of quantity supply over time. The company has had production issues and they want a safe amount of product quantity available to ensure they can meet demand. Quantity projections have been placed on the two methods.

```
proc sgplot data=sales1;
      series x=Time Y=Quantity/group=Method;
      xaxistable Projection/ labelattrs=(weight = bold size = 12)
      valueattrs=(size = 10) pad = (top = 5pct);
      run;
```

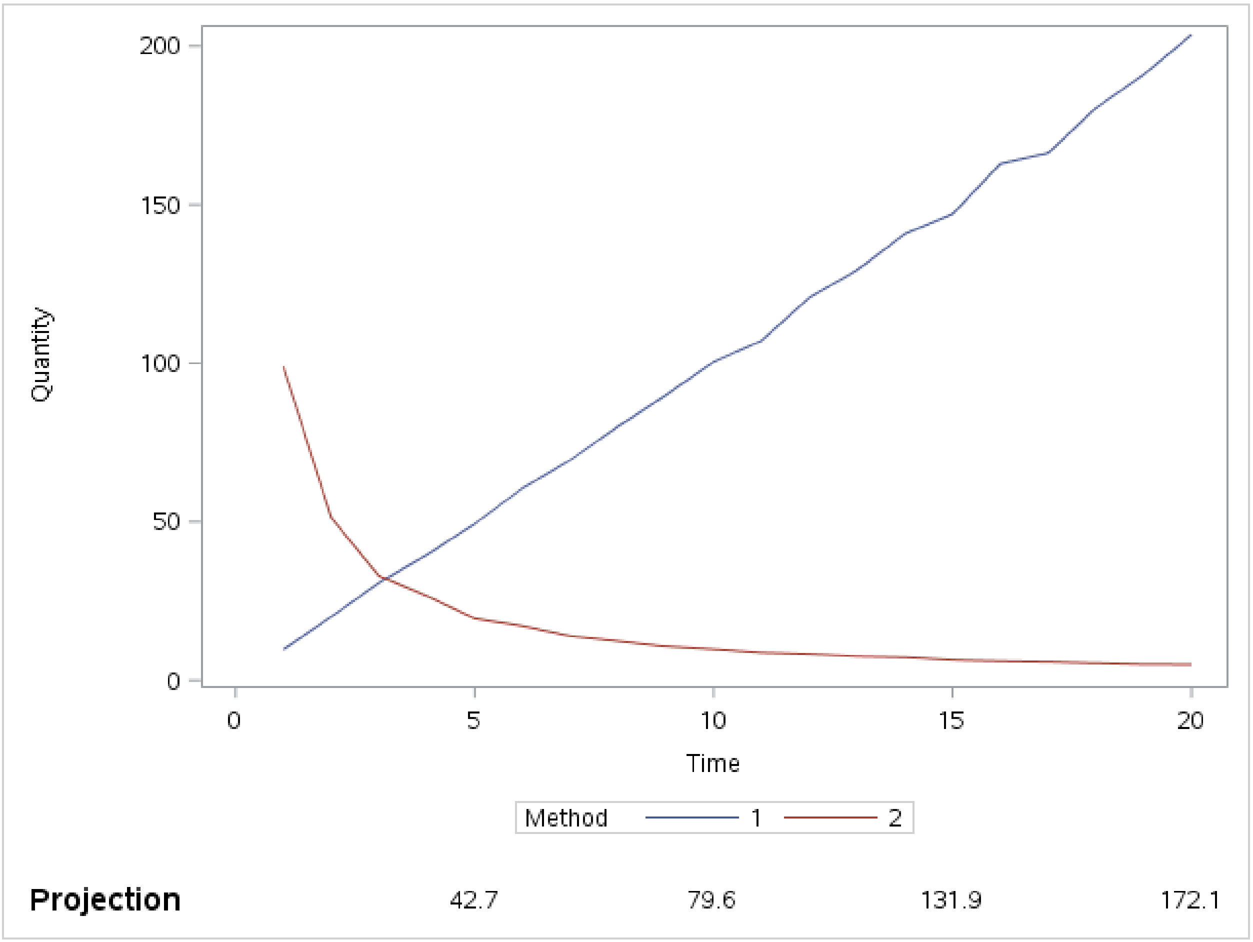

Figure 3. Displaying quantity fluctuation over time grouped by two different methods. This is with the default XAXISTABLE.

