DV03 - Replicating R figures into SAS

Antonio Rodríguez Contestí

R vs SAS: The eternal discussion

There are a lot of articles out there comparing SAS and R. One of the key areas that is typically compared is the Graphics capability to explore and display the data.

During those reads, in multiple articles I saw sentences as this one

"base on Graphics, R is the winner"

So I decided to test that affirmation.

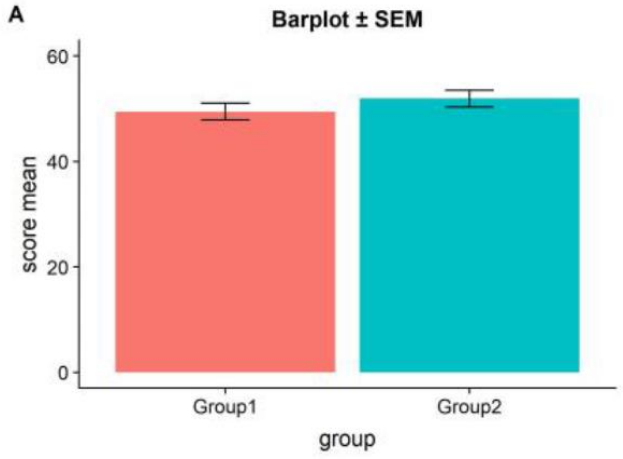
To do so, I contacted Adrian Olszewski, a Statistician and R user for a long time that always share useful information throw Linkedin, if he can provide me with a set of fancy figures in R to test if they were doable or not in SAS.

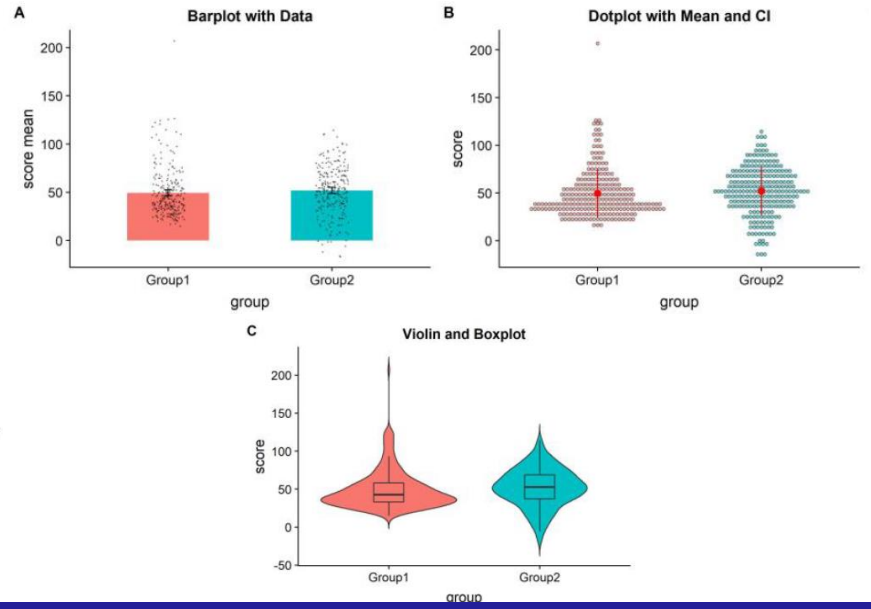
Raincloud

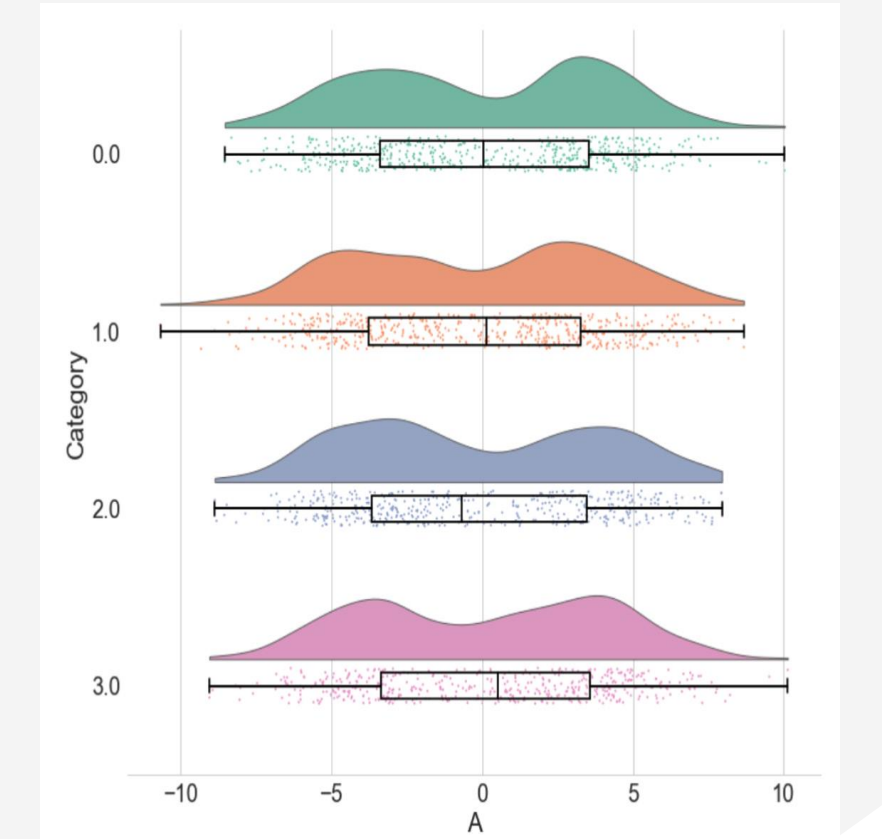
These “raincloud plots” can visualize probability density distribution (the ‘cloud’), with jittered raw data (the ‘rain’) and key summary statistics such as median, mean, and relevant confidence intervals in an appealing and flexible format with minimal redundancy.

It remove the redundancy of double info in violin plots, display data that helps to see outliers or unexpected patterns and do not depend on bins as the histograms.

This approach is the preferred over the previous purposed approaches that can be seen as follows that have different flaws

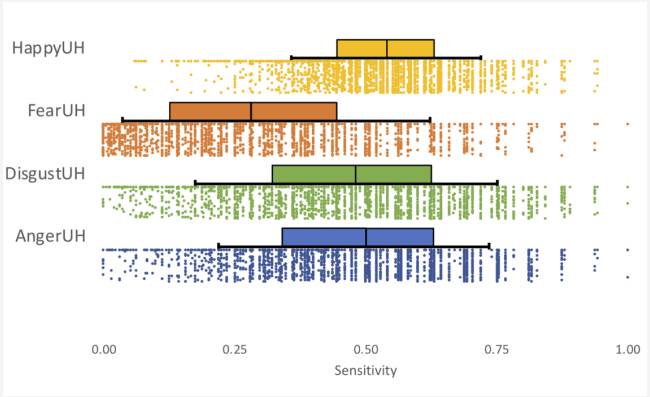


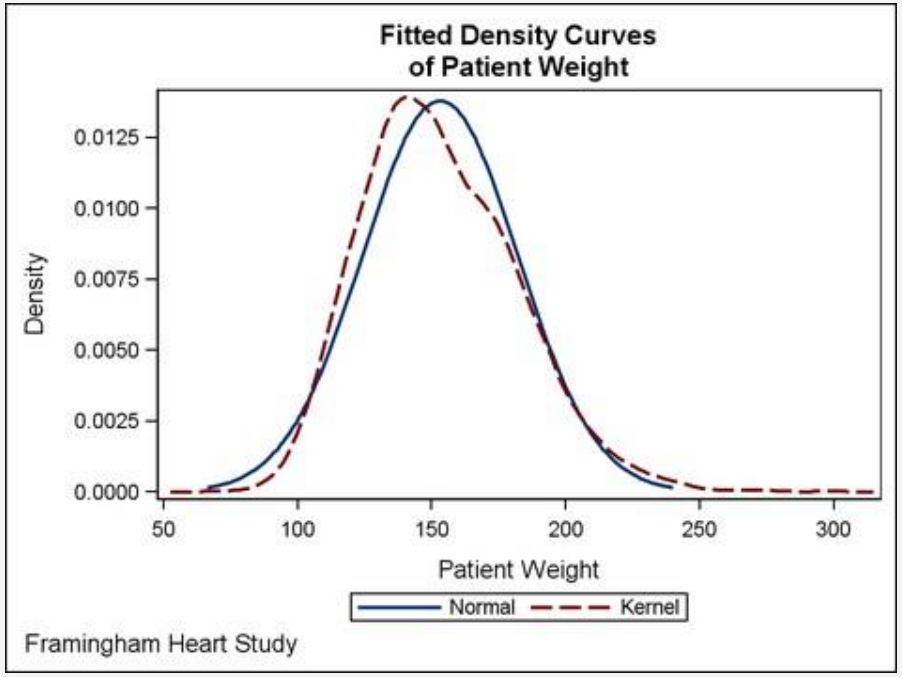




Problems to create Raincloud in SAS

- In SAS there is no option for a half box-plot, that is used on Raincloud
- SAS has a DENSITYPLOT function into GTL to display the density of the data, but do not allow to modify the position and stack multiple groups





Raincloud

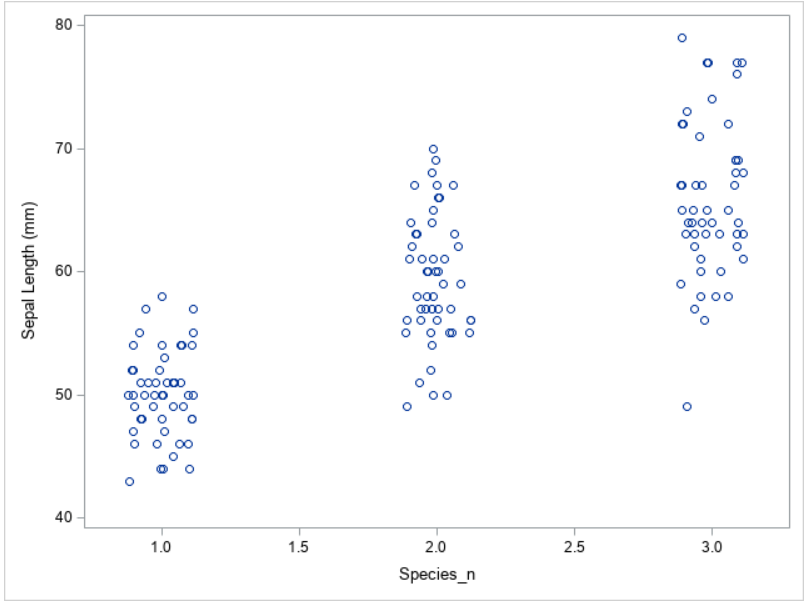
Step 1 - Rain

```
SCATTERPLOT X = column | expression
             Y = column | expression /
JITTER=NONE | AUTO
JITTEROPTS =(AXIS=AUTO | X | Y | BOTH
             UNIFORM=TRUE | FALSE
             WIDTH=positive-number);
```

Raincloud

Step 1 – Rain Code

```
PROC FORMAT;
   INVALUE Species
     'Setosa' = 1
     'Versicolor' = 2
     'Virginica' = 3;
   VALUE Species_n
     1 = 'Setosa'
     2 = 'Versicolor'
     3 = 'Virginica' ;
RUN;
DATA Iris;
  SET SASHELP.Iris;
  Species_n = INPUT(Species,Species.);
RUN;
PROC TEMPLATE;
  DEFINE STATGRAPH RainCloud;
    BEGINGRAPH;
      LAYOUT OVERLAY;
        SCATTERPLOT X = Species_n Y = SepalLength / JITTER = AUTO
                    JITTEROPTS = (AXIS = X WIDTH = 0.25 );
      ENDLAYOUT;
   ENDGRAPH;
 END;
RUN;

PROC SGRENDER DATA = Iris TEMPLATE = RainCloud;
RUN;
```



Raincloud

Step 2 – Half Box plot

As we discuss earlier, there is no way currently (SAS 9.4) to create a half box plot, so we are going to review the early days of SAS GTL.

