Paper SP14–SAS-2014

Creating and Customizing the Kaplan-Meier Survival Plot in PROC LIFETEST in the SAS/STAT® 13.1 Release

Warren F. Kuhfeld and Ying So, SAS Institute Inc.

ABSTRACT

If you are a medical, pharmaceutical, or life sciences researcher, you have probably analyzed time-to-event data (survival data). The LIFETEST procedure computes Kaplan-Meier estimates of the survivor functions and compares survival curves between groups of patients. You can use the Kaplan-Meier plot to display the number of subjects at risk, confidence limits, equal-precision bands, Hall-Wellner bands, and homogeneity test p-value. You can control the contents of the survival plot by specifying procedure options in PROC LIFETEST. When the procedure options are insufficient, you can modify the graph templates by using SAS macros. PROC LIFETEST in SAS/STAT® 13.1 provides many new options for Kaplan-Meier plot modification, and the macros have been completely redone in this release in order to provide more power and flexibility than was available in previous releases. This paper provides examples of these new capabilities.

INTRODUCTION

Data that measure lifetime or the length of time until the occurrence of an event are called *survival* data. Survival data are often medical data; examples include the survival time for heart transplant or cancer patients. Survival time is a measure of the duration of time until a specified event (such as relapse or death) occurs. Survival data consist of survival time and possibly a set of independent variables that are thought to be associated with the survival time variable. The system that gives rise to the event of interest can be biological (as with most medical data) or physical (as with engineering data). Survival analysis estimates the underlying distribution of the survival time variable and assesses the dependence of the survival time variable on the independent variables.

Standard data analysis methods are not appropriate for survival data. In general, survival times are positively skewed, and it is not reasonable to assume that data of this type have a normal distribution. Furthermore, survival times are often censored. The survival time of an individual is right-censored when the event of interest has not been observed for that individual. For example, a patient who is recruited for a clinical trial drops out of the trial or the event is not observed when the period of data collection ends. In either case, the observed time is less than the true survival time. Analysis of survival data must take censoring into account and correctly use both the censored observations and the uncensored observations.

The LIFETEST procedure in SAS/STAT is a nonparametric procedure for analyzing survival data. You can use PROC LIFETEST to compute the Kaplan-Meier (1958) curve, which is a nonparametric maximum likelihood estimate of the survivor function. You can display the Kaplan-Meier plot, which contains step functions that represent the Kaplan-Meier curves of different samples. You can also use PROC LIFETEST to compare the survivor functions of different samples by the log-rank test.

The data that are used in this paper come from 137 bone marrow transplant patients in a study by Klein and Moeschberger (1997) and are available in the BMT data set in the Sashelp library. At the time of transplant, each patient is classified in one of three risk categories: ALL (acute lymphoblastic leukemia), AML (acute myelocytic leukemia)–Low Risk, and AML–High Risk. The endpoint of interest is the disease-free survival time, which is the time in days until death, relapse, or the end of the study. The variable **Group** represents the patient's risk category, the variable **T** represents the disease-free survival time, and the variable **Status** is the censoring indicator. A status of 1 indicates an event time, and a status of 0 indicates a censored time.

All examples use the 13.1 release of SAS/STAT software from 2013. Three types of examples are provided: specifying procedure options, modifying graph templates, and modifying style templates.

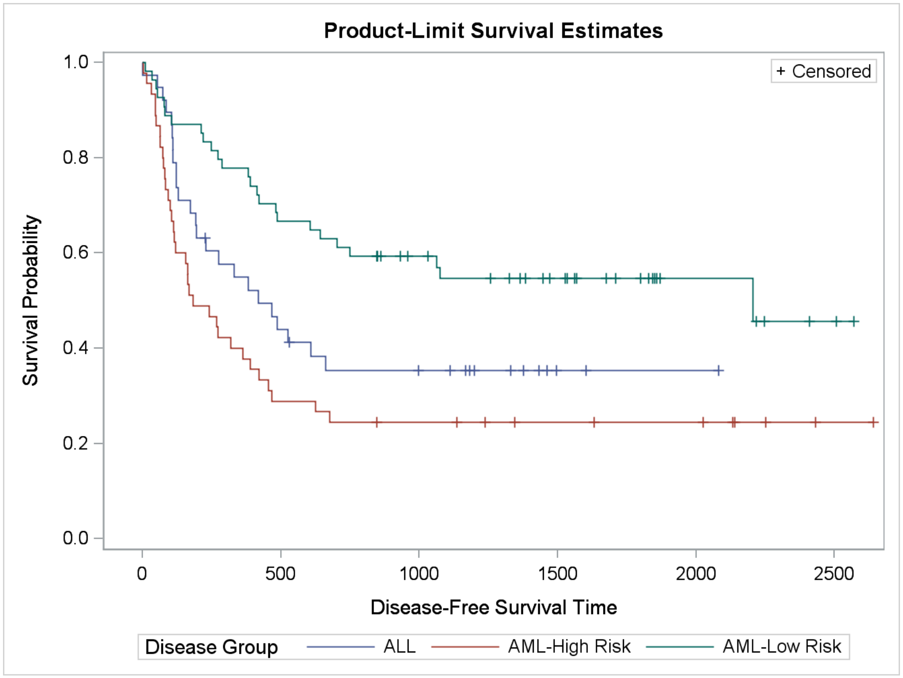
CONTROLLING THE SURVIVAL PLOT BY SPECIFYING PROCEDURE OPTIONS

You can use the following statements to enable ODS Graphics and run PROC LIFETEST:

```
ods graphics on;

proc lifetest data=sashelp.BMT;
   time T * Status(0);
   strata Group;
run;
```

The results, displayed in Figure 1, consist of three step functions, one for each of the three groups of patients. The plot shows that patients in the AML–Low Risk group have longer disease-free survival times than patients in the ALL and AML–High Risk groups.

Figure 1 Default Kaplan-Meier Plot



You specify in the TIME statement that the disease-free survival time is recorded in the variable **T**. You further specify that the variable **Status** indicates censoring and 0 indicates a censored time. Separate survivor functions are displayed for each group in the **Group** variable, which you specify in the STRATA statement.

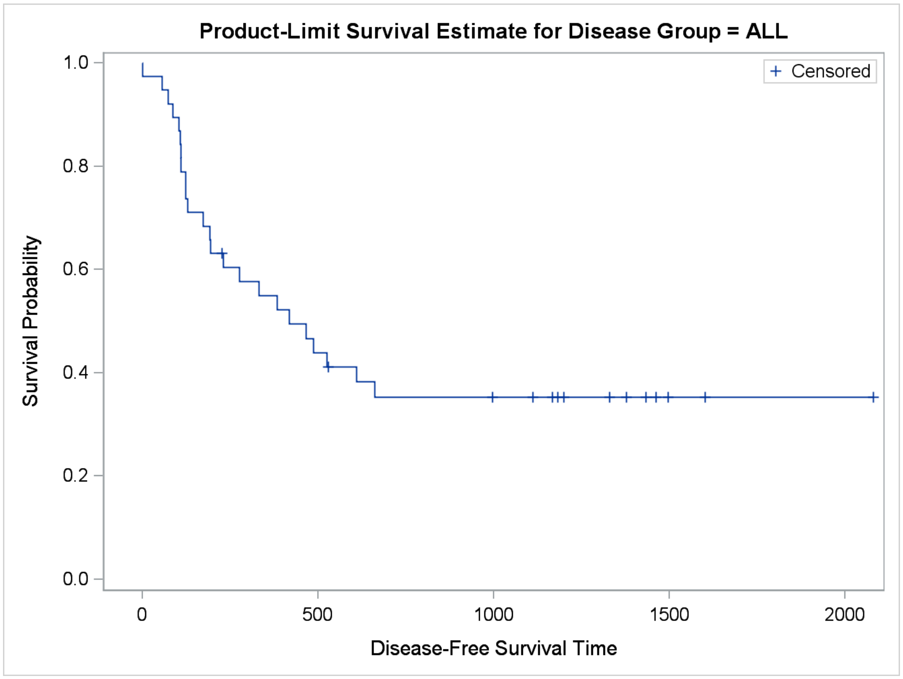
ODS Graphics is enabled for this step and all subsequent steps until it is disabled. (ODS Graphics remains enabled throughout the examples in this paper.) The survival plot is produced by default when ODS Graphics is enabled. You can add the option PLOTS=SURVIVAL to the PROC LIFETEST statement to explicitly request the SURVIVAL plot, but that is equivalent to the default. Usually, you use the PLOTS=SURVIVAL option when you want to specify plot-specific options. You can use the PLOTS= option to request nondefault graphs and specify options for some graphs. You can specify graph names (PLOTS=SURVIVAL), graph options (PLOTS=SURVIVAL(ATRISK OUTSIDE)), and suboptions (PLOTS=SURVIVAL(ATRISK OUTSIDE(0.15))).

You can use the STRATA=INDIVIDUAL option to request individual survival plots. By default, the STRATA=OVERLAY option produces the plot of overlaid step functions that is displayed in Figure 1. You can run the same analysis but request the results in three separate graphs, one per patient group, as follows:

```
proc lifetest data=sashelp.BMT plots=survival(strata=individual);
   time T * Status(0);
   strata Group;
run;
```

The first of the three survival plots is displayed in Figure 2. To conserve space, the other graphs are not displayed.

Figure 2 One of Three Individual Plots

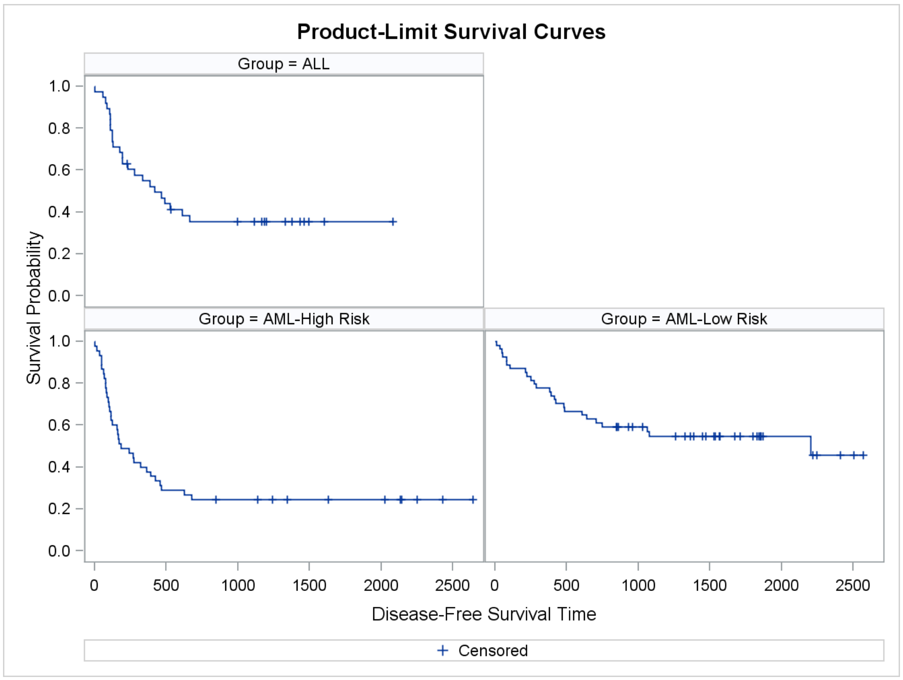


You can use the STRATA=PANEL option as follows to display the results in separate panels of a single graph:

```
proc lifetest data=sashelp.BMT plots=survival(strata=panel);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 3. The rest of this paper discusses overlaid plots such as the one displayed in Figure 1.

Figure 3 Individual Plots Displayed in a Panel

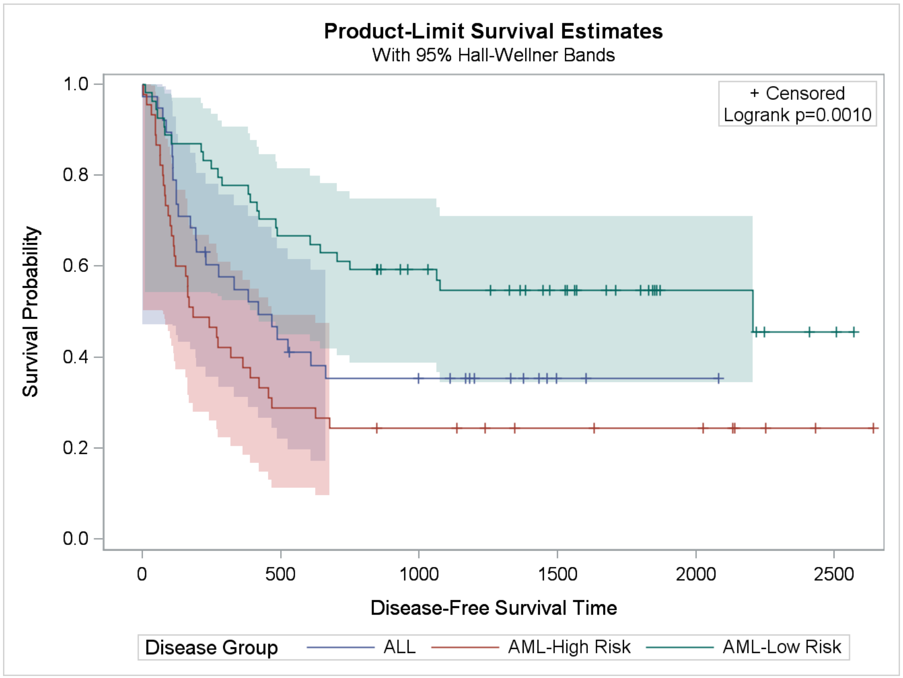


You can use the following statements to add Hall-Wellner confidence bands (Hall and Wellner 1980) to Figure 1 and display the p-value from a test that the strata are homogeneous:

```
proc lifetest data=sashelp.BMT plots=survival(cb=hw test);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 4. The Hall-Wellner confidence bands extend to the last event times. The small p-value supports rejecting the hypothesis that the groups are homogeneous.

