PharmaSUG 2024 - Paper DV- 222

Kaplan-Meier Graph: A Comparative Study using SAS vs R

Mrityunjay Kumar, Ephicacy Lifescience Analytics
Shashikant Kumar, Ephicacy Lifescience Analytics

ABSTRACT

Data Visualization plays a very important role in data analysis and reporting. Due to the increase in the high volume of data it has become essential to display data in the form of different types of graphs. Kaplan-Meier graph using PROC LIFETEST is a well-known name to display time-to-event data in the field of clinical SAS programming due to its widespread usage in reporting survival analysis.

The purpose of this paper is to explain how Kaplan-Meier (KM) graph is developed and analyzed in both SAS and R in a stepwise manner. Often, customizing the graph to include additional parameters appears challenging in SAS however its comparatively easier in R. Understanding with the help of an example based on two small treatment groups of hypothetical data, new users can understand how the process works and can lead to a pioneer discussion on leveraging open-source software like R. Shown examples also illustrate the crucially important point during the comparative analysis and may provide an alternative way of creating graphs to statistical programmers and help explore various functionalities in R.

INTRODUCTION

The Kaplan-Meier (KM) graph is commonly used to analyze time-to-event data, such as the time until death or the time until a specific event occurs. KM Plot is a widely used data visualization method for survival analysis. In clinical trials, the effect of a treatment is assessed by measuring the number of subjects survived or saved after the treatment over a period. Sometimes it is interesting to compare the survival of subjects in two or more treatments. The time starting from a defined point to the occurrence of a given event is called as the survival time and the analysis of group data as survival analysis.

The Kaplan-Meier estimate is also known as “product limit estimate”. The KM survival curve is defined as the probability of surviving in a given length of time while considering time in many small intervals. The survival probability at any particular time is calculated by the formula given below:

$$S_t = \frac{\text{Number of subjects living at the start} - \text{Number of subjects died}}{\text{Number of subjects living at the start}}$$

For each time interval, survival probability is calculated as the number of subjects surviving divided by the number of patients at risk. Subjects who have died, dropped out, or move out are not counted as “at risk” i.e., subjects who are lost are considered “censored” and are not counted in the denominator.

DATA STRUCTURE EXAMPLE FOR KM PLOT

Let's look at the example data set Bone Marrow Transplant (BMT) (Table 1) from the SASHELP.BMT to build KM plot (Table 4a). This data set will be used as an example throughout this paper. The analysis data set contains variables required to understand programming logic and has been updated to include two Stratification groups (TRTN) i.e. ALL and AML- Low Risk.

BMT data from SASHELP library has been used in this example and variables aval and cnsr are created to resemble alike ADTTE (Time to Event) data.

Annotation of variables provides additional information about the data.

USUBJID = Unique subject identifier

TRT= Stratification Group, ALL – Acute Lymphocytic Leukemia, AML – Acute Myelocytic Leukemia (Low-Risk)

TRTN = Stratification Group, 1 = ALL, 2 = AML

AVAL = Time (in days)

CNSR = Censor (0), Event (1)

Table 1: SASHELP.BMT (Bone Marrow Transplant)

Obs	usubjid	trt	aval	status	cnsr
1	1-001	ALL	2081	0	0
2	1-002	ALL	1602	0	0
3	1-003	ALL	1496	0	0
4	1-004	ALL	1462	0	0
5	1-005	ALL	1433	0	0
6	1-006	ALL	1377	0	0
7	1-007	ALL	1330	0	0
8	1-008	ALL	996	0	0
9	1-009	ALL	226	0	0
10	1-010	ALL	1199	0	0
11	1-011	ALL	1111	0	0
12	1-012	ALL	530	0	0
13	1-013	ALL	1182	0	0
14	1-014	ALL	1167	0	0
15	1-015	ALL	418	1	1
16	1-016	ALL	383	1	1
17	1-017	ALL	276	1	1
18	1-018	ALL	104	1	1
19	1-019	ALL	609	1	1
20	1-020	ALL	172	1	1

