Title: Generating Violin, eDish, Box and Mean & Standard Deviation Line graphs for Regulatory Submission: Comparison of SAS and R

Abstract:

In 2022 DV05 paper we discussed about Kaplan Meier, Forest, Bar Chart and Multi Panel graphs to observe the challenges and features that are present in both SAS and R. In order to get a deeper insight of available graph options we need to examine few more graphs (which we didn't cover earlier). This year we will discuss some of the distinct graphs like Violin, eDish, Box and Mean & Standard Deviation using line plot to highlight the differences and to compare the features that are present in both SAS and R.

Introduction:

It is time to increase familiarity with languages like R or Python (or to explore latest options that are available in SAS) and our focus in this paper is limited to SAS and R. As we know SAS has Graph Template Language (GTL) and Graphics Procedures options for graphs. R has packages like ggplot2, ggforce, lattice etc. We will use GTL for SAS graphs and ggplot2 for R graphs. And also all the analysis was done in SAS and final dataset is shared with both platforms (or we have the luxury to create R graphs (using R code) in SAS (by using the following code) instead of using R/R studio).

```
 proc iml;
       submit/r;
       install.packages("ggplot2")
       library (ggplot2)  (Cont.)
       endsubmit;
quit;
```

We will distinguish outputs using three factors i.e. Time, Quality and Resources.

Violin Plot:

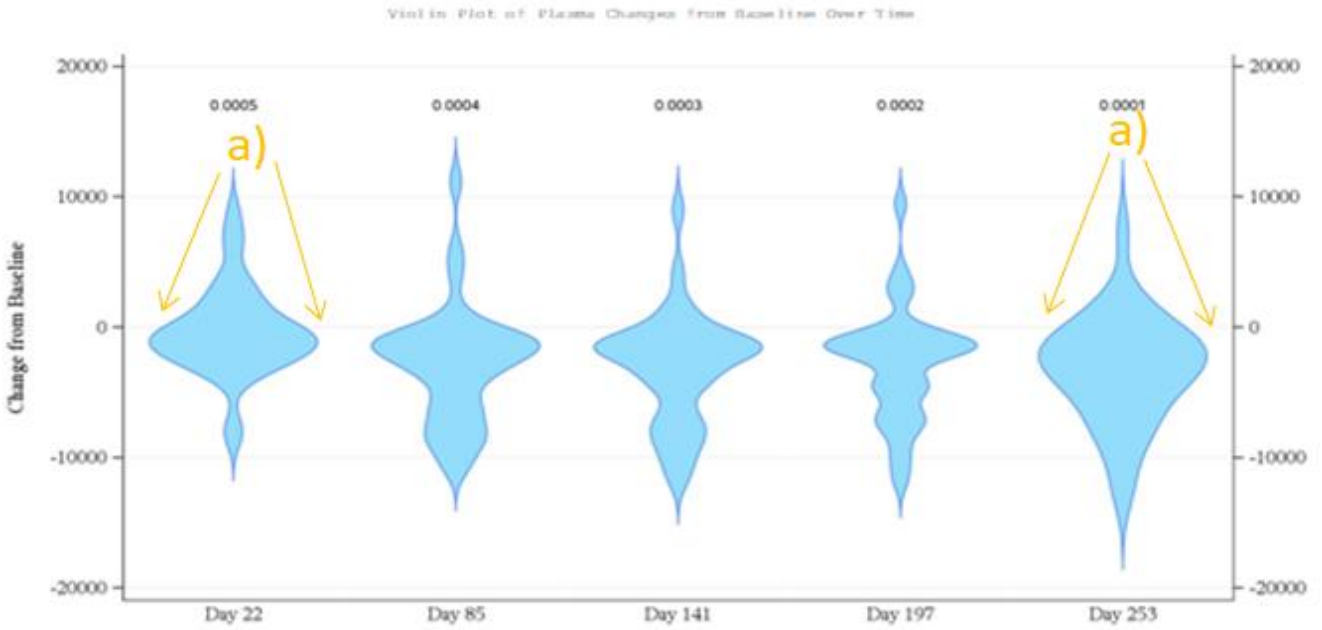
SAS Syntax: Part of GTL template code is displayed below.

```
 layout datalattice columnvar=avisit / includeMissingClass=false headerborder=false headerLabelDisplay=Value
columnheaders=bottom columnDataRange=union  rowDataRange=unionall rows=1 columnAxisOpts=(display=(line))
 rowAxisOpts=(display=all griddisplay=on displaysecondary=(line ticks tickvalues) altdisplaysecondary=(line ticks)
Label="Change from Baseline");
            layout prototype / wallDisplay=(fill);
            bandplot y=chg limitupper=density limitlower=mirror/ display=(fill outline) outlineattrs=(color=cx3A79FF)
                        type=series fillattrs=(color=cx93DCFC);
            textplot x=avisitn y=pv_pos text=pv/textattrs=(family="Albany AMT" size=8pt);
            endlayout;
endlayout;
```

