# Title: Generating Bar Chart and Mutipanel graphs using SAS vs R for Regulatory Submission

## Abstract:

In the submission package to the regulatory agencies, graphs play a vital role in interpreting the clinical data in the pictorial form. Until recently most prominently used software to create graphs was SAS but after knowing the importance of other languages (like R, Python, Spotfire etc.) which can be used for the same purpose, we tend to compare them to know the advantages and limitations of one another. Among these alternative options, R has been adopted in the pharmaceutical companies along with the traditional SAS language. In this paper we would like to discuss some of the prominent clinical graphs such as Bar chart and Multipanel plot using both SAS and R. A side-by-side comparison of both the graphs in SAS and R using their respective syntax, key differences and features that are available in both languages are highlighted in this paper.

## Introduction:

With the availability of both SAS and R softwares, we have the option to choose them based on the graphs that we need to plot. SAS is getting user friendly to accomplish the desired high quality clinical graphs with the upgraded versions of Graphic Template Language (GTL) while R is an open source software and is easy to adapt. ggplot2 package from R is a robust library which has the features to plot most of the clinical graphs. We are using GTL for SAS which might be embedded in macros and for R, we are covering all the graphs using ggplot2 package. We will use Waterfall plot to illustrate the Bar chart.

We will distinguish outputs using these three factors: Time, Quality and Resources.

**Barcharts:** We will discuss about the waterfall plot.

For data, we are going to use the following dummy data.

```
proc format;
value trtan
   1='Drug A'
   2='Drug B'
   3='Drug C'
   4='Drug A + Drug B'
   5='Drug A + Drug C'
   6='Drug B + Drug C'
   7='Placebo'
   ;
run;

data final;

   do trt01an=1 to 7;
         do i=1 to 3;

            pchg=85+trt01an*2*i;
            subjid+1;
            output;

            pchg=-100+trt01an*2*i;
            subjid+1;
            output;

         end;
   end;

run;
```

