%EventChart: A Macro to Visualize Data with Multiple Timed Events

Andrew Peng and J. Jack Lee, MD Anderson Cancer Center, Houston, TX

ABSTRACT

An event chart is a tool to visualize timeline data with multiple events. A typical event chart plots one horizontal line for each subject, with the x-axis as time intervals from a base event and the y-axis as the subject ID. Markers with different attributes are put on the line for different types of events.

This paper introduces a SAS® macro, %EventChart, which is capable of producing event charts with several user-friendly options. In the macro, the SGPLOT procedure is utilized to produce the plot and the macro takes advantage of the STYLEATTRS statement, which is newly available in SAS® 9.4, to specify colors and markers for different categories. This paper discusses the data structure of the input file, followed by an in-depth description of required and optional arguments. An application of the macro to an oncology study is provided.

INTRODUCTION

An event chart is a graph that shows multiple timed endpoints for each subject in a single chart. Figure 1 is a sample event chart that displays the results of a triathlon race. The total length of the line indicates the time in which a triathlete completes the race or the time until the athlete withdraws (DNF) or is lapped (LAP). Each interval between two consecutive events represents the time to complete that particular section. This chart comprehensively shows all timed events for every subject.

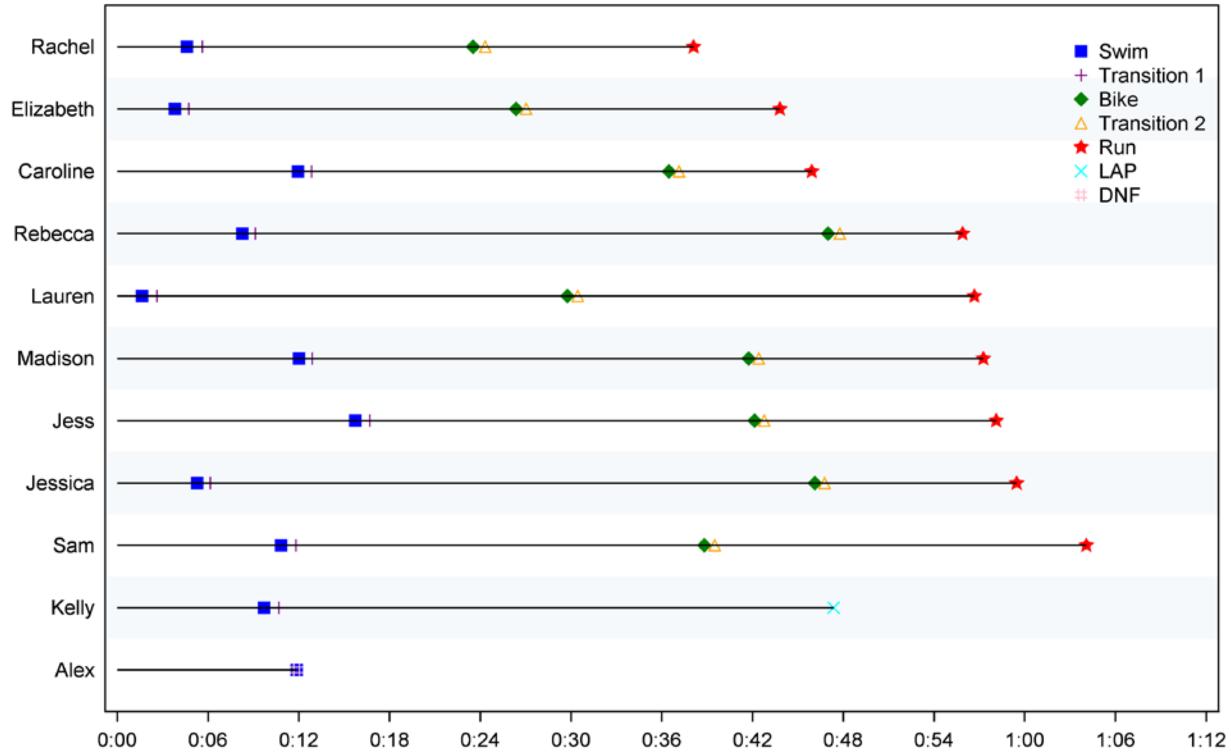


Figure 1. Sample Event Chart: Triathlon Race Results

Data with multiple events at several time points are commonly seen in clinical trials. A typical survival analysis often includes a Kaplan-Meier curve, which is useful in summarizing the time to a particular single event for the overall cohort or for separate groups within the cohort for comparison. However, it may be of interest to see multiple timed events for each subject graphically. For example, oncology clinical trial data for each patient can include the dates of randomization, first treatment, tumor response, recurrence, and departure from the study. An event chart is a solution for exhibiting this type of data.