In SAS 9.2, if you tried to overlay a Scatterplot and a boxplot, an error appeared into the log, so an approach was suggested to fix that issue, create your own boxplot statement compatible with scatterplot

https://lexjansen.com/phuse/2012/cs/CS03.pdf

The approach suggested to use proc means and a lot of VECTPLOT statements, that allow us to create lines from point A to point B, to generate the Box plot

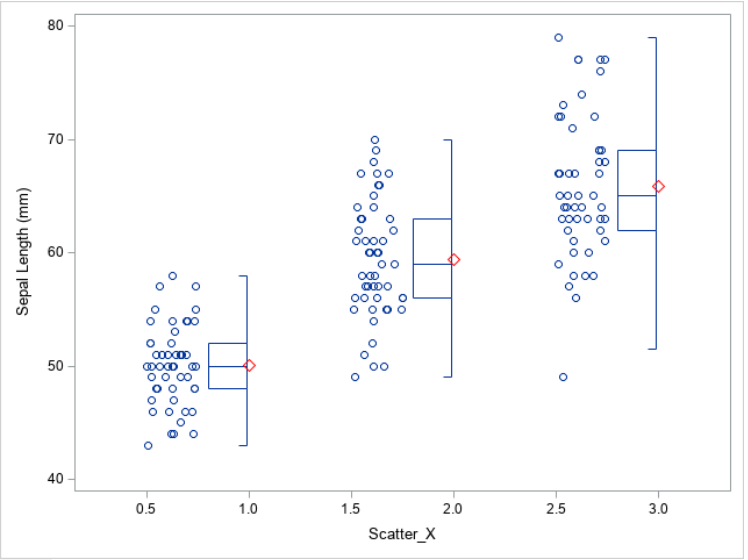
With proper instruction we can draw whatever we want, so we are going to tweak that approach to plot half boxplot only

Raincloud

Step 2 – Half Box plot Code

```
%MACRO DrawBoxPlots;
  /*  draw  median  line */
  VECTORPLOT X  =  XBoxRight   Y = median  XORIGIN  =  XBoxLeft
                   YORIGIN  =  median / ARROWHEADS = FALSE;
  /*  draw  quantile  box */
  VECTORPLOT  X  =  XBoxRight  Y = q1  XORIGIN  =  XBoxLeft
                   YORIGIN  =  q1 / ARROWHEADS = FALSE;
  VECTORPLOT  X  =  XBoxRight  Y = q3  XORIGIN  =  XBoxLeft
                   YORIGIN  =  q3 / ARROWHEADS = FALSE;
  VECTORPLOT  X  =  XBoxLeft   Y = q3  XORIGIN  =  XBoxLeft
                   YORIGIN  =  q1 / ARROWHEADS = FALSE;
  VECTORPLOT  X  =  XBoxRight  Y = q3 XORIGIN  =  XBoxRight
                   YORIGIN  =  q1 / ARROWHEADS = FALSE;
  /*  draw  whiskers */
  VECTORPLOT  X  =  XWhiskerRight Y = YWhiskerTop
             XORIGIN  = XWhiskerRight YORIGIN  =  q3 /
                                    ARROWHEADS = FALSE;
  VECTORPLOT  X  =  XWhiskerRight Y = YWhiskerTop
             XORIGIN  =  XWhiskerLeft YORIGIN  =  YWhiskerTop /
                                    ARROWHEADS = FALSE;

  VECTORPLOT  X  =  XWhiskerRight Y = YWhiskerBottom
             XORIGIN  = XWhiskerRight YORIGIN  =  q1 /
                                    ARROWHEADS = FALSE;
  VECTORPLOT  X  =  XWhiskerRight Y = YWhiskerBottom
             XORIGIN  =  XWhiskerLeft YORIGIN  =  YWhiskerBottom /
                                    ARROWHEADS = FALSE;
%MEND;
PROC MEANS DATA =  Iris NOPRINT;
 BY Species_n Species;
 VAR SepalLength;
 OUTPUT OUT = BoxPlotStats  MEAN = mean  MEDIAN = median Q1 = q1
    Q3 = q3 QRANGE = qrange  MIN = min MAX = max;
RUN;
```

```
DATA GraphData;
  SET Iris (IN = iris)
      BoxPlotStats(IN = box);
  IF iris THEN DO;
    Scatter_X = Species_n - 0.375;
  END;
  IF box THEN DO;
      XBoxLeft             =  Species_n  -  0.2;
      XWhiskerLeft         =  Species_n  -  0.05;

      /*Right side of the boxplot will be slightly left too */
      XBoxRight            =  Species_n  - 0.01;
      XWhiskerRight        =  Species_n  - 0.01;
      YWhiskerTop          =  (1.5*qrange)  +  q3;
      YWhiskerBottom       =  q1  - (1.5*qrange);
      IF max  LE  YWhiskerTop    THEN YWhiskerTop     =  max;
      IF min  GE  YWhiskerBottom THEN YWhiskerBottom  =  min;
  END;
RUN;

PROC TEMPLATE;
  DEFINE STATGRAPH RainCloud;
    BEGINGRAPH;
      LAYOUT OVERLAY;
        SCATTERPLOT X = Scatter_x Y = SepalLength / JITTER = AUTO
                          JITTEROPTS = (AXIS = X WIDTH = 0.25 );
        %DrawBoxPlots;
        SCATTERPLOT X = Species_n  Y = Mean /
            MARKERATTRS = (COLOR = RED SIZE = 9 SYMBOL = Diamond );
      ENDLAYOUT;
   ENDGRAPH;
 END;
RUN;

PROC SGRENDER DATA = GraphData TEMPLATE = RainCloud;
RUN;
```



Raincloud

Step 3 – Half Violin plot

Also as mentioned earlier, there is no way currently (SAS 9.4) to indicate to a DENSITYPLOT to be displayed in a fixed X or Y, it's always aligned with 0.

Being DENSITYPLOT suitable to our needs, we need a workaround.

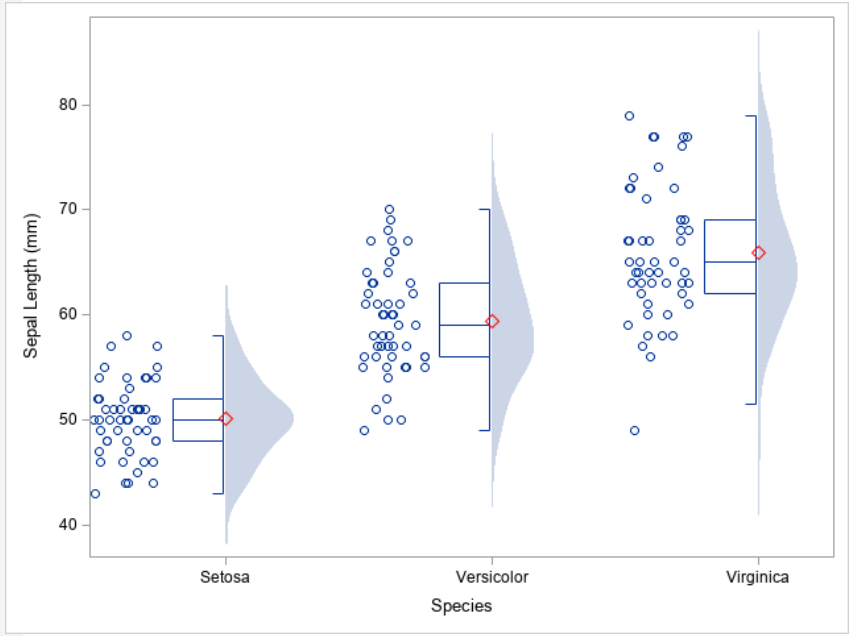
https://blogs.sas.com/content/graphicallyspeaking/2012/10/30/violin-plots/

The approach that we are going to take is to use Kernel Density Estimation (KDE) implemented in PROC KDE to calculate the density function, and later one, display it with HIGHLOW statement.