Figure 4 Confidence Bands and Homogeneity Test

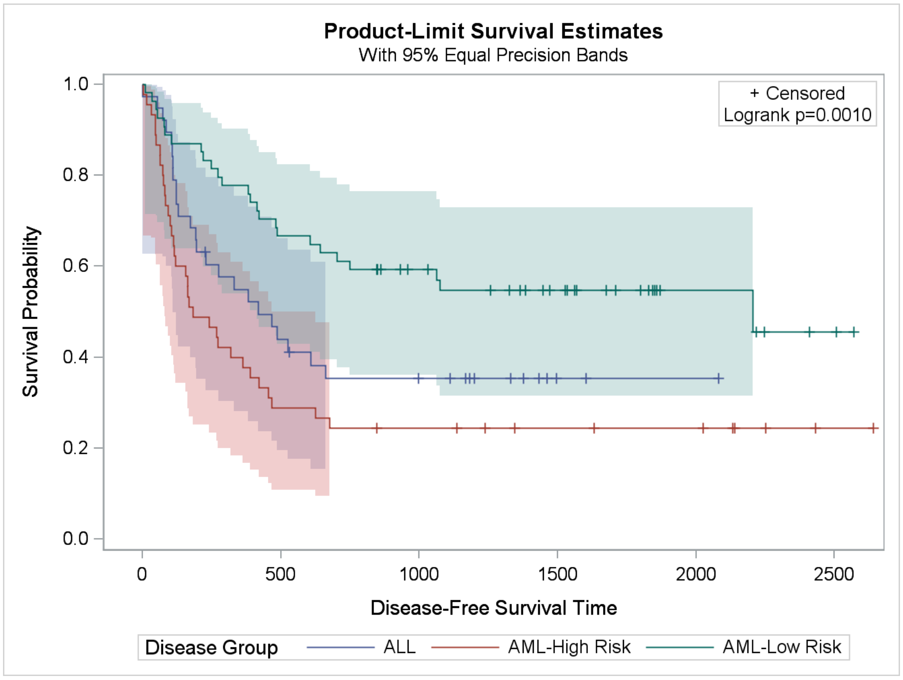


You can use the following statements to add equal-precision bands to the plot:

```
proc lifetest data=sashelp.BMT plots=survival(cb=ep test);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 5.

Figure 5 Equal-Precision Bands

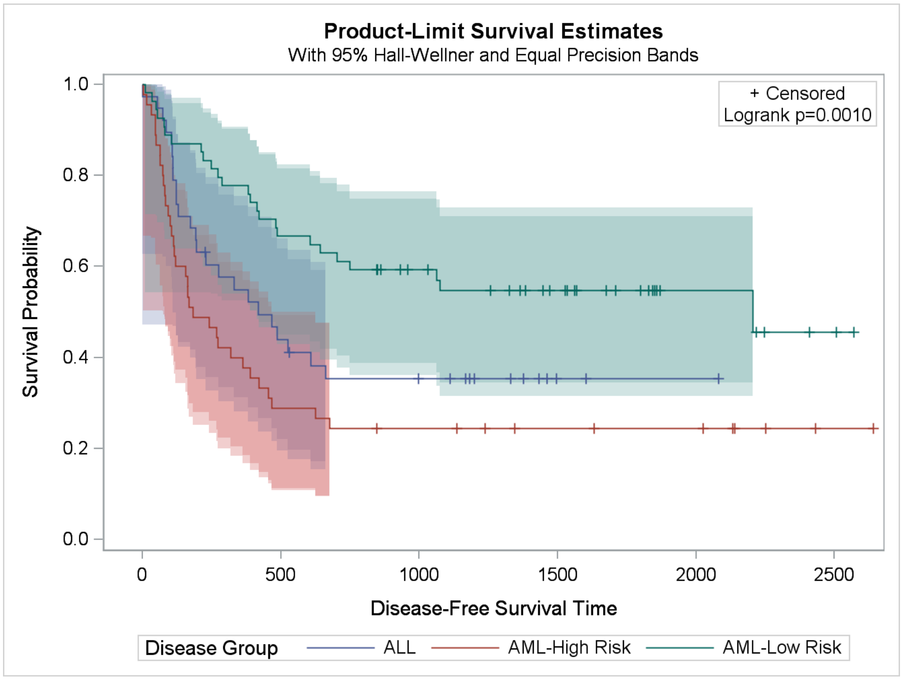


You can use the following statements to add both Hall-Wellner and equal-precision bands to the plot:

```
proc lifetest data=sashelp.BMT plots=survival(cb=all test);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 6.

Figure 6 Hall-Wellner and Equal-Precision Bands

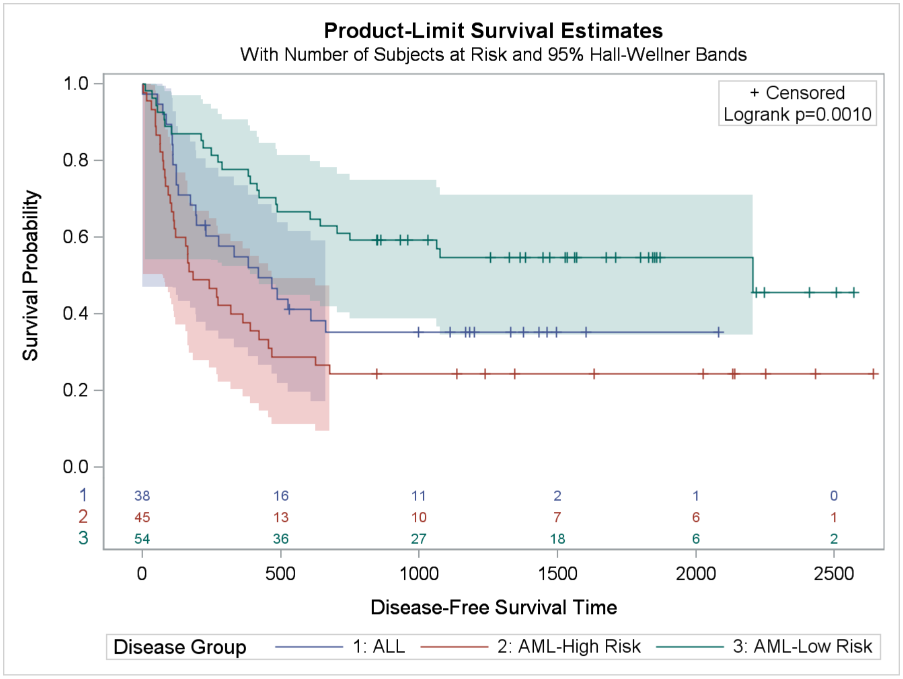


You can display the patients-at-risk table in the Kaplan-Meier plot as follows:

```
proc lifetest data=sashelp.BMT plots=survival(cb=hw test atrisk);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 7. By default, the at-risk table is displayed inside the body of the plot. This table shows the number of patients in each group who are at risk for each of the different times. For these data, the default survival times at which at-risk values are displayed are 0 to 2500 by 500. You will see how to specify other values in subsequent examples.

Figure 7 At-Risk Table inside the Plot

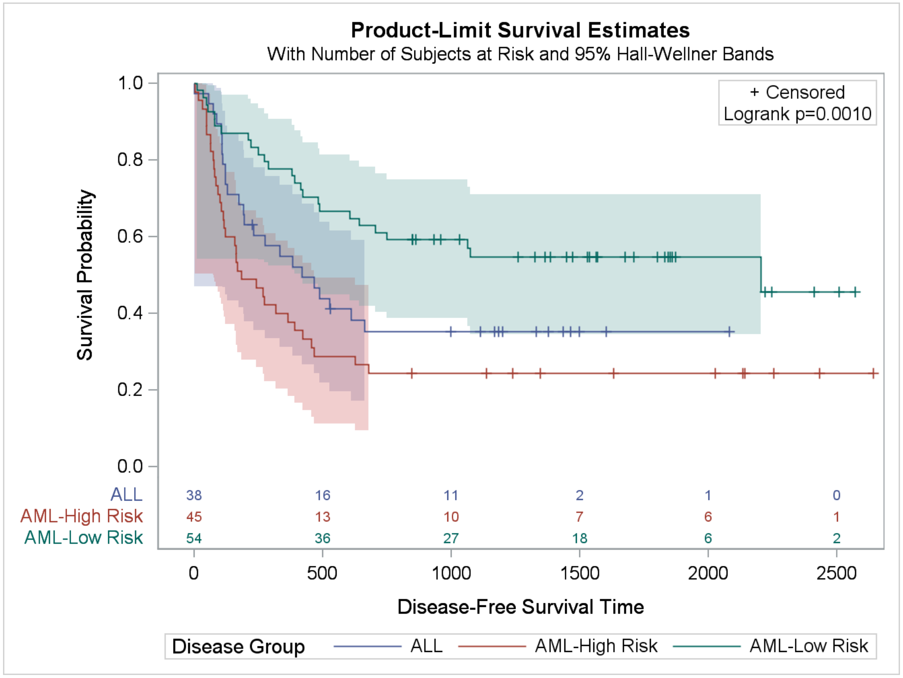


The group labels for the at-risk table are group numbers, and these numbers appear in the legend. Numbers are used rather than the actual labels because the length of the longest label (13) is greater than the default that is set by the maximum label length option (MAXLEN=12). You can display labels rather than the group numbers by specifying a MAXLEN= value equal to the maximum group label length as follows:

```
proc lifetest data=sashelp.BMT plots=survival(cb=hw test atrisk(maxlen=13));
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 8. The legend entries and the order of the rows in the at-risk table correspond to the sort order of the values of the **Group** variable.

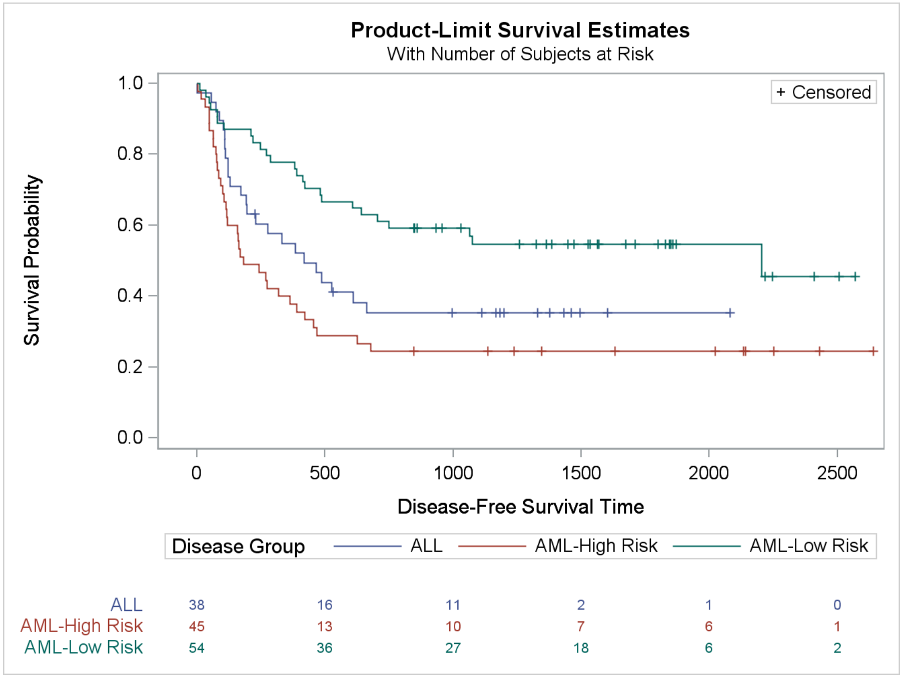
Figure 8 At-Risk Table with Labels



You can use the PLOTS=SURVIVAL(OUTSIDE) option to display the at-risk table outside the body of the plot. The option OUTSIDE(0.15) reserves 15% of the vertical graph window for the at-risk table. This example illustrates that the PLOTS= option has options nested within options and options nested within those nested options. The following step produces the plot in Figure 9:

```
proc lifetest data=sashelp.BMT
              plots=survival(atrisk(maxlen=13 outside(0.15)));
   time T * Status(0);
   strata Group;
run;
```

Figure 9 At-Risk Table outside the Plot

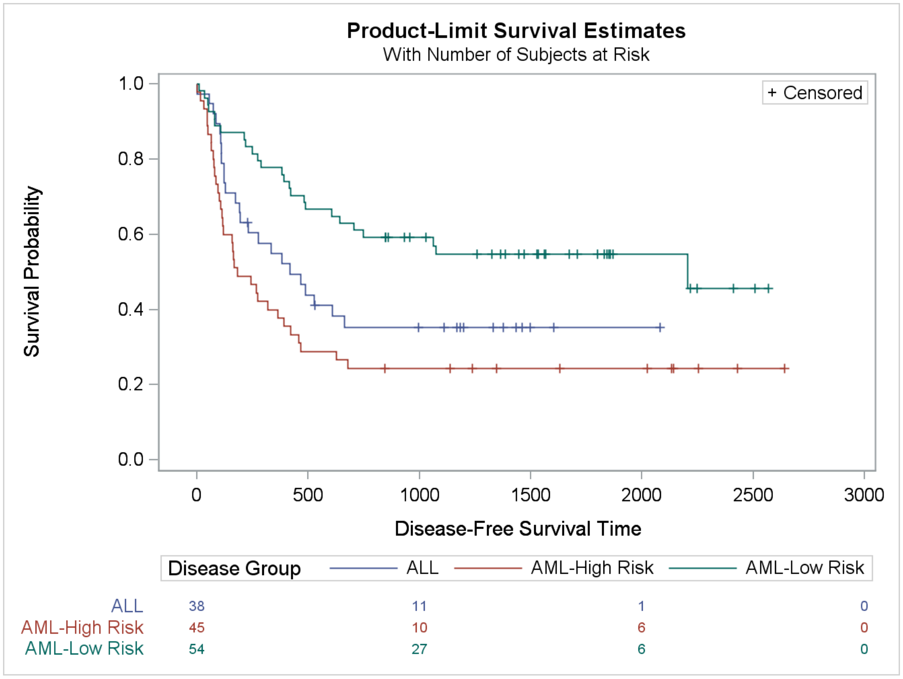


The following step explicitly controls the time values at which the at-risk values are displayed by using the PLOTS=SURVIVAL(ATRISK=0 TO 3000 BY 1000) option:

```
proc lifetest data=sashelp.BMT
              plots=survival(atrisk(maxlen=13 outside)=0 to 3000 by 1000);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 10.

Figure 10 Specifying At-Risk Values

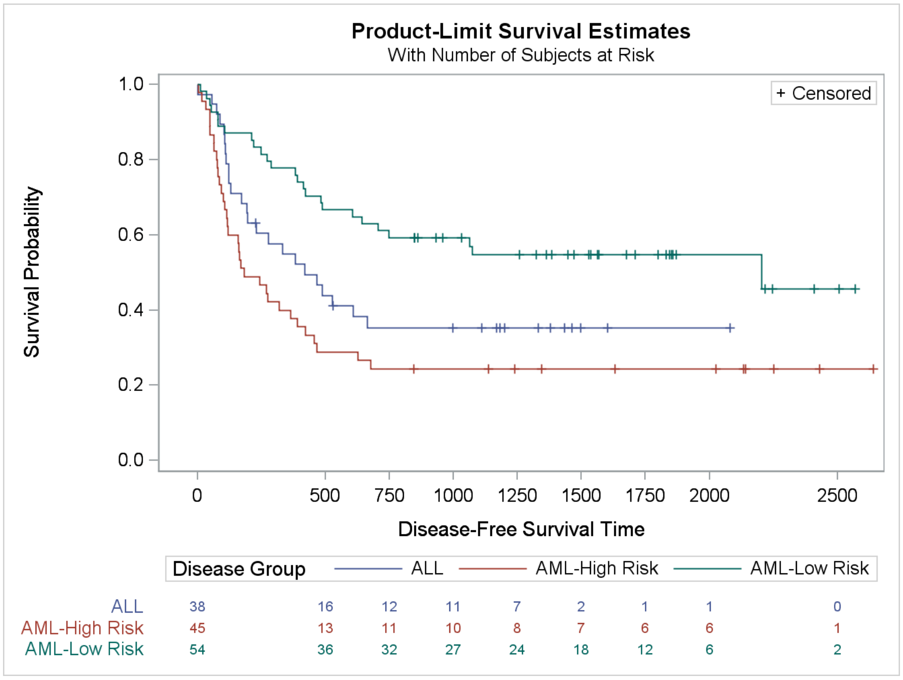


You can specify at-risk values that do not correspond to the original time axis tick marks. You can use the PLOTS=SURVIVAL(ATRISK(ATRISKTICK)) option to add tick marks that correspond to the specified at-risk values:

```
proc lifetest data=sashelp.BMT plots=survival(atrisk
   (atrisktick maxlen=13 outside)=0 500 750 1000 1250 1500 1750 2000 2500);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 11.