Obs	usubjid	trt	aval	status	cnsr
41	1-041	AML	2409	0	0
42	1-042	AML	2218	0	0
43	1-043	AML	1857	0	0
44	1-044	AML	1829	0	0
45	1-045	AML	1562	0	0
46	1-046	AML	1470	0	0
47	1-047	AML	1363	0	0
48	1-048	AML	1030	0	0
49	1-049	AML	860	0	0
50	1-050	AML	1258	0	0
51	1-051	AML	2246	0	0
52	1-052	AML	1870	0	0
53	1-053	AML	1799	0	0
54	1-054	AML	1709	0	0
55	1-055	AML	1674	0	0
56	1-056	AML	1568	0	0
57	1-057	AML	1527	0	0
58	1-058	AML	1324	0	0
59	1-059	AML	957	0	0
60	1-060	AML	932	0	0

KM PLOT USING SAS STUDIO

Let's see first how we create KM plot in SAS by running the program below and produce various output objects that may be added to the graph to enhance its appearance in detail.

We are using PROC LIFETEST and ODS OUTPUT to create output objects such as Numbers at risk, events and censored summary, median survival time, survival probability by time, p-value and Hazard ratio.

```
ods graphics on;
ods trace on;
ods output CensoredSummary=csum Survivalplot=atrisk Quartiles=median(where=(percent=50))
           HomTests=pval;
proc lifetest data=bmt method=km conftype=linear outsurv= survstat plots=survival (atrisk=0 to 2500 by 500);
  time aval*cnsr(0);
  strata trtn;
  format trtn group.;
run;
ods output close;
ods trace off;
ods graphics off;

** 1. Number of subjects at Risk **;
data atrisk_1;
  set atrisk(where=(tatrisk > .z));
  keep trt atrisk tatrisk;
  trt=stratumnum;
run;

proc transpose data=atrisk_1 out=at_risk prefix=risk_at_;
  by trt;
  var atrisk;
  id tatrisk;
  format trt group.;
run;

proc print data = at_risk;
run;
```

Obs	trt	_NAME_	_LABEL_	risk_at_0	risk_at_500	risk_at_1000	risk_at_1500	risk_at_2000	risk_at_2500
1	ALL	AtRisk	At Risk	38	16	11	2	1	0
2	AML	AtRisk	At Risk	54	36	27	18	6	2

```
** 2. Events & Censored Summary **;
data summ;
  set csum;
  pctEvnt = failed * 100 / total;
  where trtn > .z;
  format trtn group.;
run;

proc print data = summ;
run;
```

Obs	Stratum	trtn	Total	Failed	Censored	PctCens	pctEvnt
1	1	ALL	38	24	14	36.84	63.1579
2	2	AML	54	25	29	53.70	46.2963

```
** 3. Median survival time **;
proc print data = median;
run;
```

Obs	STRATUM	trtn	Percent	Estimate	Transform	LowerLimit	UpperLimit
1	1	ALL	50	418.00	LOGLOG	192.00	.
2	2	AML	50	2204.00	LOGLOG	641.00	.

```
** 4. Survival probability by timelist **;

data survstat_1;
  set survstat ;
  by stratum aval;
  retain survival_d ;
  if SURVIVAL ne . then do;
     survival_d=SURVIVAL;
  end;
run;

data survstat_2;
  set survstat_1 ;
  by stratum aval;
  retain LCL UCL;
  if SDF_LCL ne . or SDF_UCL ne . then do;
     LCL = SDF_LCL;
     UCL = SDF_UCL;
  end;
  rename aval=time ;
run;

data m_surstat(where=(dur ne .));
  set survstat_2;
      if time <= 2500 then dur=2500;
      if time <= 2000 then dur=2000;
      if time <= 1500 then dur=1500;
      if time <= 1000 then dur=1000;
      if time <= 500 then dur=500;
run;

proc sort data=m_surstat;
  by stratum trtn dur time;
run;

data survprob;
  set m_surstat;
  by stratum trtn dur time;
  sprob = put(survival_d*100,5.2)||' ('||put(LCL*100,5.2)||','||put(UCL*100,5.2)||')';
  keep trtn dur sprob;
  format trtn group.;
  if last.dur;
run;

proc transpose data=survprob out=survest prefix=surv_p_;
 by trtn;
 var sprob;
 id dur;
run;

proc print data = survest;
run;
```