Since its first introduction in SAS® 9.2, the functionality of PROC SGPLOT has been enhanced tremendously and it has become a powerful tool for producing graphs of many kinds. The new STYLEATTRS statement in SAS® 9.4 makes it a simple process to choose the colors and symbols for different categories. Prior to SAS® 9.4, the same task could be completed with PROC TEMPLATE; however, its syntax is not as user-friendly as that of PROC SGPLOT. This paper and the macro bring the STYLEATTRS statement into practical application.

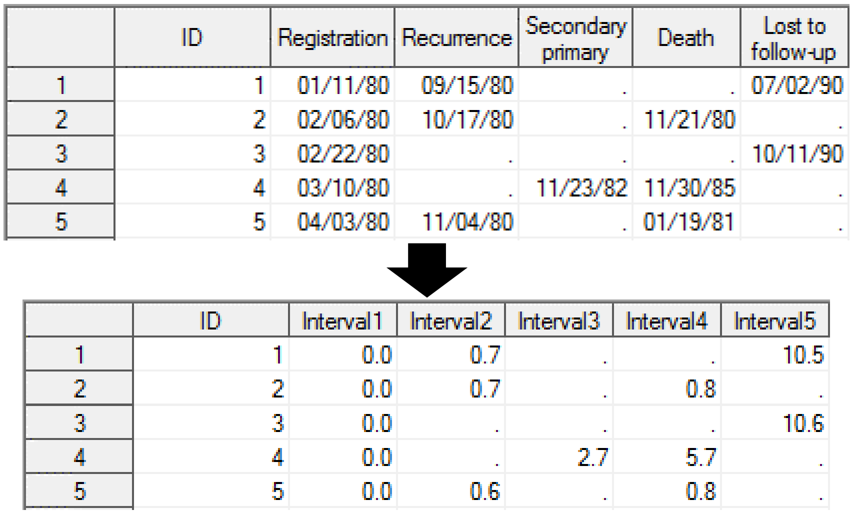
%EVENTCHART MACRO

The macro has four required arguments and five optional arguments. The user is only expected to prepare a data set of one row per subject, with a unique identifier and a few time intervals from a base event. The macro is able to care take of the work of creating the event chart. Below is the general appearance of the macro.

```
%EventChart(InputDst=,
            IDvar=,
            IntervalVars=,
            IntervalLabels=,
            Colors=,
            Symbols=,
            XAxisOpts=,
            YAxisOpts=,
            ODSGraphicsOpts=);
```

PREPARING THE INPUT FILE

The event chart demonstrates multiple timed events for each subject and the time is plotted on the x-axis. First of all, the input file needs to be one row per subject and to have a column for unique identifiers. The unique identifier variable can be either numeric or a character. Further, the macro will plot the subjects on the y-axis in the order of the input file so users can sort the data in any way it needs to be presented. As for the timed events, most commonly a raw file includes the dates of events, but the macro only intakes columns of time intervals from the base event, e.g., the registration date in a clinical trial. The time interval variables can be generated by subtracting the base time point from all the events of interest. The unit of time (day, week, etc.) needs to be taken into account when creating time intervals.



