PhUSE 2007

Paper IS03

Automating GREPLAY to display multiple Graphs per page

KarolAnne Fitzpatrick, ICON Clinical Research, Dublin, Ireland
Erika Daly, ICON Clinical Research, Dublin, Ireland

ABSTRACT

In Phase I studies, particularly with PK data, we often have a lot of information to summarize for each subject. Graphs are the natural choice for presenting this data. It may be desirable to compare responses across subjects or even within subjects across time. PROC GREPLAY in SAS allows you to present several separate graphs on the same page. In this paper I will describe some of the ways that GREPLAY can contribute to the presentation of graphical information. I describe the code and options necessary to implement GREPLAY and also outline a macro to automatically repeat the procedure for multiple pages. In addition, some guidelines on preparing the component plots will also be given, including axis scaling and annotations.

INTRODUCTION

In Phase I studies we often have a lot of information to summarize for each subject, particularly with PK data on drug absorption and release. Displaying this data in tabular form makes it very dense to review: graphs are the natural choice for presenting this data, with each subject's response graphed separately. However, it may be desirable to compare responses across subjects or even within subjects across time. Rather than having several pages each with one graph on them, PROC GREPLAY in SAS allows you to present several separate graphs on the same page. In this paper I will describe some of the ways that GREPLAY can contribute to the presentation of graphical information.

ONE PAGE OUTPUT

Assume we want to display graphs for 4 subjects on the same page. Firstly we tell PROC GREPLAY to create a template (called PK1) with four "panels" for graphs within one larger panel (the page).

```
proc greplay igout=work.gseg gout=work.gseg tc=work.tmplt nofs;
    tdef PK1 1 / llx=0   lly=0    /*full panel for main titles */
                 ulx=0   uly=100
                 lrx=100 lry=0
                 urx=100 ury=100

             2 / llx=0   lly=50   /*left upper panel*/
                 ulx=0   uly=88
                 lrx=50  lry=50
                 urx=50  ury=88

             3 / llx=50  lly=50   /*right upper panel*/
                 ulx=50  uly=88
                 lrx=100 lry=50
                 urx=100 ury=88

             4 / llx=0   lly=6    /*left lower panel*/
                 ulx=0   uly=44
                 lrx=50  lry=6
                 urx=50  ury=44

             5 / llx=50  lly=6    /*right lower panel*/
                 ulx=50  uly=44
                 lrx=100 lry=6
                 urx=100 ury=44
                 ;
```

```
run;
quit;
```

TDEF defines the template, PK1. The first panel template defines the area of the whole page (0 – 100%). Each of the following panel templates defines the location and size of one of the four graphs to be displayed on the page. The lower left hand corner of the panel is defined by the x and y co-ordinates (llx, lly), whilst co-ordinates (ulx, uly) define the upper left corner. Similarly, the lower right corner of the panel is defined by co-ordinates (lrx, lry), whilst co-ordinates (urx, ury) define the upper right corner. The co-ordinate values are all percentages of the total area. Hence panel 1 being 0-100 as this is the complete page.

The IGOUT= option specifies the input catalogue which contains the graphs that you want to present, in this case the GSEG catalog, which is automatically created by SAS on running a graph, whilst the GOUT = option specifies the output catalog. The TC= option defined the name of the SAS catalogue where the template is to be stored so that this can be used at a later time. To ensure that the GSEG catalogue contains only the graphs that you will be working with it is a good idea to delete any old graphs that may be in the catalogue using the following code prior to plotting the data:

```
proc greplay igout=work.gseg nofs;
      delete _all_;
run;
```

CREATING THE PLOTS

Now that SAS knows the layout we want, we need to create the graphs to populate the panels. These graphs are created using the standard SAS graph commands.