Obs	trtn	_NAME_	surv_p_500	surv_p_1000	surv_p_1500	surv_p_2000	surv_p_2500
1	ALL	sprob	43.94 (27.97,59.90)	35.31 (19.76,50.85)	35.31 (19.76,50.85)	35.31 (19.76,50.85)	35.31 (19.76,50.85)
2	AML	sprob	66.67 (54.09,79.24)	59.26 (46.15,72.36)	54.70 (41.17,68.24)	54.70 (41.17,68.24)	45.58 (25.75,65.42)

```
** 5. Log-rank p-value **;

   proc print data = pval;
   run;
```

Obs	Test	ChiSq	DF	ProbChiSq
1	Log-Rank	4.7298	1	0.0296
2	Wilcoxon	4.9938	1	0.0254
3	-2Log(LR)	8.9297	1	0.0028

```
** 6. Hazard ratio correct way using REF = option in class **;

ods output ParameterEstimates = hr_wo_ref;
proc phreg data = bmt;
  class trtn; ** No reference group given **;
  model aval*cnsr(0) = trtn / risklimits;
  format trtn group.;
run;

proc print data = hr_wo_ref;
run;
```

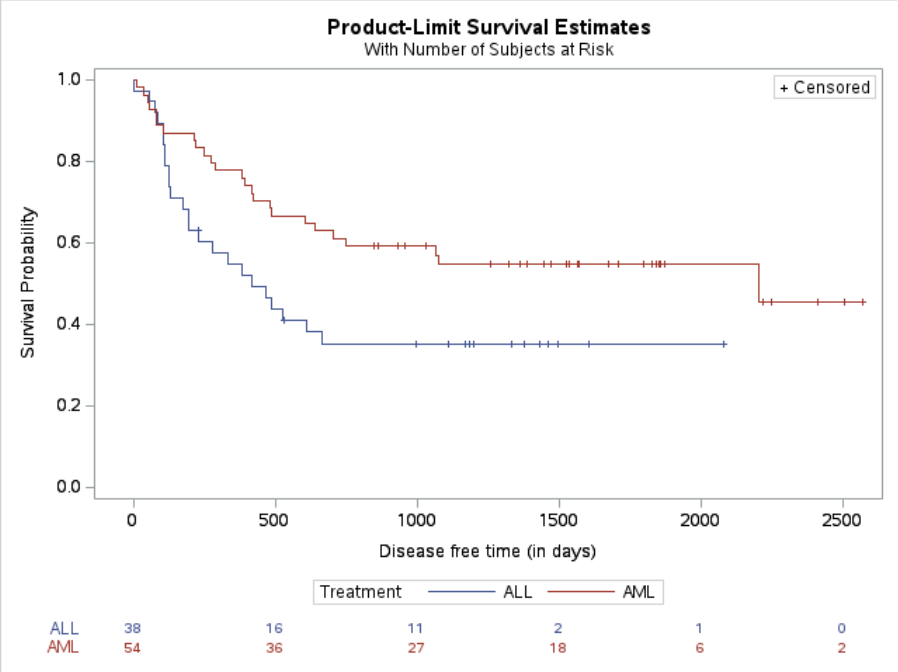
Obs	Parameter	ClassVal0	DF	Estimate	StdErr	ChiSq	ProbChiSq	HazardRatio	HRLowerCL	HRUpperCL	Label
1	trtn	ALL	1	0.62166	0.29040	4.5825	0.0323	1.862	1.054	3.290	Treatment ALL

```
ods output ParameterEstimates = hr_w_ref;
proc phreg data = bmt;
  class trtn(ref='ALL'); ** 'ALL' group is used as reference **;
  model aval*cnsr(0) = trtn / risklimits;
  format trtn group.;
run;

proc print data = hr_w_ref;
run;
```

