GGPLOT2 Advanced

Emily Cheng

Aug 18, 2022

GGPLOT2 Advanced

1. GGPLOT2 Syntax Overview
2. Case 1: Color, Legend and Text Modification
3. Case 2: KM Plot
4. Case 3: Forest Plot
5. Summary

2022 PharmaSUG
Pre-conference Seminar

1. Syntax Overview

With ggplot2, you begin a plot with a call to the function ggplot(). Then you can build a graph upon that coordinate system, layer by layer.

The following seven elements, or layers, constitute what is referred to as “the layered grammar of graphics”.

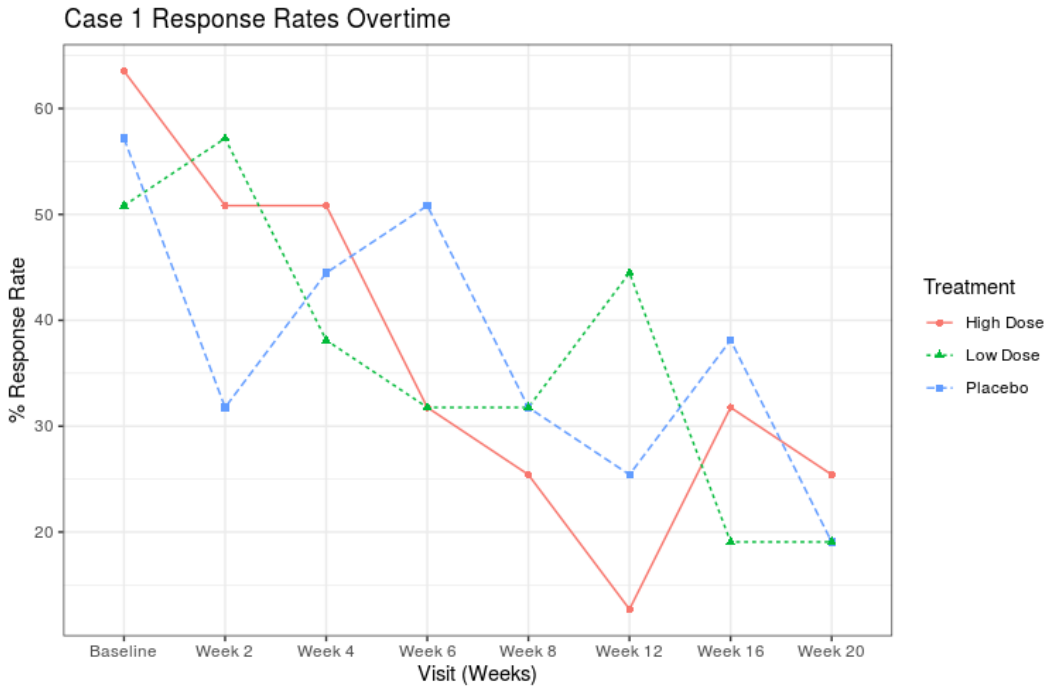
PharmaSUG Seminar

Graphical Elements	
Data	The data frame you want to plot
Aesthetics	The scales onto which we wish to map our data (e.g., x, y).
Geometries	The visual elements used in our data (i.e., how the plot will look)
Coordinates	The space on which data will be plotted
Statistics	Representations of our data to aid understanding
Facets	Plotting small multiples
Themes	All non-data ink

Focus: first three layers (required by ggplot2)