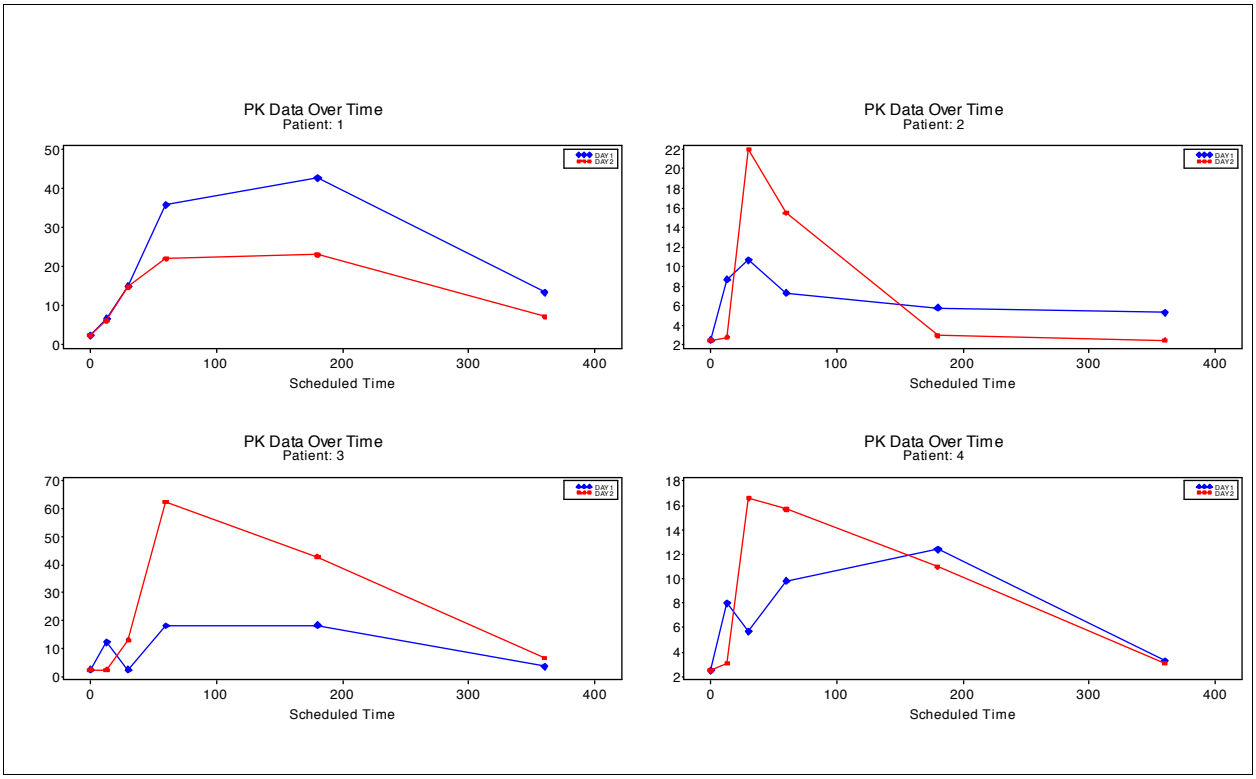
```
proc gplot data=dataset;
    plot response*time=day / frame legend=none vaxis=axis1 haxis=axis2;
    by subject;
run;
quit;
```

These plots are automatically numbered in the SAS catalogue. The first output from PROC GPLOT is called “GPLOT”, the second is called “GPLOT1”, the third “GPLOT2” and so on. The order of the outputs depends on the order of the “by” variables (in this case, subject number).

Now that we have created the template and the graphs, we use PROC GREPLAY to put the graphs into the appropriate panels of the template.

```
proc greplay igout=work.gseg gout=work.gseg tc=work.tmplt nofs;
    template PK1;
    treplay 2:gplot
            3:gplot1
            4:gplot2
            5:gplot3
            ;
quit;
```

This code assigns the first output produced (“GPLOT”) to the space defined by the second panel (the first panel is the whole page).



