PharmaSUG 2021 - Paper EP-029

A Macro for Point Estimate and Confidence Interval of the Survival Time Percentiles

Hengwei Liu, Daiichi Sankyo, Inc.

ABSTRACT

The PROC LIFETEST in SAS® creates a table called quartiles. It contains the point estimate and confidence interval for the 25, 50 and 75 percentiles for the survival time. PROC LIFETEST doesn't have an option to calculate the point estimate and confidence interval for any other percentiles for the survival time. In this paper a macro is set up to calculate the point estimate and confidence interval for the survival time percentiles. The development is done with SAS version 9.4_M4 with SAS/STAT 14.2 in Linux.

INTRODUCTION

The code below will look familiar to programmers who have worked on survival analysis:

```
ods output ProductLimitEstimates=km_curv Quartiles=quartiles;
proc lifetest data=adtte method=km conftype=loglog alphaqt=0.05;
by strata;
time aval*cnsr(1);
run;
```

The ods table ProductLimitEstimates provides the product-limit survival estimates; the table Quartiles shows point estimate and confidence interval for the quartiles of the survival times. We will show how to use the table ProductLimitEstimates to calculate the point estimate and confidence interval for any percentile, not just the quartiles.

First off, we give a short discussion of the mathematical background, which is taken from SAS documentation for SAS/STAT 14.2 (reference [1]).

Let $t_1 < t_2 < \cdots < t_D$ represent the distinct event times.
The general formula for estimating the 100*p*th percentile point is

$q_p = \min\{t_j | \hat{S}(t_j) < 1- p\}$

where $\hat{S}(t_j)$ is the Kaplan-Meier estimate of the survivor function at t_j. If $\hat{S}(t_j)$ is exactly equal to 1 - *p* from t_j to t_{j+1}, the 100*p*th percentile is taken to be $(t_j + t_{j+1})/2$.

The 100(1 – α)% confidence interval for the 100*p*th percentile is the set of all points that satisfy

$$\left|\frac{g(\hat{S}(t)-g(1-p)}{g'\left(\hat{S}(t)\right)\hat{\sigma}\left(\hat{S}(t)\right)}\right| \leq z_{1-\frac{\alpha}{2}}$$

Where g is a transformation specified by CONFTYPE= option in PROC LIFETEST, $g'(x)$ is the first derivative of g(x), $\hat{\sigma}\left(\hat{S}(t)\right)$ is the estimate of the standard error for $\hat{S}(t)$, and $z_{1-\frac{\alpha}{2}}$ is the (100(1-α/2))th percentile of the standard normal distribution.

The confidence interval is extended but does not include the next event time based on Brookmeyer and Crawley (see reference [2]).

These algorithms are implemented in a SAS macro.

THE DETAILS

There is an article about how to calculate the point estimate (see reference [3]). The main effort of this paper is to calculate the confidence limits.

To calculate the confidence interval, we look at the CONFTYPE= option for PROC LIFETEST.

Based on the SAS/STAT 14.2 documentation, user can choose one of five possible transformations for the CONFTYEP= option.

In the ods table ProductLimitEstimates there are variables called Survival and StdErr. We use those variables to specify the formula for the calculation of the confidence interval.

The formulae are shown in table 1.

Transformation	**Formula for Transformation**	**The formula for** $\frac{g(\hat{S}(t)-g(1-p))}{g'(\hat{S}(t))\hat{\sigma}(\hat{S}(t))}$
ASINSQRT	$g(x) = sin^{-1}(\sqrt{x})$	$\frac{\arcsin(\sqrt{survival}) - arcsin(\sqrt{1-\frac{percentile}{100}})}{\frac{stderr}{2\sqrt{survival(1-survival)}}}$
LOGLOG	$g(x) = \log(-\log(x))$	$\frac{\log(-\log(survival) - \log(-\log(1-\frac{percentile}{100}))}{\frac{stderr}{survival * \log(survival)}}$
LINEAR	$g(x) = x$	$\frac{survival - (1-\frac{percentile}{100})}{stderr}$
LOG	$g(x) = \log(x)$	$\frac{\log(survival) - \log((1-\frac{percentile}{100})}{\frac{stderr}{survival}}$
LOGIT	$g(x) = \log(\frac{x}{1-x})$	$\frac{\log(\frac{survival}{1-survival}) - \log(\frac{1-\frac{percentile}{100}}{\frac{percentile}{100}})}{\frac{stderr}{survival * (1-survival)}}$

Table 1. Formulae to calculate the confidence interval

PARAMETERS FOR THE MACRO

The macro takes five parameters.

INDATA=the input dataset. The table 2 shows the variables expected in the input dataset.