Figure 3 above shows the default XAXISTABLE specification. Notice anything wrong with Figure 3? We have projection quantity estimates at specified time points. But, there are two methods but only one projection. It would not be clear to the company what these projection estimate values represent. Therefore, it is necessary to see the projection values by method. The next example shows how this would be achieved.

```
proc sgplot data=sales1;
      format Method method.;
      series x=Time Y=Quantity/group=Method;
      xaxistable Projection/class = Method
      labelattrs=(weight = bold size = 12)
      valueattrs=(size = 10)
      colorgroup=Method
      location=inside x = time;
run;
```

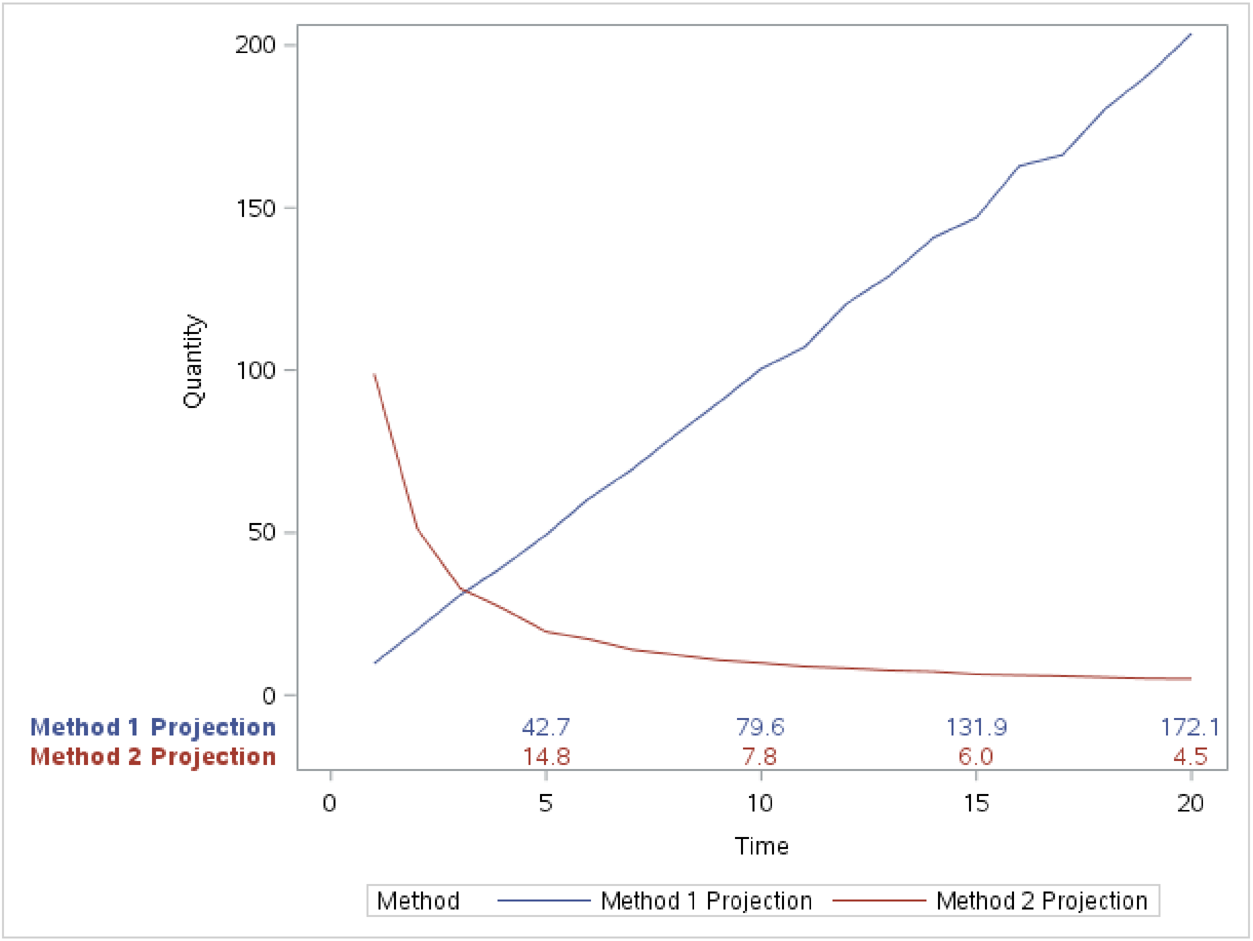

Figure 4. Displaying quantity fluctuation over time grouped by two different methods. Using relevant options in XAXISTABLE.

Notice in Figure 4 that the company would now be able to view their projections by method, all neatly within the figure and at the corresponding time point. The matching colors are also a nice feature. These grouping colors are achieved by the ‘`colorgroup=Method`’ code.

USING XAXISTABLE TO CREATE AN AT RISK TABLE

While these features to SGPLOT are enough to demonstrate an improvement, one of the most practical uses of the XAXISTABLE statement comes when customizing survival curves plots. Certainly the crux of the survival plot, the curves will always be paramount, but there can be additional information such as the cumulative number of events, as well as the numbers at risk that can improve a survival plot. With the flexibility that the SGPLOT procedure offers compared to that of the atrisk option in the LIFETEST procedure, it is easy to see why outputting curves to SGPLOT can be useful.

A useful feature itself, the LIFETEST procedure is able to include an at risk table in the generated plot using the atrisk option:

```
proc lifetest data=survhelp notable
      plots=survival(atrisk=(0,500,1000,1500,2000,2500,3000));
      time time*surv(0);
      strata group;
      format group groupI.;
run;
```