Figure 11 Adding At-Risk Tick Marks

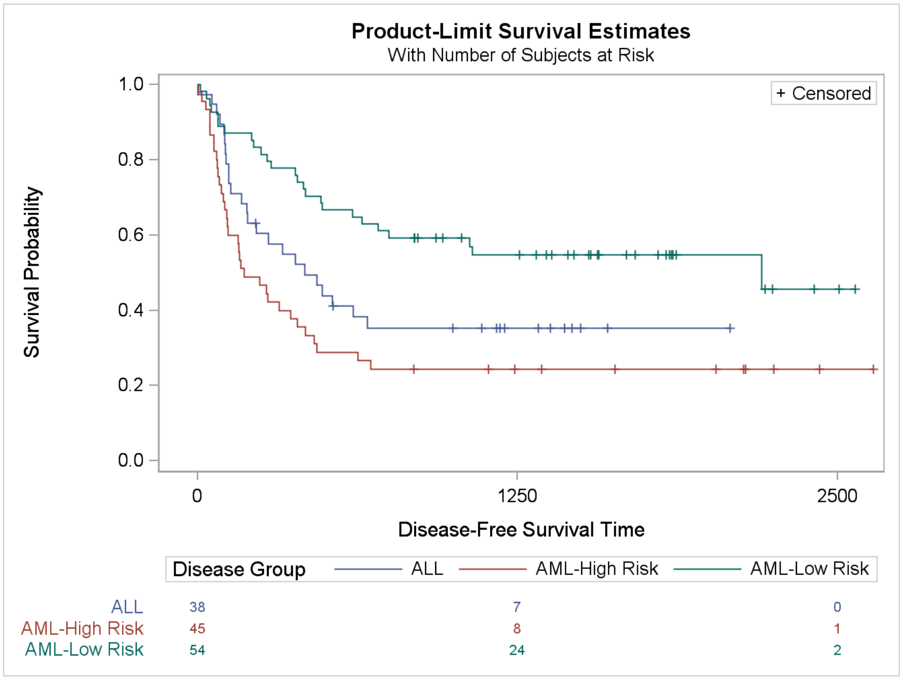


You can display tick values only at those times that are given in the ATRISK= list:

```
proc lifetest data=sashelp.BMT plots=survival(atrisk
      (atrisktickonly maxlen=13 outside)=0 1250 2500);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 12.

Figure 12 Limiting At-Risk Tick Marks

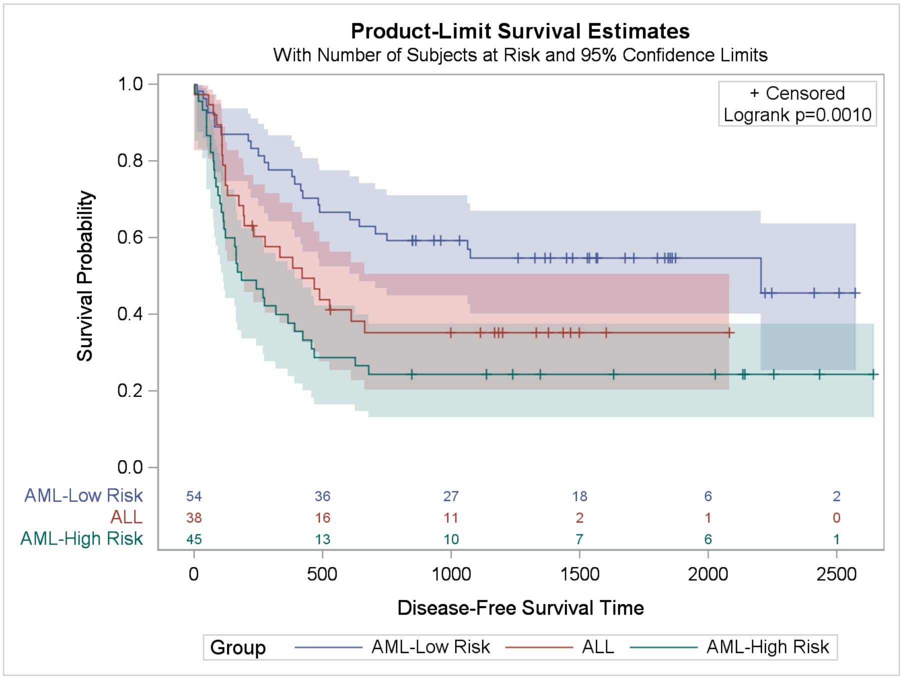


You can change the order of the legend entries by first changing each original group value to a new value in the desired order and then running the analysis by using a FORMAT statement to provide the original values. In this example, the order is changed to AML–Low Risk (the top curve), followed by ALL (the middle curve), followed by AML–High Risk. With this ordering, there is a clearer correspondence between the curves, the at-risk table, and the legend. The following steps illustrate this reordering:

```
proc format;
   invalue bmtnum   'AML-Low Risk' = 1  'ALL' = 2  'AML-High Risk' = 3;
   value   bmtfmt   1 = 'AML-Low Risk'  2 = 'ALL'  3 = 'AML-High Risk';
run;

data BMT(drop=g);
   set sashelp.BMT(rename=(group=g));
   Group = input(g, bmtnum.);
run;

proc lifetest data=BMT plots=survival(cl test atrisk(maxlen=13));
   time T * Status(0);
   strata Group / order=internal;
   format Group bmtfmt.;
run;
```

The PROC FORMAT step has two statements. The INVALUE statement creates an informat that maps the values of the original **Group** variable into integers that have the correct order. The VALUE statement creates a format that maps the integers back to the original values. The informat is used along with the INPUT function in the DATA step to create a new integer **Group** variable. The FORMAT statement assigns the BMTFMT format to the **Group** variable so that the actual risk groups are displayed in the analysis. You specify the ORDER=INTERNAL option in the STRATA statement to sort the **Group** values based on internal order (the order specified by the integers, which are the internal unformatted values). This example also illustrates the CL option, which displays pointwise confidence limits for the survival curve (instead of the Hall-Wellner confidence bands). The results are displayed in Figure 13.

Figure 13 At-Risk Table Row Order Matches Profiles

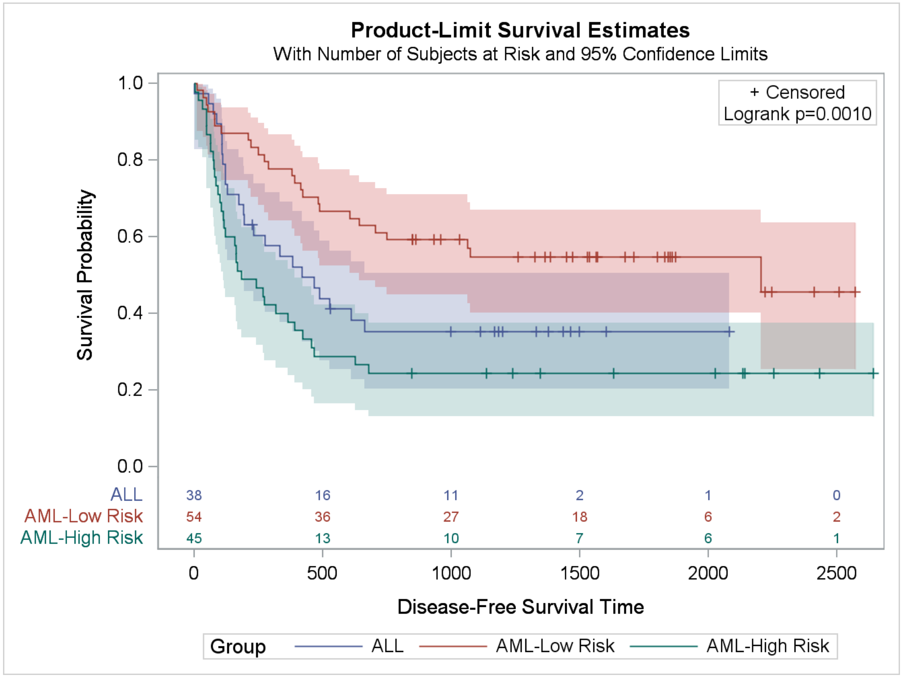


You can submit the following steps to display ALL first, followed by AML–Low Risk and then AML–High Risk:

```
proc format;
   invalue bmtnum 'ALL' = 1  'AML-Low Risk' = 2  'AML-High Risk' = 3;
   value   bmtfmt 1 = 'ALL'  2 = 'AML-Low Risk'  3 = 'AML-High Risk';
run;

data BMT(drop=g);
   set sashelp.BMT(rename=(group=g));
   Group = input(g, bmtnum.);
run;

proc lifetest data=BMT plots=survival(cl test atrisk(maxlen=13));
   time T * Status(0);
   strata Group / order=internal;
   format Group bmtfmt.;
run;
```

The results are displayed in Figure 14.

Figure 14 Controlling Legend Order

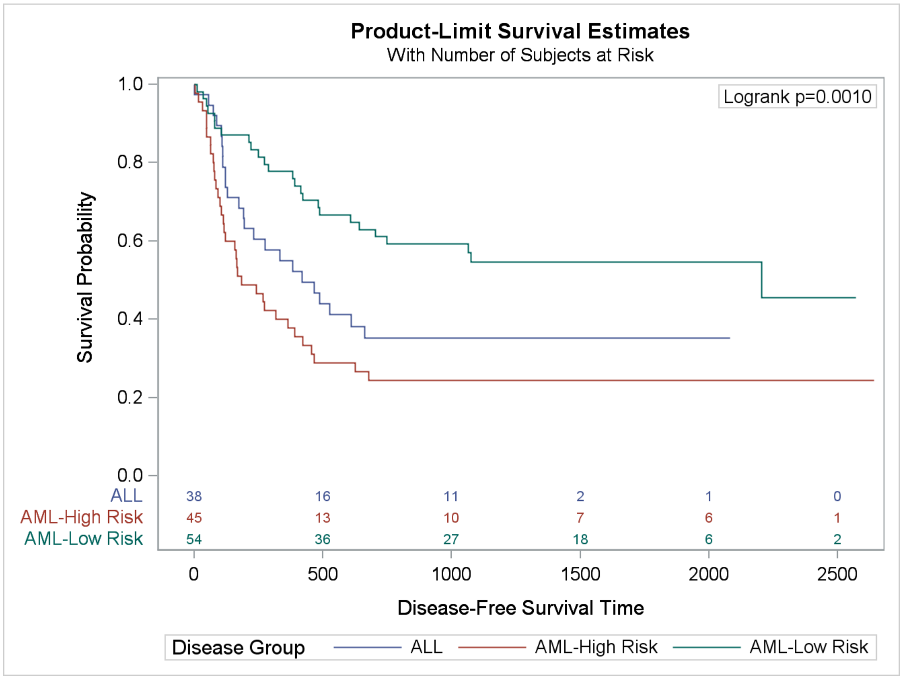


You can use the PLOTS=SURVIVAL(NOCENSOR) option to suppress the display of censored observations as follows:

```
proc lifetest data=sashelp.BMT
              plots=survival(nocensor test atrisk(maxlen=13));
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 15.