Raincloud

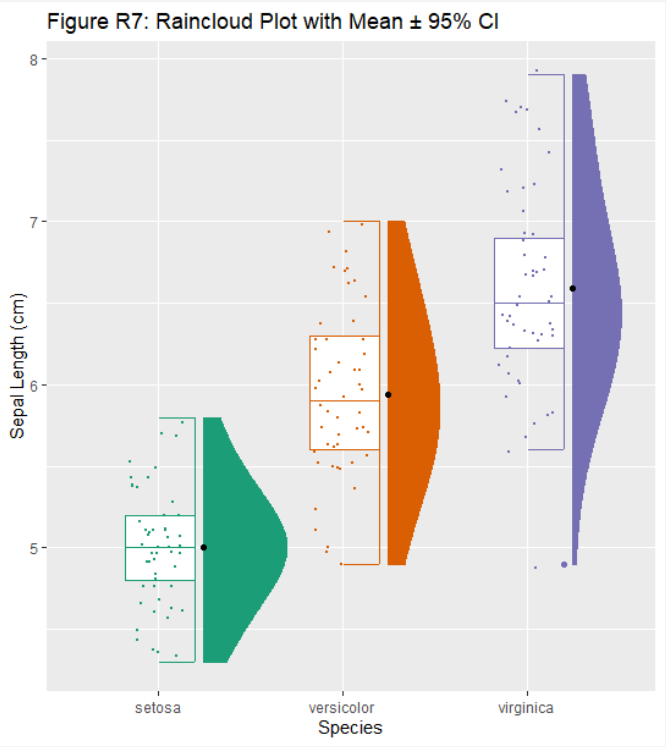
Step 3 – Half Violin plot

```
/* Kernel Density Estimation */
PROC KDE DATA = Iris;
BY Species_n;
UNIVAR SepalLength / NGRID = 1000
   UNISTATS PERCENTILES
  PLOTS = NONE OUT = Density
(RENAME = (value = SepalLength)
  DROP = var);
RUN;
/* Get maximum Density value*/
PROC SQL NOPRINT;
  SELECT MAX(density) INTO :max_dens
  FROM Density;
QUIT;
/* Modify density values to have same proportion*/
DATA Density_scaled;
  SET Density;
  Origin = Species_n;
  Density = Species_n + Density * 0.25/&max_dens.;
RUN;

DATA GraphData;
  SET Iris (IN = iris)
      BoxPlotStats(IN = box)
      Density_scaled;
  IF iris THEN DO;
    Scatter_X = Species_n - 0.375;
  END;
  IF box THEN DO;
      XBoxLeft              =  Species_n  -  0.2;
      XWhiskerLeft          =  Species_n  -  0.05;

      /*Right side of the boxplot will be slightly left too */
      XBoxRight             =  Species_n  - 0.01;
      XWhiskerRight         =  Species_n  - 0.01;
      YWhiskerTop           =  (1.5*qrange)  +  q3;
      YWhiskerBottom        =  q1  - (1.5*qrange);
      IF max  LE  YWhiskerTop    THEN YWhiskerTop     =  max;
      IF min  GE  YWhiskerBottom THEN YWhiskerBottom  =  min;
  END;
RUN;

PROC TEMPLATE;
  DEFINE STATGRAPH RainCloud;
    BEGINGRAPH;
      LAYOUT OVERLAY/ XAXISOPTS = (Label = "Species" LINEAROPTS = (
            TICKVALUEFORMAT = Species_n. TICKVALUELIST = (1 2 3)
                    ) OFFSETMIN = 0.05 OFFSETMAX = 0.05) ;

        SCATTERPLOT X = Scatter_x Y = SepalLength / JITTER = AUTO
                    JITTEROPTS = (AXIS = X WIDTH = 0.25 );
        %DrawBoxPlots;
        HIGHLOWPLOT Y = SepalLength HIGH = Density LOW = Origin /
                    DISPLAY = (FILL) TYPE = BAR BARWIDTH = 1
                    INTERVALBARWIDTH = 1 DATATRANSPARENCY = 0.5;
        SCATTERPLOT X = Species_n  Y = Mean /
            MARKERATTRS = (COLOR = RED SIZE = 9 SYMBOL = Diamond );
      ENDLAYOUT;
    ENDGRAPH;
  END;
RUN;

PROC SGRENDER DATA = GraphData TEMPLATE = RainCloud;
RUN;
```



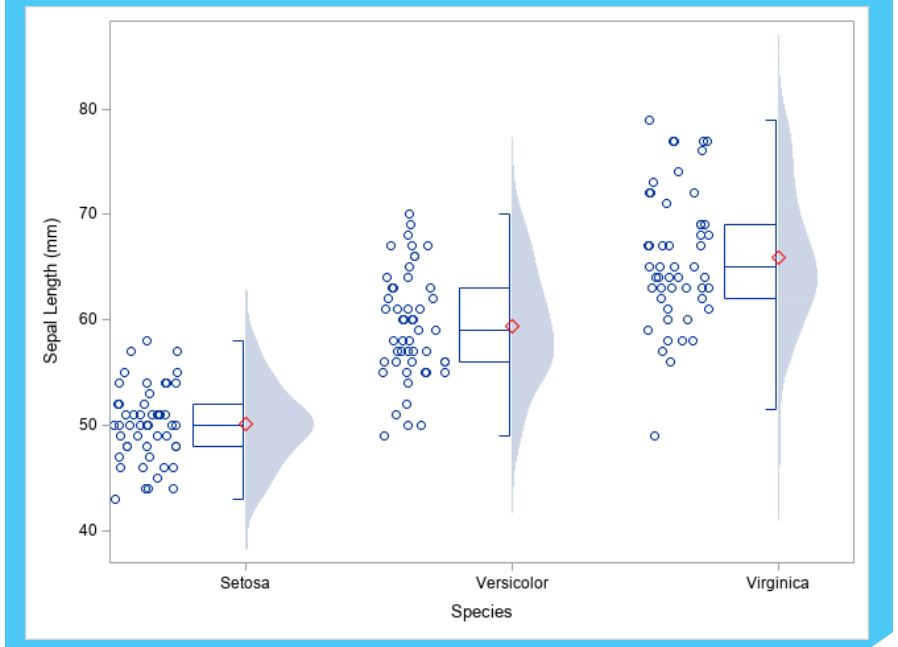
Raincloud R code

```
packages <- c("cowplot", "readr", "ggplot2", "dplyr", "lavaan", "smooth",
"Hmisc","plyr","PupillometryR"," "gghalves")
if (length(setdiff(packages, rownames(installed.packages()))) > 0) {
  install.packages(setdiff(packages, rownames(installed.packages())))
}

# Load packages ----
library(ggplot2)
library(PupillometryR)
library(gghalves)

#Raincloud s with mean and confidence interval
summarySE <- function(data = NULL, measurevar, groupvars = NULL, na.rm = FALSE,
                      conf.interval = .95, .drop = TRUE) {
  library(plyr)

  # New version of length which can handle NA's: if na.rm==T, don't count them
  length2 <- function(x, na.rm = FALSE) {
    if (na.rm) {
      sum(!is.na(x))
    } else {
      length(x)
    }
  }

  # This does the summary. For each group's data frame, return a vector with
  # N, mean, median, and sd
  datac <- plyr::ddply(data, groupvars, .drop=.drop,
                       .fun = function(xx, col) {
                         c(N      = length2(xx[[col]], na.rm=na.rm),
                           mean   = mean(xx[[col]], na.rm=na.rm),
                           median = median(xx[[col]], na.rm=na.rm),
                           sd     = sd(xx[[col]], na.rm=na.rm)
                         )
                       },
                       measurevar
  )

  # Rename the "mean" and "median" columns
  datac <- plyr::rename(datac, c("mean" = paste(measurevar, "_mean", sep = "")))
  datac <- plyr::rename(datac, c("median" = paste(measurevar, "_median", sep = "")))

  datac$se <- datac$sd / sqrt(datac$N)  # Calculate standard error of the mean

  # Confidence interval multiplier for standard error
  # Calculate t-statistic for confidence interval:
  # e.g., if conf.interval is .95, use .975 (above/below), and use df=N-1
  ciMult <- qt(conf.interval / 2 + .5, datac$N - 1)
  datac$ci <- datac$se * ciMult

  return(datac)
}
myiris = iris[,c(1,5)]
summary_iris <- summarySE(myiris, measurevar = "Sepal.Length",
                          groupvars = c("Species"))

p7 <- ggplot(iris,aes(x=Species,y=Sepal.Length, colour = Species))+
 geom_flat_violin(position = position_nudge(x = .25, y = 0),adjust =2, aes(fill=Species))+
  geom_half_boxplot( size = .15,position = position_nudge(x = .2, y = 0)) +
  geom_point(position = position_jitter(width = .15), size = .5)+
  geom_point(data = summary_iris, aes(x = Species, y = Sepal.Length_mean), position = position_nudge(.25), colour = "BLACK")+
  ylab('Sepal Length (cm)')+xlab('Species')+guides(fill = "none", colour = "none") +
  scale_colour_brewer(palette = "Dark2")+
  scale_fill_brewer(palette = "Dark2")+
  ggtitle("Figure R7: Raincloud Plot with Mean ± 95% CI")

p7
```