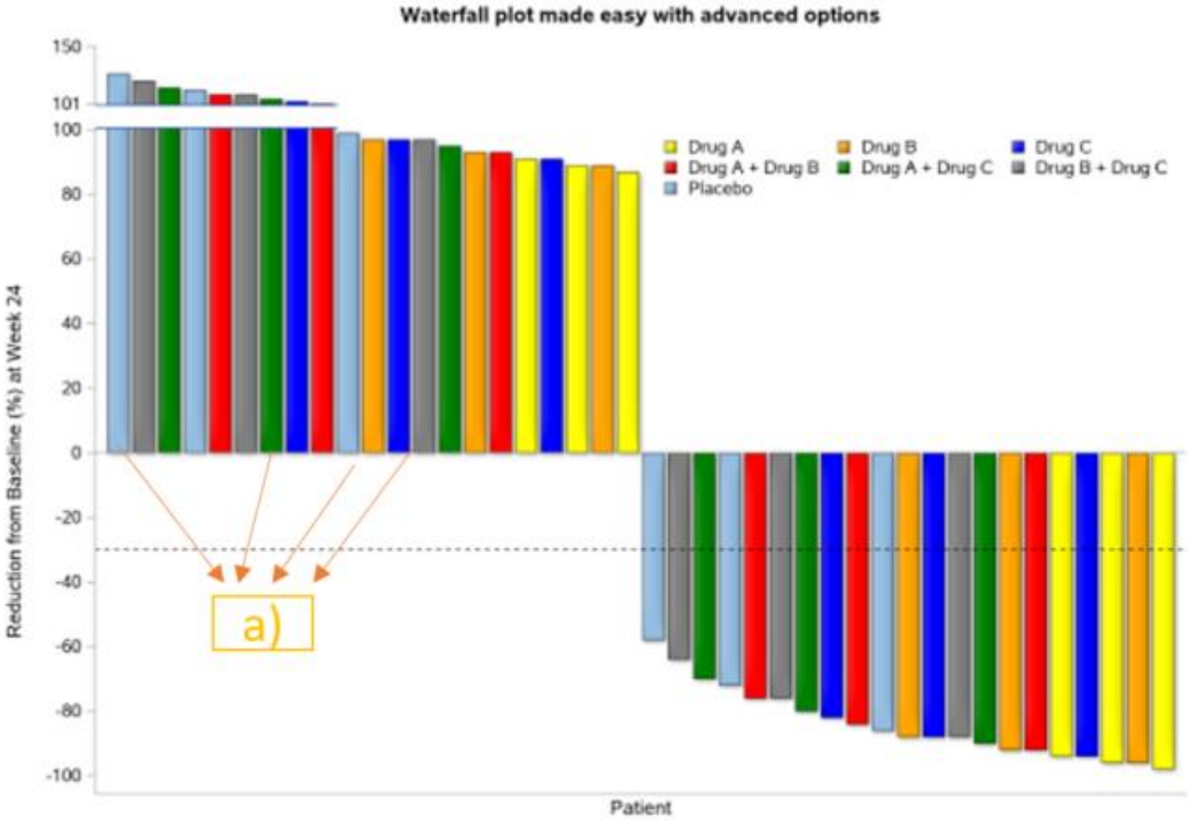
SAS Syntax: Parts of GTL template code is displayed below.

```
     layout overlay / border=false walldisplay=(fill)
                    yaxisopts=(display=(line ticks tickvalues) griddisplay=off linearopts=(viewmin=101 viewmax=150
                    tickvaluelist=(101 150) ))  xaxisopts=(display=none linearopts=(viewmin=1) ) ;
       barchart X=xvar Y=yvar11 / index=trt01an stat=sum group=trt01an fillattrs=multicolor dataskin=matte
outlineattrs=(color=black) orient=vertical;
       drawline x1=-2.2 y1=-1.5 x2=21 y2=-1.5/
          x1space=datapercent y1space=datapercent x2space=DATAPERCENT y2space=datapercent
              lineattrs=(THICKNESS=1px);
     endlayout;

     layout overlay / border=false walldisplay=(fill)
             yaxisopts=(label='Reduction from Baseline (%) at Week 24'
                             display=(line label ticks tickvalues) griddisplay=off
                             linearopts=(viewmin=-105 viewmax=100 tickvaluelist=(-100 -80 -60 -40 -20 0 20 40 60 80 100)))
                    xaxisopts=(display=(line label) label='Patient'  linearopts=(viewmin=1));
       barchart X=xvar Y=yvar12 / index=trt01an stat=sum group=trt01an name='a' fillattrs=multicolor dataskin=matte
outlineattrs=(color=black) orient=vertical;
       drawline x1=-2.2 y1=100.2 x2=21 y2=100.2/
          x1space=datapercent y1space=datapercent x2space=DATAPERCENT y2space=datapercent
              lineattrs=(THICKNESS=1px);
       referenceline y=-30 / lineattrs=(color=black pattern=2);
       discretelegend 'a' /across=3 location=inside border=false halign=right valign=top opaque=true;
```

