ERROR BARS WITH LINE GRAPHS AND BAR CHARTS

Shuching Shaw, NSI Technology Services Inc.
Bernard Most, NSI Technology Services Inc.

ABSTRACT

It is useful to have full control of the plotting of error bars when displaying statistical data. For example, a problem often arises in overlay plotting of two or more curves which illustrate mean response and confidence intervals over the same set of abscissas("times"): the confidence intervals may overlap leading to graphic clutter and/or ambiguity in interpretation. An algorithm which removes such clutter according to well defined rules and enhances interpretability is described and illustrated. Implementing the algorithm entailed development of code to plot the data using PROC GPLOT in SAS/GRAPH which leads to other benefits.

Disclaimer: Although the research described in this document has been supported by the United States Environmental Protection Agency through contract number 68-02-4450 to NSI Technology Services Inc., and subjected to Agency technical peer review, it does not necessarily reflect the views of the Agency and no official endorsement should be inferred. Mention of trade names or commercial products does not constitute endorsement or recommendation for use.

Introduction :

A problem often arises in overlay plotting of two or more curves which illustrate mean response and error bars over the same set of abscissas ("times"): the error bars may overlap leading to graphic clutter and/or ambiguity in interpretation. The algorithm described below removes such clutter according to well defined rules and may enhance interpretability. It is applied independently at each "time" value of interest. More generally the "error bars" of interest depict confidence intervals but the error bar terminology is often used below for convenience.

Sections 1 through 4 refer to line graphs. Section 5 applies the basic plotting techniques developed for line graphs to bar charts; here there is no question of overlapping error bars. Section 6 points out some advantages and limitations of the techniques described in addition to making a comment on interpretation of the graphical results.

1. Data Input :

The main macro expects observations consisting of group identifier, abscissa value, response mean, and response interval half-length (e.g. one standard error of the mean). For the typical situation in which the data consist of groups of (x,y) pairs where x is, say, time and y is, say, response with a third class variable, say, dose in each observation being a group identifier, a preliminary step is to use PROC MEANS to get the mean and standard error of response at each time and dose.

2. Algorithm for removing clutter :

At any given abscissa there are several intervals(and means) being represented. Each has an upper limit(UL), a mean(M), and a lower limit(LL). Each M is plotted. Rank and consider intervals sequentially according to UL in descending order. Test for "including" the next UL: include it if it is lower than all preceeding LL's (clearly we "include" the highest UL). Test for including the lower limit of the current interval: do not include it if it is between any LL and UL pair; else include it. Draw lines between each mean and its corresponding UL and/or LL which have been included.

Clearly, the algorithm for removing the "clutter" is not unique. If, for example, pairwise comparisons between each response and a control dose response were of primary interest, an alternate algorithm would be preferable.

3. Graphical Output :

The figures show several ways to plot the same sample data. Preprocessing the raw data (say, using PROC MEANS) to input the error bar lengths directly, the HILOTJ option yields figure 1 (note that symbols can't be used). In figure 2 the overlay plots have been enhanced with symbols and in figure 3 the algorithm has been applied to remove clutter(note

that all means are still shown but that only selected error bars are shown).

4. **Outline of SAS Code :**

The following steps outline the SAS code to draw line graphs with clutter removed:

- Read input data; perform preliminary calculations(e.g. using PROC MEANS) if required.
- Compute upper and lower limits of confidence bands.
- Implement algorithm to flag "error bars" which will be plotted and those which will not be plotted.
- Overlay plot(using PROC GPLOT): connected means, included error bars, tops/bottoms of included error bars. Option SKIPMISS in GPLOT is used to skip the missing values.

5. **Bar Charts :**

The techniques used to draw the line graphs described above are readily applied to drawing bar charts. The BAR Function in ANNOTATE is used to draw the bar patterns with PROC GPLOT (Figure 4).