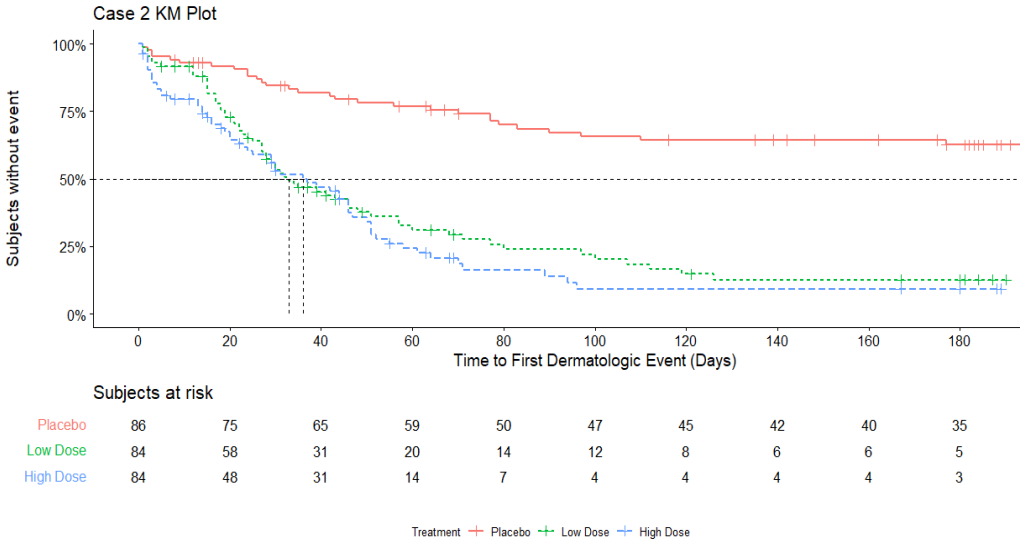
1. Syntax Overview

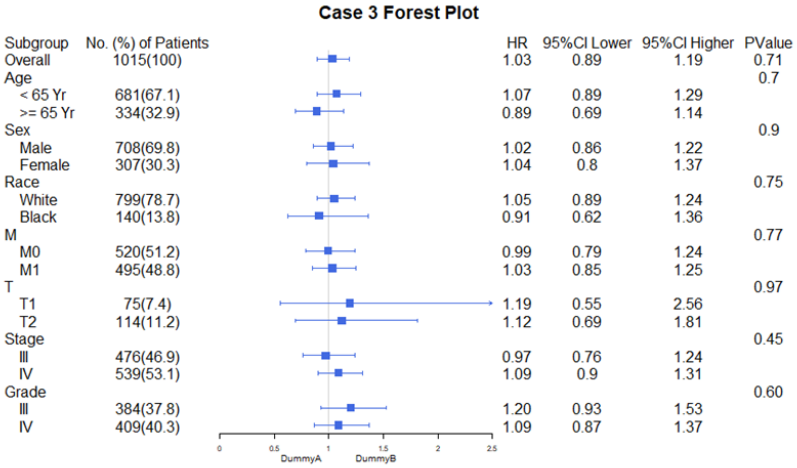
```
ggplot(data=rspsum, aes(y=RESPONSE, x=AVISIT, group=TRT, color=TRT)) +
  geom_line(aes(linetype=TRT) ) +
  geom_point(aes(shape=TRT) ) +
  labs(title = 'Case 1 Response Rates Overtime',
            x = 'Visit (Weeks)',
            y = '% Response Rate',
        shape = 'Treatment',
    linetype = 'Treatment',
        color = 'Treatment') +
  theme_bw()
```



2022 PharmaSUG Pre-conference Seminar

After this session, you will be able to create several figures...