	ID	Registration	Recurrence	Secondary primary	Death	Lost to follow-up
1	1	01/11/80	09/15/80	.	.	07/02/90
2	2	02/06/80	10/17/80	.	11/21/80	.
3	3	02/22/80	.	.	.	10/11/90
4	4	03/10/80	.	11/23/82	11/30/85	.
5	5	04/03/80	11/04/80	.	01/19/81	.

	ID	Interval1	Interval2	Interval3	Interval4	Interval5
1	1	0.0	0.7	.	.	10.5
2	2	0.0	0.7	.	0.8	.
3	3	0.0	.	.	.	10.6
4	4	0.0	.	2.7	5.7	.
5	5	0.0	0.6	.	0.8	.

Display 1. Raw Data with Event Dates (top);
Processed Data with Time Intervals from Base Event (bottom)

Display 1 is an example of creating five time intervals from the original event dates. The unit of time is years. Note that creating a “dummy” first time interval of all zeros is optional. It gives users a chance to show the beginning time point, i.e., the date of registration to the study in this example.

REQUIRED ARGUMENTS

The macro has four required arguments. *InputDst* takes the prepared input data set and *IDVar* takes the identifier variable. All time intervals are listed in *IntervalVars*, separated by a space. Lastly, users need to specify a label for each event in *IntervalLabels*, separated by an underscore (_). The labels in *IntervalLabels* and their corresponding events in *IntervalVars* have to be in the same order. Table 1 lists all required arguments and their descriptions.

Argument	Description	Delimiter
InputDst	Input data set	N/A
IDVar	Unique identifier; can be numeric or character	N/A
IntervalVars	Variables of the time intervals	Space
IntervalLabels	Labels of the events	Underscore (_)

Table 1. Required Arguments

OPTIONAL ARGUMENTS

There are five optional arguments for the macro (Table 2). The *Colors* and *Symbols* are used to respectively assign colors and symbols to the events; they will be allocated in the same order as in *IntervalVars* (and *IntervalLabels* as well). Users can refer to https://goo.gl/YUJCw6 for a list of color names and https://goo.gl/3bcgxn for a full list of symbols supported by SAS®. *XAxisOpts* and *YAxisOpts* are used to insert options in XAXIS/YAXIS in PROC SGPLOT. The use of %STR function is highly recommended to mask special characters such as the equal sign to avoid errors during macro processing. By default, the author applies COLORBANDS and COLORBANDSATTRS options to create alternating light-colored bands, which visually helps to distinguish subjects. Finally, any desired options in the ODS GRAPHICS system statement can be put into *ODSGraphicsOpts*. %STR is also recommended to enclose the argument.

Argument	Description	Default value
Colors	Colors for the events	Blue Red Green Orange Purple Cyan Pink Brown
Symbols	Symbols for the events	Plus SquareFilled Triangle DiamondFilled X CircleFilled Asterisk StarFilled
XAxisOpts	Option(s) in XAXIS statement in PROC SGPLOT	(none)
YAxisOpts	Option(s) in YAXIS statement in PROC SGPLOT	%str(display=(noticks) colorbands=even colorbandsattrs=(color=cx94BDE1 transparency=.9))
ODSGraphicsOpts	Option(s) in ODS GRAPHICS system statement	%str(width=8in height=6in noborder)

Table 2. Optional Arguments

EXAMPLE CHART IN AN ONCOLOGY STUDY

Data are presented for 20 patients selected from a study of head and neck squamous cell carcinoma. The data set includes dates of registration to the study, tumor recurrence, secondary primary tumor (SPT), death, and loss to follow-up. The chart graphically summarizes all events of interest.