6. **NOTES on the techniques :**

- The code size can be reduced significantly if the graphics output device is not capable of generating color graphics.
- Not shown, due to space limitations, is code (similar to that necessitated to implement the algorithm) to produce a correct overlay plot of curves which have non-identical sets of abscissas.
- Alternate algorithm for removing clutter (but, say, requiring at least one half-bar connected to each mean) may be implemented using the given code as a basis.
- The subject of just how long to make the error bars so that non-overlap is readily interpretable has not been addressed. This falls under the general heading of multiple comparison problems in statistics. A simple special case is worth noting; namely, a pairwise comparison of two means each observed with error which is distributed normally with the same variance. Error bars of half-length one sem(standard error of the mean) correspond to a 68.26% confidence interval for each mean; taking non-overlap of such error bars as being statistically significant corresponds to testing the null hypothesis of no significant difference between the means at an alpha level of 0.157. For an alpha level of 0.05 the error bar half-lengths would have to be 1.386 * sem. These figures are based on the fact that the standard error of the difference of the observed means under the null hypothesis is $\sqrt{2}$ times the individual sem's.

Authors:

Shuching Shaw/Bernard Most
NSI Technology Services, Inc.
P. O. Box 12313
RTP, NC 27709
(919)541-1837/(919)541-2390


```
/*-------------------------------------------------------------
Data set demo.dat. Each observation includes time,
trt, mean and standard error of the mean
-------------------------------------------------------------*/
3 1 336.1 2.2
3 2 330.3 3.4
3 3 327.7 2.5
3 4 324.5 2.5
4 1 344.5 4.4
4 2 336.9 4.8
4 3 330.8 3.7
4 4 323.9 4.3
5 1 351.7 4.5
5 2 344.2 2.2
5 3 340.9 2.4
5 4 336.2 3.5
6 1 349.3 3.2
6 2 344.3 3.0
6 3 337.4 5.0
6 4 322.2 4.2
7 1 363.9 5.0
7 2 354.0 2.0
7 3 351.0 2.0
7 4 330.2 3.6
```