Obs	Parameter	ClassVal0	DF	Estimate	StdErr	ChiSq	ProbChiSq	HazardRatio	HRLowerCL	HRUpperCL	Label
1	trtn	AML	1	-0.62166	0.29040	4.5825	0.0323	0.537	0.304	0.949	Treatment AML

We can see from above results that correct way to compute ***Hazard ratio*** is to explicitly define reference group using REF= option in the class statement otherwise SAS considers second group as reference and first group as active group when group is numerically coded in ascending order.

Figure 1. KM Plot in SAS Studio



KM PLOT USING R STUDIO

Now let's see how we create KM plot in R by running the program below and produce similar output objects created by SAS that may be added to the graph and then we can see comparison.

The following step by step approach helps the user to create KM plot in R.

Below listed packages need to be installed where "survival" and "survminer" packages are meant for survival analysis and KM Plot customization. However, "dplyr" package is used for data manipulation, packages "readxl" and "writexl" are used for importing and exporting of data in excel format and package "gtsummary" is used to print summary of regression results in publication-ready tabular format. After installation, each package needs to be loaded into R session using library() function.

```
# 1. Install packages into R

install.packages("survival")
install.packages("survminer")
install.packages("gtsummary")
install.packages("readxl")
install.packages("writexl")
install.packages("dplyr")
```

BMT data set from SASHELP library was exported in excel and then imported in R using function 'read_excel' and 'head' function prints first 10 observations.

```
# 2. Read data into R

library(readxl)
bmt <- read_excel("bmt.xlsx")
head(bmt, 10)
```

```
> head(bmt, 10)
# A tibble: 10 × 3
    trtn  aval  cnsr
   <dbl> <dbl> <dbl>
 1     1  2081     0
 2     1  1602     0
 3     1  1496     0
 4     1  1462     0
 5     1  1433     0
 6     1  1377     0
 7     1  1330     0
 8     1   996     0
 9     1   226     0
10     1  1199     0
```

Surv() – creates a survival data set like SAS output data set from OUTSURV = DATA in PROC LIFETEST. The result of the Surv() is usually used as input by other survival package functions. By default, the event indicator expects the following format: 0=censor, 1=dead. The object 'surv_object' contains the time and events details whether the event occurred or not at a given time.

Note: R functions are case-sensitive, so this function should always start with a capital "S".

```
# 3. Create survival object after loading 'survival' package
library(survival)
surv_object <- Surv(time = bmt$aval, event = bmt$cnsr )
surv_object
```

We can see in below results plus (+) symbol, which indicates censored observations like SAS output.

```
> surv_object
  [1] 2081+ 1602+ 1496+ 1462+ 1433+ 1377+ 1330+  996+  226+ 1199+ 1111+  530+
 [13] 1182+ 1167+  418   383   276   104   609   172   487   662   194   230
 [25]  526   122   129    74   122    86   466   192   109    55     1   107
 [37]  110   332  2569+ 2506+ 2409+ 2218+ 1857+ 1829+ 1562+ 1470+ 1363+ 1030+
 [49]  860+ 1258+ 2246+ 1870+ 1799+ 1709+ 1674+ 1568+ 1527+ 1324+  957+  932+
 [61]  847+  848+ 1850+ 1843+ 1535+ 1447+ 1384+  414  2204  1063   481   105
 [73]  641   390   288   421    79   748   486    48   272  1074   381    10
 [85]   53    80    35   248   704   211   219   606
```