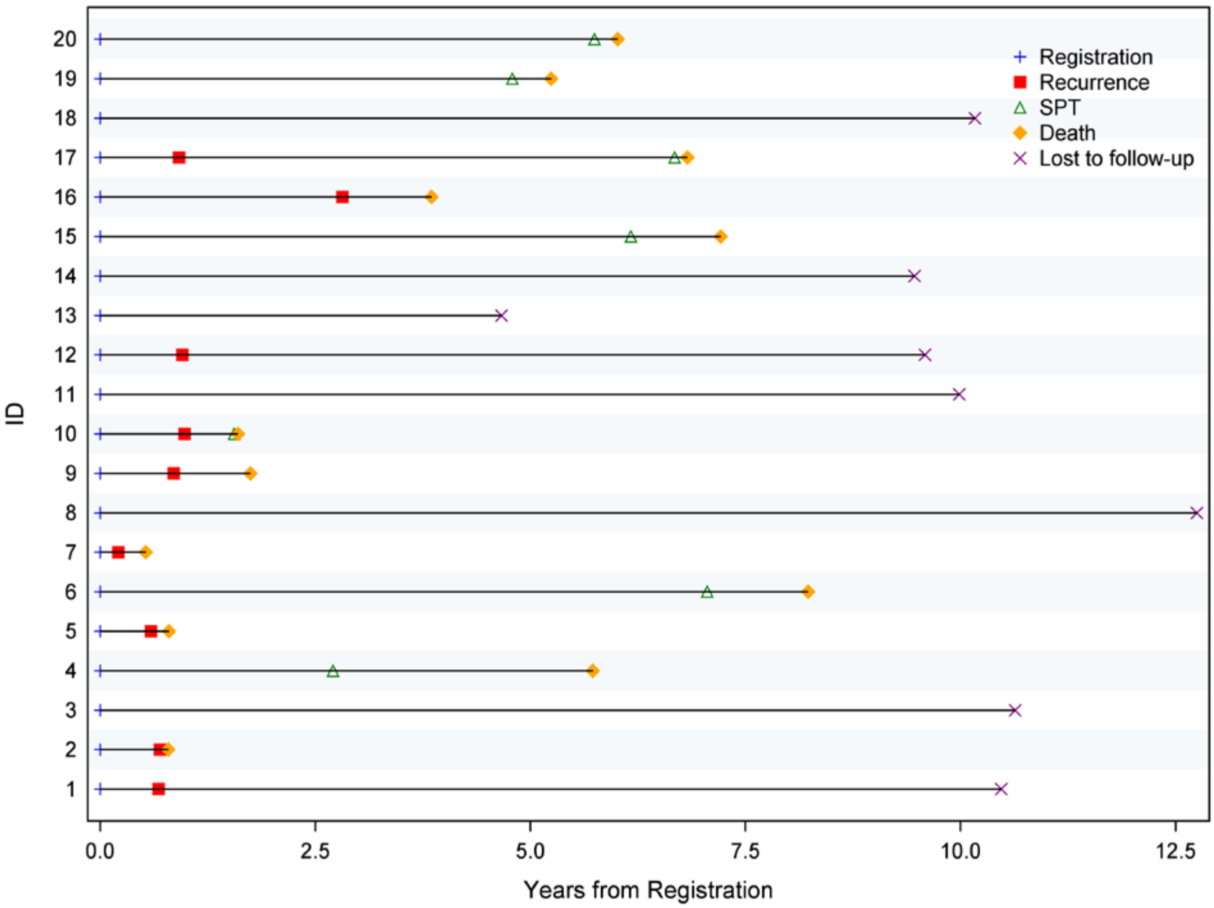


Figure 2. An Event Chart for an Oncology Study

```
%EventChart(
    InputDst=oncology,
    IDvar=ID,
    IntervalVars=Interval1 Interval2 Interval3 Interval4 Interval5,
    IntervalLabels=Registration_Recurrence_SPT_Death_Lost to follow-up,
    XAxisOpts=%str(label="Years from Registration"));
```

The default setting of the macro nicely creates the chart. The only optional argument specified here is the x-axis label in *XAxisOpts*. Occasionally users need to extend the graph horizontally by specifying larger VALUES= in the *XAxisOpts* in order to create more space to accommodate the automatic legend of the events. Note that the macro put the IDs in the order of the input data set. In this example, the input data set happens to be sorted by ID. Users can sort the input file in any order they want and the macro will display it that way. Also note that not every subject has data for all five events; that does not impede plotting the graph.