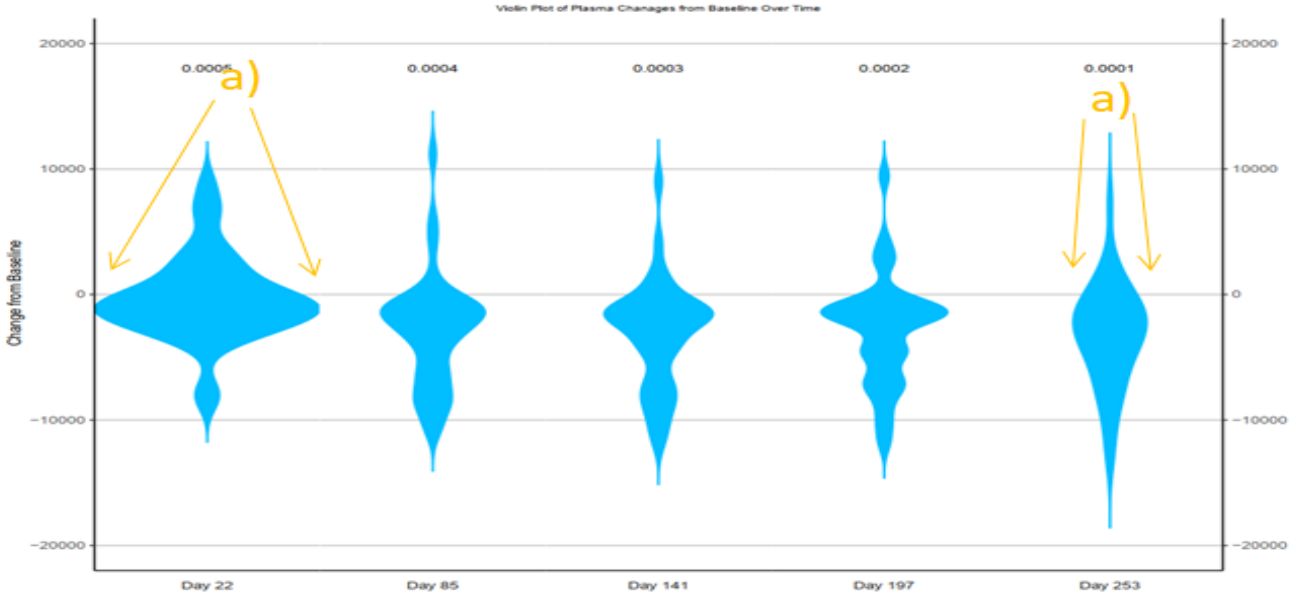
R Syntax: Load the required libraries including ggplot2 etc. before you run the following code.

```
vioR1ba <- ggplot(viods, aes(x=reorder(AVISIT,AVISITN),y=CHG)) + geom_ribbon(aes(y=CHG, xmin=mirror,
                xmax=density),fill="blue") + facet_wrap(~reorder(AVISIT,AVISITN), nrow=1,ncol=5,
                strip.position="bottom") + theme_classic() + ylab("\n Change from Baseline") + scale_x_discrete() +
                labs(subtitle="Violin Plot of Plasma Chanages from Baseline Over Time") +geom_text
                (data=(subset(voids,!is.na(densityq1))), aes(label=as.character(pv),x=reorder(AVISIT,AVISITN),
                 y=18000),size=3) + scale_y_continuous(limits = c(-20000, 20000), breaks = seq(-20000,
                 20000,10000),sec.axis = sec_axis(~.*1) ) + theme(axis.ticks.x=element_blank(), strip.switch.pad.wrap =
                 unit (0.1,"cm"), axis.line.x= element_line(linetype = 1), panel.spacing = unit(0, "lines"),panel.border =
                element_blank(),strip.background = element_rect(linetype = 0), strip.placement="outside",
                panel.grid.major.y = element_line(linetype=1, color="gray"),axis.text.x=element_blank(),
                plot.subtitle = element_text(hjust = 0.5,size=7),axis.title.y=element_text(size=10))
```

output: SAS



output: R



Conclusion:

Differences: Some of the observed limitations are indicated below.

a) Width is wider or narrower in R when compared to SAS.

Time: To create, R took less time when compared to SAS.
Quality: It depends on the colors that we use.
Resources: As R is open source, R has an edge.

EDISH Plot:

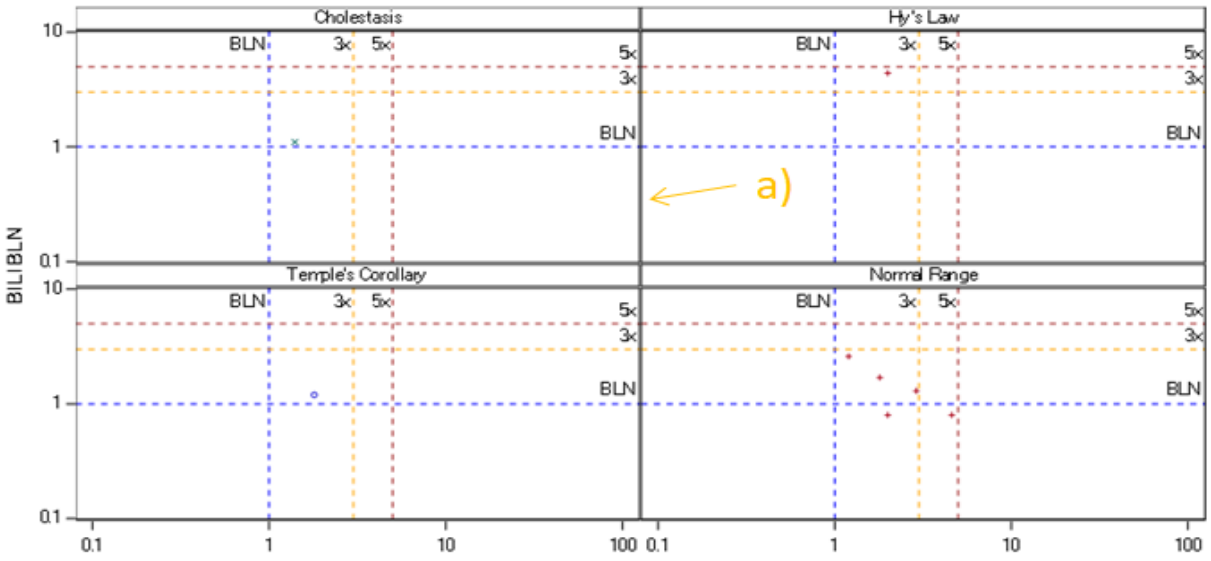
SAS Syntax: Part of GTL template code is displayed below.

```
layout datapanel classvars=(quadrant) / includeMissingClass=TRUE headerLabelDisplay=Value rows=2
columns=2  columnAxisOpts=(display=all type=log logopts=(tickvaluelist=(0.1 1 10 100) viewmin=0.1
 viewmax=100))  rowAxisOpts=(display=all type=log logopts=(tickvaluelist=(0.1 1 10) viewmin=0.1
 viewmax=10) offsetmax=0.01 offsetmin=0.01);
 layout prototype / walldisplay=all;

            referenceline x=1 / curvelabellocation=inside curvelabel='BLN' lineattrs=(color=blue pattern=2);
            referenceline x=3 / curvelabellocation=inside curvelabel='3x' lineattrs=(color=orange pattern=2);
            referenceline x=5 / curvelabellocation=inside curvelabel='5x' lineattrs=(color=brown pattern=2);
            referenceline y=1 / curvelabellocation=inside curvelabel='BLN' lineattrs=(color=blue pattern=2);
            referenceline y=3 / curvelabellocation=inside curvelabel='3x' lineattrs=(color=orange pattern=2);
            referenceline y=5 / curvelabellocation=inside curvelabel='5x' lineattrs=(color=brown pattern=2);

            scatterplot x=&param.bln y=bilibln / name= 'srm' group=quadbase index=b_quadn markerattrs=(size=8px);
endlayout;
```