Figure 1

Usual Plot with HILOTJ option

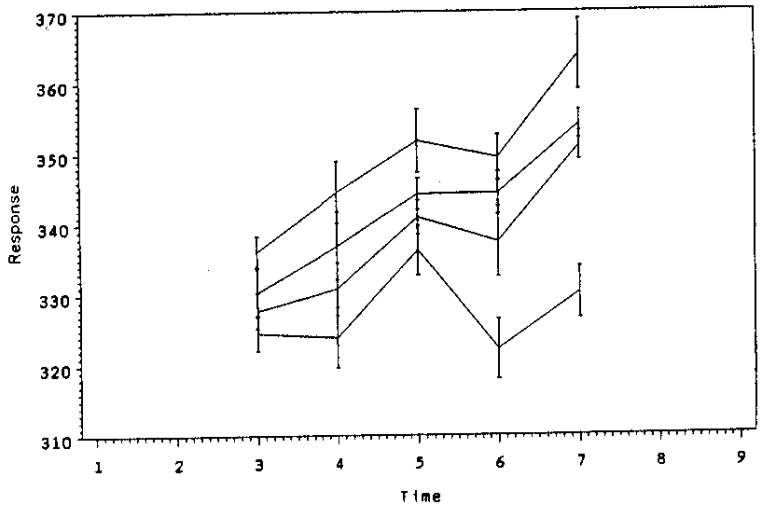


Figure 2

Enhanced Plot – with Overlap

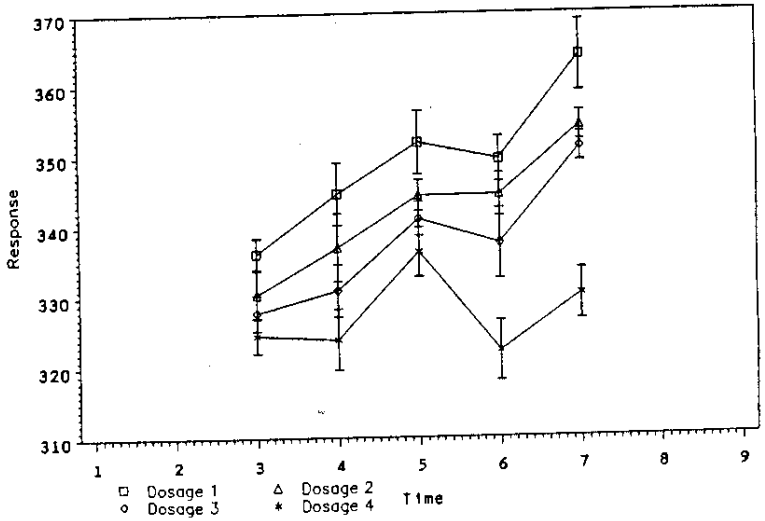


Figure 3

Enhanced Plot – Uncluttered

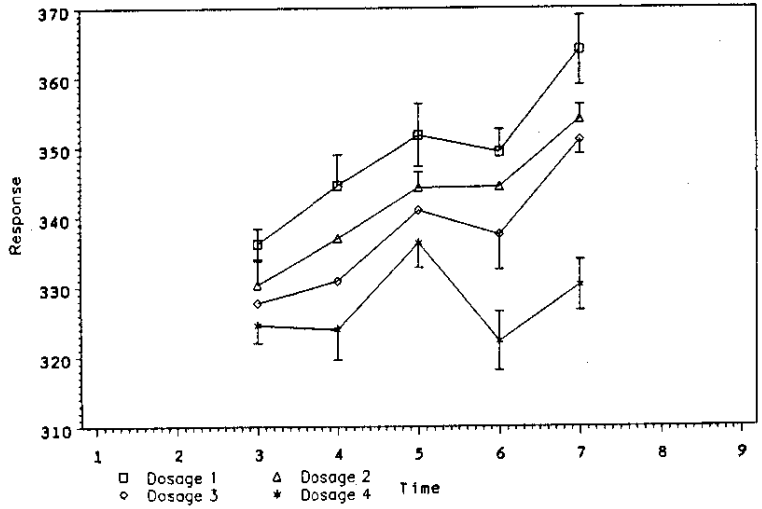
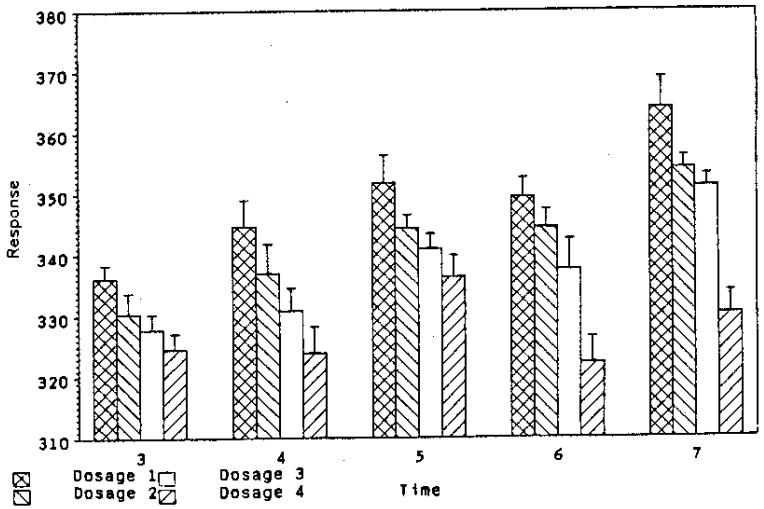