survfit() – function creates survival curves using the Kaplan-Meier method based on a formula. The survfit() function uses the results from Surv() function as a response variable in the model formula.

```
# Create survival statistics
kmfit <- survfit(surv_object ~ trtn, data = bmt)
kmfit
```

```
> kmfit
Call: survfit(formula = surv_object ~ trtn, data = bmt)

        n events median 0.95LCL 0.95UCL
trtn=1 38     24    418     194      NA
trtn=2 54     25   2204     704      NA
```

surv_summary() function provides survival statistics for each timepoint, and head() function prints the first 10 observations from the output. Survival statistics summary is then exported by using write_xlsx() function into excel file “evnt_sum.xlsx”.

```
# print summary for survival statistics
evnt_sum <- surv_summary(kmfit, data=bmt)
head(evnt_sum,10)
library(writexl)
write_xlsx(evnt_sum,"evnt_sum.xlsx")
```

	A	B	C	D	E	F	G	H	I	J
1	time	n.risk	n.event	n.censor	surv	std.err	upper	lower	strata	trtn
2	1	38	1	0	0.973684	0.026669	1	0.924097	trtn=1	1
3	55	37	1	0	0.947368	0.038236	1	0.878967	trtn=1	1
4	74	36	1	0	0.921053	0.047494	1	0.839185	trtn=1	1
5	86	35	1	0	0.894737	0.055641	0.997832	0.802293	trtn=1	1
6	104	34	1	0	0.868421	0.063145	0.982832	0.767329	trtn=1	1
7	107	33	1	0	0.842105	0.070244	0.966403	0.733795	trtn=1	1
8	109	32	1	0	0.815789	0.077086	0.948842	0.701394	trtn=1	1
9	110	31	1	0	0.789474	0.083771	0.930344	0.669934	trtn=1	1
10	122	30	2	0	0.736842	0.096946	0.891035	0.609332	trtn=1	1

survdiff() function is used to find out the difference between the survival curves. We see that the survival curves are placed distantly apart, and the log-rank test p-value (0.03) provides evidence of a survival difference between ALL and AML.

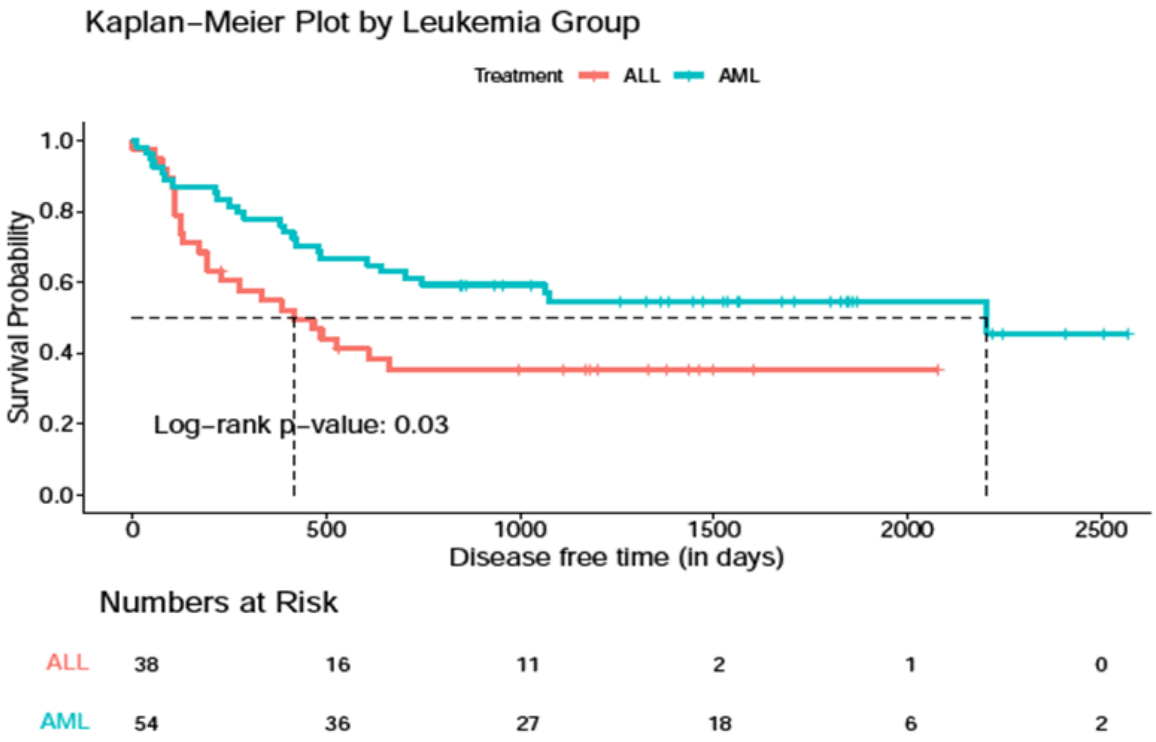
```
# Log-rank p value
survdiff(Surv(aval,cnsr)~trtn, data=bmt)
```

```
> survdiff(Surv(aval,cnsr)~trtn, data=bmt)
Call:
survdiff(formula = Surv(aval, cnsr) ~ trtn, data = bmt)

        N Observed Expected (O-E)^2/E (O-E)^2/V
trtn=1 38       24     16.8      3.03      4.73
trtn=2 54       25     32.2      1.59      4.73

 Chisq= 4.7  on 1 degrees of freedom, p= 0.03
```