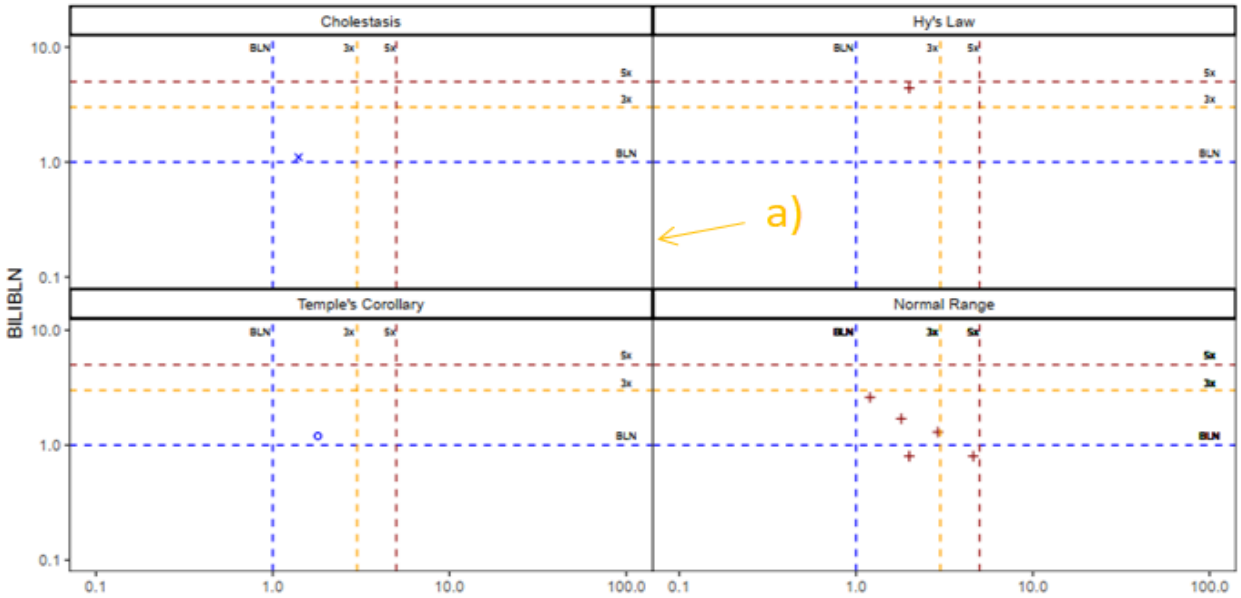
R Syntax: Load the required libraries including ggplot2 etc. before you run the following code.

```
edishGraph <- ggplot(subset(edishds,!is.na(ALTBLN)), aes(ALTBLN, BILIBLN, color= Quadrant, shape= Quadrant,
                group = Quadbase)) + geom_point(show.legend=F) + facet_wrap(~quadrant1,nrow = 2, dir="h") +
                geom_text(aes(label="BLN",y=9.7,x=0.85),size=2,color="black") + geom_text(aes(label="3x",
                 y=9.7,x=2.7), size=2,color="black") + geom_text(aes(label="5x",y=9.7,x=4.6), size=2, color="black") +
                geom_text(aes(label="BLN",y=1.2,x=100),size=2,color="black") geom_text(aes(label="3x",y=3.5,
                 x=100),size=2,color="black") + geom_text(aes(label="5x",y=6,x=100), size=2, color="black") +
                 scale_shape_manual("", values=c(4,3,3,1)) + scale_color_manual("", values=c("blue","darkred",
                 "darkred",  "blue")) + geom_hline(aes(yintercept=1), color="blue", linetype=2) +
                 geom_hline(aes(yintercept=3), color="orange", linetype=2) + geom_hline(aes(yintercept=5),
                 color="brown", linetype=2) + geom_vline(aes(xintercept=1), color="blue", linetype=2) +
                 geom_vline(aes(xintercept=3), color="orange", linetype=2) + geom_vline(aes(xintercept=5),
                 color="brown", linetype=2) + scale_y_log10(limits=c(0.1,10)) + scale_x_log10(limits=c(0.1,100)) +
                 theme_bw() + theme(strip.background = element_rect(fill = "white", colour = "black", linewidth =
                 rel(2)), panel.grid = element_blank(),panel.spacing = unit(0, "lines"),strip.switch.pad.grid = unit(0,
                 "cm"))
```

output: SAS



output: R



Conclusion:

Differences: Some of the observed limitations are indicated below.

a) In R we have the option to increase gap between the panels without freeing the panels.