CONCLUSION

An event chart is quite handy for visualizing timed events, especially when the sample size is reasonably small. These graphs can be applied to any field with timeline data. In medical research, the common Kaplan-Meier plot provides summarized information of a single time-to-event endpoint. The event chart offers additional detailed looks at multiple time-to-event endpoints for every subject. The event chart can also be used to check the data for errors in the timed events before running an analysis.

The macro presented in this paper allows SAS® users to generate event charts in a straightforward manner. Several optional arguments are available so users have great flexibility to customize the plots. Prior to invoking the macro, users can specify other statements to enhance the plot, such as TITLE or ODS HTML.

REFERENCES

- Goldman, A. I. (1992). EVENTCHARTS: Visualizing Survival and Other Time-Events Data. *The American Statistician*, 46(1), 13-18.
- Lee, J.J., Hess, K.R., & Dubin, J.A. (2000). Extensions and Applications of Event Charts. *The American Statistician*, 54(1), 63-70.
- SAS® 9.4 ODS Graphics: Procedures Guide, Fifth Edition.

RECOMMENDED READING

Matange, Sanjay “Graphically Speaking - Data Visualization with a Focus on SAS ODS Graphics” http://blogs.sas.com/content/graphicallyspeaking/

CONTACT INFORMATION

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

Andrew Peng
The University of Texas MD Anderson Cancer Center
speng1@mdanderson.org

Appendix: %eventchart macro code

```
%macro EventChart(
    InputDst=,
    IDvar=,
    IntervalVars=,
    IntervalLabels=,
    Colors=blue red green orange purple cyan pink brown,
    Symbols=Plus SquareFilled Triangle DiamondFilled X CircleFilled Asterisk
StarFilled,
    XAxisOpts=,
    YAxisOpts=%str(display=(noticks) colorbands=even
colorbandsattrs=(color=cx94BDE1 transparency=.9)),
    ODSGraphicsOpts=%str(width=8in height=6in noborder));
*** clean residual _plot_ data set;
proc datasets lib=work nowarn nolist;
    delete _plot_;
quit;

*** prepare data for plotting;
data _plot_;
    set &InputDst;
    length _ECGroup_ $50;

    *** Specify both ends for the highlow chart;
    array max_inter{*} &IntervalVars;
    _HighLowStart_ = 0;
    _HighLowEnd_ = max(of max_inter{*});

    *** Output one subject and one interval per row;
    %let i = 1;
    %do %while (%scan(&IntervalVars, &i) ne );
    %let IntervalVar = %scan(&IntervalVars, &i);
    %let IntervalLabel = %qscan(&IntervalLabels, &i, '_');
    _ECInt_ = &IntervalVar;
    _ECGroup_ = "&IntervalLabel";
    output;
    %let i = %eval(&i + 1);
    %end;
run;

ods graphics / attrpriority=none &OdsGraphicsOpts;
PROC SGPLOT DATA=_plot_;
    STYLEATTRS datacontrastcolors=(&Colors) datasymbols=(&Symbols);
    SCATTER y=&IDvar x=_ECInt_ / group=_ECGroup_ grouporder=data
name='scatter';
    HIGHLOW y=&IDvar high=_HighLowEnd_ low=_HighLowStart_;
    KEYLEGEND 'scatter' / title=' ' location=inside noborder across=1;
    XAXIS &XAxisOpts;
    YAXIS type=discrete &YAxisOpts;
RUN;
%mend;
```