Color, Legend and Text Modification

Aug 18, 2022

2. Color, Legend and Text Modification – Case 1

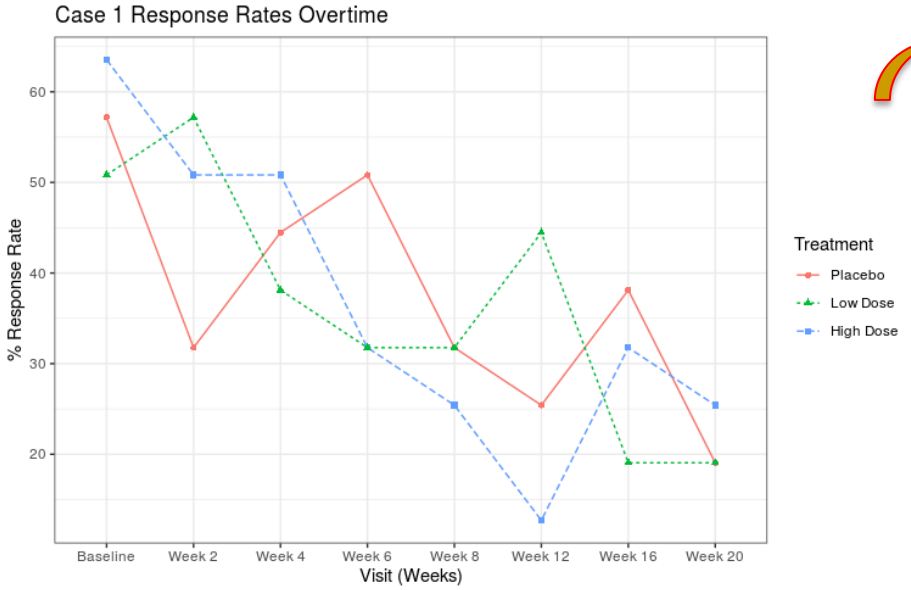
In this case, we are going to create graphs with customized modification in color, legend and text. Let's start with a basic plot.

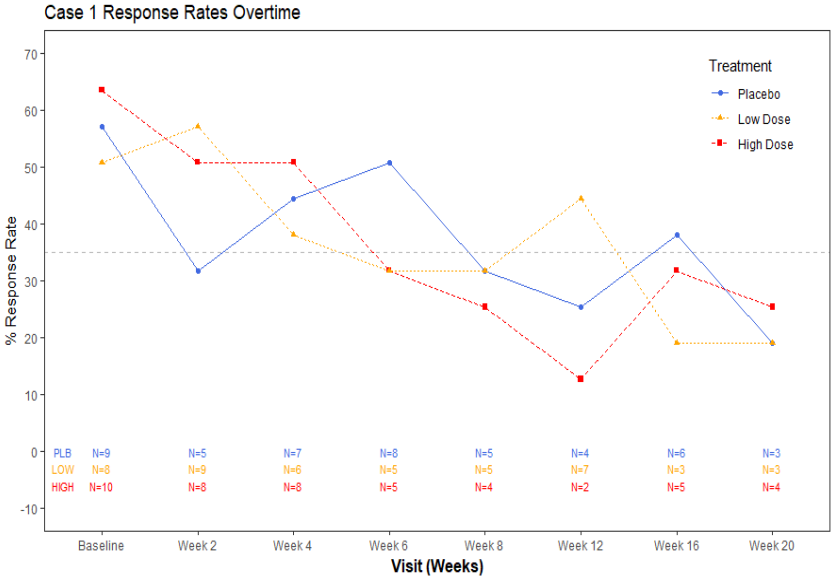
rspsum

AVISITN	AVISIT	TRT	RESPONSE	COUNT
0	Baseline	Placebo	57.17916137	9
0	Baseline	High Dose	63.53240152	10
0	Baseline	Low Dose	50.82592122	8
2	Week 2	Placebo	31.76620076	5
2	Week 2	High Dose	50.82592122	8

```
library(ggplot2)
library(dplyr)
library(haven)
```