Figure 4

Bar Chart with Error Bars



1421

```
/*-------------------------------------*/
/*                                     */
/*            lineplot                 */
/*                                     */
/* This program makes an overlay line  */
/* graph with error bars. Code         */
/* includes the "compare" macro.       */
/*                                     */
/*-------------------------------------*/

/*-------------------------------------
   read input data. Re-arrange the data
   file if necessary, so trt will have
   the values 1,2,3...etc.
-------------------------------------*/
data data1;
infile '[]demo.dat';
input time trt trtmean std;

/*----c is the number of groups-------*/
%let c=4;

/*-------------------------------------
   Compute the upper and lover limits of
   each groups. Each observation
   includes time, mean, upper limit and
   lower limit of each group.
-------------------------------------*/
proc sort data=data1;
by time trt;

data data2;
array mean{&c} mean1-mean&c;
array ul{&c} ul1-ul&c;
array ll{&c} lll-ll&c;
do i=1 to &c;
   set data1;
   by time;
   mean{i}= trtmean;
   ul{i}=trtmean+std;
   ll{i}=trtmean-std;
   if last.time then return;
end;
keep time mean1-mean&c ul1-ul&c
   lll-ll&c;

/*-------------------------------------
   This macro applies a well definied
   algorithm to examine whether to
   include the upper or lover limits in
   the plot
-------------------------------------*/
%include compare;
/*------use ANNOTATE to enhance the
   graphical output--------------------*/

proc gplot data=data4;
plot

/*----plot the mean of each curve-----*/
mean1*time mean2*time mean3*time
   mean4*time

/*-connect the upper and lover limits-*/
hh1*time hh2*time hh3*time hh4*time

/*---put symbol - at the upper and lover
   limits-------*/
pp1*time pp2*time pp3*time pp4*time

/haxis=axis1 vaxis=axis2 overlay
   skipmiss  frame annotate=legend;
symbol1 i=join l=1 v=square c=black;
symbol2 i=join l=1 v=triangle c=green;
symbol3 i=join l=1 v=diamond c=red;
symbol4 i=join l=1 v=star c=blue;
symbol5 i=join l=1 v=none c=black;
symbol6 i=join l=1 v=none c=green;
symbol7 i=join l=1 v=none c=red;
symbol8 i=join l=1 v=none c=blue;
symbol9 i=none f=cgreek v=- c=black;
symbol10 i=none f=cgreek v=- c=green;
symbol11 i=none f=cgreek v=- c=red;
symbol12 i=none f=cgreek v=- c=blue;

/*-customize axes' labels and titles--*/


/*-------------------------------------*/
/*                                     */
/*            compare                  */
/*                                     */
/* This macro uses the algorithm to    */
/* examine whether to include the      */
/* upper or lover limit in the plot    */
/*                                     */
/*-------------------------------------*/

/*-------------------------------------
   Apply the algorithm here. If an
   original overlay plot is desired,
   skip to next data set (data4; set
   data2)
-------------------------------------*/
data data3; set data2;
array c{&c} c1-c&c;
array ul{&c} ul1-ul&c;
array ll{&c} lll-ll&c;
array mean{&c} mean1-mean&c;
array mflag{&c} mflag1-mflag&c;
array a{&c} a1-a&c;
array b{&c} b1-b&c;
by time;
/*-------------------------------------
   Sort the upper limits of all curves
   in descending order. Flag to 0 after
   the sorting. C{k} is the rank of
   curve k.
-------------------------------------*/
do i=1 to &c;
   mflag{i} = 1;
end;

do k=1 to &c;
   ma=-9999;
   do j=1 to &c;
      if mflag{j} = 1 then
         do;
         ma=max(ma,ul{j});
         if (ma=ul{j}) then
            c{k}=j;
         end;
   end;
   mflag{c{k}}=0;
end;
/*-------------------------------------
   Do not include the lover limit if it
   is between any pair of upper and
   lover limits
-------------------------------------*/
```

```
do k=1 to &c;
   a{k}=1;
   do l=1 to &c;
      if ll{k} < ul{l} and
         ll{k} > ll{l} then
         a{k}=0;
   end;
end;
/*-------------------------------------
   Include the highest upper limit.
   Include the upper limit if it is
   lower than all preceeding lower
   limit. i.e. do not include it if it
   is larger than any preceeding lower
   limit.
-------------------------------------*/
b{1}=1;
do k=2 to &c;
   b{k}=1;
   do g=k-1 to 1 by -1;
      if ul{c{k}} > ll{c{g}} then
         b{k}=0;
   end;
end;

do k=1 to &c;
   if a{k}=0 then ll{k}=.;
   if b{k}=0 then ul{c{k}}=.;
end;
keep time mean1-mean&c ul1-ul&c
   lll-ll&c;

proc sort data=data3;
   by time;

/*-------------------------------------
   Assign UL, MEAN, LL and missing
   values to array HH (lines); and
   assign UL and LL to array PP.
-------------------------------------*/
data data4;
array ul{&c} ul1-ul&c;
array ll{&c} lll-ll&c;
array mean{&c} mean1-mean&c;