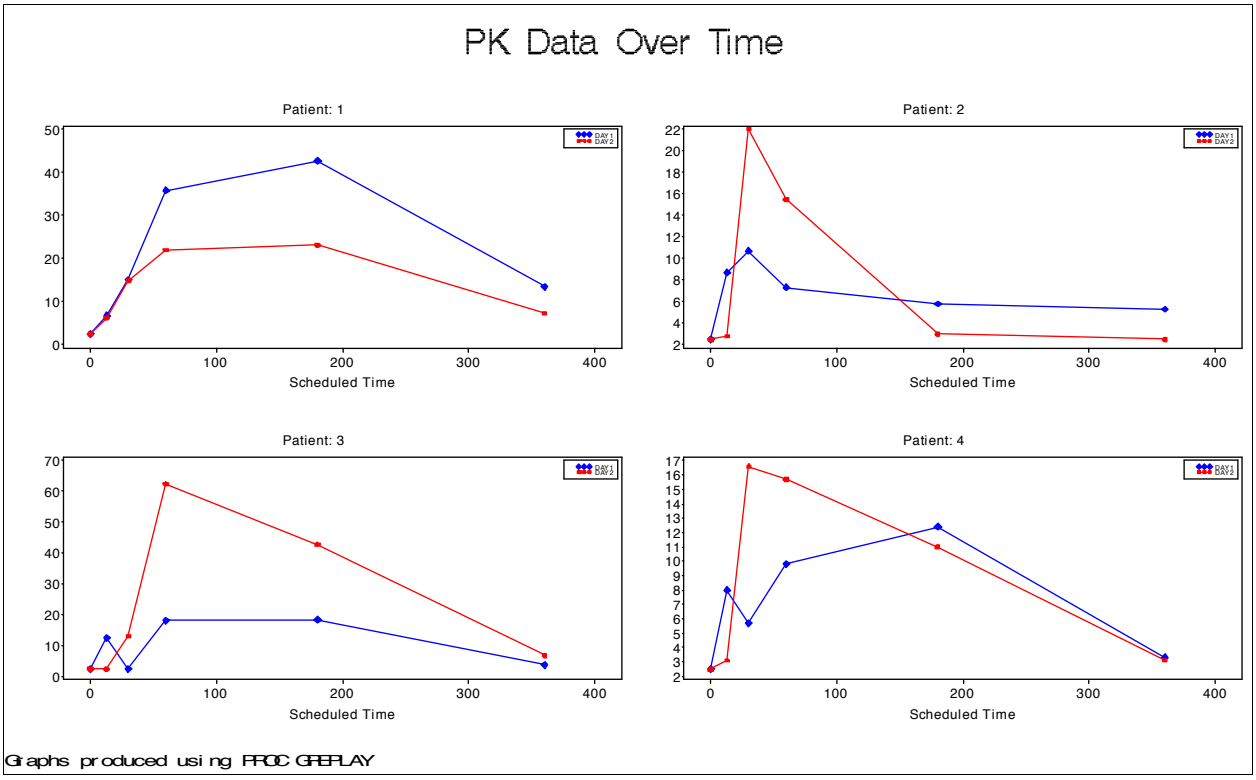
Instead of having the subject number and graph description on each of the four graphs, we could put one title across the whole page by editing the layout of the first panel, the "page" that the four graphs appear on. To format the first panel, we use PROC GSLIDE to control its appearance.

```
proc gslide;
    title1    h=2 justify=center "TITLE OF GRAPH" ;
    footnote1 h=1 justify=left   "Graphs produced using PROC GREPLAY";
run;
quit;
```