PharmaSUG China Seminar

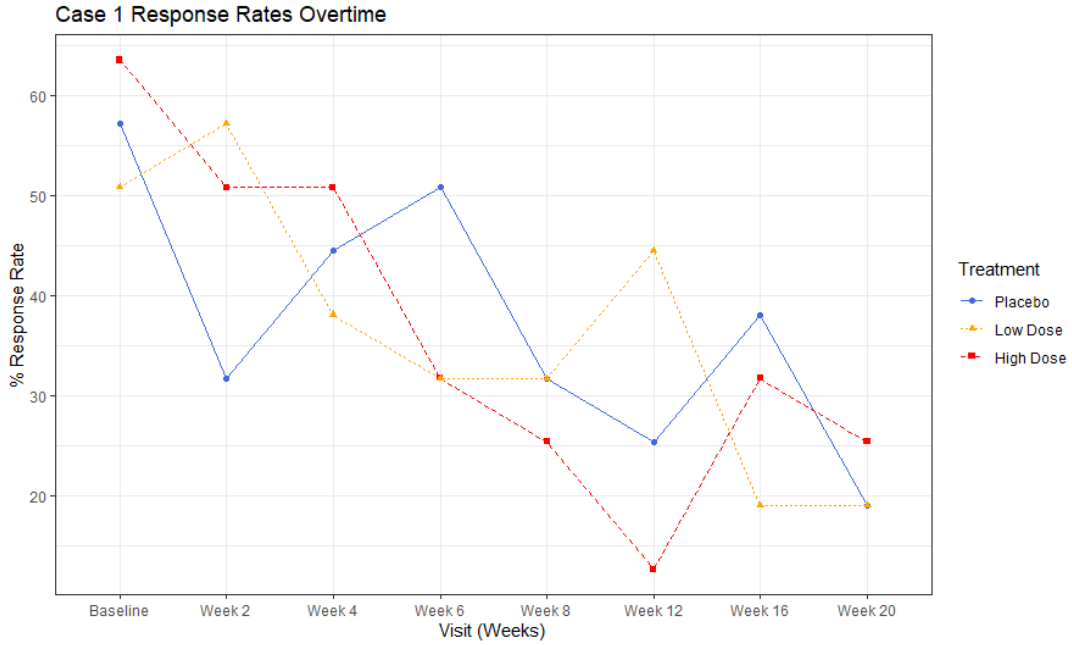




2.1 Color

I want to change default colors.

```
p + scale_color_manual(values=c("royalblue", "orange", "red"))
```

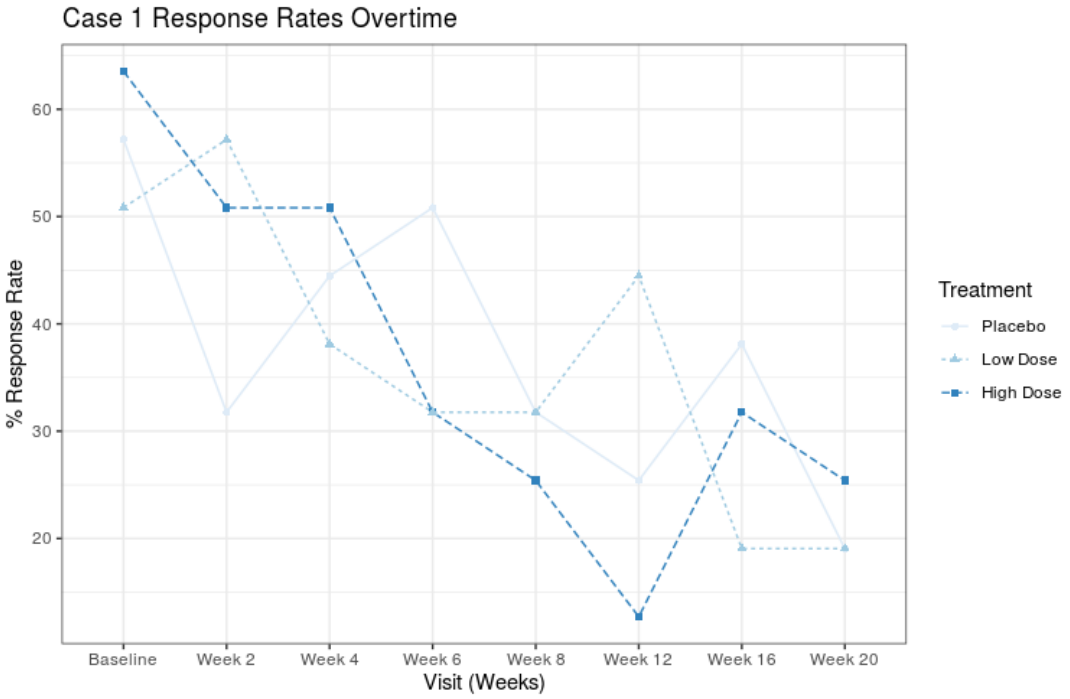


PharmaSUG

2.1 Color

I want to change colors to gradient ones.

```
p + scale_color_brewer(palette="Blues",direction=1)
```