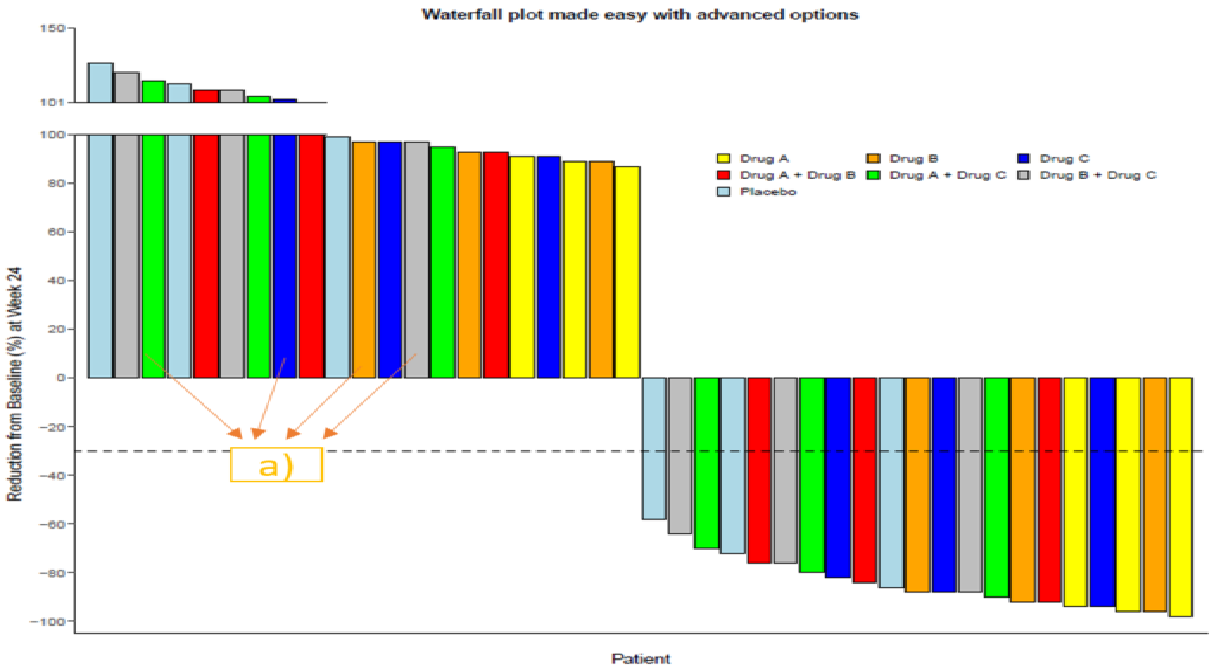
R Syntax: Load the required libraries including ggplot2, patchwork etc. before we run the following code.

```
wf0$trt01an <- factor(wf0$trt01an,levels=c(1,2,3,4,5,6,7),
                  labels=c('Drug A','Drug B','Drug C','Drug A + Drug B','Drug A + Drug C','Drug B + Drug C','Placebo'))

wf1 <- ggplot(wf0,aes(y=yvar1,x=xvar,fill=trt01an)) + geom_bar(stat='identity',show.legend=F,color="black") +
      scale_fill_manual("",values=c('Drug A'='yellow','Drug B'='orange','Drug C'='blue','Drug A + Drug B'='red',
                                   'Drug A + Drug C'='green', 'Drug B + Drug C'='grey','Placebo'='lightblue')) +
      geom_segment(y=101, x=-1, yend=101, xend=9.6) +  scale_y_continuous(breaks=seq(101,150,49),expand =
c(0,0)) +  scale_x_continuous(breaks=seq(0,43,1), limits=c(0,43),expand = c(0,0)) + xlab(" ") + ylab(" ") +
      theme_bw() +  labs(subtitle="Waterfall plot made easy with advanced options") +
      theme(legend.background = element_blank(), legend.key = element_blank(),
       plot.subtitle = element_text(face="bold",hjust=0.5), plot.background =element_blank(),
       panel.background =element_blank(), panel.grid = element_blank(), panel.border = element_blank(),
       axis.line.x = element_blank(), axis.ticks.x = element_blank(), axis.text.x = element_blank(),
       axis.line.y = element_line(colour = "black")) + coord_cartesian(ylim = c(100.5, 150),xlim=c(0,43))

wf2 <- ggplot(wf0, aes(y=yvar1,x=xvar,fill=trt01an)) + geom_bar(stat='identity',color="black") +
      geom_hline(yintercept = -30,linetype=2,color="black") +
      scale_fill_manual("", labels=c('Drug A','Drug B','Drug C','Drug A + Drug B','Drug A + Drug C','Drug B + Drug
C','Placebo'). values=c("yellow","orange","blue","red","green", "grey","lightblue")) +
      geom_segment(y=100, x=-1, yend=100, xend=9.6) +
      scale_y_continuous(breaks=seq(-100,150,20), limits=c(-105,150), expand = c(0,0)) +
      scale_x_continuous(breaks=seq(0,43,1), limits=c(0,43), expand = c(0,0)) +
      xlab("\nPatient") + ylab("\nReduction from Baseline (%) at Week 24") + theme_bw() +
      theme(legend.justification=c(0,0.5), legend.position=c(0.55, 0.92),
       legend.direction = "horizontal",legend.text.align = 0, legend.key.size = unit(0.28, 'cm'),
       legend.background = element_blank(), legend.key = element_blank(), plot.background =element_blank(),
       panel.background =element_blank(), panel.grid = element_blank(), panel.border = element_blank(),
       axis.ticks.x = element_blank(), axis.text.x = element_blank(), axis.line.x = element_line(colour = "black"),
       axis.line.y = element_line(colour = "black")) + coord_cartesian(ylim = c(-105, 100),xlim=c(0,43)) +
      guides(fill = guide_legend(ncol = 3, nrow = 3, byrow = T))

wffinal <- wf1 + wf2 + plot_layout(ncol=1, heights=c(0.15,1))

ggsave(wffinal, file="./Waterfallplot.pdf", width = 10, height=8, dpi=300)
```