Figure 15 Censored Values Not Displayed

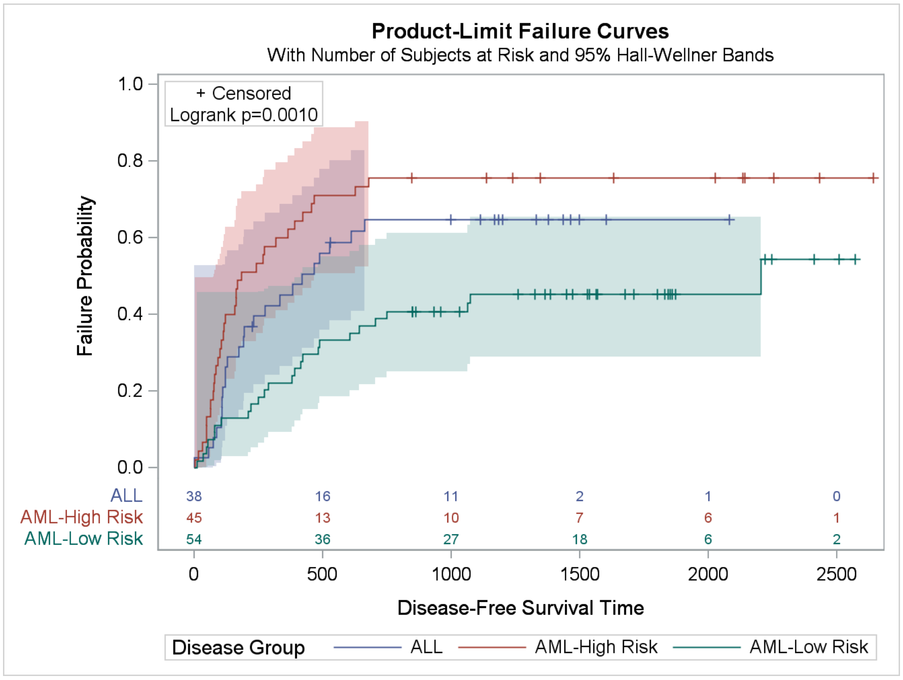


All the discussion up to this point has been about survival plots. You can instead plot failure probabilities by using the PLOTS=SURVIVAL(FAILURE) option as follows:

```
proc lifetest data=sashelp.BMT
     plots=survival(cb=hw failure test atrisk(maxlen=13));
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 16.

Figure 16 Failure Plot



CONTROLLING THE SURVIVAL PLOT BY MODIFYING GRAPH TEMPLATES

The preceding section illustrates the PLOTS= options for controlling the survival plot. If you need to make modifications that are not shown there, this section shows how to modify the survival plot by using macros and macro variables to modify graph templates. Before you modify anything, you must submit the %ProvideSurvivalMacros macro definition from the sample library to SAS. You can both store the macros in a temporary file and submit them to SAS by submitting the following statements:

```
data _null_;
   %let url = //support.sas.com/documentation/onlinedoc/stat/ex_code/131;
   infile "http:&url/templft.html" device=url;
   file 'macros.tmp';
   input;
   if index(_infile_, '</pre>') then stop;
   if pre then put _infile_;
   if index(_infile_, '<pre>')  then pre + 1;
run;

%inc 'macros.tmp' / nosource;
```

You begin by running the following macro:

```
%ProvideSurvivalMacros
```

Running this macro provides the default macros and macro variables (or restores them if you have previously submitted the %ProvideSurvivalMacros macro).[1] The %ProvideSurvivalMacros macro also runs the %CompileSurvivalTemplates macro and hence replaces any compiled survival plot templates that you might have created in the past. You can recompile the templates by submitting the following macro:

```
%CompileSurvivalTemplates
```

This macro runs PROC TEMPLATE and compiles the templates from all the macros and macro variables in the %ProvideSurvivalMacros macro along with any that you modified. Running this macro produces two compiled templates that are stored in a special SAS data file called an item store. Assuming that you have not modified your ODS path by using an ODS PATH statement, compiled templates are stored in an item store in the Sasuser library. Files in the Sasuser library persist across SAS sessions until they are deleted. When you are done with a modified template, it is wise to clean up all remnants of it by restoring the default macros and by deleting the modified templates from the Sasuser template item store. You can delete the modified templates (so that SAS can find only the original templates) by running the following step:

```
proc template;
   delete Stat.Lifetest.Graphics.ProductLimitSurvival  / store=sasuser.templat;
   delete Stat.Lifetest.Graphics.ProductLimitSurvival2 / store=sasuser.templat;
run;
```

This step deletes the compiled templates from the item store Sasuser.Templat. You can omit the STORE= option if you are using the default ODS path, but it is good practice to explicitly control which templates are deleted. Deleting the compiled templates does not change any of the macros or macro variables. Only the compiled templates (not the macros or macro variables) affect the graph when you run PROC LIFETEST.

Here is a simple, complete program (except for retrieving and including the %ProvideSurvivalMacros macro from the sample library) with setup, macro variable modifications to change the title, and cleanup:

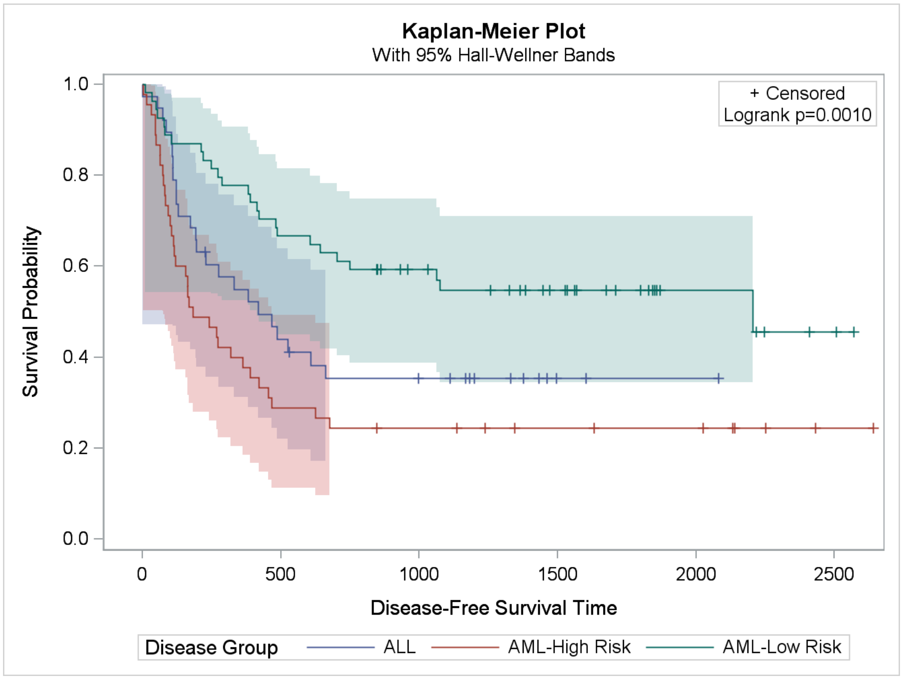
```
/*-- Original Macro Variable Definitions ----------------------------------
%let TitleText0 = METHOD " Survival Estimate";
%let TitleText1 = &titletext0 " for " STRATUMID;
%let TitleText2 = &titletext0 "s";
-------------------------------------------------------------------------*/

                                               /* Make the macros and macro      */
%ProvideSurvivalMacros                         /* variables available.           */
```

[1]Semicolons are not needed after a macro call like this one, so they are not used in these examples.

```
%let TitleText0 = "Kaplan-Meier Plot";     /* Change the title.               */
%let TitleText1 = &titletext0 " for " STRATUMID;
%let TitleText2 = &titletext0;

%CompileSurvivalTemplates                   /* Compile the templates with      */
                                            /* the new title.                  */

proc lifetest data=sashelp.BMT              /* Perform the analysis and make   */
              plots=survival(cb=hw test);   /* the graph.                      */
   time T * Status(0);
   strata Group;
run;

%ProvideSurvivalMacros                      /* Optionally restore the default */
                                            /* macros and macro variables.     */

proc template;                              /* Delete the modified templates. */
   delete Stat.Lifetest.Graphics.ProductLimitSurvival  / store=sasuser.templat;
   delete Stat.Lifetest.Graphics.ProductLimitSurvival2 / store=sasuser.templat;
run;
```

The results are displayed in Figure 17. You can see that the graph title is now "Kaplan-Meier Plot".

Figure 17 Kaplan-Meier Plot Title Modification



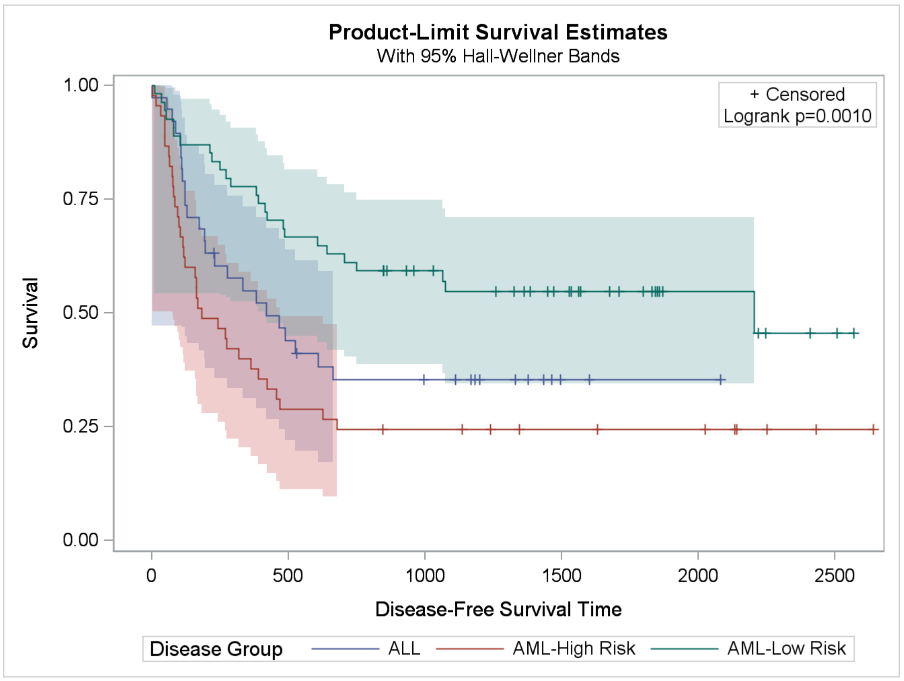
There are multiple title macro variables because two different types of plots are defined in the survival plot templates. The first macro variable, **TitleText0**, contains the text that is the same for both types of plots. The second macro variable, **TitleText1**, contains the title for the single-stratum case. The third macro variable, **TitleText2**, contains the title for the multiple-strata case. Both **TitleText1** and **TitleText2** use the common text that is defined in **TitleText0**. Both **TitleText0** and **TitleText2** were changed from their original definition; the definition of **TitleText1** was copied from the %ProvideSurvivalMacros macro. You must provide all relevant %LET statements when you modify **TitleText0**. In this case it is **TitleText0** and **TitleText2**, but it is easy to copy all three and then just modify what you need. Alternatively, when you know the number of strata, you can modify only **TitleText1** or **TitleText2**.

The following statements modify the default tick value list for the Y axis from the default increment of 0.2 to have an increment of 0.25 and also change the Y-axis label to “Survival”:

```
/*-- Original Macro Variable Definitions ---------------------------------
%let yOptions   = label="Survival Probability" shortlabel="Survival"
                  linearopts=(viewmin=0 viewmax=1
                              tickvaluelist=(0 .2 .4 .6 .8 1.0));
--------------------------------------------------------------------------*/

%ProvideSurvivalMacros

%let yOptions = label="Survival"
                linearopts=(viewmin=0 viewmax=1
                            tickvaluelist=(0 .25 .5 .75 1));

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT plots=survival(cb=hw test);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 18.

Figure 18 Y-Axis Modification

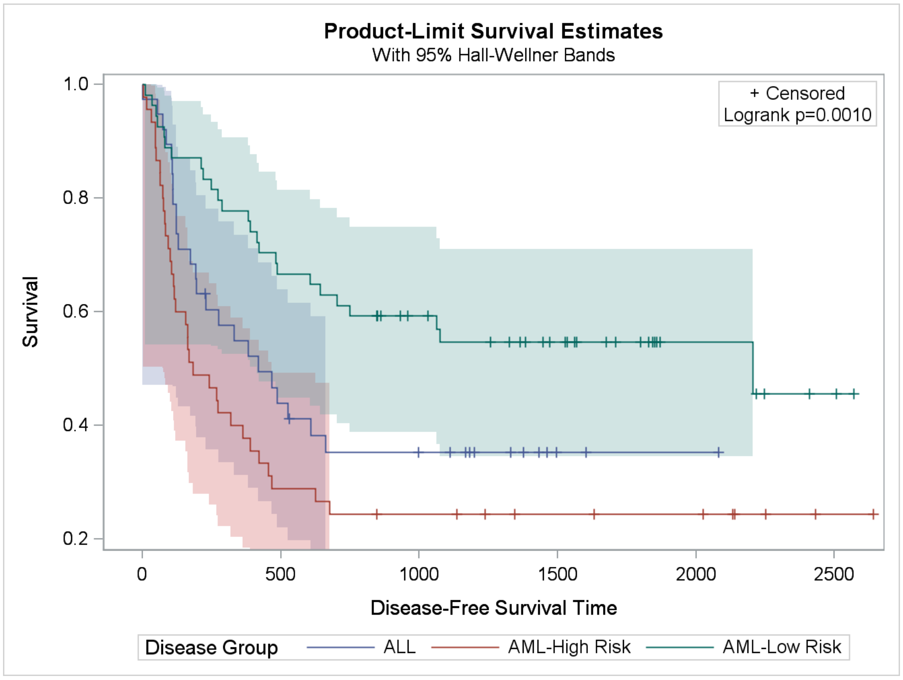


The following statements modify the Y axis so that tick marks start at 0.2:

```
%ProvideSurvivalMacros

%let yOptions = label="Survival"
                linearopts=(viewmin=0.2 viewmax=1
                            tickvaluelist=(0 .2 .4 .6 .8 1.0));

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT plots=survival(cb=hw test);
   time T * Status(0);
   strata Group;
run;
```

You only need to change the value of the VIEWMIN= option, in this case from 0 to 0.2. You do not need to modify the tick value list. The VIEWMIN= option (not the tick value list) controls the smallest value shown on the axis. The results are displayed in Figure 19.