2.1 Color

```
scale_color_brewer(palette='Blues', direction=1)
```

There are 3 types of palettes
- sequential
- diverging
- qualitative

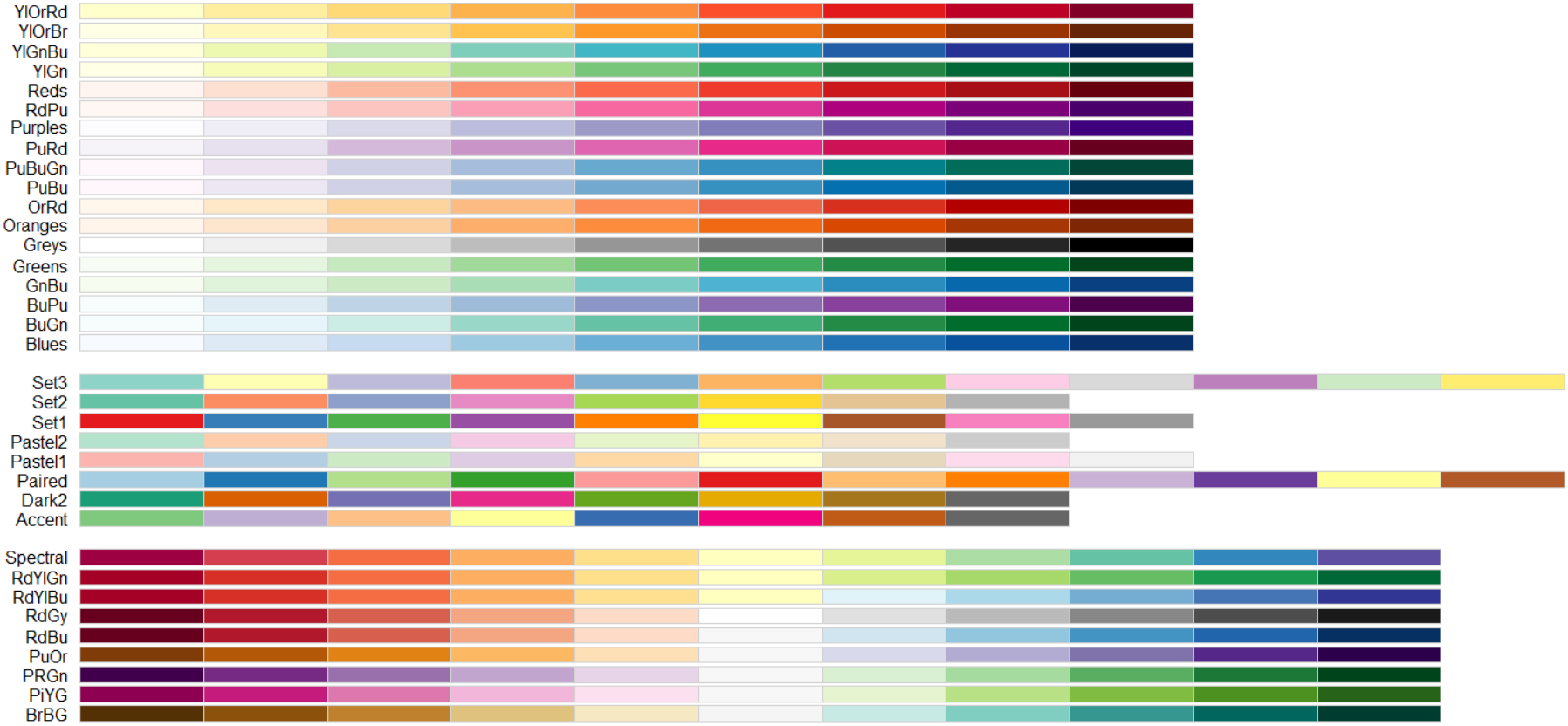
Two options for direction:
- 1
- -1, reversed