output: SAS



output: R



Conclusion:

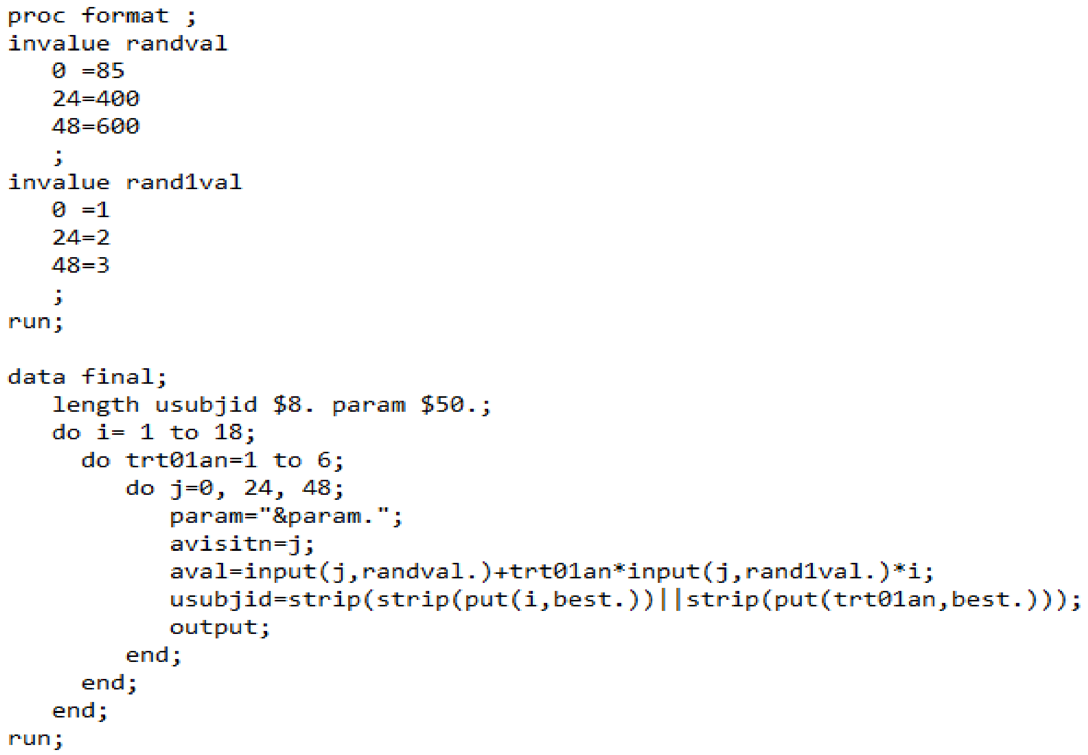
Differences: Some of the observed limitations are indicated below.

a) SAS has dataskin options like CRISP, GLOSS, MATTE, PRESSED, and SHEEN, which adds a heightened visual effect to two-dimensional plots and these options are missing in R ggplot2.

Time: To create the above plot, R took less time when compared to SAS.
Quality: Almost similar but SAS has an edge with the heightened visual effects.
Resources: R has an advantage as it an open source software and free updates are available from time to time.

## Multipanel plot:

For data, we are going to use the dummy data and some of the rows are displayed below.



```
proc format ;
invalue randval
   0 =85
   24=400
   48=600
   ;
invalue rand1val
   0 =1
   24=2
   48=3
   ;
run;

data final;
   length usubjid $8. param $50.;
   do i= 1 to 18;
     do trt01an=1 to 6;
        do j=0, 24, 48;
           param="&param.";
           avisitn=j;
           aval=input(j,randval.)+trt01an*input(j,rand1val.)*i;
           usubjid=strip(strip(put(i,best.))||strip(put(trt01an,best.)));
           output;
        end;
     end;
   end;
run;
```

SAS Syntax: Parts of GTL template code is displayed below.