Figure 19 Y Axis, First Tick Change

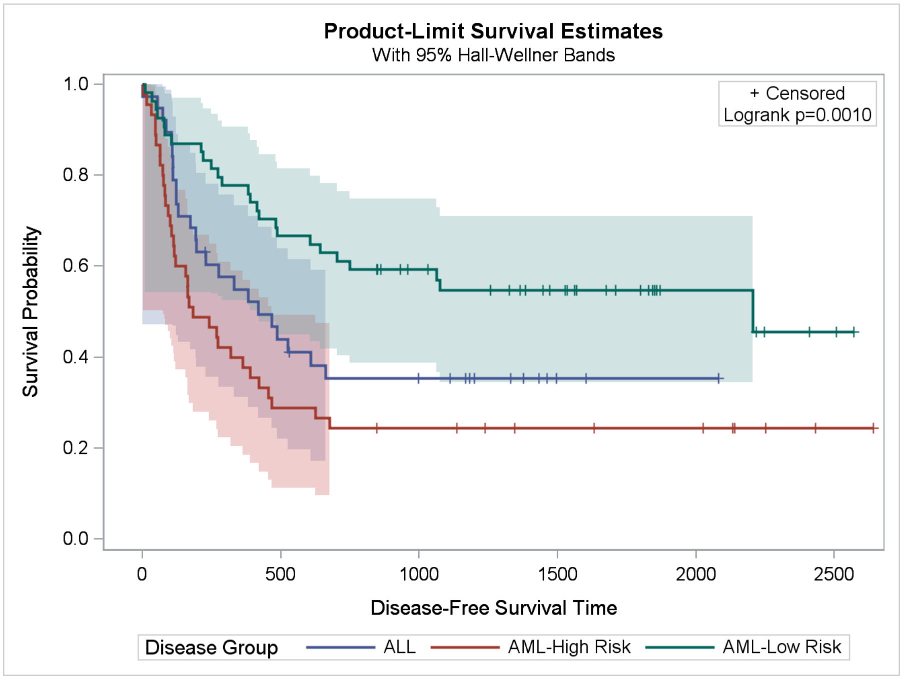


The following steps modify the line thickness for the step functions in the survival plot:

```
%ProvideSurvivalMacros

%let StepOpts = lineattrs=(thickness=2.5);

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT plots=survival(cb=hw test);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 20.

Figure 20 Changing Line Thickness



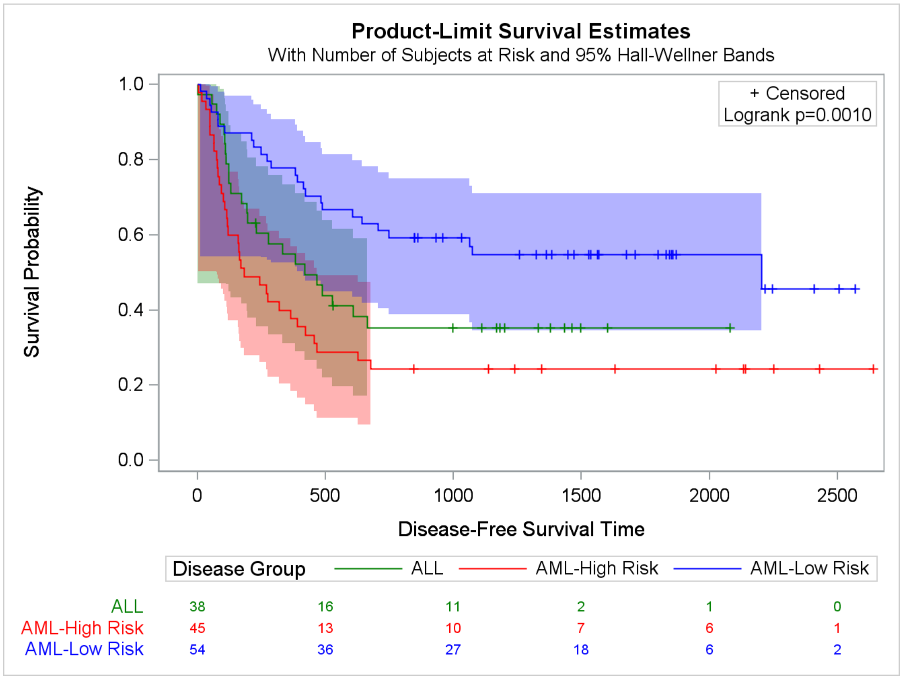
By default, the **StepOpts** macro variable is null.

SAS styles control the colors that are displayed in graphs. The style elements `GraphData1`, `GraphData2`, ..., `GraphData12` control the appearance of groups of observations, such as the groups of patients displayed in each step plot in the Kaplan-Meier plot. You can override these colors by using the **GraphOpts** macro variable (which is null by default). By default, the colors for the first three groups in the HMTLBlue style are shades of blue, red, and green. You can change them to a pure green, red, and blue as follows:

```
%ProvideSurvivalMacros

%let GraphOpts = DataContrastColors=(green red blue)
                 DataColors=(green red blue);

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT
              plots=survival(cb=hw test atrisk(outside maxlen=13));
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 21. The DATACONTRASTCOLORS= option specifies the contrast colors, which are used for markers and lines. The DATACOLORS= option specifies the colors, which are used for shaded areas such as confidence bands.

Figure 21 Named Color Specifications



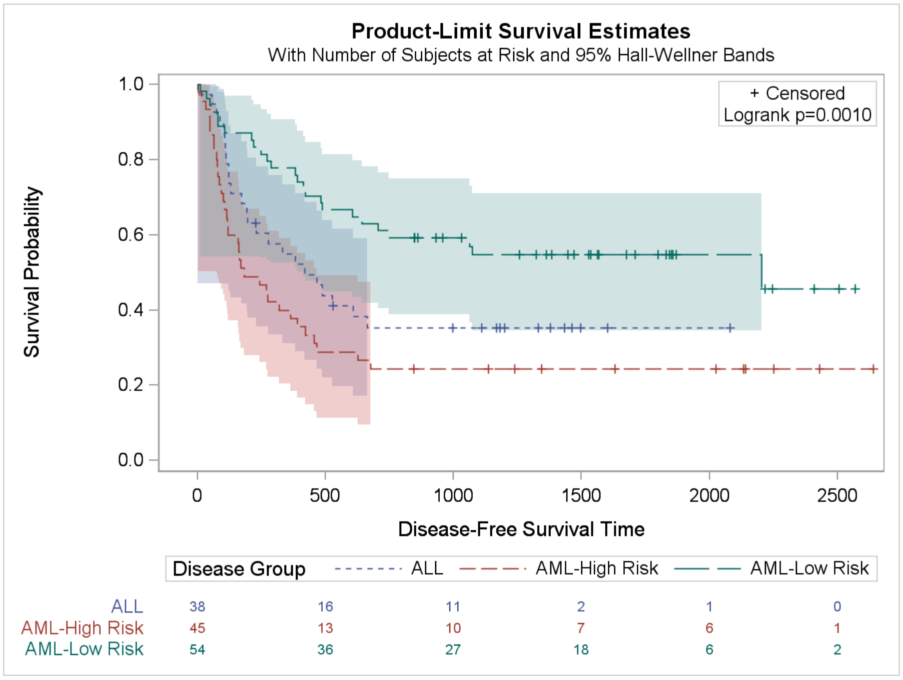
The original colors (as shown in Figure 22) are more subtle than those shown in Figure 21. If you want to change the order of the original colors by using this approach, then you need to know what they are so that you can specify them. The graph colors for the HTMLBlue and Statistical styles are extracted from the style and displayed in Figure 35. The %Reorder macro on page 33 shows you how to change a style template to specify the original colors in a different order (without having to extract and specify the color names).

You can change the line patterns as follows:

```
%ProvideSurvivalMacros

%let GraphOpts = attrpriority=none
                 DataLinePatterns=(ShortDash MediumDash LongDash);

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT
              plots=survival(cb=hw test atrisk(outside maxlen=13));
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 22.

Figure 22 Changing the Line Patterns



You must use the option ATTRPRIORITY=NONE when you want to have varying line patterns in an ATTRPRIORITY=COLOR style like HTMLBlue or Pearl. In an ATTRPRIORITY=COLOR style, groups are not distinguished by line patterns, and the line patterns for second and subsequent groups match the line pattern for the first group.

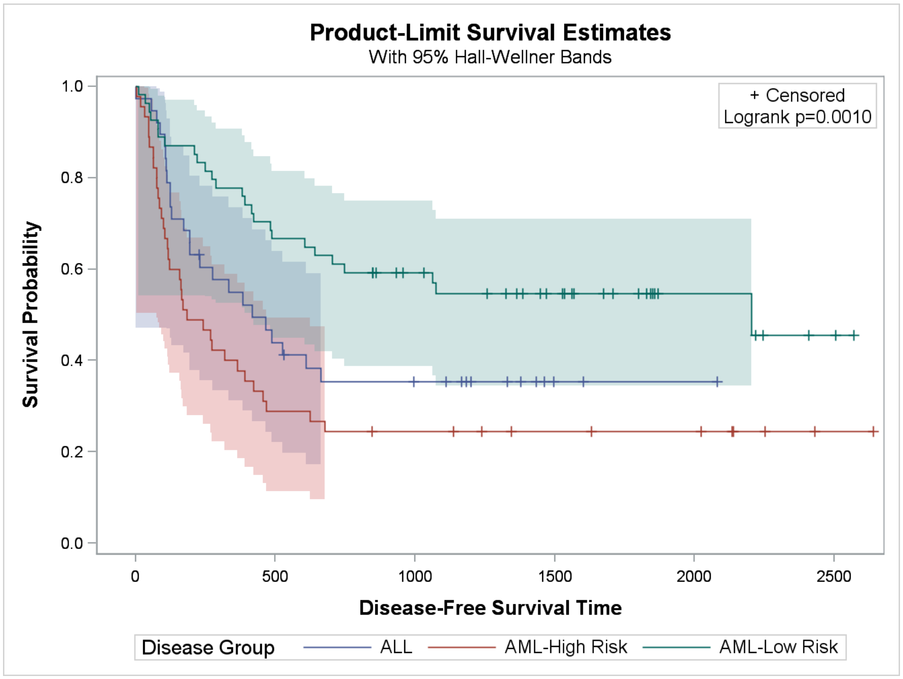
You can change the fonts for the Y-axis, X-axis, and title as follows:

```
%ProvideSurvivalMacros

/*-- Original Macro Variable Definitions ---------------------------------
%let TitleText0 = METHOD " Survival Estimate";
%let TitleText1 = &titletext0 " for " STRATUMID;
%let TitleText2 = &titletext0 "s";
%let yOptions   = label="Survival Probability"
                  shortlabel="Survival"
                  linearopts=(viewmin=0 viewmax=1
                              tickvaluelist=(0 .2 .4 .6 .8 1.0));
%let xOptions   = shortlabel=XNAME
                  offsetmin=.05
                  linearopts=(viewmax=MAXTIME tickvaluelist=XTICKVALS
                              tickvaluefitpolicy=XTICKVALFITPOL);
-------------------------------------------------------------------------*/

%let tatters    = textattrs=(size=12pt weight=bold family='arial');
%let TitleText0 = METHOD " Survival Estimate";
%let TitleText1 = &titletext0 " for " STRATUMID / &tatters;
%let TitleText2 = &titletext0 "s" / &tatters;

%let yOptions   = label="Survival Probability"
                  shortlabel="Survival"
                  labelattrs=(size=10pt weight=bold)
                  tickvalueattrs=(size=8pt)
                  linearopts=(viewmin=0 viewmax=1
                              tickvaluelist=(0 .2 .4 .6 .8 1.0));

%let xOptions   = shortlabel=XNAME
                  offsetmin=.05
```

```
                    labelattrs=(size=10pt weight=bold)
                    tickvalueattrs=(size=8pt)
                    linearopts=(viewmax=MAXTIME tickvaluelist=XTICKVALS
                                tickvaluefitpolicy=XTICKVALFITPOL);

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT plots=survival(cb=hw test);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 23.

Figure 23 Changing the Fonts



Font options include the following:

COLOR=*style-reference* | *color*
FAMILY=*style-reference* | *'string'*
SIZE=*style-reference* | *dimension*
STYLE=*style-reference* | **NORMAL** | **ITALIC**
WEIGHT=*style-reference* | **NORMAL** | **BOLD**

Fonts vary from installation to installation. Sample font strings include "Times New Roman", "Courier New", "Arial", and "Calibri". For more information about text and label attribute options, see *SAS Graph Template Language: Reference*.

This example shows you how to move the legend inside the plot (to the top right) and move the homogeneity test and censored value legend to the bottom right of the plot:

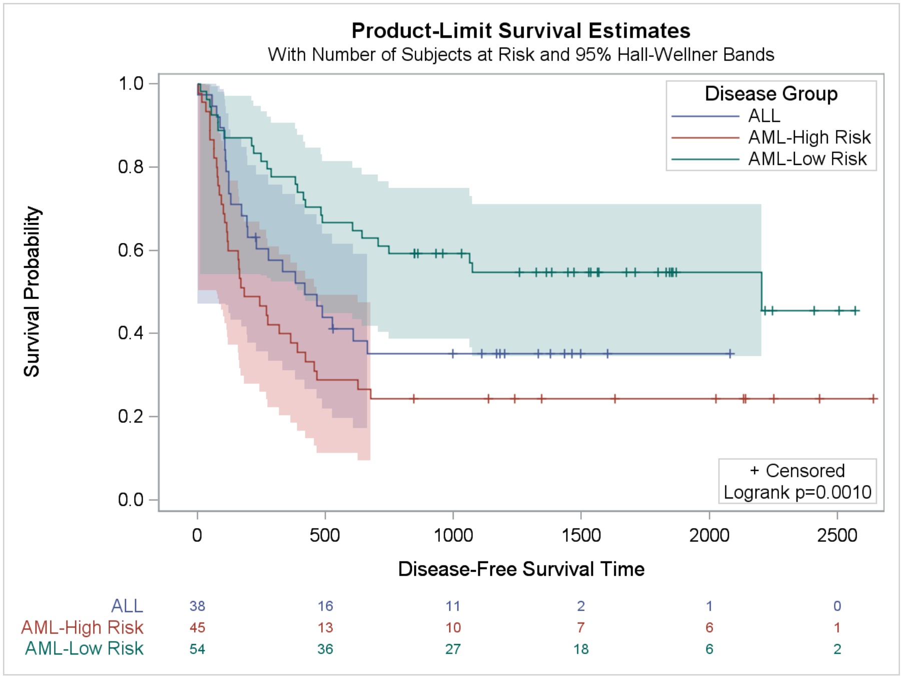
```
%ProvideSurvivalMacros

/*-- Original Macro Variable Definitions ----------------------------------
%let InsetOpts  = autoalign=(TOPRIGHT BOTTOMLEFT TOP BOTTOM)
                  border=true BackgroundColor=GraphWalls:Color Opaque=true;
%let LegendOpts = title=GROUPNAME location=outside;
-------------------------------------------------------------------------*/
```

```
%let InsetOpts   = autoalign=(BottomRight)
                   border=true BackgroundColor=GraphWalls:Color Opaque=true;

%let LegendOpts = title=GROUPNAME location=inside across=1 autoalign=(TopRight);

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT
              plots=survival(cb=hw test atrisk(outside maxlen=13));
   time T * Status(0);
   strata Group;
run;
```

This example shows you how to replace the AUTOALIGN=(TOPRIGHT BOTTOMLEFT TOP BOTTOM) option in the macro variable **InsetOpts** with AUTOALIGN=(BOTTOMRIGHT) and add the AUTOALIGN=(TOPRIGHT) option to the **LegendOpts** macro variable. You can also add the option ACROSS=1 to the **LegendOpts** macro variable to stack all legend entries vertically (to have just one element in each row).

The results are displayed in Figure 24.