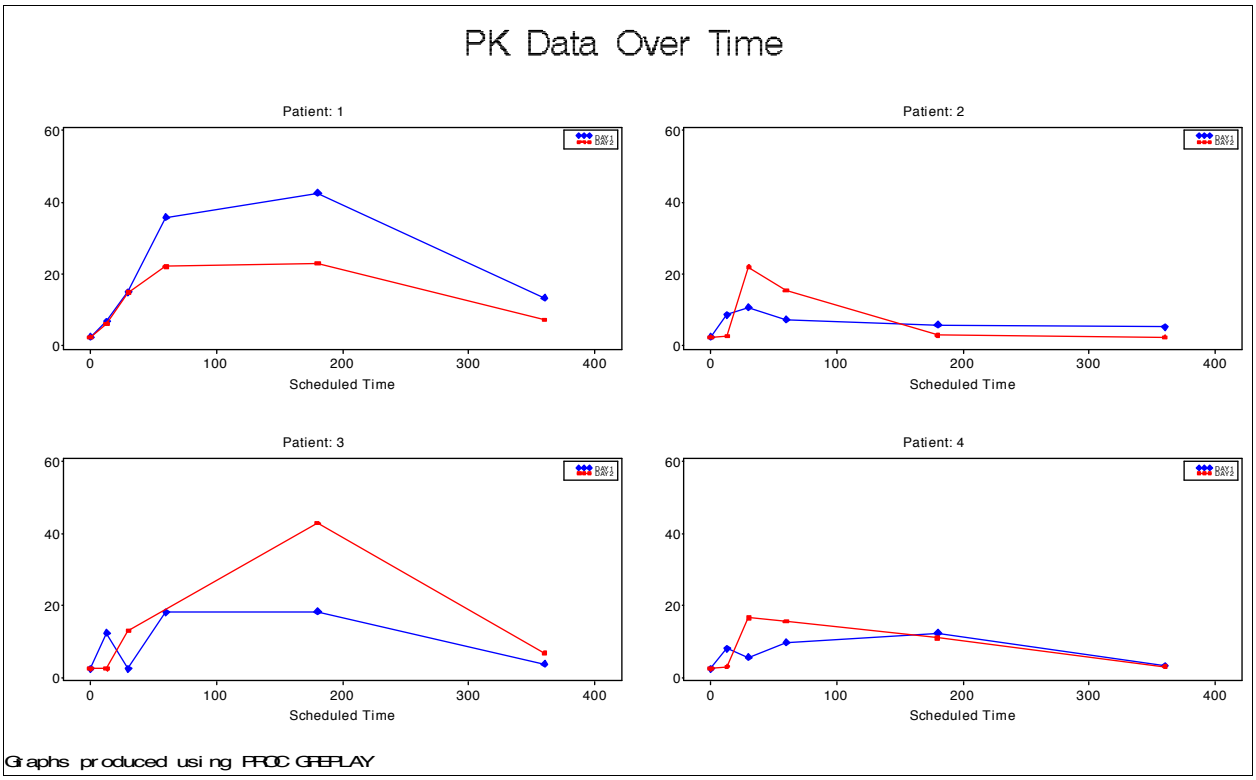
The title statements for each plot are removed or, in this case, updated to present only the subject number as the overall title will be presented using the code above. The output created from the GSLIDE procedure is called "GSLIDE". To add this to the overall page, we re-run GREPLAY including this slide as the output to fill panel 1 applet.

```
proc greplay igout=work.gseg gout=work.gseg tc=work.tmplt nofs;
    template PK1;
    treplay 1:gslide
            2:gplot
            3:gplot1
            4:gplot2
            5:gplot3
            ;
quit;
```

Now the page looks like this:



Now that these graphs are all on one page, comparison is made a lot easier. However, the scale on the Y-axis differs between graphs. In order to compare these graphs visually and with ease the graph axis should be to the same scale. To do this we find the maximum and minimum values for the Y-axis and scale the graphs accordingly.
If we re-run the graphs and greplay we get the following:



ANNOTATING THE PLOTS

To label the graphs we can use the SAS Graph annotate facility. This is a dataset which contains graphic commands which are used to customize graphs. These commands are contained within macros which are collectively known as the annotate macros. Prior to using any of the annotate macros the following must be run:

```
%annomac;
```

In our example the x-axis has tick marks at 100 mark intervals. The data points are at irregular and non-uniform intervals. Therefore to aid the reading of the graph it would be better to have each data point labeled. We are going to present and label tick marks of the x-axis which are the timepoints at which the measurements occur.

These values are formatted values and we are going to apply this format to the points using one of the annotate macros.

```
proc format;
    value timep
        0 = '- 1m'
        15 = '+ 15m'
        30 = '+ 30m'
        60 = '+ 1h'
        180 = '+ 3h'
        360 = '+ 6h'
    ;
run;
```

The annotate facility contains a number of macros, such as `%label, %line`, some of which can be seen below:

```
data annodata;
    set final;
    by subject;
    length text $ 16;
    xsys='2'; ysys='5';
    %label(timep, 13, "I", black, 0, 0, 2, 'Arial', E); * tick marks;
    xsys='2'; ysys='5';
    if timep = 0 then do;
        %label(timep-4, 9, put(timep,timep.),black,45,0,2, 'Arial', D);
    end;
    else if timep = 15 or timep = 30 then do;
        %label(timep-1, 9, put(timep,timep.),black,45,0,2, 'Arial', D);
    end;
    else do;
        %label(timep-2, 9, put(timep,timep.), black,45,0,2, 'Arial', D);
    end;
    if last.subject then do;
        %label(180, 0, "Scheduled Time", black,0,0,2, 'Arial', 8);
    end;
run;
```

The `%LABEL` macro takes the following format:

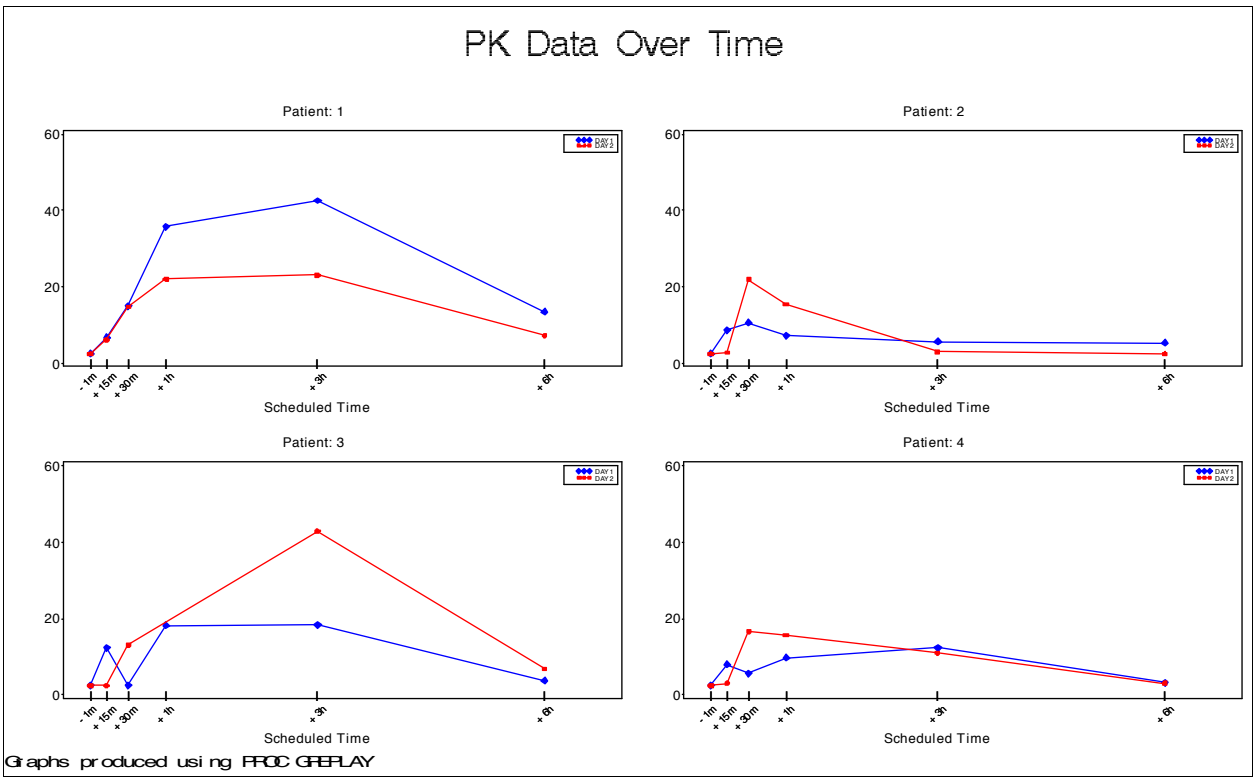
```
%LABEL (x, y, text, color, angle, rotate, size, style, position);
```

See SAS/Graph User's Guide for details on all `%label` parameter explanations.