Time: To create, SAS took less time when compared to R.
Quality: Almost similar to SAS
Resources: R has an advantage as it is open source software and free updates are available from time to time.

BOX Plot:

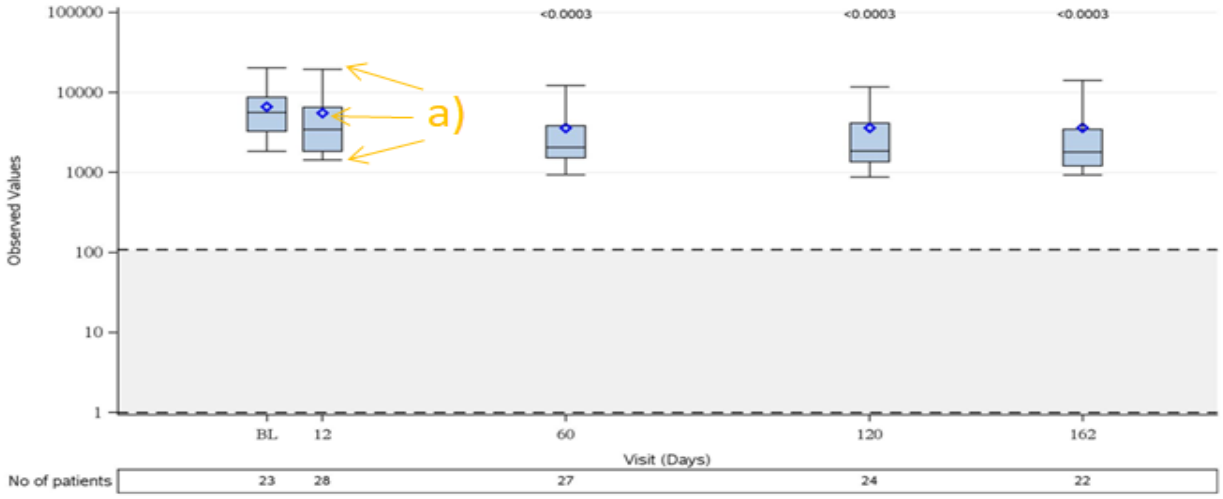
SAS Syntax: Part of GTL template code is displayed below.

```
layout lattice / rows=2 columns=1 columngutter=10 rowweights=(0.95 0.05) columndatarange=unionall;
layout overlay / walldisplay=none xaxisopts=(display=(label line ticks tickvalues) label='Visit (Days)'
tickvalueattrs=(size=9pt) linearopts=(tickvaluelist=(%str(&incrm)) viewmin=&minvis1 viewmax=&maxvis1
tickvalueformat=Xaxfmt. tickvaluefitpolicy=stagger) labelattrs=(family="Albany AMT" size=9pt))
yaxisopts=(griddisplay=on type=log logopts=(base=10 tickintervalstyle=logexpand thresholdmin=1
thresholdmax=1 tickvalueformat=yaxi.) label='Observed Values' labelattrs=(family="Albany AMT" size=9pt)
offsetmin=0 offsetmax=0) ;

        bandplot x=ord2 limitupper=max2 limitlower=min2/ fillattrs=(color=cxf0f0f0);
        referenceline y=min2 / name='r1' lineattrs=(color=black pattern=dash);
        referenceline y=max2 / name='r2' lineattrs=(color=black pattern=dash);
        blockplot x=ord222 block=p_v / display=(values) repeatedvalues=true valuehalign=start valuevalign=top
        includemissingclass=false labelposition=top valuefitpolicy=shrink valueattrs=(family="Albany AMT"
        SIZE=8pt color=black) labelattrs=(SIZE=9pt family="Albany AMT" color=black);
        boxplot x=ord2 y=aval / boxwidth=0.7 groupdisplay=cluster clusterwidth=.5
        name="c1" capshape=line extreme=true SPREAD=true meanattrs=(symbol=diamond color=blue size=8
      weight=bold) outlierattrs=(symbol=asterisk color=blue size=5);
      endlayout;

      layout overlay / yaxisopts=(display=none) xaxisopts=(display=none) pad=(top=0px) ;

      blockplot x=ord22 block=n / display=(values label) repeatedvalues=true valuehalign=start
      valuefitpolicy=shrink valueattrs=GRAPHVALUETEXT(family="Albany AMT" SIZE=8pt color=black)
      labelattrs=graphvaluetext(SIZE=9pt family="Albany AMT" color=black);
      endlayout; endlayout;
```