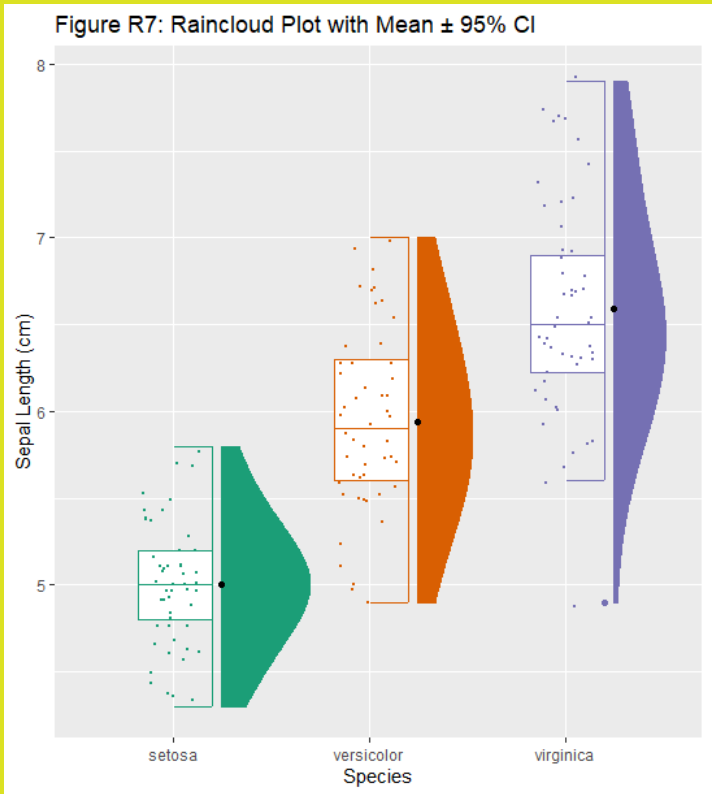


Comparison

SAS

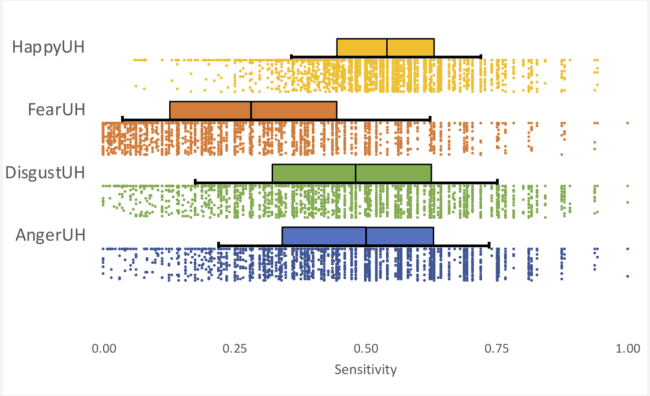


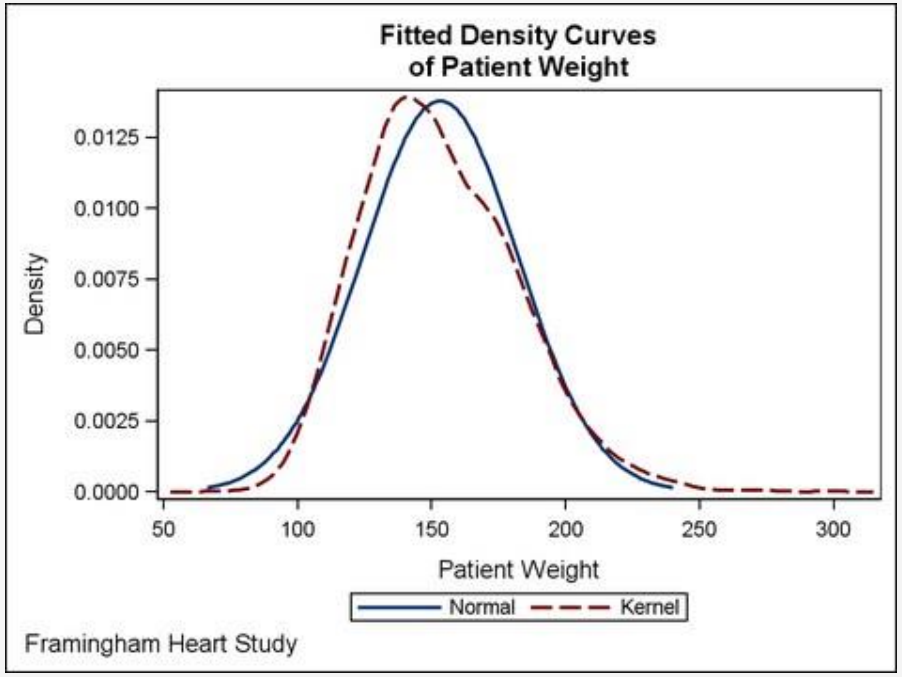
R



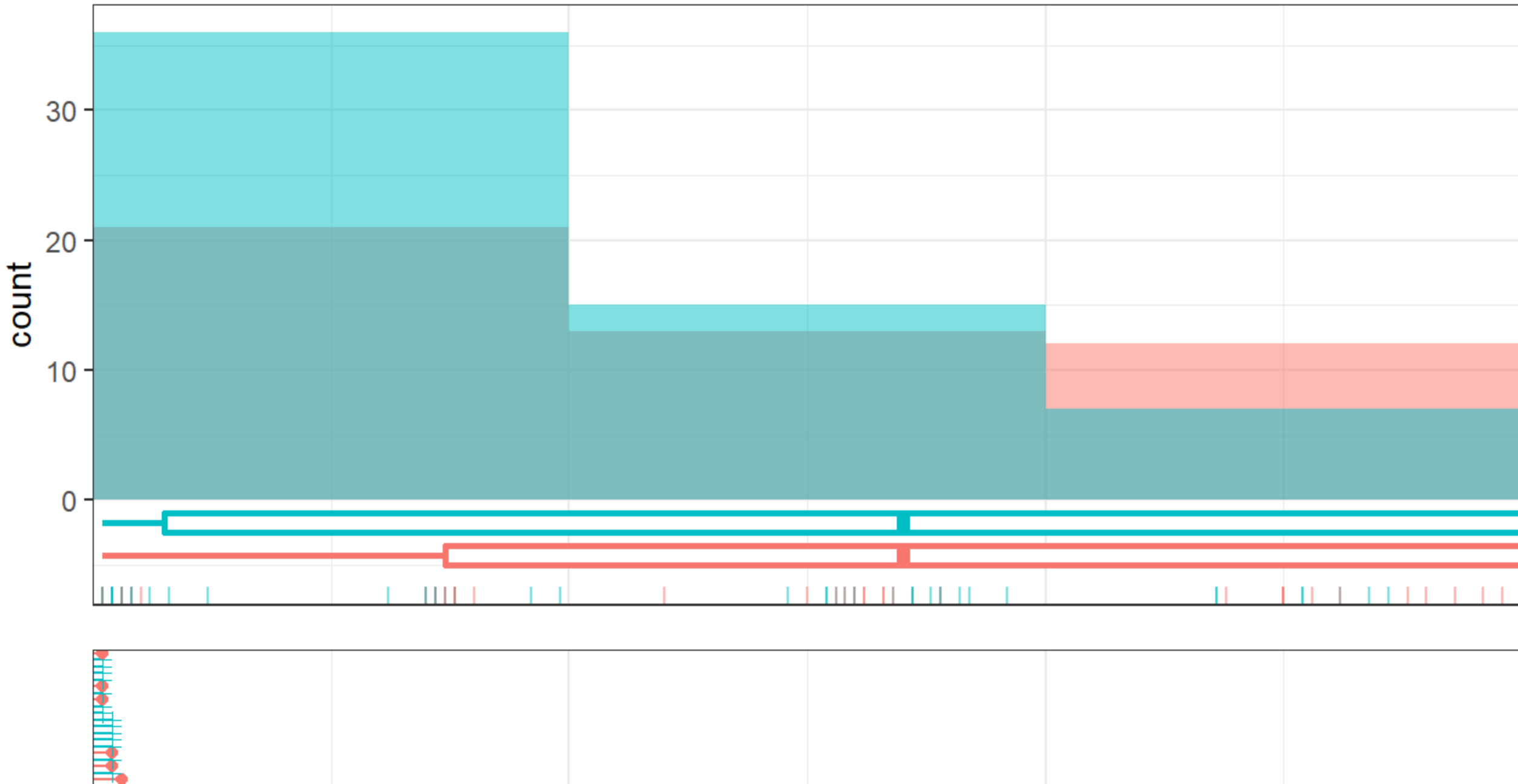
Improvement for SAS

- Option to display only half box
- Add XORIGIN alike option to being able to select the origin of the density



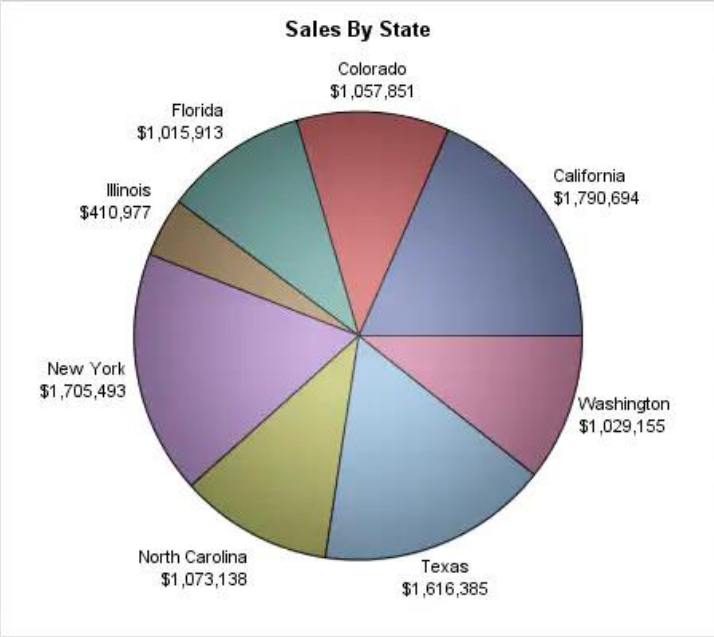


Fraction of events to censored observations



Problems to create exploratory figure in SAS

- Despite SAS has available to create pie charts, it needs to be generated standalone with a LAYOUT REGION



Exploratory survival plot

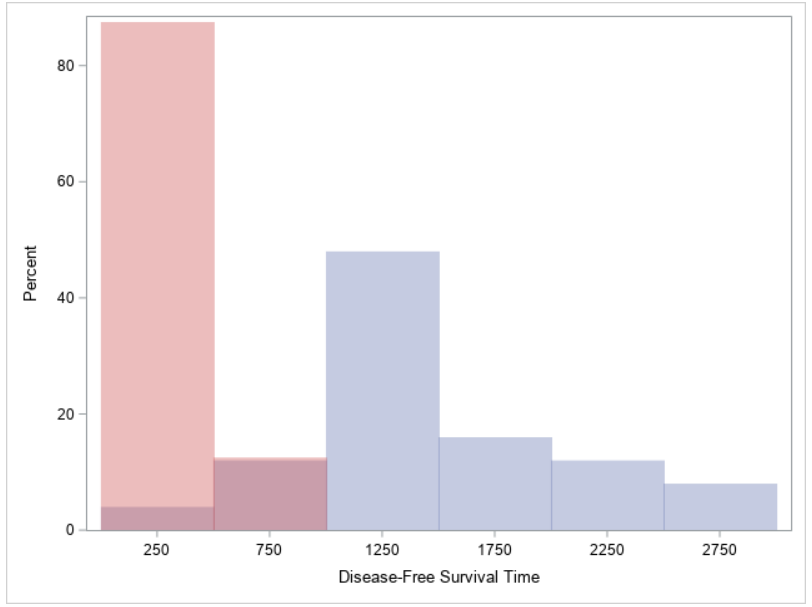
Step 1 - Histogram

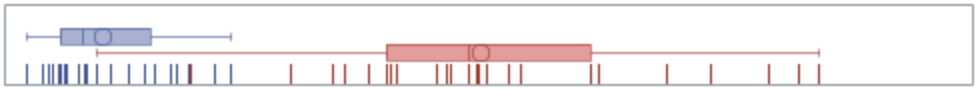
```
HISTOGRAM numeric-column | expression /
     BINSTART = number BINWIDTH = number
     NBINS = integer
     DATATRANSPARENCY = number
     DISPLAY = ( display-options )
     GROUP = column | expression;
```

Exploratory survival plot

Step 1 – Histrogram

```
DATA Survival;
   SET SASHELP.Bmt;
   IF _N_ < 50;
RUN;

PROC TEMPLATE ;
  DEFINE STATGRAPH PieChart;
  BEGINGRAPH;
    LAYOUT OVERLAY;
       HISTOGRAM T / BINSTART = 250 BINWIDTH = 500 NBINS = 6
                DISPLAY = ( FILL ) GROUP = Status
                DATATRANSPARENCY = 0.6;
    ENDLAYOUT;
   ENDGRAPH;
 END;
RUN;

PROC SGRENDER DATA = Survival TEMPLATE = PieChart;
RUN;
```





Exploratory survival plot

Step 2 – Box plot and fingerplot

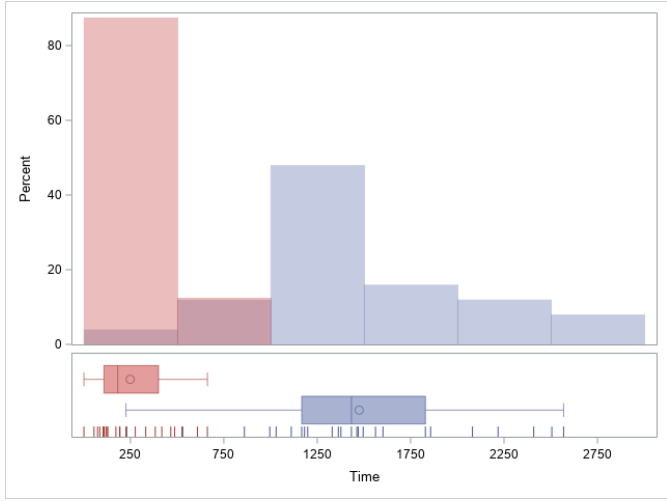
This is easily obtained using the LAYOUT OVERLAY and the two following statements

```
BOXPLOT X = column | expression Y = column | expression /
                ORIENT = VERTICAL | HORIZONTAL
                DATATRANSPARENCY = number
                GROUP = column | discrete-attr-var | expression ;
FRINGEPLOT numeric-column /
                GROUP = column | discrete-attr-var | expression;
```