Let's look at ggsurvplot() function from the survminer package which has been used to create KM plot and enhance its appearance by adding risk-table, log-rank p-value, survival median line etc. Various options under ggsurvplot() help in customizing KM plot as per user and the link has been provided in reference section at the end of the paper.

```
# 4. Load package 'survminer' for KM plot
library(survminer)

# create KM plot using 'ggsurvplot'
ggsurvplot(kmfit,
title = "Kaplan-Meier Plot by Leukemia Group",
xlab = "Disease free time (in days)",
ylab = "Survival Probability",
legend.title = "Treatment",
legend.labs = c("ALL", "AML"),
linetype = 1,
pval = "Log-rank p-value: 0.03",
pval.method = TRUE,
conf.int = FALSE,
risk.table = "absolute",
risk.table.title = "Numbers at Risk",
tables.theme = theme_cleantable(),
risk.table.pos = "out",
fontsize = 4,
surv.median.line = "hv"
)
```

Figure 2. KM Plot in RStudio



HAZARDS RATIO (HR):

Cox proportional hazards regression is one of the most popular regression techniques for survival analysis.

Here, the measure of effect is the ***Hazard rate*** (HR), which is the risk of failure (or the risk of event i.e death in our example), given that the participant has survived up to a specific time.

Usually, we are interested in comparing ***independent*** groups with respect to their hazards, and we use a hazard ratio, which is analogous to an odds ratio in the setting of multiple logistic regression analysis.

coxph() function from survival package can be used to compute ***Hazard ratio***. ***HR*** above 1 indicates a covariate that is positively associated with the event probability, and thus negatively associated with the length of survival.

- HR = 1: No effect
- HR < 1: Reduction in the hazard
- HR > 1: Increase in Hazard

```
# 5. Hazard ratio using Cox proportional regression
hr <- coxph(Surv(aval,cnsr) ~ trtn, data = bmt)
hr
```

```
> hr
Call:
coxph(formula = Surv(aval, cnsr) ~ trtn, data = bmt)

        coef exp(coef) se(coef)      z      p
trtn -0.6224    0.5367   0.2904 -2.143 0.0321

Likelihood ratio test=4.52  on 1 df, p=0.03343
n= 92, number of events= 49
```

tbl_regression() function from gtsummary package helps to print ***Hazard ratio*** in publication ready tabular format. This function can also be used to produce data summary reports in tabular format.

```
# Print Hazard ratio
library(gtsummary)
tbl_regression(hr, exp = TRUE)
```