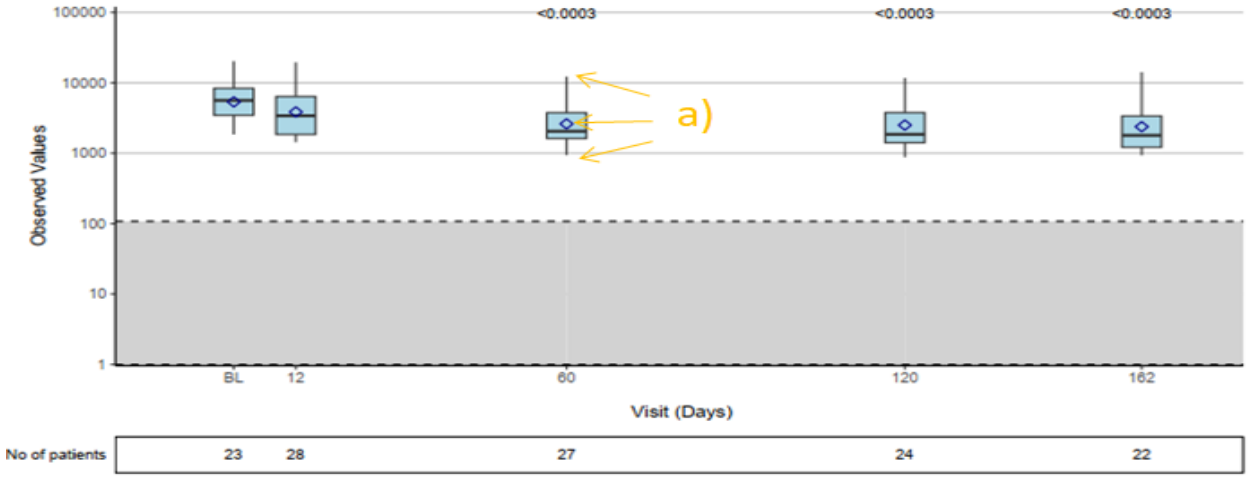
R Syntax: Load the required libraries including ggplot2, patchwork etc. before we run the following code.

```
boxGraphpt1 <- ggplot(boxplotDS,aes(x=ord2, y=AVAL)) + geom_boxplot(data =subset(boxplotDS,
               !is.na(AVAL)),aes(group=ord2),fill="lightblue",width=7) + geom_ribbon( aes(x=AVISITN,
               ymax=max2,ymin=min2),fill="lightgray",show.legend=F) + geom_hline( aes(yintercept=max2),
               linetype=2) + geom_hline(aes(yintercept=min2),linetype=2) + geom_text(aes(label=p_v,
               x=ord2,y=97000), show.legend=F, size=3) + stat_summary(data =subset(boxplotDS, !is.na(AVAL)),
               fun.y=mean, geom="point", shape=5, size=2, color="darkblue") scale_y_log10(breaks=c(1,10,
               100,1000,10000, 100000), limits=c(0.96,120000), labels=c("1","10","100","1000","10000",
               "100000"), expand = c(0,0)) + scale_x_continuous(breaks=c(1,12, 60,120, 162), limits=c(- 20 , 180),
               labels=c("BL","12","60","120","162"),expand = c(0,0)) +  xlab("\n Visit (Days)") +
               ylab("Observed Values") + theme_classic() + theme(axis.title.y = element_text(hjust =0.5,vjust=-10),
               panel.grid.major.y  = element_line(color="gray"))
boxplotDS$atrisklbl <- "No of patients"
boxGraphpt2 <- ggplot(subset(boxplotDS, !is.na(n))) +geom_text(aes(label=n, x=ord22,y=atrisklbl), show.legend=F,
               size=3) + scale_x_continuous(breaks=c(1,12,60,120,162),limits=c(-20,180),labels= c("BL","12","60",
               "120","162"),expand = c(0,0)) + scale_y_discrete() + ylab(" ") + xlab(" ") + theme_bw() +
               theme(panel.grid.major = element_blank(),  panel.grid.minor = element_blank(),axis.line=
               element_blank(), axis.text.x = element_blank(), axis.ticks = element_blank(), panel.border=
               element_rect(color="black",fill="NA"),axis.text.y = element_text(color="black"))

boxGraph <- boxGraphpt1 + boxGraphpt2 + plot_layout(ncol=1, heights=c(1,0.10))
```