Exploratory survival plot

Step 2 – Half Box plot Code

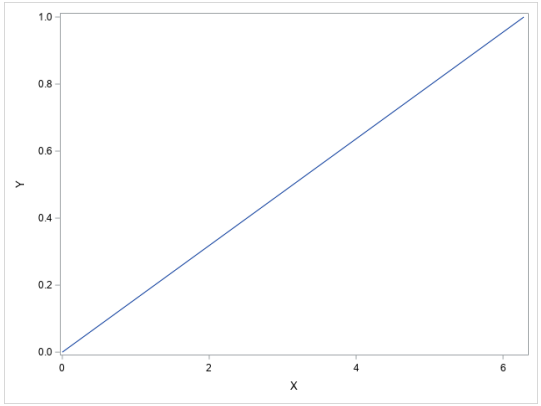
```
PROC TEMPLATE;
  DEFINE STATGRAPH PieChart;
  BEGINGRAPH;
    LAYOUT LATTICE / ROWS = 2 ROWWEIGHTS = (0.8 0.2)
                     COLUMNDATARANGE = UNION;
      COLUMNAXES;
        COLUMNAXIS / DISPLAY =( TICKS TICKVALUES LABEL ) LABEL = " Time ";
      ENDCOLUMNAXES;
      LAYOUT OVERLAY;
        HISTOGRAM T / BINSTART = 250 BINWIDTH = 500 NBINS = 6
                      DISPLAY = ( FILL ) GROUP = Status
                      DATATRANSPARENCY = 0.6;
      ENDLAYOUT;
      LAYOUT OVERLAY / YAXISOPTS =( DISPLAY = ( LINE ));
        BOXPLOT Y=T X = Status / ORIENT = HORIZONTAL BOXWIDTH =.9
                                DATATRANSPARENCY =0.4 GROUP = Status;
        FRINGEPLOT T / GROUP = Status;
      ENDLAYOUT;

      SIDEBAR / ALIGN = RIGHT;
        DISCRETELEGEND " Survival " / TITLE = " Status " BORDER = FALSE
                      ACROSS = 1;
      ENDSIDEBAR;
    ENDLAYOUT;
  ENDGRAPH;
  END;
RUN;

PROC SGRENDER DATA = Survival TEMPLATE = PieChart;
RUN;
```

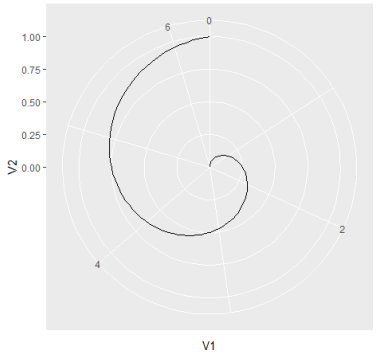


Exploratory survival plot

Step 3 – Pie chart overlapped

As we mention, this is the part that SAS can not do by default, to acomplish that we are going to use polar coordenates and the POLYGON Statement.



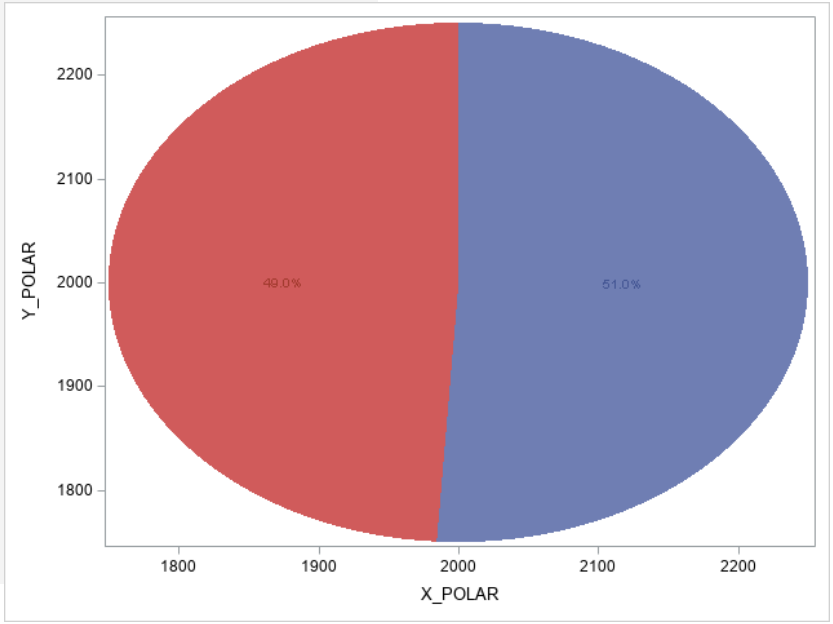


Exploratory survival plot

Step 3 – Pie chart

```
PROC FREQ DATA = Survival ;
  TABLE Status / OUT = Freqout OUTCUM ;
RUN ;

DATA Pie ;
  SET Freqout ;
  /* Origin of arc calculation in percentage */
  RETAIN Start ;
  IF MISSING ( Start ) THEN Start = 0;
  ELSE Start = Cum_PCT - Percent ;
  /* End origin arc in percentage */
  End = Cum_PCT ;
  /* Circumference parameters */
  Center_X = 2000;
  Center_Y = 2000;
  Radius = 250;
  /* Label to display */
  Label = PUT ( Percent ,4.1) ||"%";
*/* To create a smooth circumference for each 0.01% create a dot
  DO i = ROUND (Start ,0.01) TO ROUND (End ,0.01) BY 0.01;
    X_Polar = Center_X +
    Radius * COS( CONSTANT ("PI")/2-i *2*( CONSTANT ("PI")) /100) ;
    Y_Polar = Center_Y +
    Radius * SIN( CONSTANT ("PI")/2-i *2*( CONSTANT ("PI")) /100) ;
    OUTPUT ;
  END ;
*/* To close the circumference , output the center of the figure
  X_Polar = Center_X ;
  Y_Polar = Center_Y ;
  OUTPUT ;
RUN ;
```

```
PROC TEMPLATE;
  DEFINE STATGRAPH Pie;
  BEGINGRAPH;
      LAYOUT OVERLAY ;
        POLYGONPLOT X = X_Polar Y = Y_Polar ID = Status /
           DISPLAY = ( FILL ) GROUP = Status LABEL = Label ;
      ENDLAYOUT;
  ENDGRAPH;
  END;
RUN;

PROC SGRENDER DATA = Pie TEMPLATE = Pie;
RUN;
```