Following code is the important part of template code.

```
     cell;cellheader;entry "Drug A";endcellheader;
     layout overlay/ walldisplay=(fill) yaxisopts= ( tickvalueattrs=(size=1) linearopts=(includeranges=(&ylow-&ybrk
&ybrkcnt-&ymax)))
     xaxisopts=( label=' '  display=(line ticks) offsetmin=0.1 offsetmax=0.1 tickvalueattrs=(size=8)
linearopts=(tickvaluelist=(&xvaluelist) viewmax=&xmax) );
     seriesplot y=trt1 x=avisitn/group=usubjid display=(markers) datatransparency=0.5;
     referenceline y=&ybrk/lineattrs=(color=red pattern=2) curvelabel="&ybrk" curvelabelattrs=(color=red size=1);
     endlayout;
     endcell;
```

Note: Please repeat the above step by sub setting the data to the other drugs as well.

R Syntax: Load the required libraries including ggplot2, grid, cowplot etc. before we run the following code.

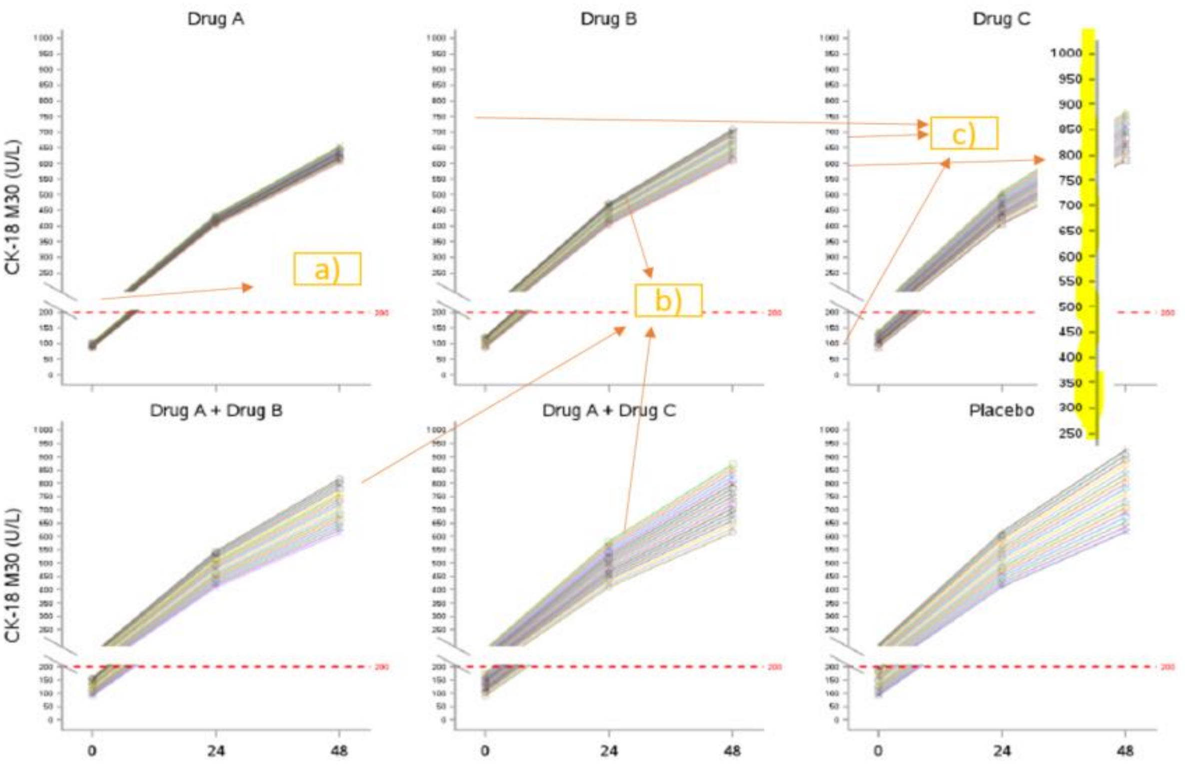
```
mp0$trt01an <- factor(mp0$trt01an,levels=c(1,2,3,4,5,6),
                    labels=c('Drug A','Drug B','Drug C','Drug A + Drug B','Drug A + Drug C','Placebo'))
mp0$xdscrt <- factor(mp0$xdscrt,levels=c(0, 24, 48),labels=c("BL","W24","W48"))

mp1 <- ggplot(subset(mp0, mp0$trt01an == "Drug A")) + geom_line(aes(x = avisitn,y = aval,
                                                                  group=usubjid, color=as.factor(usubjid)),show.legend=F) +
geom_text(aes(47.5,200,label=200,vjust=1.4),size=3.5,color="red") +
 geom_hline(aes(yintercept = 200),color="red",linetype=2) + theme_classic() + scale_colour_brewer(palette="Set3") +
  scale_x_continuous("",breaks=seq(0,48,24),minor_breaks=NULL) +  theme(axis.text.x = element_blank()) +
  scale_y_continuous("",breaks=seq(0,200,25),expand = c(0,0)) + coord_cartesian(ylim=c(0,200), xlim=c(0,48))

mp2 <- ggplot(subset(mp0, mp0$trt01an == "Drug A")) + geom_line(aes(x = avisitn,y = aval,
                                                                 group=usubjid, color=as.factor(usubjid)),show.legend=F) +
  theme_classic() + scale_colour_brewer(palette="Set3") + scale_x_continuous("",breaks=c(0,24,48)) +
  scale_y_continuous("",breaks=c(201,400,600,800,1000),expand = c(0,0)) +
  coord_cartesian(ylim=c(201,1000)) + theme(axis.line.x = element_blank(),
                           axis.ticks.x = element_blank(),axis.text.x=element_blank(),plot.title = element_text(hjust = 0.5))

mp1a <- plot_grid(mp2,mp1, ncol = 1, align = "v",rel_heights=c(1,1))
```