These are particularly well suited to display discrete values on a map.
You can view all available palettes through http://www.colorbrewer.org or typing

```
library("RColorBrewer")
RColorBrewer::display.brewer.all()
```

2022 PharmaSUG Pre-conference Seminar

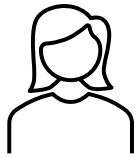
2.1 Color

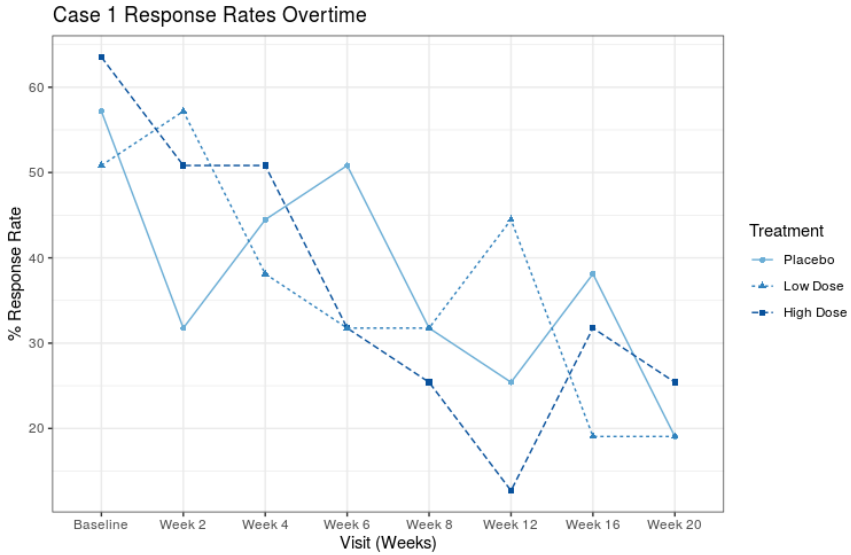

YlOrRd
YlOrBr
YlGnBu
YlGn
Reds
RdPu
Purples
PuRd
PuBuGn
PuBu
OrRd
Oranges
Greys
Greens
GnBu
BuPu
BuGn
Blues
Set3
Set2
Set1
Pastel2
Pastel1
Paired
Dark2
Accent
Spectral
RdYlGn
RdYlBu
RdGy
RdBu
PuOr
PRGn
PiYG
BrBG

2.1 Color

The lightest color is too faint to see. Can I change default colors from palette?

```
library("RColorBrewer")

mypalette<-brewer.pal(5, "Blues")

mypalette[3:5]

p + scale_color_manual(values=mypalette[3:5])
```





2022 PharmaSUG Pre-Conference Seminar

2.1 Color: Quick Summary

- Specify your own set of colors

```
p + scale_color_manual(values=c("royalblue",  "orange", "Red"))
```

- For discrete_scale

```
p + scale_color_brewer(palette="Blues",direction=1)
```

- For continuous_scale

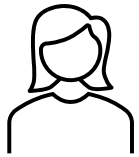
```
p + scale_colour_gradient(low = "white", high = "black")
p + scale_colour_gradient2(low = "red", mid= "white", high = "blue")
p + scale_colour_gradientn(colours = terrain.colors(10))
```