output: SAS



output: R



Conclusion:

Differences: Some of the observed limitations are indicated below.

a) Whiskers and mean symbols are not generated using geom_box in R. For displaying mean, we have to use other statistical options in addition to geom_box, which is not in agreement with SAS stats.

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

Quality: Almost similar to SAS

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

Mean and standard Deviation Line Plot:

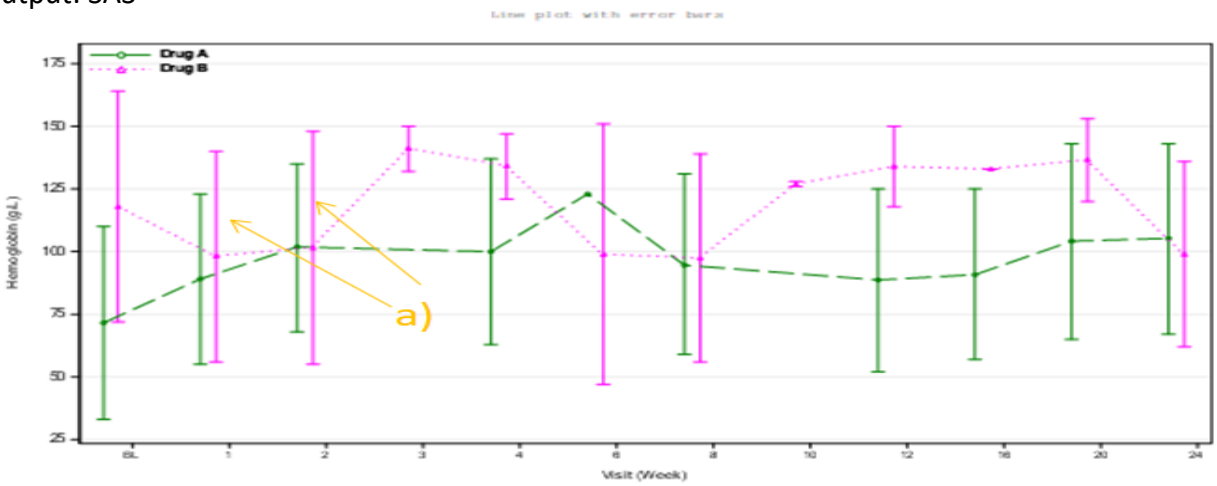
SAS Syntax: Part of GTL template code is displayed below.

```
layout overlay / xaxisopts=(display=(label ticks tickvalues) label='Visit (Week)' tickvalueattrs=(size=7pt)
linearopts=(tickvaluefitpolicy=STAGGER tickvaluelist=x_incrm viewmin=x_minvis viewmax=x_maxvis)
labelattrs=(family="Albany AMT" size=9pt))  yaxisopts=(griddisplay=on label=_byval2_
labelattrs=(family="Albany AMT" size=9pt) offsetmax=0.05 offsetmin=0.01 linearopts=
(thresholdmin=1 thresholdmax=1));
            scatterplot x=nomnum y=mean / name='p1'  group=cohortcd index=cohortcd groupdisplay=cluster
            clusterwidth=.15 yerrorlower=lower yerrorupper=upper;
            seriesplot x=nomnum y=mean / name='p2' index=cohortcd  group=cohortcd;
            discretelegend 'p1' 'p2' / merge=true location=inside across=1 valign=top halign=right border=false
            valueattrs=graphvaluetext(family="Albany AMT" Weight=Bold SIZE=8pt);
endlayout;
```