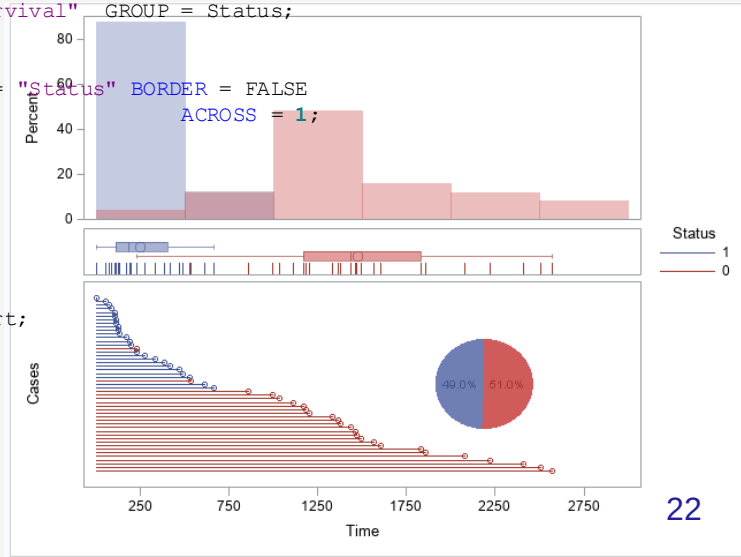


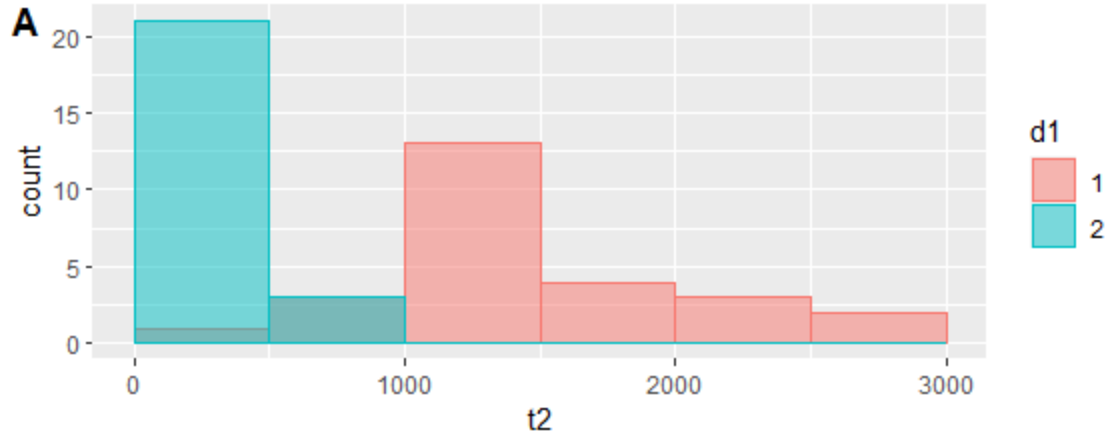
Exploratory survival plot

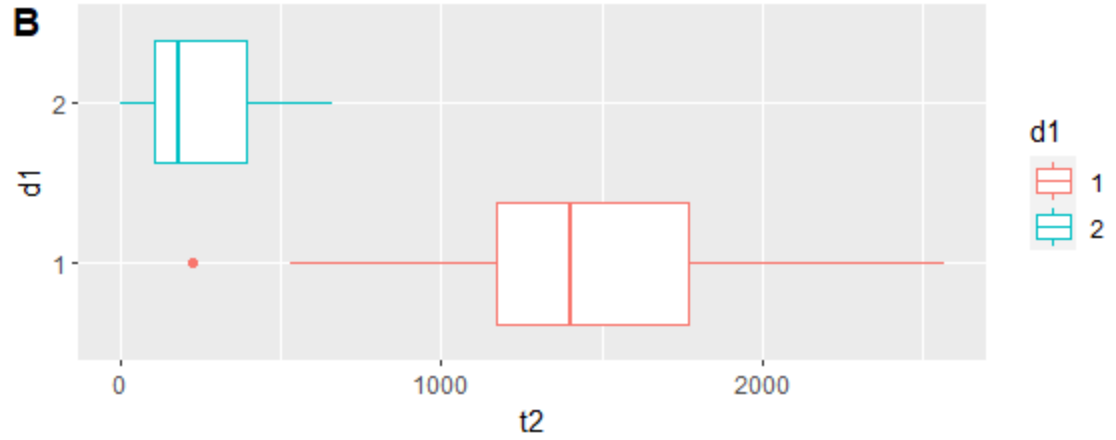
```
DATA Survival;
  SET SASHELP.Bmt;
  IF _N_ < 50;
RUN;


PROC FREQ DATA = Survival;
  TABLE Status / OUT = Freqout OUTCUM;
RUN;


DATA Pie;
  SET Freqout;


  /* Origin of arc calculation in percentage*/
  RETAIN Start;
  IF MISSING(Start) THEN Start = 0;
  ELSE Start = Cum_PCT - Percent;


  /* End origin arc in percentage*/
  End      = Cum_PCT;


  /* Circumference parameters */
  Center_X = 2000;
  Center_Y = 2000;
  Radius   = 250;


  /* Label to display */
  Label    = PUT(Percent,4.1)||"%";
```

```
  /* To create a smooth circumference for each 0.01%
create a dot */
  DO i = ROUND(Start,0.01) TO ROUND(End,0.01) BY 0.01;
    X_Polar = Center_X +
        Radius * COS(CONSTANT("PI")/2-
i*2*(CONSTANT("PI"))/100);
    Y_Polar = Center_Y +
        Radius * SIN(CONSTANT("PI")/2-
i*2*(CONSTANT("PI"))/100);
    OUTPUT;
  END;
  /* To close the circumference, output the center of
the figure */
  X_Polar = Center_X;
  Y_Polar = Center_Y;
  OUTPUT;
RUN;

PROC SORT DATA = Survival;
  BY T;
RUN;

DATA All;
  SET Survival (IN = Obs)
      Pie;
  IF Obs THEN DO;
    /* Origin of survival time */
    Origin = 1;
    SubjectID = _N_;
  END;
RUN;
```

```
PROC TEMPLATE;
  DEFINE STATGRAPH PieChart;
    BEGINGRAPH ;
    LAYOUT LATTICE / ROWS = 3 ROWWEIGHTS = (0.45 0.1 0.45)
                                  COLUMNDATARANGE = UNION;
      COLUMNAXES;
        COLUMNAXIS / DISPLAY=(TICKS TICKVALUES LABEL) LABEL = "Time";
      ENDCOLUMNAXES;
      LAYOUT OVERLAY;
        HISTOGRAM T / BINSTART = 250 BINWIDTH = 500 NBINS = 6
                                    DISPLAY = (FILL) GROUP = Status
                                              DATATRANSPARENCY = 0.6;
      ENDLAYOUT;
                LAYOUT OVERLAY / YAXISOPTS =(DISPLAY = (LINE));
                  BOXPLOT Y=T X = Status / ORIENT=HORIZONTAL BOXWIDTH=.9
                                    DATATRANSPARENCY=0.4 GROUP = Status;
                  FRINGEPLOT T / GROUP = Status ;
                ENDLAYOUT;
      LAYOUT OVERLAY / Y2AXISOPTS = (DISPLAY = (LINE)
                        LINEAROPTS=(VIEWMIN = 1500 VIEWMAX = 2500))
                       X2AXISOPTS = (DISPLAY = (LINE)
                          LINEAROPTS=(VIEWMIN = 0 VIEWMAX = 2750))
                       YAXISOPTS = (DISPLAY = (LINE LABEL)
                                    LABEL = "Cases" REVERSE = TRUE);
        POLYGONPLOT X = X_Polar Y = Y_Polar ID = Status /
                    DISPLAY = (FILL) GROUP = Status LABEL = Label
                                                XAXIS = X2 YAXIS = Y2;
        SCATTERPLOT X = T Y = SubjectID / GROUP = Status;
        HIGHLOWPLOT LOW = Origin HIGH = T Y = SubjectID /
                                    NAME = "Survival"  GROUP = Status;
      ENDLAYOUT;
      SIDEBAR / ALIGN = RIGHT;
        DISCRETELEGEND "Survival" / TITLE = "Status" BORDER = FALSE
                                    ACROSS = 1;

      ENDSIDEBAR;
    ENDLAYOUT;
    ENDGRAPH;
  END;
RUN;

/* Generate the graph */
PROC SGRENDER DATA = Pie TEMPLATE = PieChart;
RUN;
```

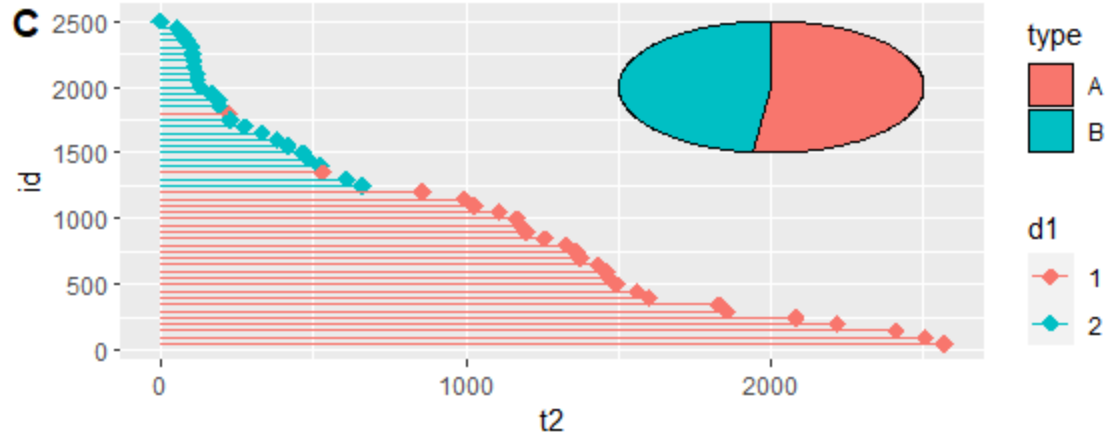




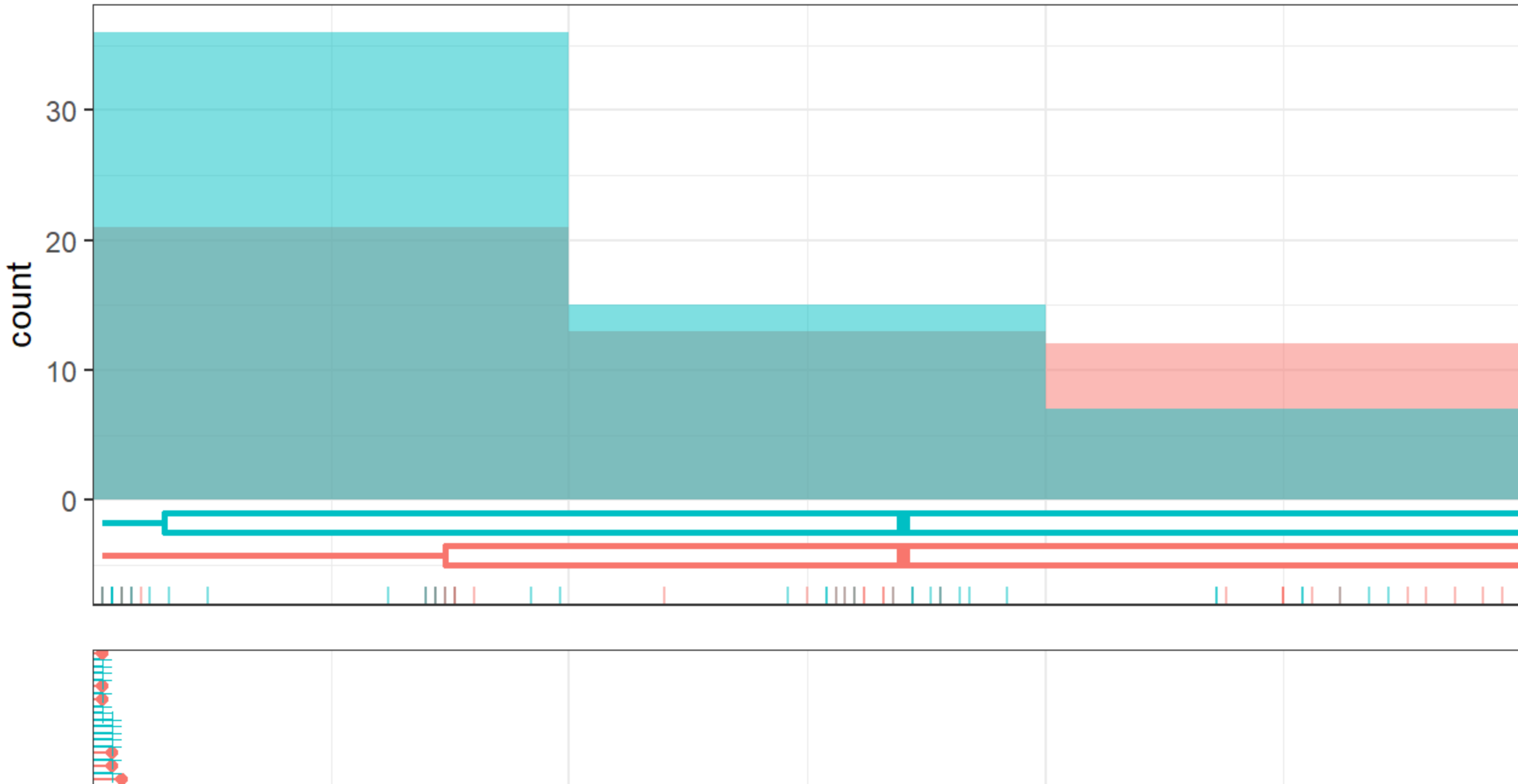


.5,position="identity")

gment(aes(xend=0,yend=id)) +

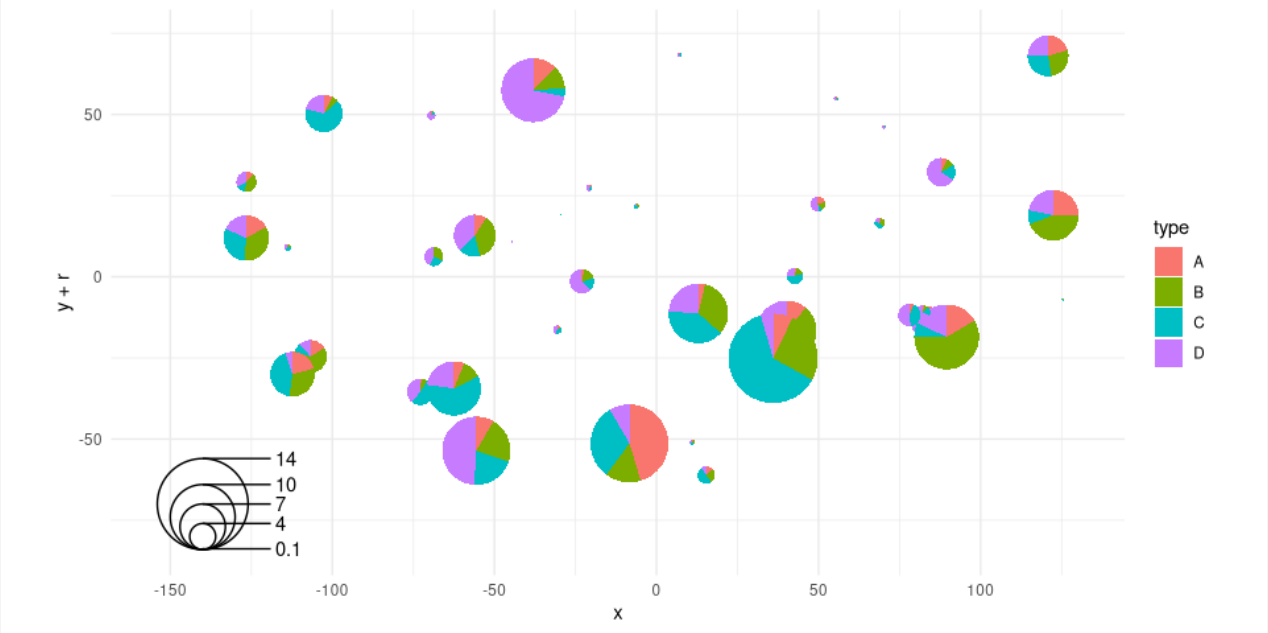


Fraction of events to censored observations



Improvement for SAS

- Option to overlay pie charts too (For comparison, in R you can add an scatterpie, where the coordinates and radius can be specified)