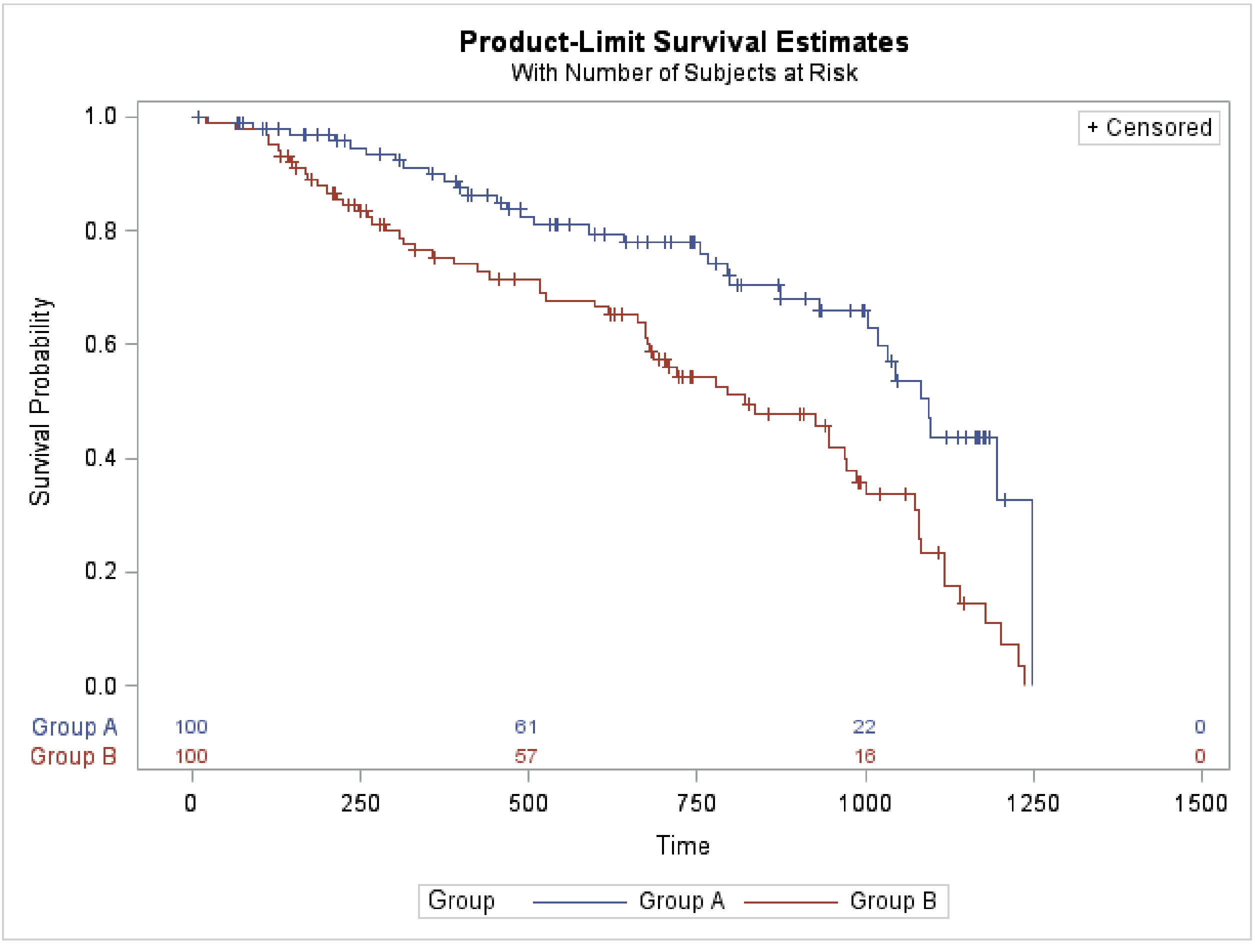

Figure 5. Displaying Kaplan Meier curves produced with an atrisk table using PROC LIFETEST.

By outputting the plot data, the plot becomes much easier to customize. Combining a couple of different SGPLOT statements, it is quite easy to generate a near identical plot:

```
proc sgplot data=flags;
      step x=time y=survival / group=stratumnum name='s';
      scatter x=time y=censored / markerattrs=(symbol=plus) name='c';
      scatter x=time y=censored / markerattrs=(symbol=plus)
      GROUP=stratum;
      xaxistable atrisk / x=tatrisk class=stratumnum
      colorgroup=stratumnum location=inside;
      keylegend 'c' / location=inside position=topright;
      keylegend 's' / linelength=20;
      format stratumnum groupI.;
run;
```

The step and scatter plots should both be familiar, but it is the use of the XAXISTABLE statement that produces the at risk table, similarly to the LIFETEST procedure. The use of the x, class, and location options are apparent here as well, and are necessary to replicate the LIFETEST plot.