Note: Please repeat the above step by sub setting the data to the other drugs as well and concatenate them using the following step.

```
mpfinal <- grid.arrange(arrangeGrob(mp1a, top='\n     Drug A',left="CK-18 M30 (U/L)"),
                      arrangeGrob(mp1b, top='\n     Drug B'), arrangeGrob(mp1c, top='\n     Drug C'),
                      arrangeGrob(mp1d, top='         Drug A + Drug B',left="CK-18 M30 (U/L)"),
                      arrangeGrob(mp1e, top='         Drug A + Drug C'), arrangeGrob(mp1f, top='     Placebo'),
                      top = "Figure 1. CK-18 M30 (U/L) By Treatment Among Patients with Abnormal Baseline",
                      ncol = 3, nrow=3)
ggsave(mpfinal, file=".//Multipanelplot.pdf", width = 8, height=10, dpi=300)
```

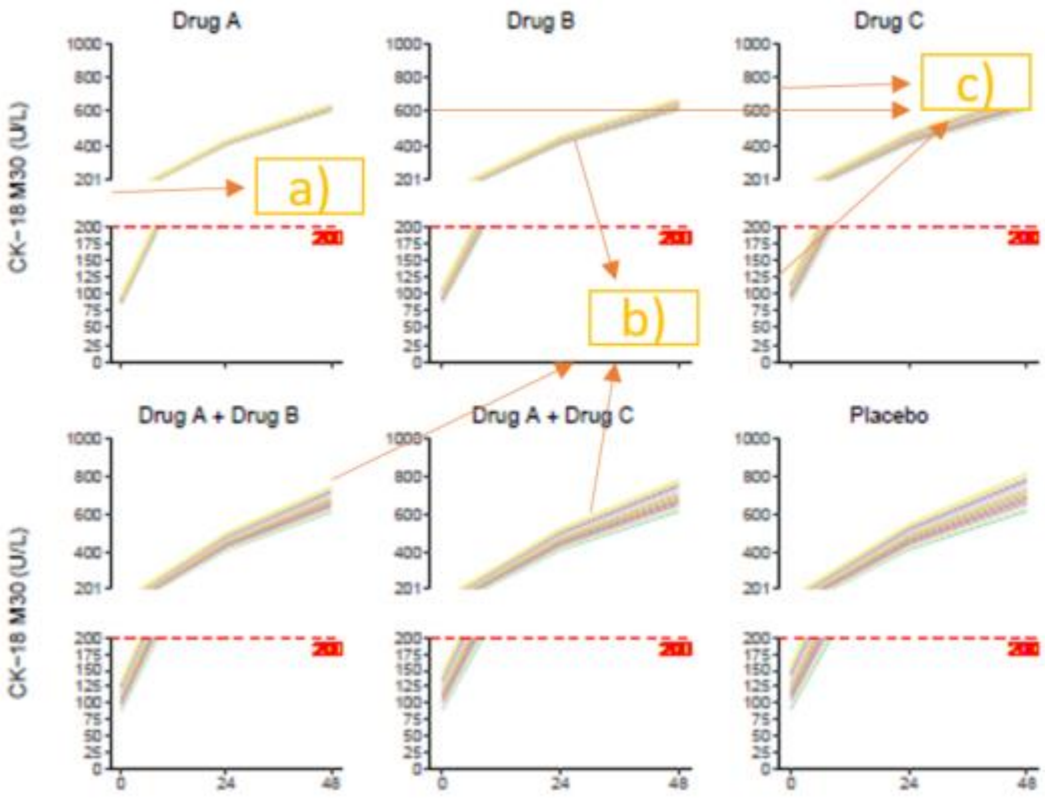
output: SAS

Figure 1. CK-18 M30 (U/L) By Treatment Among Patients with Abnormal Baseline



output: R

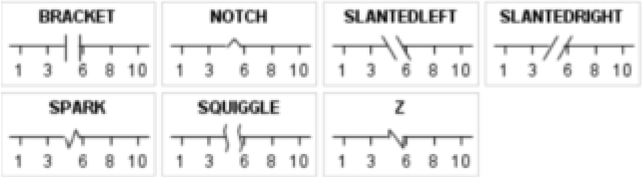
Figure 1. CK−18 M30 (U/L) By Treatment Among Patients with Abnormal Baseline



Conclusion:

Differences: Some of the observed limitations are indicated below.

a) SAS has following pre-specified broken axis options (with some limited restrictions while displaying the data). R doesn't have these options so we followed the similar technique that we used for Waterfall plot.



b) SAS is adjusting the thickness of the series plot with R it is not the case.

c) For the y-axis tickvalues, in SAS because of the limitations tickvalue increments has to be consistent across the y-axis. With R, after the y-axis gap, lower limit (i.e. 201) is clearly visible and y-axis increments can be easily altered as shown in the above plot.