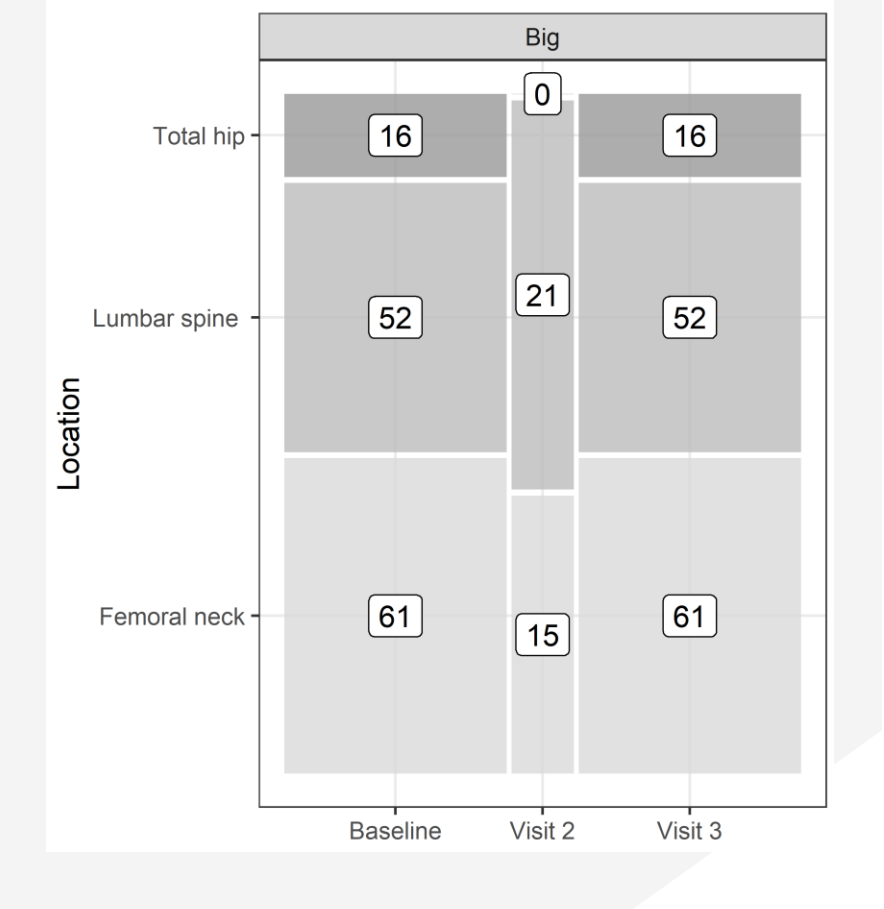
Mosaic plot

Mosaic plots (Hartigan and Kleiner, 1981;) are used for exploratory data analysis of categorical data. Mosaic plots have been available for decades in SAS products such as JMP, SAS/INSIGHT. Since SAS 9.3v2 with base SAS we have access to these outputs with GTL.

To put frequencies over the figure, annotations functionality was needed. This has the drawback of 2 datasets to validate.

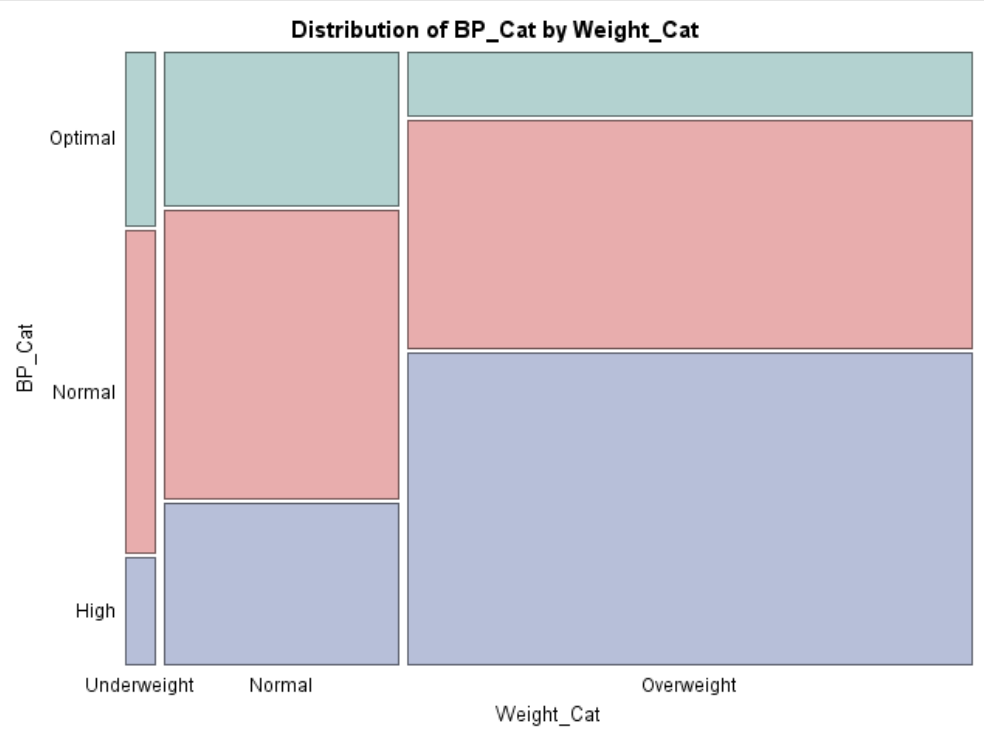
Annotation method can be found

https://blogs.sas.com/content/iml/2019/07/08/add-annotation-mosaic-plot-sas.html



Problems to create Mosaic plot in SAS

- MOSAICPLOT by default do not allow to add notation without extra tools, annotation and an additional dataset are needed to create the annotation and thus, two datasets to validate

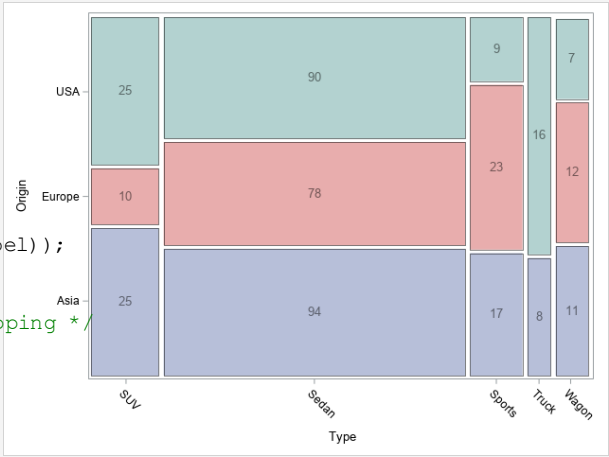


Mosaic SAS code

```
PROC FREQ DATA = SASHELP.Cars;
   WHERE Type ne 'Hybrid';
   TABLES Type / OUT = TypeFreq OUTCUM ;
RUN;
PROC FREQ DATA = SASHELP.Cars;
   WHERE Type ne 'Hybrid';
   TABLES Origin * Type / OUT = CrossFreq
    (WHERE=(PERCENT ne .)) OUTPCT;
RUN;
DATA TypeFreq;
  SET TypeFreq;
  Start_X = Cum_Pct - Percent;
  End_X   = Cum_Pct;
  Midpoint_X = (Start_X + End_X) / 2;
  KEEP Type Start_X End_X Midpoint_X;
RUN;
PROC SORT DATA = CrossFreq;
  BY Type Origin;
RUN;
DATA CrossFreq;
  SET CrossFreq;
  BY Type;
  RETAIN Cum_Pct 0 MosaicID 0;
  MosaicID = MosaicID + 1;
  IF first.Type THEN Cum_Pct = 0;
  Cum_Pct = Cum_Pct + Pct_Col;
  Start_Y = Cum_Pct - Pct_Col;
  End_Y   = Cum_Pct;
  /* Midpoint on first Y category only */
  IF Type = "SUV" THEN Midpoint_Y = (Start_Y + End_Y) / 2;
  KEEP Type Origin Start_Y End_Y Count MosaicID Midpoint_Y;
RUN;
```