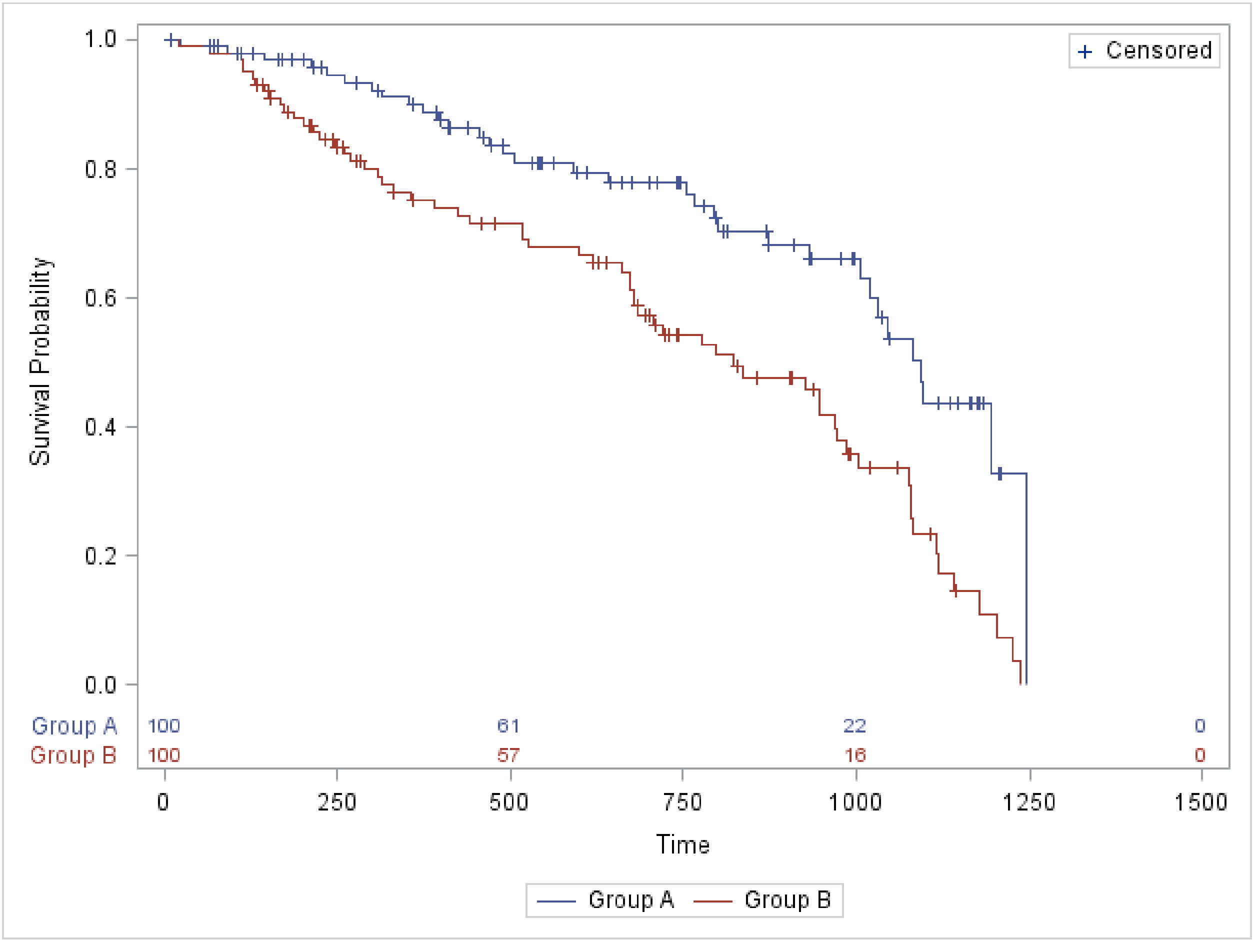

Figure 6. Displaying Kaplan Meier curves produced with an atrisk table using PROC SGPLOT.

Certainly if this was all that was needed, outputting the LIFETEST plot to a SGPLOT would be an unnecessary step. However, altering the output dataset for a particular need and incorporating that into the plot is where this becomes beneficial. By creating the cumulative number of events, as well as determining the median survival time, the plot is immediately improved by just a couple of changes to the SGPLOT code:

```
proc sgplot data=COMBINED;
      step x=time y=survival / group=stratumnum name='s';
      scatter x=time y=censored / markerattrs=(symbol=plus) name='c';
      scatter x=time y=censored / markerattrs=(symbol=plus)
      GROUP=stratum;
      xaxistable ATRISK_EVENT / x=tatrisk class=stratumnum
      colorgroup=stratumnum TITLE = "Number at Risk (Events)"
      location=inside;
      yaxistable yhelp/ y=SURVProb1 class=stratumnum nomissingchar
      classorder=descending colorgroup=stratumnum  TITLE = "Median
      Survival Time";
      keylegend 'c' / location=inside position=topright;
      keylegend 's' / linelength=20;
      format stratumnum groupI.;
run;
```