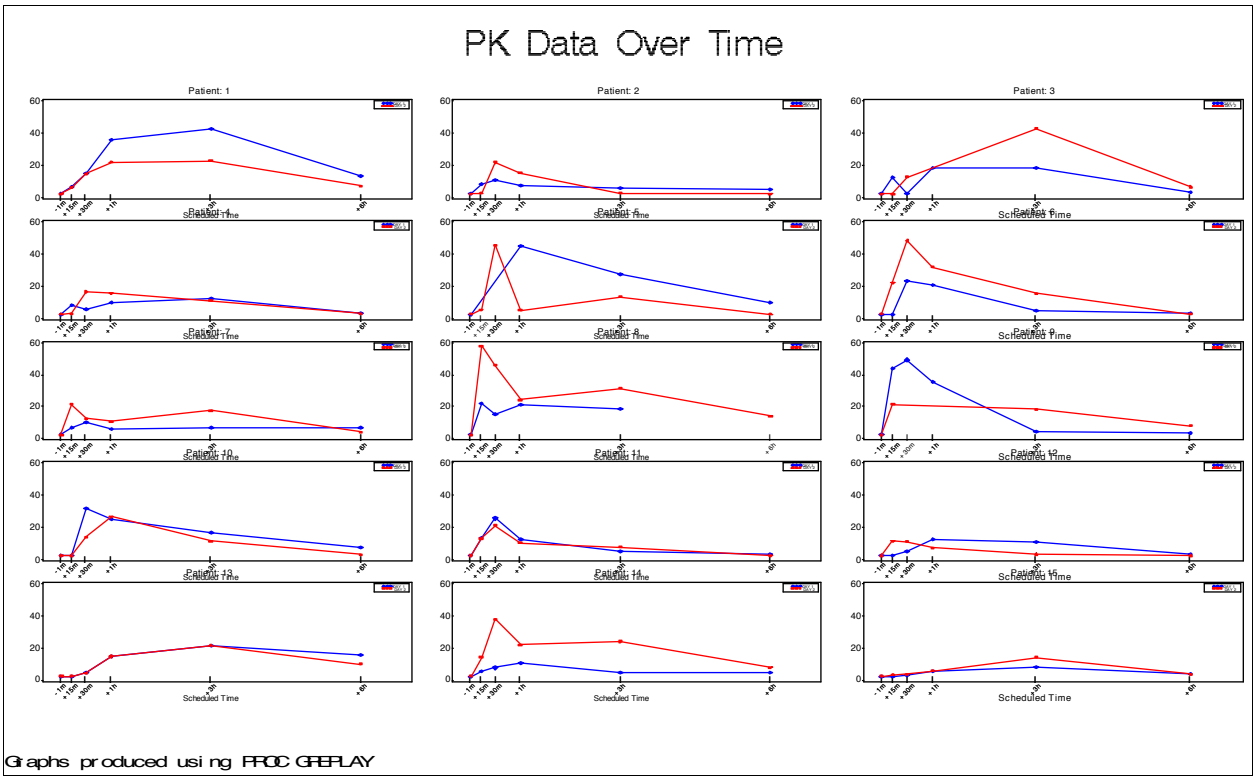
This dataset contains text which is assigned to the tick marks. The annotate dataset is now called in the GPLOT code:

```
proc greplay igout=work.gseg nofs;
      delete _all_;
run;
proc gplot data=dataset;
    plot response*time=day / frame legend=none vaxis=axis1 haxis=axis2
                             annotate=annodata;
    by subject;
run;
quit;
```

We get the following:



When greplay is used in conjunction with the annotate facility it is possible to present a large number of graphs on the one page. In the case of a large number of graphs the scales and any text on the graph can be very hard, almost impossible to read. This is where the annotate facility can come into play. For example if we present 15 graphs on the one page, we get the following:



However as the graphs are all small and the axis individual information is harder to read, we might want to remove this information if the objective is to compare overall patients data the individual information may not be that

important. In this case if we remove the axis information and use the annotate facility and the following code we have a lot clearer presentation of data:

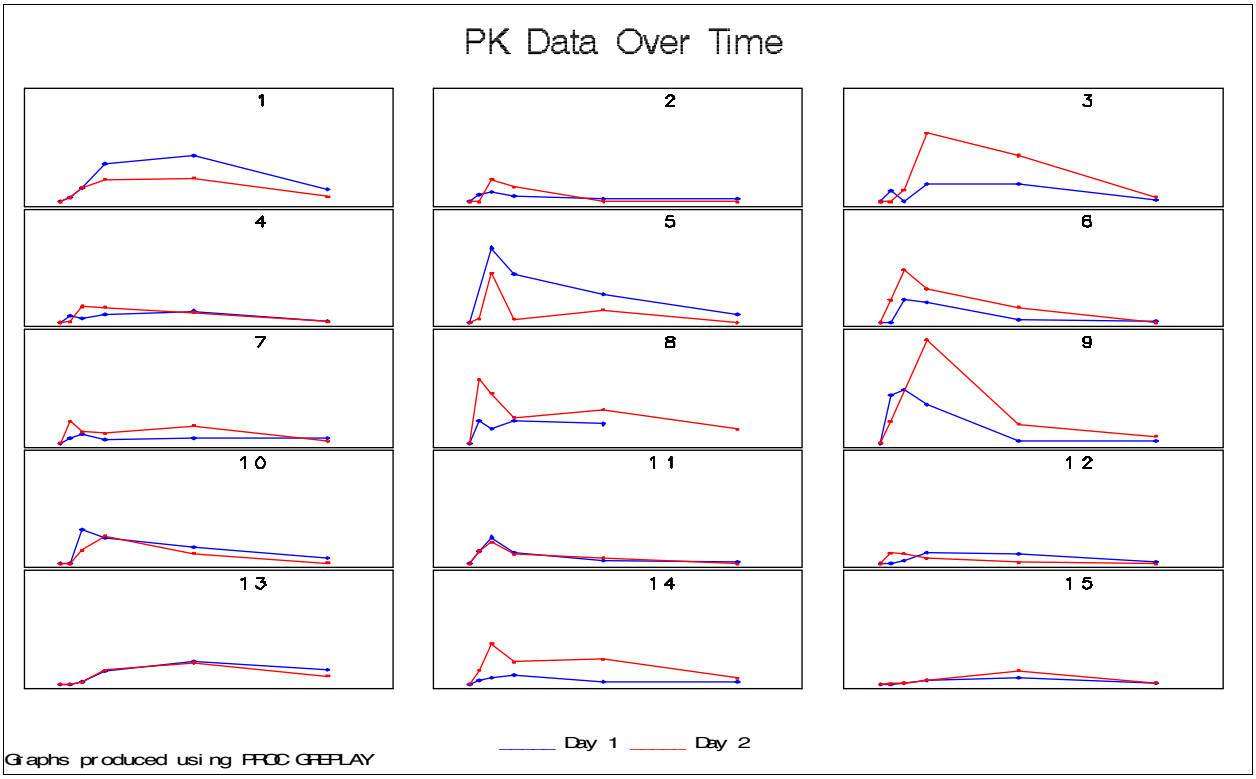
```
%annomac;

data annodata;
    set dataset;
    by subject;
    length text $ 16;
    xsys='2'; ysys='5';
    x=90; y=90;
    size=5;
    text=usubjid;
run;

options nobyline;

proc greplay igout=work.gseg nofs;
     delete _all_;
run;

proc gplot data=dataset;
    plot response*time=day / nolegend annotate=annodata;
    by subject;
run;
quit;
```



CREATING MULTIPLE PAGES

In the case of having multiple graphs that are to be output to the template discussed previously of having 4 graphs per page the following macro will count the number of graphs, order correctly and output them to the template.

```
%macro greplay(path=, out=);
    proc sql noprint;
        select count(distinct subject) into :npats
        from dataset;
    quit;

   %put Number of Patients is &npats;

    %do i = 1 %to &npats %by 4;
        %if %eval(&i) eq 1 %then %let j = 1;
                           %else %let j = %eval((&i / 4)+1);

        filename grafic "&PATH\&OUT..&j..emf";

        proc greplay igout=work.gseg gout=work.gseg tc=work.tmplt nofs;
            template PK1;

            %if %eval(&i+3) le &npats %then %do;
                treplay 1:gslide
                        %if %eval(&i) eq 1 %then 2:gplot;
                                           %else 2:gplot%eval(&i-1);
                        3:gplot%eval(&i)
                        4:gplot%eval(&i+1)
                        5:gplot%eval(&i+2)
                        ;
            %end;
            %else %do;
                treplay 1:gslide
                        %if %eval(&i)   le &npats %then 2:gplot%eval(&i-1);
                        %if %eval(&i+1) le &npats %then 3:gplot%eval(&i);
                        %if %eval(&i+2) le &npats %then 4:gplot%eval(&i+1);
                        %if %eval(&i+3) le &npats %then 5:gplot%eval(&i+2);
                ;
            %end;
        quit;
    %end;
%mend greplay;
```

where PATH= location of output file and OUT= name of the output file.

This macro code reads the graphs, from 1 to number of graphs (npats), and uses proc greplay to present these graphs in the already defined template PK1. There are two possible outcomes having entered the proc grplay step, one is having 4 graphs to output as per the template, the second deals with having less than 4 graphs, which would be the final page.

CONCLUSION

PROC GREPLAY can be a very powerful tool in improving the quality and readability of graphical outputs in clinical trials, in particular for Phase I PK data.

REFERENCES

[1] SAS Institute Inc. *SAS/Graph* ®User's Guide, Release 6.03 Edition.

CONTACT INFORMATION

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

KarolAnne Fitzpatrick
Phone: +353 1 2912225
Email: FitzpatrickK@iconirl.com

Erika Daly,
Phone: +353 1 2912364
Email: DalyE@iconirl.com

Biostatistics Department
ICON Clinical Research
South County Business Park
Leopardstown
Dublin 18
Ireland
Fax: +353 1 2912723
Web: www.iconclinical.com