2022 PharmaSUG Pre-conference Seminar

2.2 Text

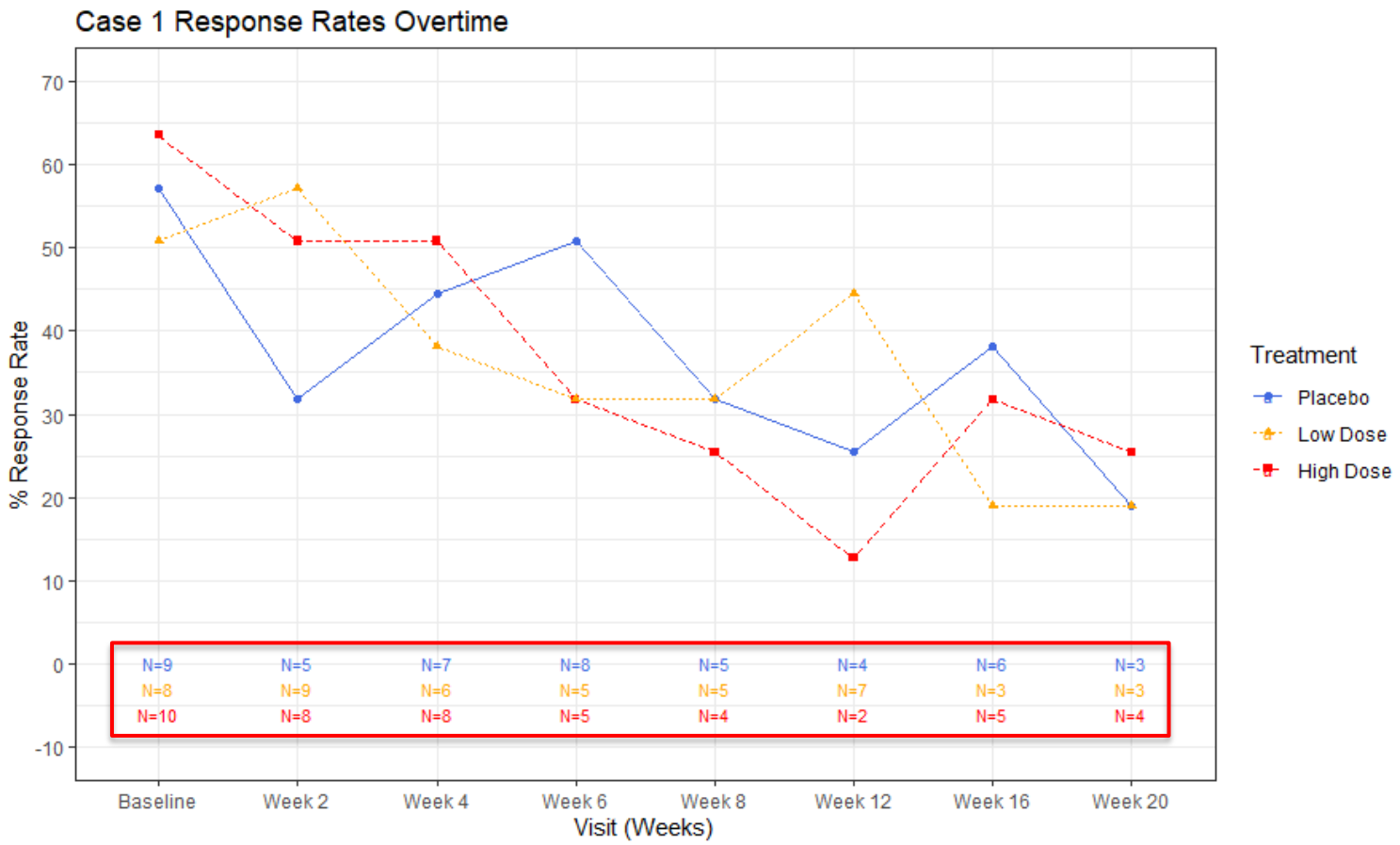
I want to add text in plot to provide more information.

```
p + geom_text()
p + geom_label()
p + annotate()
```