AVAL	The survival time
CNSR	censor (1 is for censored case, 0 is for event)
Strata variable	If the analysis is done by strata, then there will also be a variable for strata.

Table 2. the Variables Expected in the Input Dataset for the Macro

PERCSTR= the list of percentiles for which we need to calculate the point estimate and confidence interval. It can be one percentile, or multiple percentiles separated by comma.

ALPHAPT= specifies the significance level α for the 100(1- α)/% confidence intervals for the percentiles of the survival time. The default is 0.05.

CONFTYPE= Specifies the transformation applied to the survivor function to obtain confidence limits. User can choose one of the five transformations in table 1. The default is LOGLOG.

STRATA= the strata variable for the PROC LIFETEST. Leave this parameter blank if there is no strata variable for PROC LIFETEST.

WORKFLOW OF THE MACRO

The workflow of the macro is straightforward.

1. Process the macro parameter PERCSTR to determine the number of percentiles the user requested. This number will be used to set up a loop over each percentile value.
2. Check if the macro parameter STRATA is blank. If it is blank, add a variable with value equal to 1 to the input dataset and assign macro parameter STRATA to be this variable name.
3. Run the PROC LIFETEST and output the ods table ProductLimitEstimates. Let's call the output dataset KM_CURV.
4. Based on the transformation specified in the macro variable CONFTYPE, use the variable SURVIVAL and STDERR from the dataset KM_CURV and the percentile specified in the macro parameter PERCSTR to calculate the $\frac{\boldsymbol{g}(\hat{\mathbf{S}}(\mathbf{t})-\mathbf{g}(\mathbf{1}-\mathbf{p}))}{\boldsymbol{g}'\left(\hat{\mathbf{S}}(\mathbf{t})\right)\hat{\boldsymbol{\sigma}}\left(\hat{\mathbf{S}}(\mathbf{t})\right)}$. The formula is displayed in the third column of table 1.
5. Use the condition $\left|\frac{g(\hat{S}(t)-g(1-p))}{g'\left(\hat{S}(t)\right)\hat{\sigma}\left(\hat{S}(t)\right)}\right| \leq z_{1-\frac{\alpha}{2}}$ to flag the values of AVAL, where α is specified by the macro parameter ALPHAPT.
6. The smallest value of AVAL is the lower confidence limit. The upper confidence limit is up to but not including the event time right after the largest value of AVAL.
7. Calculate the point estimate. We adapt the code from reference [3].
8. Print out the result.

VALIDATION OF THE MACRO WITH R

We run a small R program to validate the output of the SAS macro. These are the sample codes for the R program:

```
library(survival)

library(haven)

adtte <- read_sas("C:\\xxx\\xxx\\adtte.sas7bdat")

adtte <- adtte[(adtte$PARAMCD=="PFS" & adtte$PARQUAL=="CENTRAL" &
adtte$FASFL=="Y"),]


adtte$AVAL2 <- adtte$AVAL/(365.25/12)

adtte$CNSR2 <- ifelse(adtte$CNSR==1,0,1)
```

```
adtte <- adtte[c("AVAL2","CNSR2","COHORTN")]
fit <- survfit(Surv(AVAL2, CNSR2) ~ COHORTN,
               data = adtte, conf.type="log-log")
fit
quantile(fit, 0.75)
```

There are some interesting points about the R codes:

The survfit function offers three transformations. See reference [4]. The conf.type can take the value “plain”, “log”, and “log-log”, which are equivalent to the SAS option CONFTYPE=linear, log and loglog respectively.

In the formula for survfit function, the censor flag=0 is for censored case, and censor flag=1 is for event.

The upper confidence limits in the output produced by the R program don’t match the result from our SAS macro in some cases.

CONCLUSION

The point estimate and confidence limits for survival time percentiles other than the quartiles are not readily available from PROC LIFETEST. The SAS macro discussed in this paper was developed to get that work done quickly.

REFERENCES

[1] SAS/STAT 14.2 Users’ Guide, the LIFETEST Procedure

https://support.sas.com/documentation/onlinedoc/stat/142/lifetest.pdf

[2] Brookmeyer, R., and Crowley, J. (1982). “A Confidence Interval for the Median Survival Time.” Biometrics 38:29–41.

[3] Smith, Kris. 2019. The Kaplan-Meier Survival Time Percentiles. Available at

https://www.linkedin.com/pulse/kaplan-meier-survival-time-percentiles-chris-smith/

[4] R documentation for survfit

https://www.rdocumentation.org/packages/survival/versions/2.11-4/topics/survfit

ACKNOWLEDGMENTS

The author thanks the management at Daiichi Sankyo, Inc. for the support.