Time: To create the above plot, SAS took less time when compared to R.

Quality: Almost similar.

Resources: R has an advantage as it an open source software and free updates are available from time to time.

Final conclusion: Bar chart and Multipanel graphs can be plotted using SAS and R. Since R is open source and newer generations of software engineers are getting exposed to R courses (in the academic institutions) when compared to traditional SAS (which is used prevalently in Biostatistics courses offered in public health degree), this gives an added advantage to R. On the other hand, understanding the uses of R language in pharmaceutical industry, all the SAS programmers are getting (or will get) trained in R, in due course of time. Small and mid-size CRO companies might adopt to R for their daily usage because of their budgetary constraints. Over time, some of these alternative options (like R, Python) can be used along with SAS or organizations/users will have the ability to choose the software. As programmers, we might need to adapt to changing environment by learning new languages like R, Python (which has already emerged as an alternative), Spotfire, etc.

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

Author Name: Javvaji, Girija Manohar
Labcorp Drug Development Inc. (previously Covance)
4000 Centergreen Way # 300
Cary, NC 27513
Work Phone: 224-280-4518
Email: girija.javvaji@labcorp.com

Co-Author Name: D’Souza, Alistair
Labcorp Drug Development Inc. (previously Covance)
4000 Centergreen Way # 300
Cary, NC 27513
Work Phone: 646-244-2336
Email: Alistair.dsouza@labcorp.com