Characteristic	HR[1]	95% CI[1]	p-value
trtn	0.54	0.30, 0.95	0.032

[1] HR = Hazard Ratio, CI = Confidence Interval

COMPARISON: KM PLOT USING SAS VS R

We have seen the results produced by both SAS and R in a stepwise manner. It is observed that both softwares are capable of producing high quality graphics. However, we also see that customizing graphics in RStudio appears easier due to multiple options available in the R packages. Although, the same result can also be achieved by modifying the template available in SAS using PROC TEMPLATE or creating annotation data set while using PROC SGPLOT. Not much difference was observed in execution time, however it also depends on size of data. Computation logic of *Hazard ratio* was found differently in SAS as compared to R due to reference group selection. By default, SAS treats 'AML coded as 2' as reference group whereas R treats 'ALL coded as 1' as reference group hence explicit reference must be given to avoid erroneous results.

CONCLUSION

We have seen the functionalities of both softwares SAS and R in producing results for survival analysis in the form of KM Plot. While SAS has been widely used and accepted in the pharmaceutical industry for producing high quality reports, now with several packages available within R, it can also be used as an alternative approach for producing reports, especially graphs.

On the other hand, this paper has shown that both softwares can be used to produce KM plot outputs, thus it is advisable to explore new technologies and their functions from statistical programming perspective. Exploring graphics packages from R softwares to create customized graphs like KM plot and other graphs can be a more efficient and time saving approach.

Statistical programmers interested in exploring visualization with enhanced graphics features may find this paper useful. Learning new software for doing the same task can always be more fun just like driving to the same destinations by different cars.

REFERENCES

Manish Kumar Goel, Pardeep Khanna, Jugal Kishore, *"Understanding survival analysis: Kaplan-Meier estimate".* https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3059453/pdf/IJAR-1-274.pdf

Oleksandr Babych, 2023, PHUSE Conference (US Connect) - *"How to Customize Survival* Curves in R and Add Interactivity*"*. https://www.lexjansen.com/phuse-us/2023/st/PAP_ST16.pdf

Warren F. Kuhfeld, Ying So, SAS Institute Inc. 2014, PharmaSUG Conference – *"Creating and Customizing the Kaplan-Meier Survival Plot in PROC LIFETEST in the SAS/STAT® 13.1 Release".* https://www.lexjansen.com/pharmasug/2014/SP/PharmaSUG-2014-SP14-SAS.pdf

Drawing Survival Curves Using ggsurvplot: https://rpkgs.datanovia.com/survminer/reference/ggsurvplot.html

UCLA - Statistical Methods and Data Analytics: https://stats.oarc.ucla.edu/sas/examples/asa2/applied-survival-analysis-by-hosmer-lemeshow-and-maychapter-2-descriptive-methods-for-survival-data/

Emily C. Zabor, "*Survival Analysis in R*". https://www.emilyzabor.com/tutorials/survival_analysis_in_r_tutorial.html)

ACKNOWLEDGMENTS

Authors would like to extend their sincere thanks to Ephicacy Lifescience Analytics for giving an opportunity to write this paper. Any brand and product names are trademarks of their respective companies.

RECOMMENDED READING

- *SAS® Studio For new User® https://welcome.oda.sas.com/*
- *RStudio User Guide: https://docs.posit.co/ide/user/ide/get-started/*
- *R Package: https://cran.r-project.org/web/packages/*

CONTACT INFORMATION

Your comments and questions are more valued and encouraged. You may contact authors at:

Mrityunjay Kumar, Associate Director - Biostatistics & Programming
Ephicacy Lifescience Analytics
Bengaluru, India
E-mail: mrityunjay.kumar@ephicacy.com
www.ephicacy.com

Shashikant Kumar, Director - Biostatistics & Programming
Ephicacy Lifescience Analytics
Bengaluru, India
E-mail: shashikant.kumar@ephicacy.com
www.ephicacy.com