array hh{&c} hh1-hh&c;
array pp{&c} pp1-pp&c;
set data3;
do i=1 to &c;
   mean{i}=mean{i};
   hh{i}=ul{i};
   pp{i}=ul{i};
end;
output;
do i=1 to &c;
   hh{i}=mean{i};
   pp{i}=ll{i};
end;
output;
do i=1 to &c;
   hh{i}=ll{i};
end;
output;
do i=1 to &c;
   hh{i}=.;
end;
output;
keep time mean1-mean&c hh1-hh&c
   pp1-pp&c;
```

```
/*-------------------------------------*/
/*                                     */
/*              bar                    */
/*                                     */
/* This program draws a bar chart with*/
/* error bars using ANNOTATE with PROC*/
/* GPLOT. Code includes the "chart"   */
/* macro.                             */
/*                                     */
/*-------------------------------------*/

/*-------------------------------------
   input data file, re-scale the data
   set if necessary, so the time and trt
   values are 1,2,3...etc.
-------------------------------------*/
data data1;
infile '[]demo.dat';
input time trt mean std;

/*-------------------------------------
   ntreat is the number of bars in each
   group
   ngroup is the number groups
   vmin is the minimum range of vertical
   axis, this value will be used in the
   ANNOTATE data set to create bar
   pattern
-------------------------------------*/
%let ntreat=4;
%let ngroup=5;
%let vmin=310;

/*-------------------------------------
   this macro uses "bar" in an ANNOTATE
   data set to construct bars in the
   plot
-------------------------------------*/
%include chart;

/*----use ANNOTATE to enhance the
   graphical output-------------------*/

proc gplot data=data2;
plot

/*-connect the upper and lower limits-*/
hh1*day hh2*day hh3*day hh4*day

/*--put symbol - at the upper and lover
   limits-------*/
hh1*day hh2*day hh3*day hh4*day

/frame skipmiss overlay vaxis=axis2
   haxis=axis1 annotate=leg;

symbol1 l=1 i=join v=none c=black;
symbol2 l=1 i=join v=none c=green;
symbol3 l=1 i=join v=none c=red;
symbol4 l=1 i=join v=none c=blue;
symbol5 i=none f=cgreek v=- c=black;
symbol6 i=none f=cgreek v=- c=green;
symbol7 i=none f=cgreek v=- c=red;
symbol8 i=none f=cgreek v=- c=blue;
```

```
/*-customize the axes' labels and
   titles--*/
axis1 label=(c=black 'Time')
      order=0 to &ran by 1
      minor=none
      major=none
      offset=(0.5 cm, 0.5 cm)
      value=('' '' '3' '' '' '' '' ''
   '4' '' '' '' '' '' '5' '' '' '' '' ''
   '6' '' '' '' '' '' '7' '' '');
axis2 label =none
      offset=(0 cm, 0 cm)
      order= &vmin to 380 by 10;


/*-------------------------------------*/
/*                                     */
/*             chart                   */
/*                                     */
/* This macro uses function "BAR" in   */
/* an ANNOTATE data set to construct   */
/* the bars in the plot.               */
/*                                     */
/*-------------------------------------*/

/*-------------------------------------
   this data set assigns two macro
   variables, va and ran, to control the
   scale for all combination of # of
   groups with # of treatments in each
   group
-------------------------------------*/
data _null_;
    t = &ntreat+2;
    r=t*&ngroup-2;
    call symput('va',t);
    call symput('ran',r);
run;

/*-------------------------------------
   Assign UL and the mean to array HH.

   The X(abscissa) axis must be rescaled
   to accommodate multiple 'groups' at
   each 'time'.; a new variable, day, is
   created for each (group,time)
   pair(the X axis values will range
   from 0 to (ntreat+2)*ngroup-2).  Here
   the days are set to the ordinal
   numbers with horizontal ends of each
   bar defined as 0.5 units from the
   center of the bar; some care is then
   required in an axis label statement
   to insert proper labels(see example
   in bar program).
-------------------------------------*/
data data2(keep=day hh1-hh&ntreat);
array hh{&ntreat} hh1-hh&ntreat;
set data1;
day=&va*(time-1)+trt-0.5;
hh{trt}=mean+std; output;
hh{trt}=mean; output;
hh{trt}=.; output;

proc sort data=data1;
by trt;
```

```
/*-------------------------------------
   construct bars using function BAR in
   ANNOTATE. Refer to SAS/GRAPH User's
   Guide version 5, page 58 for the
   pattern selections
-------------------------------------*/
data leg(keep=function style color x y
   xsys ysys line when);
length function $8 color $8;
xsys='2'; ysys='2'; when='A';
set data1;
by trt;

function='move'; x=&va*(time-1)+trt-1;
   y=&vmin; style=''; output;

function='bar'; x=&va*(time-1)+trt;
   y=mean; line=0;

if trt=1 then do; style='X1';
   color='black'; end;
if trt=2 then do; style='L1';
   color='green'; end;
if trt=3 then do; style='E';
   color='red'; end;
if trt=4 then do; style='R1';
   color='blue'; end;
output;
```