Figure 24 Inset Legend

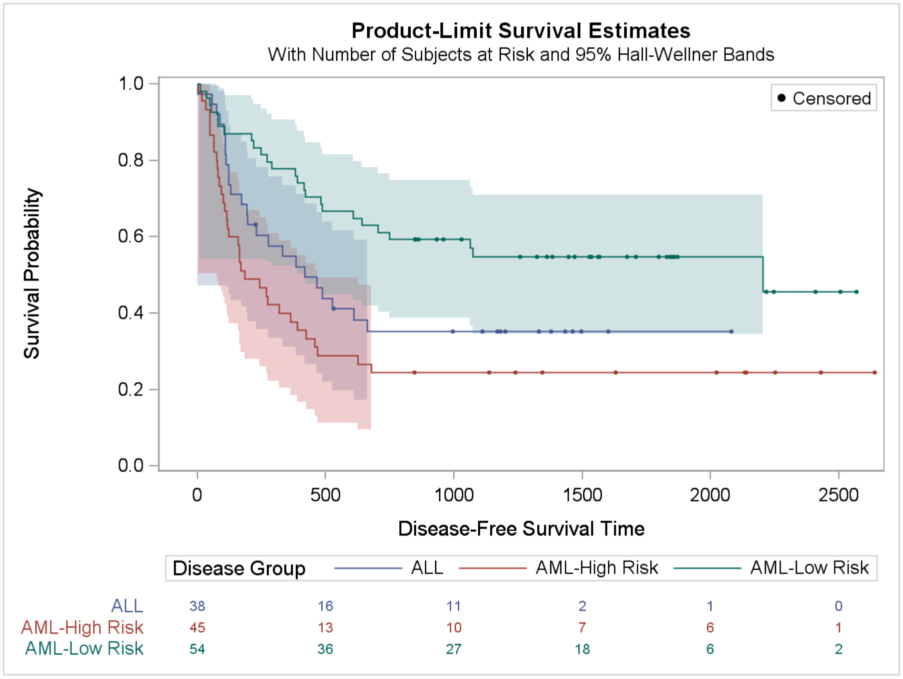


By default, PROC LIFETEST displays a plus sign to indicate censoring. This example illustrates how to change the plus sign to a small filled circle in both the step plots and the inset box. The following steps change the template and create the plot in Figure 25:

```
/*-- Original Macro Variable Definitions ----------------------------------
%let Censored   = markerattrs=(symbol=plus);
%let CensorStr  = "+ Censored";
-------------------------------------------------------------------------*/

%ProvideSurvivalMacros

%let censored  = markerattrs=(symbol=circlefilled size=3px);
%let censorstr = "(*ESC*){Unicode '25cf'x} Censored"
                 / textattrs=GraphValueText(family=GraphUnicodeText:FontFamily);
```

```
%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT plots=survival(cb=hw atrisk(outside maxlen=13));
   time T * Status(0);
   strata Group;
run;
```

Figure 25 Censored Values Modification



The Unicode Consortium (`http://unicode.org/`) provides a list of character codes. Also see the chapter "ODS Graphics Template Modification" in *SAS/STAT User's Guide* for information about the Unicode specification for other markers. Although some Unicode characters are supported in some fonts, you should always specify a Unicode font when using special characters.

You can add a horizontal reference line to the survival plot by adding the following statement to the template:

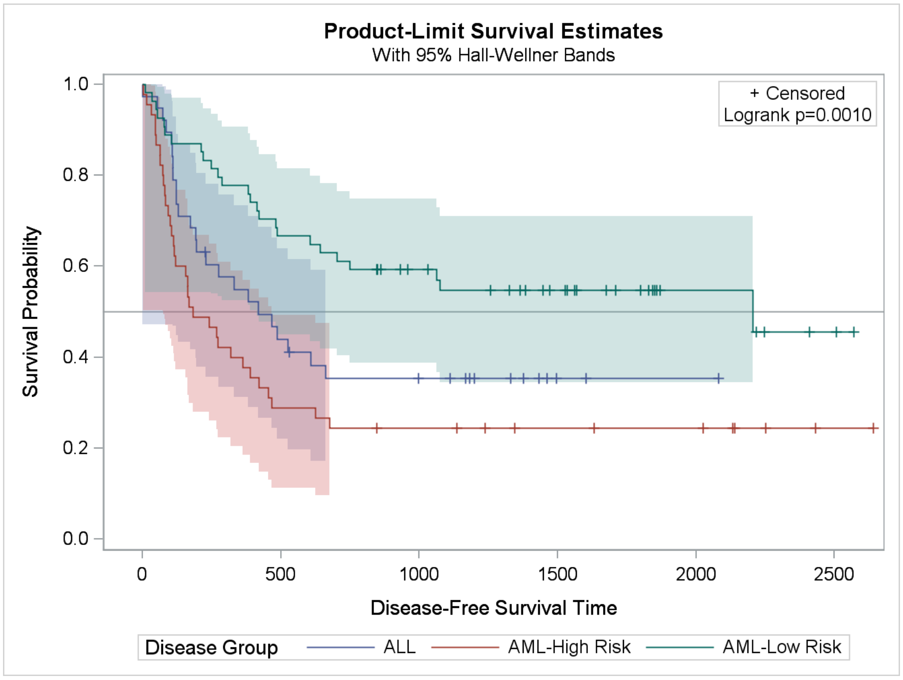
```
referenceline y=0.5;
```

You can do this by using the %StmtsTop macro. By default, this macro is empty. You can use the %StmtsTop macro to add new statements to the beginning of the block of statements that define the appearance of the graph. In contrast, you can use the %StmtsBottom macro to provide statements at the end of the statement block. ODS Graphics draws statements in the order in which they appear; therefore, reference lines should be drawn first so they do not obscure other parts of the graph.

The following step creates the plot in Figure 26:

```
%ProvideSurvivalMacros

%macro StmtsTop;
   referenceline y=0.5;
%mend;

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT plots=survival(cb=hw test);
   time T * Status(0);
   strata Group;
run;
```

Figure 26 Horizontal Reference Line



This example modifies the contents of the %pValue macro. The original %pValue macro definition is as follows:

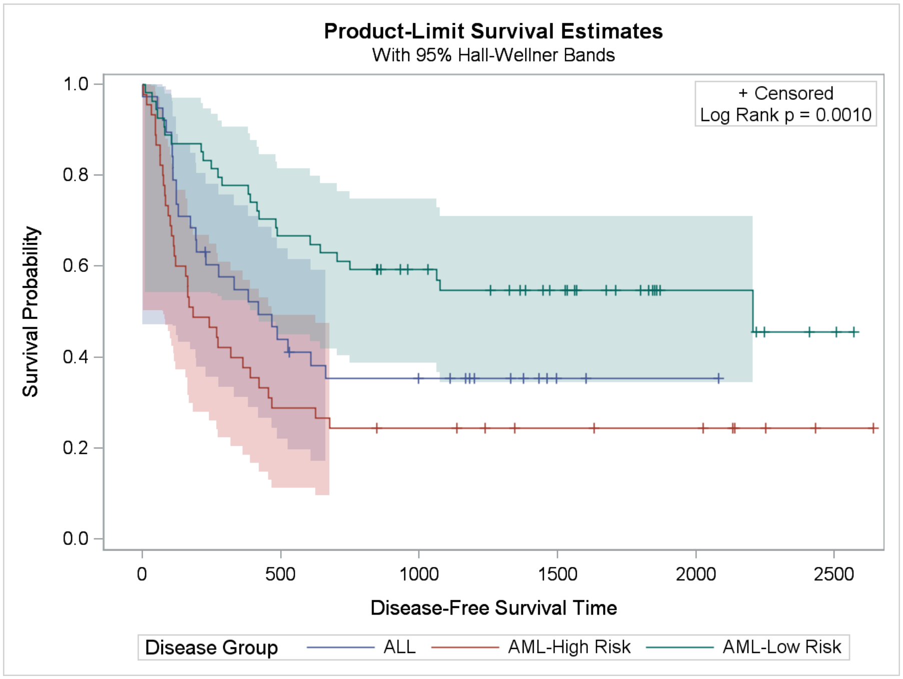
```
%macro pValue;
   if (PVALUE < .0001)
      entry TESTNAME " p " eval (PUT(PVALUE, PVALUE6.4));
   else
      entry TESTNAME " p=" eval (PUT(PVALUE, PVALUE6.4));
   endif;
%mend;
```

The following example directly specifies the test name (replacing the internal name “Logrank” with “Log Rank”) and adds blank spaces around the equal sign:

```
%ProvideSurvivalMacros

%macro pValue;
   if (PVALUE < .0001)
      entry "Log Rank p "   eval (PUT(PVALUE, PVALUE6.4));
   else
      entry "Log Rank p = " eval (PUT(PVALUE, PVALUE6.4));
   endif;
%mend;

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT plots=survival(cb=hw test);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 27.

Figure 27 Cosmetic Inset Entry Change



Because this template modification replaces a character string that is more appropriately set by PROC LIFETEST, you should clean up afterward as follows:

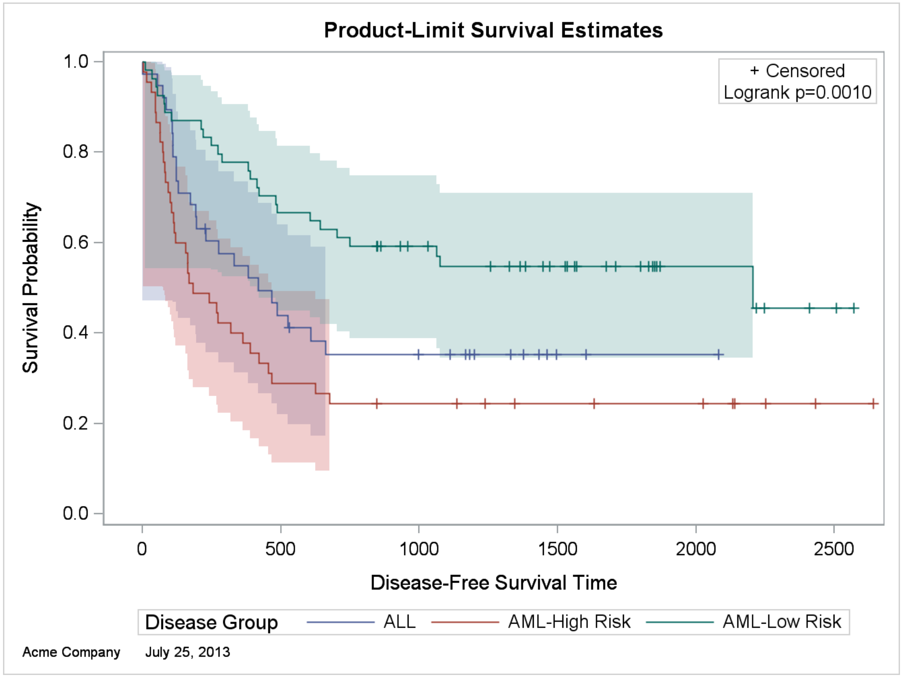
```
%ProvideSurvivalMacros

proc template;
   delete Stat.Lifetest.Graphics.ProductLimitSurvival  /
          store=sasuser.templat;
   delete Stat.Lifetest.Graphics.ProductLimitSurvival2 /
          store=sasuser.templat;
run;
```

The following steps add an ENTRYFOOTNOTE statement to the %StmtsBeginGraph macro and suppress the second title:

```
%ProvideSurvivalMacros

%let ntitles = 1;
%macro StmtsBeginGraph;
   entryfootnote halign=left "Acme Company %sysfunc(date(),worddate.)" /
                 textattrs=GraphDataText;
%mend;

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT
              plots=survival(cb=hw test);
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 28. By default, the **nTitles** macro variable is 2, and all titles are displayed. Setting **nTitles** to 1 suppresses the second title. You can add titles or footnotes to the plot by adding them to the %StmtsBeginGraph macro. This example adds a footnote that consists of a company name followed by the current date, which is formatted by using the WORDDATE format. The **`GraphDataText`** style element is used; it appears in a smaller font than the default style element, **`GraphFootnoteText`**.

Figure 28 Footnote but No Second Title



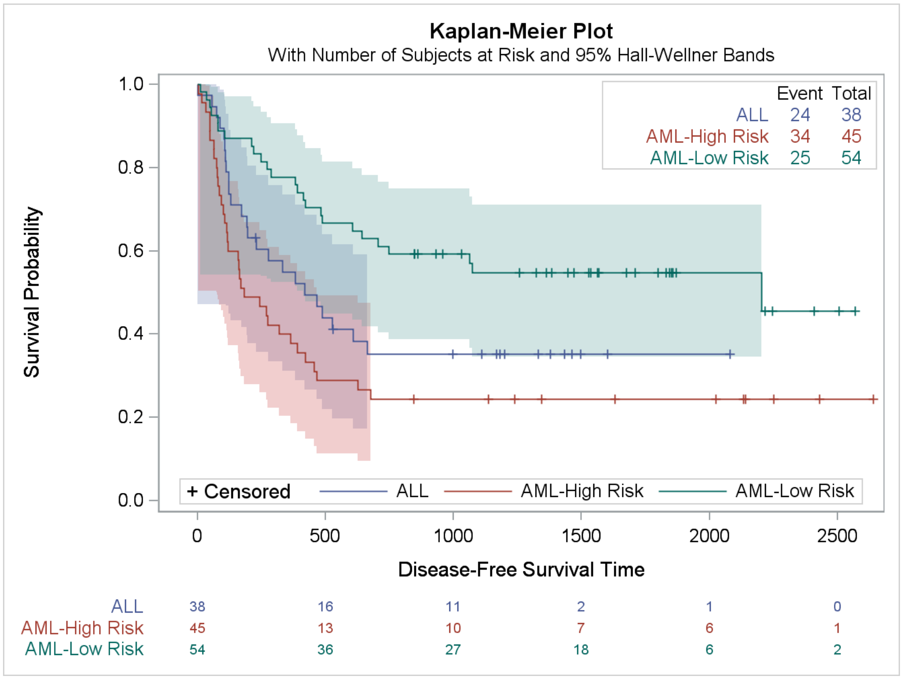
This example shows you how to modify the template to produce the plot displayed in Figure 29. This new plot has an inset table in the top right corner that shows the number of observations and the number of events in each stratum. The legend has been moved inside the plot and combined with the old inset table that shows the marker for censored observations.[2] Also, the title is changed to “Kaplan-Meier Plot”.

```
%ProvideSurvivalMacros

%let TitleText2 = "Kaplan-Meier Plot";
%let LegendOpts = title="+ Censored"
                  location=inside autoalign=(Bottom);
%let InsetOpts  = ;

%macro StmtsBottom;
   dynamic %do i = 1 %to 3; StrVal&i NObs&i NEvent&i %end;;
   layout gridded / columns=3 border=TRUE autoalign=(TopRight);
      entry ""; entry "Event"; entry "Total";
      %do i = 1 %to 3;
         %let t = / textattrs=GraphData&i;
         entry halign=right Strval&i &t; entry NEvent&i &t; entry NObs&i &t;
      %end;
   endlayout;
%mend;

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT plots=survival(cb=hw atrisk(outside maxlen=13));
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 29.

[2]This legend is wide and might not be displayed if your graph is small. If the legend is not displayed, try increasing the size of the graph by specifying the WIDTH= or HEIGHT= option in the ODS GRAPHICS statement.

Figure 29 Inset Event Table



The macro variable **TitleText2**, which controls the title for the multiple-strata plot, is changed. You can change all three title macro variables, as is done in the construction of the plot in Figure 17, or you can change only **TitleText2** when you have multiple overlaid strata, as in this example. The **LegendOpts** macro variable value was changed from TITLE=GROUPNAME LOCATION=OUTSIDE to display the censored value legend in place of the legend title and to display the legend inside the bottom of the plot. When the **InsetOpts** macro variable is null, the usual inset that contains the censored value and the p-value is not displayed.

The %StmtsBottom macro (null by default) is replaced by a macro that creates the new inset table. This macro adds statements to the bottom of the templates. If you ignore for a moment most of the options, the core of the generated statements is as follows:

```
dynamic StrVal1 NObs1 NEvent1 StrVal2 NObs2 NEvent2 StrVal3 NObs3 NEvent3;
layout gridded / columns=3;
   entry "";        entry "Event";    entry "Total";
   entry Strval1;   entry NEvent1;    entry NObs1;
   entry Strval2;   entry NEvent2;    entry NObs2;
   entry Strval3;   entry NEvent3;    entry NObs3;
endlayout;
```

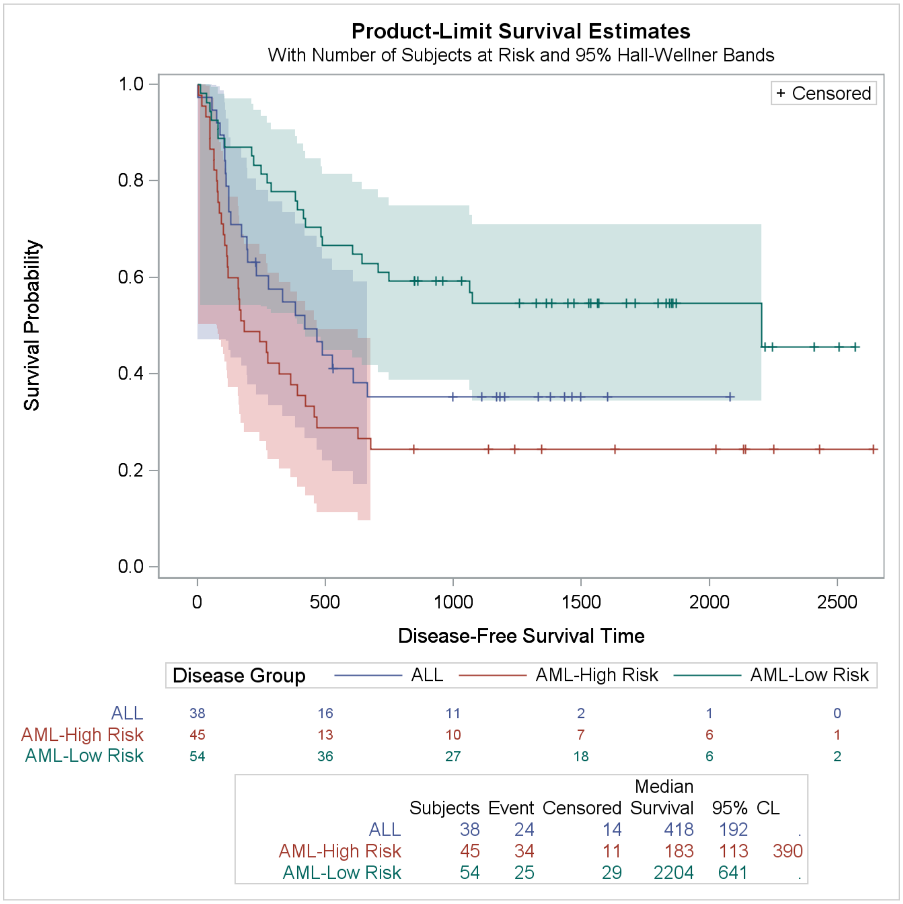
The macro first constructs a DYNAMIC statement that includes the names of the dynamic variables that contain some of the results. PROC LIFETEST creates these dynamic variables and sets them to values, but you must declare them in your template before using them. The macro then constructs a 4×3 grid that contains a table consisting of a title line and a row for each stratum (which consists of the stratum label, the number of events, and the total number of subjects). The full layout that the %StmtsBottom macro generates, with all the options, is as follows:

```
dynamic StrVal1 NObs1 NEvent1 StrVal2 NObs2 NEvent2 StrVal3 NObs3 NEvent3;
layout gridded / columns=3 border=TRUE autoalign=(TopRight);
   entry "";
   entry "Event";
   entry "Total";
   entry halign=right Strval1 / textattrs=GraphData1;
   entry NEvent1 / textattrs=GraphData1;
   entry NObs1 / textattrs=GraphData1;
   entry halign=right Strval2 / textattrs=GraphData2;
   entry NEvent2 / textattrs=GraphData2;
   entry NObs2 / textattrs=GraphData2;
   entry halign=right Strval3 / textattrs=GraphData3;
   entry NEvent3 / textattrs=GraphData3;
   entry NObs3 / textattrs=GraphData3;
endlayout;
```

This example adds a table to the plot that displays a summary of event information. The following statements create the plot in Figure 30:

```
%ProvideSurvivalMacros

%let GraphOpts = DesignHeight=DefaultDesignWidth;

%SurvivalSummaryTable

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT
              plots=survival(cb=hw atrisk(outside maxlen=13));
   time T * Status(0);
   strata Group;
run;
```

Figure 30 External Event Table



	0	500	1000	1500	2000	2500
ALL	38	16	11	2	1	0
AML-High Risk	45	13	10	7	6	1
AML-Low Risk	54	36	27	18	6	2

	Subjects	Event	Censored	Median Survival	95% CL	
ALL	38	24	14	418	192	.
AML-High Risk	45	34	11	183	113	390
AML-Low Risk	54	25	29	2204	641	.

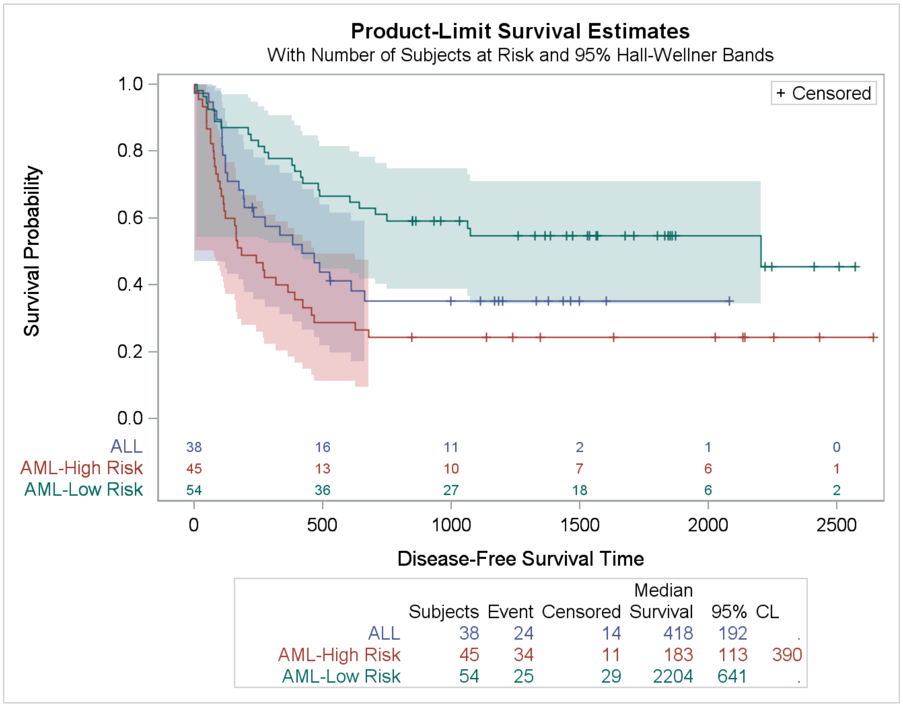
The **GraphOpts** macro variable specifies the option DESIGNHEIGHT=DEFAULTDESIGNWIDTH. This option sets the graph height to the default graph width of 640 pixels. These sizes are *design sizes*. The graph is designed for a 640 × 640 pixel display, but it can be expanded or shrunk. The macro %SurvivalSummaryTable adds new statements to the graph templates that display the number of subjects, number of events, number of censored observations, median survival time, and 95% confidence limits for the median survival time.

The plot in Figure 30 has a legend. However, the plot displays values in the tables by using colors that match the colors of the step functions, so you do not need the legend. The next statements show how to remove the legend:

```
%ProvideSurvivalMacros

%let GraphOpts  = DesignHeight=500px;
%let LegendOpts = ;

%SurvivalSummaryTable

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT
              plots=survival(cb=hw atrisk(maxlen=13));
   time T * Status(0);
   strata Group;
run;
```

The legend is suppressed when the **LegendOpts** macro variable is null. This example also illustrates changing the design height to 500 pixels and moving the at-risk table back inside the body of the plot. The results are displayed in Figure 31.

Figure 31 Suppressing the Legend



	Subjects	Event	Censored	Median Survival	95% CL	
ALL	38	24	14	418	192	.
AML-High Risk	45	34	11	183	113	390
AML-Low Risk	54	25	29	2204	641	.

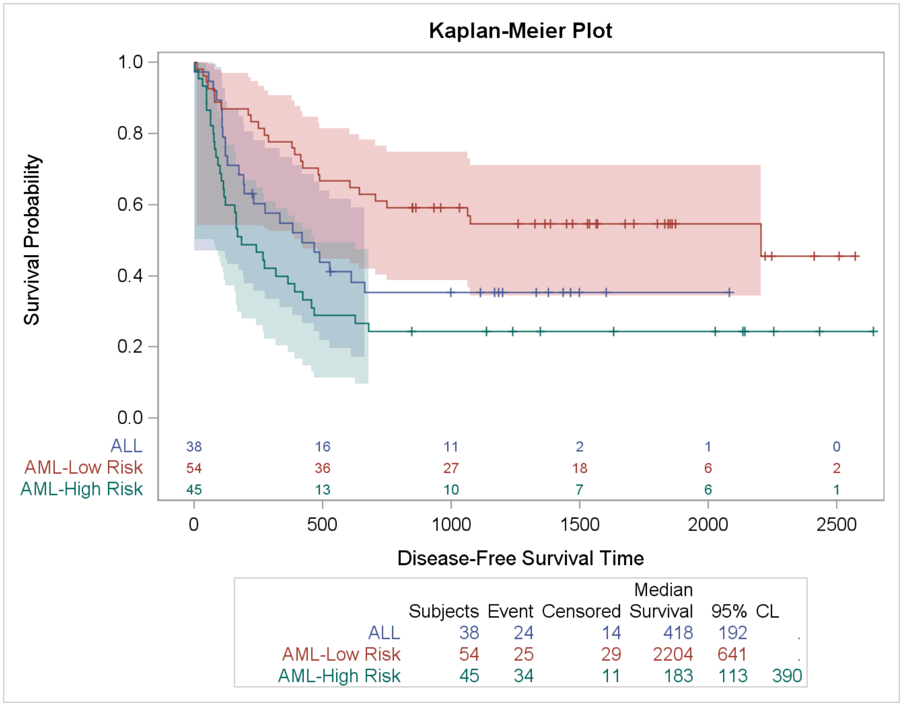
This example combines a number of features from previous examples. The order of the strata levels in the tables is ALL, AML–Low Risk, and AML–High Risk; the title is set to “Kaplan-Meier Plot”; the second title line is suppressed; the graph height is set to 500 pixels; the legend and the inset box that contains the legend for censored values are both suppressed; the event table is displayed outside the plot; and the at-risk table is displayed inside the plot.

```
proc format;
   invalue bmtnum 'ALL' = 1  'AML-Low Risk' = 2  'AML-High Risk' = 3;
   value   bmtfmt 1 = 'ALL'  2 = 'AML-Low Risk'  3 = 'AML-High Risk';
run;
```

```
data BMT(drop=g);
   set sashelp.BMT(rename=(group=g));
   Group = input(g, bmtnum.);
run;

%ProvideSurvivalMacros

%let TitleText2 = "Kaplan-Meier Plot";
%let nTitles    = 1;
%let GraphOpts  = DesignHeight=500px;
%let LegendOpts = ;
%let InsetOpts  = ;

%SurvivalSummaryTable

%CompileSurvivalTemplates

proc lifetest data=BMT plots=survival(cb=hw atrisk(maxlen=13));
   time T * Status(0);
   strata Group / order=internal;
   format Group bmtfmt.;
run;
```

The results are displayed in Figure 32.

Figure 32 Extensive Customizations



	Subjects	Event	Censored	Median Survival	95% CL	
ALL	38	24	14	418	192	.
AML-Low Risk	54	25	29	2204	641	.
AML-High Risk	45	34	11	183	113	390

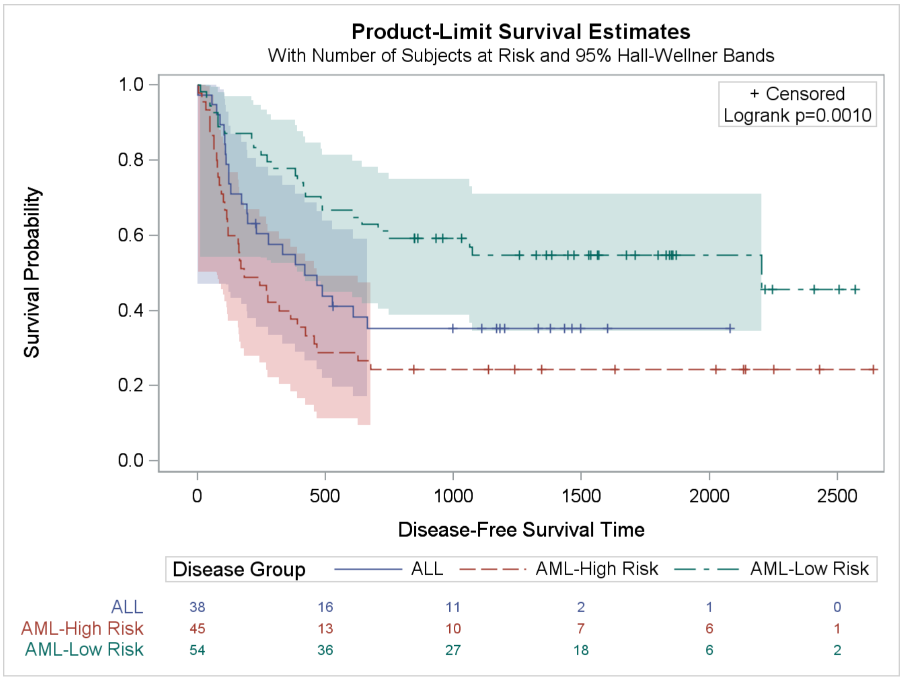
CONTROLLING THE SURVIVAL PLOT BY MODIFYING STYLE TEMPLATES

The graphs that have been displayed up to this point were all produced by using the HTMLBlue style, which is the default style for the HTML destination in the SAS windowing environment. This is an all-color style (because of the ATTRPRIORITY='Color' option); it does not rely on line style or marker changes to differentiate groups. You can switch to a style that varies colors, markers, and lines by specifying the STYLE= option in an ODS destination statement. You can use the HTMLBlueCML style as follows to make a graph whose line patterns differ:

```
ods html style=htmlbluecml image_dpi=300;
proc lifetest data=sashelp.BMT
              plots=survival(cb=hw test atrisk(outside maxlen=13));
   time T * Status(0);
   strata Group;
run;
ods html close;
```

The results are displayed in Figure 33. This example also illustrates specifying the IMAGE_DPI= option to control the resolution (measured in dots per inch, or DPI) of the image. All images in this paper are created at 300 DPI. The default setting for the HTML destination is 100 DPI. Images that are created at 300 DPI are clearer than images created at 100 DPI, but they require about nine times as much disk space.

Figure 33 Line and Color Group Differentiation



You can use the HTMLBlue or Pearl style when you want to differentiate groups only by color. Alternatively, you can easily modify any other style to be an all-color style like HTMLBlue or Pearl by using the ATTRPRIORITY='Color' option (or the ATTRPRIORITY=COLOR option in the ODS GRAPHICS statement):[3]

```
proc template;
   define style styles.ListingColor;
      parent = styles.Listing;
      style Graph from Graph / attrpriority = "Color";
   end;
run;
```

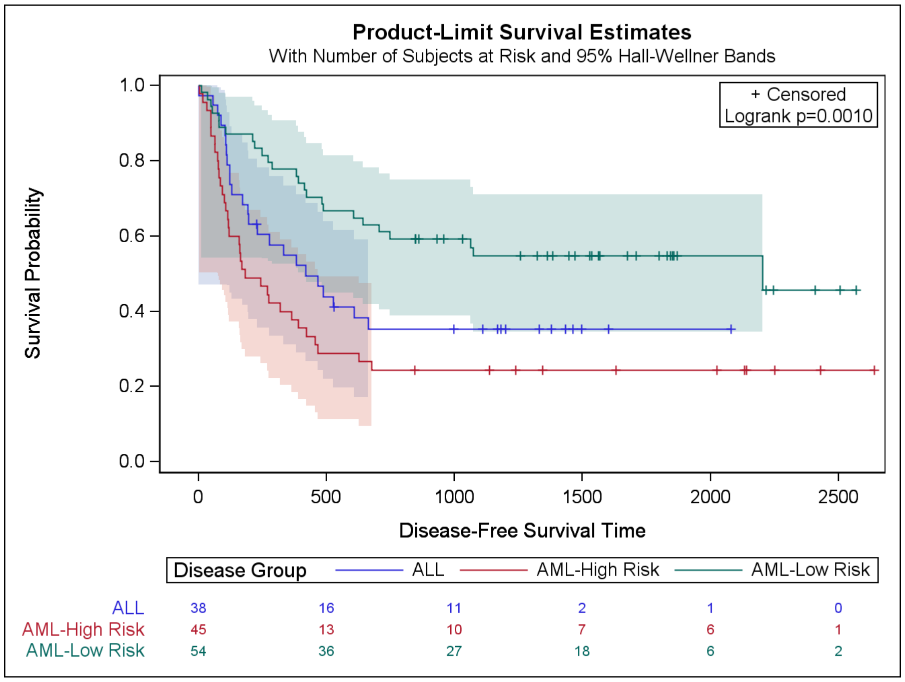
You need to specify the new style name in an ODS destination statement, as in the following:

```
ods html style=ListingColor image_dpi=300;
proc lifetest data=sashelp.BMT
              plots=survival(cb=hw test atrisk(outside maxlen=13));
   time T * Status(0);
   strata Group;
run;
ods html close;
```

[3]The style option is ATTRPRIORITY=*quoted-string*, whereas the GTL and ODS GRAPHICS statement option is ATTRPRIORITY=*keyword*.

The results are displayed in Figure 34.

Figure 34 ATTRPRIORITY='Color' Style



You can use the SOURCE statement in PROC TEMPLATE to display a style as follows:

```
proc template;
   source styles.htmlblue;
run;
```

The results of this step, which are not shown, include the option PARENT=STYLES.STATISTICAL and do not include definitions of the colors (`gData1`, `gData2`, ..., `gData12`) and contrast colors (`gcData1`, `gcData2`, ..., `gcData12`). These are the color definitions that are used in the style elements `GraphData1`, `GraphData2`, ..., `GraphData12`. You can examine the parent Statistical style as follows:

```
proc template;
   source styles.statistical;
run;
```

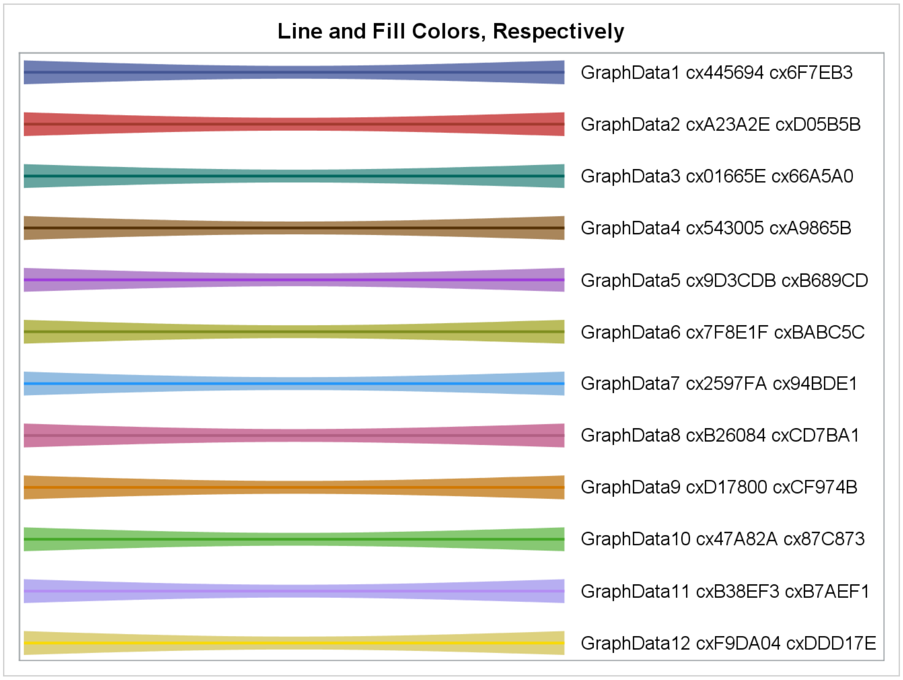
The results of this step are not shown because they are hard to interpret in their raw form, but the desired color definitions are included. You can submit the following statements to display the colors for the Statistical (and hence HTMLBlue) style in a more understandable form:

```
proc template;
   source styles.statistical / file='style.tmp';
run;

data colors;
   infile 'style.tmp';
   input;
   if index(_infile_, 'data') then do;
      element = scan(_infile_, 1, ' ');
      Color = scan(_infile_, 3, ' ;');
      Type = ifc(index(element, 'gc'), 'Line', 'Fill') || ' Colors';
      i = input(compress(element, 'gcdat'';'), ?? 2.);
      if i then output;
   end;
run;
```

```
proc sort; by descending type i; run;

data display;
   array y[12] y1 - y12;
   do i = 1 to 12; y[i] = i;      end;
   do x = 1 to 10; output;        end;
   do i = 1 to 12; y[i] = i + .5; end;
   do x = 1 to 10; output;        end;
run;

data _null_;
   set colors;
   call symputx(compress(type || put(i, 2.)), color);
run;

proc sgplot noautolegend data=display;
   %macro reg;
      title 'Line and Fill Colors, Respectively';
      %do i = 1 %to 12;
         reg y=y%eval(13-&i) x=x / lineattrs=GraphData&i clmattrs=GraphData&i
                          nomarkers clm curvelabelpos=max
                          curvelabel="  GraphData&i &&LineColors&i &&FillColors&i";
      %end;
   %mend;
   %reg
   xaxis display=none;
   yaxis display=none;
run;
title;
```

The results are displayed in Figure 35. The colors in Figure 35 are richer than the colors in the bands in the survival plots because of the DATATRANSPARENCY= options in the BANDPLOT statements in the survival-plot templates.

Figure 35 HTMLBlue Style Colors Display

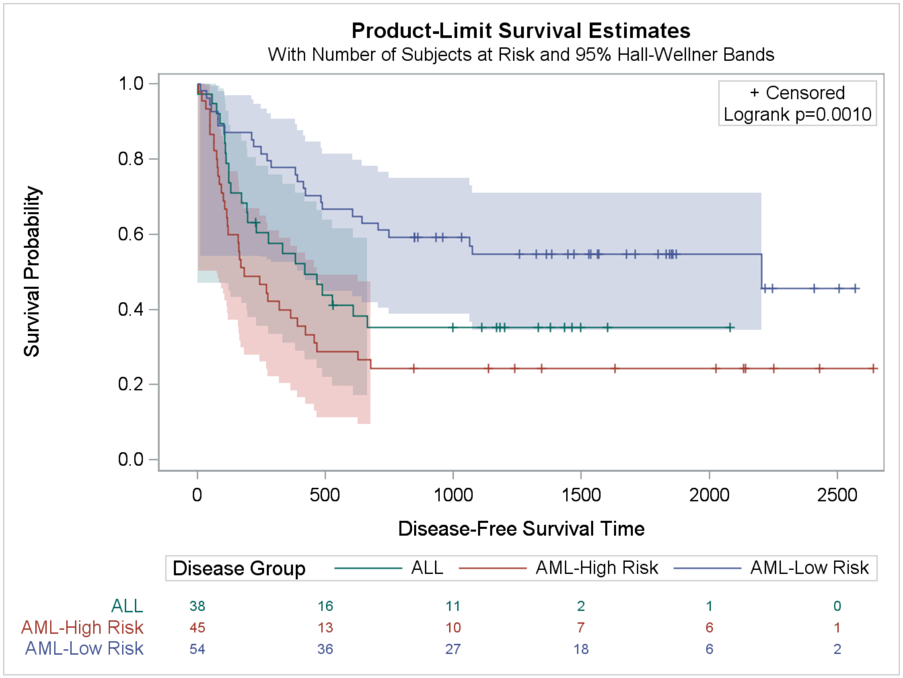


You can use the information in Figure 35 to specify the desired colors in the graph template. You can copy the third, second, and first colors from each list and switch the colors as follows:

```
%ProvideSurvivalMacros

%let GraphOpts = DataContrastColors=(cx01665E cxA23A2E cx445694)
                 DataColors=(cx66A5A0 cxD05B5B cx6F7EB3);

%CompileSurvivalTemplates

proc lifetest data=sashelp.BMT
              plots=survival(cb=hw test atrisk(outside maxlen=13));
   time T * Status(0);
   strata Group;
run;
```

The results are displayed in Figure 36. The familiar colors are used, but they are now in a different order. The next section shows you how to modify a style template to change the color order without having to extract the original color names.

Figure 36 Swapping Colors from the Style



You can use the information in Figure 35 to modify the style template, but the next example shows an easier way. You can modify the colors in a style as follows:

```
%macro reorder(from, to, list);
   proc template;
      define style styles.&to;
         parent=styles.&from;
         %do i = 1 %to 12;
            %let s = %scan(&list, &i);
            %if &s ne %then %do;
               style GraphData&i from GraphData&i /
                     contrastcolor = GraphColors("gcdata&s")
                     color = GraphColors("gdata&s");
            %end;
         %end;
      end;
   run;
%mend;
```

```
%reorder(htmlblue,   /* Parent style.                               */
         MyStyle,    /* New style to create. Specify it in an ODS   */
                     /* destination statement.                      */
         3 2 1)      /* Replace the first few GraphData colors      */
                     /* (1 2 3) with the colors from the specified  */
                     /* GraphData style elements (3 2 1).           */
                     /* You can specify up to 12 integers in the    */
                     /* range 1 - 12.                               */
```

The %Reorder macro creates a new style called MyStyle that inherits most of its attributes from the HTMLBlue style. However, in the new style, the colors for groups 1, 2, and 3 have been replaced by the colors for groups 3, 2, and 1. In other words, the colors for **GraphData1** and **GraphData3** have been switched.

The following step creates the plot:

```
ods html style=mystyle;
proc lifetest data=sashelp.BMT
              plots=survival(cb=hw test atrisk(outside maxlen=13));
   time T * Status(0);
   strata Group;
run;
ods html close;
```

The survival plot is not shown, but it matches the plot in Figure 36.

You can delete the new style template as follows:

```
proc template;
   delete Styles.MyStyle / store=sasuser.templat;
run;
```

CONCLUSIONS

You have many methods available to you for modifying the PROC LIFETEST survival plot. You can use the PLOTS= option. Some of the options discussed in this paper are new in SAS/STAT 12.1. If you find that PLOTS= option modifications are not sufficient, you can modify the graph templates by using the macros. You can control colors and overall appearance by using a different SAS output style or by modifying a style. You can find more information about modifying the PROC LIFETEST survival plot in the *SAS/STAT User's Guide*, in the procedure chapter on PROC LIFETEST (`http://support.sas.com/documentation/onlinedoc/stat/131/lifetest.pdf`) and the introductory chapter on Kaplan-Meier survival plot modification (`http://support.sas.com/documentation/onlinedoc/stat/131/kaplan.pdf`). You can find both on this support.sas.com documentation page: `http://support.sas.com/documentation/onlinedoc/stat/index.html`.

SOFTWARE CREDITS

The LIFETEST procedure was designed and programmed by Ying So, Principal Research Statistician at SAS.

ACKNOWLEDGMENTS

The authors are grateful to Ed Huddleston of SAS Institute Inc. for his valuable assistance in the preparation of this paper.

CONTACT INFORMATION

Warren F. Kuhfeld
SAS Institute Inc.
S6018 SAS Campus Drive
Cary, NC, 27513
(919) 531-7922
Warren.Kuhfeld@sas.com

Ying So
SAS Institute Inc.
S6048 SAS Campus Drive
Cary, NC, 27513
(919) 531-7925
Ying.So@sas.com

REFERENCES

Hall, W. J. and Wellner, J. A. (1980), "Confidence Bands for a Survival Curve from Censored Data," *Biometrika*, 67, 133–143.

Kaplan, E. L. and Meier, P. (1958), "Nonparametric Estimation from Incomplete Observations," *Journal of the American Statistical Association*, 53, 457–481.

Klein, J. P. and Moeschberger, M. L. (1997), *Survival Analysis: Techniques for Censored and Truncated Data*, New York: Springer-Verlag.