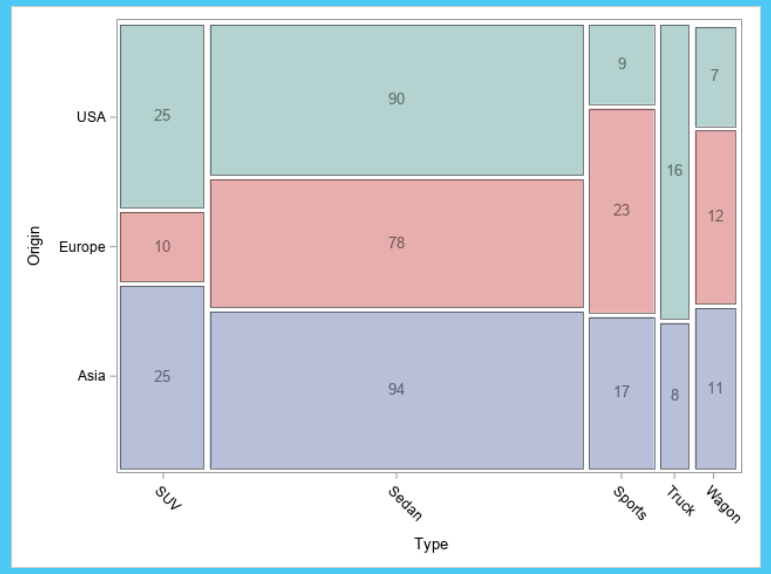
```
DATA All;
  MERGE CrossFreq
        TypeFreq;
  BY Type;
  /* Space between blocs */
  Gap = 1;
  /* Create rectangles for each category */
  X = Start_X;
  IF Start_X ne 0 THEN X = X + Gap/2;
  Y = Start_Y;
  IF Start_Y ne 0 THEN Y = Y + Gap/2;
  OUTPUT;
  X = Start_X;
  IF Start_X ne 0 THEN X = X + Gap/2;
  Y = End_Y;
  IF End_Y ne 100 THEN Y = Y - Gap/2;
  OUTPUT;
  X = End_X;
  IF End_X ne 100 THEN X = X - Gap/2;
  Y = Start_Y;
  IF End_Y ne 100 THEN Y = Y - Gap/2;
  OUTPUT;
  X = End_X;
  IF End_X ne 100 THEN X = X - Gap/2;
  Y = Start_Y;
  IF Start_Y ne 0 THEN Y = Y + Gap/2;
  OUTPUT;
  LABEL X = "Type"
        Y = "Origin";
RUN;
DATA TypeFmt;
   LENGTH Label $8;
   SET TypeFreq (RENAME = (Type = Label));
   RETAIN FmtName 'Type_n' Type 'n';
   /* Round to avoid issues with overlapping */
   Start = ROUND(End_X,0.001);
   End   = ROUND(End,0.001);
   OUTPUT;
   KEEP Start End Label FmtName;
RUN;

PROC FORMAT LIBRARY=WORK CNTLIN=TypeFmt;
RUN;
```

```
DATA OriginFmt;
   LENGTH Label $8;
   SET CrossFreq (RENAME = (Origin = Label));
   WHERE Type = "SUV";
   RETAIN FmtName 'Origin_n' Type 'n';
   /* Round to avoid issues with overlapping */
   Start = ROUND(Start_Y,0.001);
   End   = ROUND(End_Y,0.001);
   OUTPUT;
   KEEP Start End Label FmtName;
run;

PROC FORMAT LIBRARY=WORK CNTLIN=OriginFmt;
RUN;

PROC SQL NOPRINT;
  SELECT Midpoint_X INTO :TypeList SEPARATED BY " "
  FROM TypeFreq;
QUIT;

PROC SQL NOPRINT;
  SELECT Midpoint_Y INTO :OriginList SEPARATED BY " "
  FROM CrossFreq WHERE Type = "SUV";
QUIT;

PROC TEMPLATE;
  DEFINE STATGRAPH MyMosaic;
    BEGINGRAPH;
LAYOUT OVERLAY /
      XAXISOPTS = (LINEAROPTS = (TICKVALUEFORMAT = Type_n.
                                 TICKVALUELIST = (&Typelist.)
                                 TICKVALUEFITPOLICY = ROTATE))
      YAXISOPTS = (LINEAROPTS = (TICKVALUEFORMAT = Origin_n.
                                 TICKVALUELIST = (&Originlist.)));
         POLYGONPLOT X=X Y=Y ID = MosaicID  /  DATATRANSPARENCY = 0.5
                  DISPLAY=(FILL OUTLINE) GROUP = Origin LABEL = Count
                  LABELATTRS=(COLOR = BLACK SIZE = 10)
                  OUTLINEATTRS=(COLOR = BLACK);
    ENDLAYOUT;
  ENDGRAPH;
  END;
RUN;

PROC SGRENDER DATA = All TEMPLATE = MyMosaic;
RUN;
```

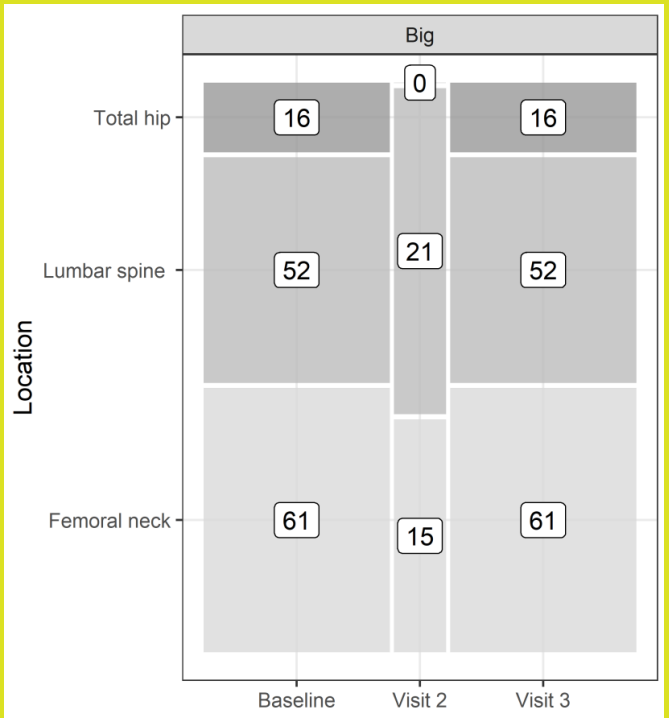


Comparison

SAS

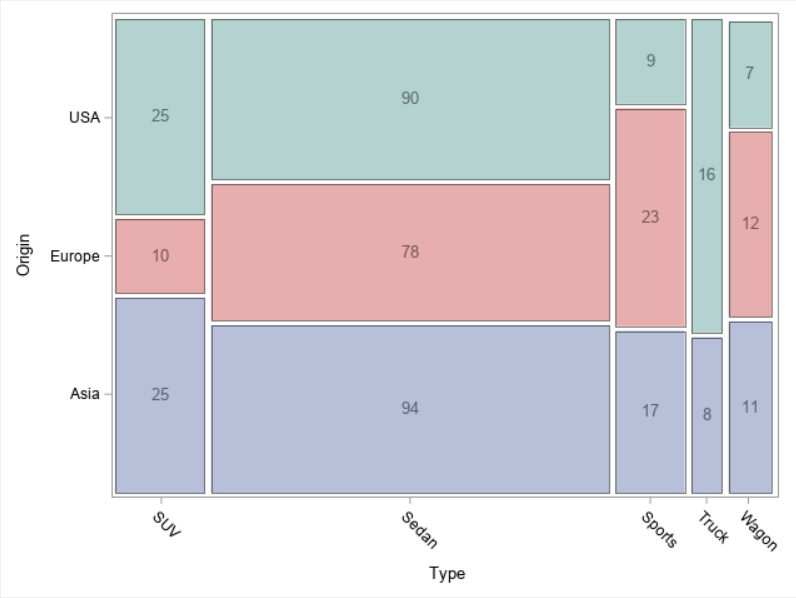


R



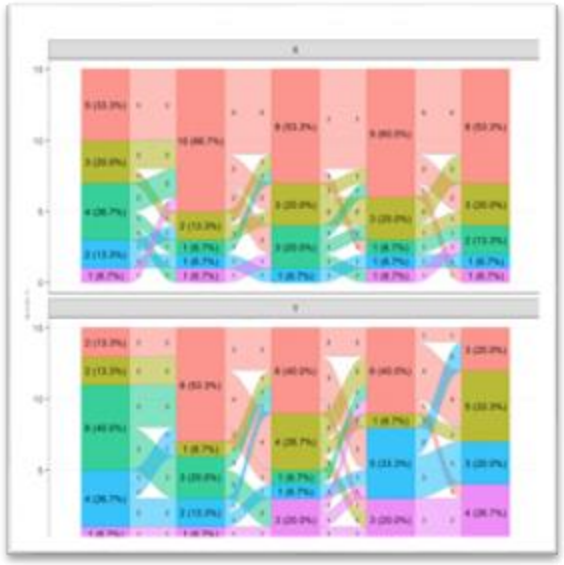
Improvement for SAS

- Include to MOSAICPLOT statement a LABEL option to be able to include content



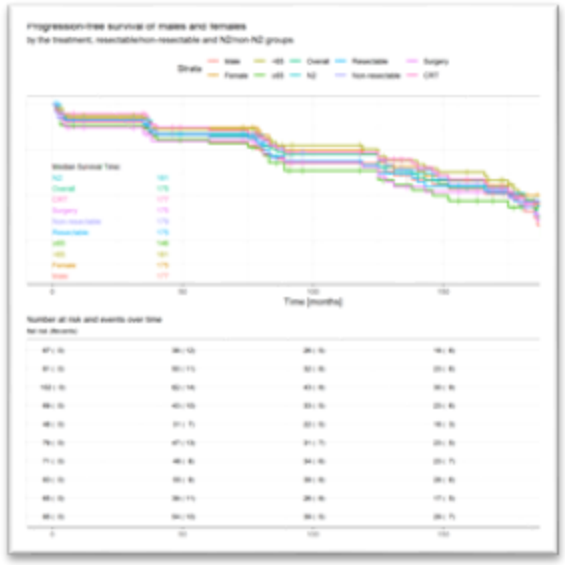
Other figures discussed

The set of figures shown where only a subset of the ones reproduced. Other figures mentioned where QQ plot with CI, different spaghetti plots or Cullen-Frey among others.



Alluvial plot

https://github.com/RhoInc/sas-sankeybarchart [1,8]



Survival plot + table

They can be including a XAXISTABLE to a survival plot [7]

Survival panel

It can be created using PROTOTYPE

Conclusion

- Both programs are equally capable

- That said, R incorporate new tools for plot creation with more frequency an those are more flexible to be used in a wider spectrum of scenarios that allow for a more straightforward approach to certain figure

References

[1] Ryan Bailey and Shane Rosanbalm. sas-sankeybarchart. 2017. https://github.com/RhoInc/sas-sankeybarchart

[2] John Fox. Applied Regression Analysis and Generalized Linear Models. SAGE Publications, 2008.

[3] Sakshi Gupta. SAS vs R: Which is Better? Feb. 2020. https://in.springboard.com/blog/sas-vs-r/

[4] Volker Harm Catalina Meja Herrera. Grouped Jittered Box Plots in SAS 9.2 and SAS 9.3. 2012. https://lexjansen.com/phuse/2012/cs/CS03.pdf

[5] Sanjay Matange. Violin Plots. Oct. 2012. url: https://blogs.sas.com/content/graphicallyspeaking/2012/10/30/violin-plots/

[6] Priya Pedamkar. Dierence Between SAS and R. url: https://www.educba.com/sas-vs-r/

[7] Antonio Rodriguez and Sebastia Barcelo. Guide for Producing Figures with Graphic Template Language (GTL) Using SAS. Lulu, 2020.

[8] Shane Rosanbalm. \Getting Sankey with Bar Charts". In: PharmaSUG (2015).

[9] SAS vs R: Which One is Better Statistics Language. Aug. 2019. https://www.codeavail.com/blog/sas-vs-r/

[10] Rick Wicklin. How to add an annotation to a mosaic plot in SAS. July 2019.

https://blogs.sas.com/content/iml/2019/07/08/add-annotation-mosaic-plot-sas.html.

Questions?