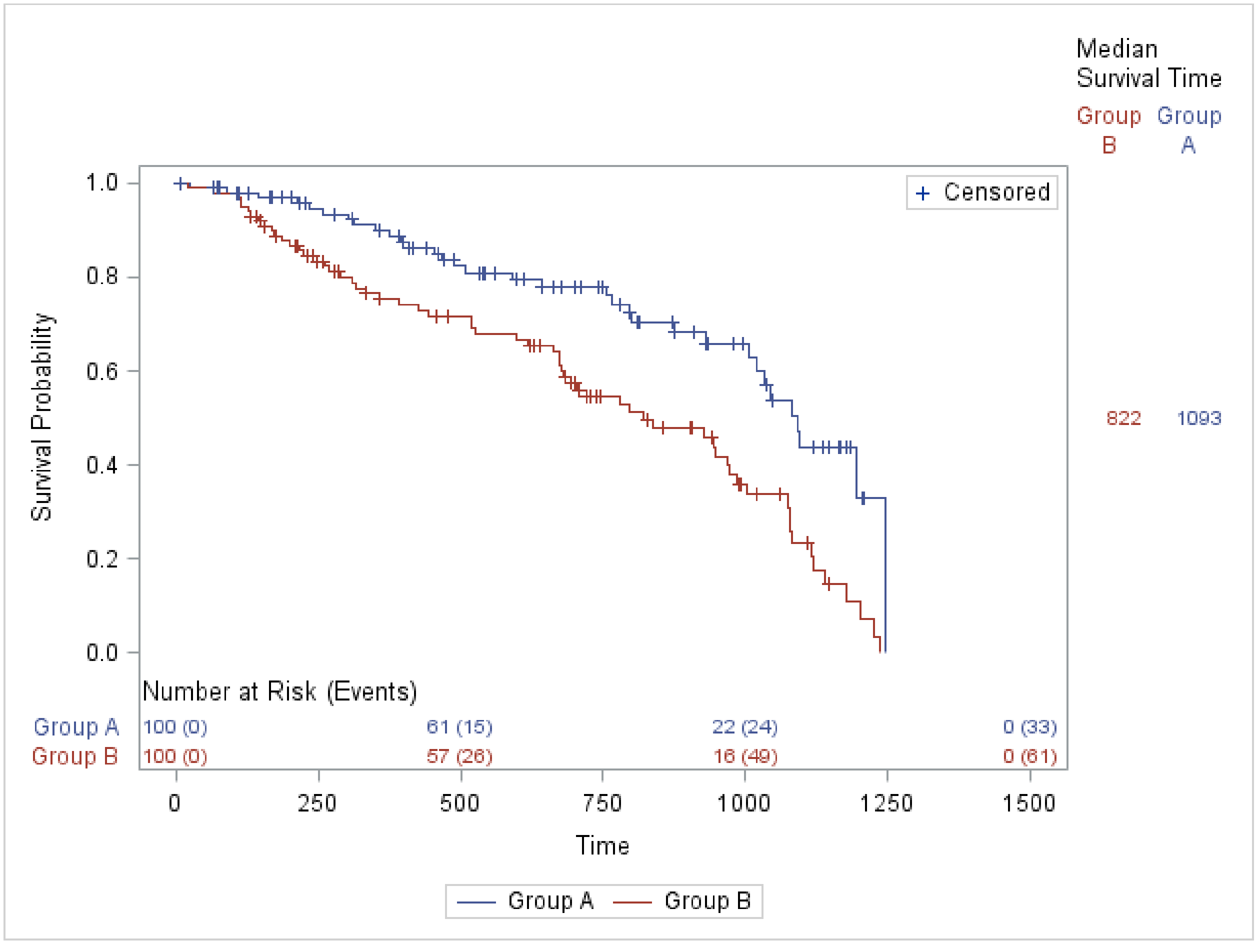

Figure 7. Displaying Kaplan Meier curves produced with an atrisk table and median survival times by group.

Figure 7 provides much more information than is originally afforded in the LIFETEST procedure. Not only are the cumulative number of events shown in the graph, the user does not have to guess at the median survival time.

CONCLUSION

Ultimately, the XAXISTABLE and YAXISTABLE statements are tools to add to an ever growing belt. Their use allows for additional information to be incorporated into a graph, and their practicality has been demonstrated in this paper. They provide options similar to that of many other SGPLOT statements which should be easily transferable. As demonstrated, a very beneficial use of these statements is survival plot customization. The task of creating an at risk table is useful. But, this application could extend to other metrics such as hazard ratios from a corresponding cox model. A key factor in quality visuals is to allow for all information to be displayed in an entirely interpretable manner for a first time observer. The XAXISTABLE and YAXISTABLE statements allow this factor to be accomplished in only a few lines of code.