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

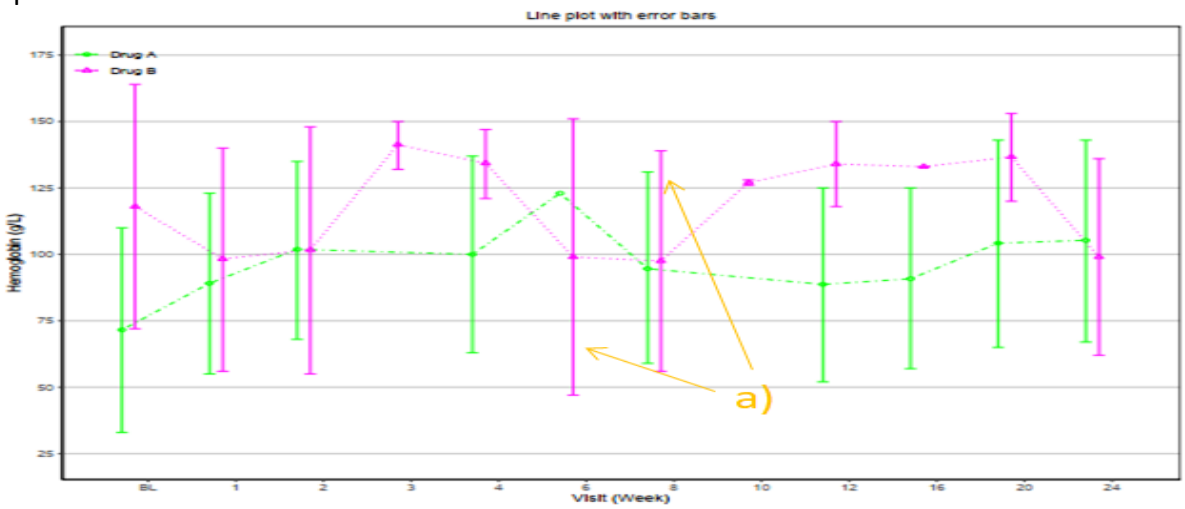
```
linegraph <- ggplot(lineds, aes(x = nomnum, y = mean, linetype= cohortcd1, shape= cohortcd1, colour= cohortcd1,
            group=cohortcd1)) + geom_line() + geom_point()+  geom_errorbar(aes(ymax = upper, ymin
            = lower),linetype=1) + scale_linetype_manual("", values=c(4,3))+  scale_shape_manual("",
            values=c(1,2)) + scale_color_manual("", values=c("green","magenta")) + ylab("Hemoglobin (g/L)") +
            xlab("Visit (Week)") + theme_classic() + scale_y_continuous (limits = c(23, 178), breaks =
            seq(25,175,25)) + labs(subtitle="Line plot with error bars") + scale_x_continuous (limits = c(-2, 56),
            breaks=seq(0,55,5), labels =  c("BL","1","2","3","4","6","8","10","12","16","20","24")) +  theme
            (panel.border =element_rect(color="black",fill = NA), legend.justification=c(0,1), legend.position=
            c(0.001, 0.9999), legend.direction = "vertical",legend.text.align = 0.5, panel.grid.major.y  =
            element_line(color="gray"),  legend.background = element_blank(), plot.subtitle =
            element_text(hjust = 0.5))
```

output: SAS



,

output: R



Conclusion:

Differences: Some of the observed limitations are indicated below.

a) In R we have the options to match the error bar line types with treatment line types where as in SAS error bars are solid lines. In order to match SAS graph we have overwritten the existing options in R.

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

Quality: Almost similar to SAS

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

Final conclusion: Whatever graphs we have created so far using SAS for regulatory submissions, those can be plotted using R as well. Plotting with R is easier when compare to SAS. However, options like innermargin, colorbands and heightened visual appearances (of filled bars) to name a few are considered special in SAS. R is an open source software 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 course, offered in public health degree), which 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 R for their daily use because of their budgetary constraints. 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
City, State: Chicago, IL
Phone: 919-637-6937
Email: girija.javvaji@gmail.com