CONTACT INFORMATION

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

Hengwei Liu
Daiichi Sankyo, Inc.
211 Mount Airy Road
Basking Ridge, NJ 07920
Hengwei_liu@yahoo.com

APPENDIX

The SAS macro is presented in the appendix.

```
%macro ptile(indata=, percstr=, alphapt=0.05, conftype=LOGLOG,
strata=);
** process the percstr ** ;
data perc;
length percentile $3.;
perc="&percstr";
num=countw(perc,',');
if num=0 then do;
percentile=perc;
output;
end;
else do k=1 to num;
percentile=scan(perc,k,',');
output;
end;
run;

data _null_;
set perc end=eof;
i+1;
call symput(compress('percentile'||put(i,best.)),
trim(left(percentile)));
if eof then call symput('total', put(_n_, best.));
run;

%if &strata eq %then %do;
%let strata=addstra;
data adtte; set adtte;
addstra=1;
run;
%end;

proc sort data=adtte; by &strata;
run;

ods output ProductLimitEstimates=km_curv quartiles=quartiles;
proc lifetest data=&indata conftype=&conftype alphaqt=&alphapt
method=KM;
strata &strata;
time aval*cnsr(1); * 1 is censored;
run;

%do k=1 %to &total;
** consider five different transformations ** ;

%if %sysfunc(upcase(&conftype))=LOGLOG %then %do;
data km; set km_curv;
a1=log((-1)*log(survival))-log((-1)*log(%sysevalf(1-
&&percentile&k/100)));
```

```
a2=stderr/(survival*log(survival));
tranform=a1/a2;
run;
%end;

%else %if %sysfunc(upcase(&conftype))=LOG %then %do;
data km; set km_curv;
a1=log(survival)-log(%sysevalf(1-&&percentile&k/100));
a2=stderr/survival;
tranform=a1/a2;
run;
%end;

%else %if %sysfunc(upcase(&conftype))=LINEAR %then %do;
data km; set km_curv;
a1=survival-%sysevalf(1-&&percentile&k/100);
a2=stderr;
tranform=a1/a2;
run;
%end;

%else %if %sysfunc(upcase(&conftype))=LOGIT %then %do;
data km; set km_curv;
a1=log(survival/(1-survival))-log(%sysevalf(1-
&&percentile&k/100)/%sysevalf(&&percentile&k/100));
a2=stderr/(survival*(1-survival));
tranform=a1/a2;
run;
%end;

%else %if %sysfunc(upcase(&conftype))=ASINSQRT %then %do;
data km; set km_curv;
a1=arsin(sqrt(survival))-arsin(sqrt(%sysevalf(1-&&percentile&k/100)));
a2=stderr/(2*sqrt(survival)*sqrt(1-survival));
tranform=a1/a2;
run;
%end;

data km; set km;
where censor=0; * 0 is for event;
if (-1)*probit(%sysevalf(1-&alphapt/2))<=tranform<=probit(%sysevalf(1-
&alphapt/2)) then flag='Y';
obs=_n_;
run;

** get the lower limit ** ;
data km1(keep=&strata lower); set km;
where flag='Y';
by &strata;
if first.&strata;
lower=aval;
```

```
** get the upper limit **;
data km2(keep=&strata obs); set km;
where flag='Y';
by &strata;
if last.&strata;
obs=obs+1;

data km3(keep=&strata upper);
merge km km2(in=a);
by &strata obs;
if a;
upper=aval;
run;

** get the point estimate ** ;
data calcptiles(keep=&strata ptiles);
set km_curv(where=(censor eq 0)); * 0 is for event;
by &strata;
cutoff=%sysevalf(&&percentile&k/100);
retain _survival ptiles exact;

if first.&strata then do;
ptiles=.;
exact=.;
_survival=.;
end;
if survival>.z then _survival=survival;

if .z<round(_survival,10**-12)<=1-cutoff then do;
ptiles=min(aval, ptiles);

if exact=1 then do;
exact=.;
ptiles=(aval+ptiles)/2;
end;
if round(survival,10**-12) = 1-cutoff then exact=1;
end;
if last.&strata;
run;

data final&&percentile&k;
merge km1 km3 calcptiles;
by &strata;
percent="&&percentile&k";
transform="&conftype";
run;

data final;
set %if &k=1 %then %do; final&percentile1 %end; %else %do; final
final&&percentile&k; %end;;
run;
```

```
%end;

proc report data=final;
columns &strata percent ptiles transform lower upper;
define &strata/"Stratum";
define percent/"Percent";
define ptiles/"Estimate";
define transform/"Transform";
define lower/"LowerLimit";
define upper/"UpperLimit";
run;

%mend;

*example;
*%ptile(indata=adtte, percstr=%str(15,25,35,45,50, 55, 65, 75, 85,
95), alphapt=0.05, conftype=logit, strata=cohort);
```