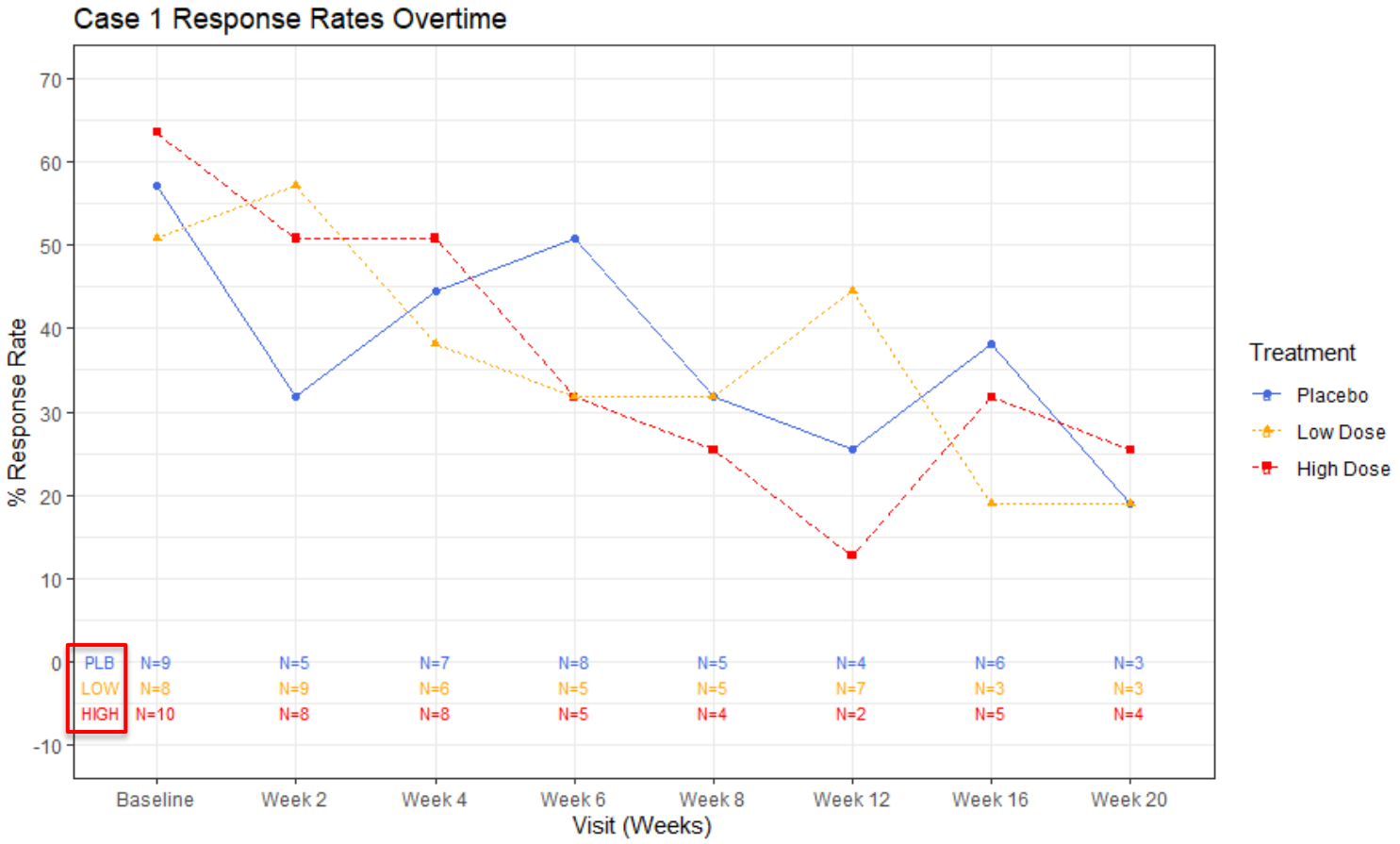


2022 PharmaSUG Pre-conference Seminar

```
p + geom_text(aes(y=ytext,x=AVISIT,label=paste0("N=",COUNT)),size=3)+
    scale_y_continuous(limits=c(-10,70),breaks=seq(-10,70,by=10))
```



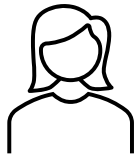
```
p + annotate("text",x=0.6,y=0 ,label="PLB", size=3, color="royalblue") +
    annotate("text",x=0.6,y=-3,label="LOW", size=3, color= "orange"  ) +
    annotate("text",x=0.6,y=-6,label="HIGH",size=3, color= "red"     )
```

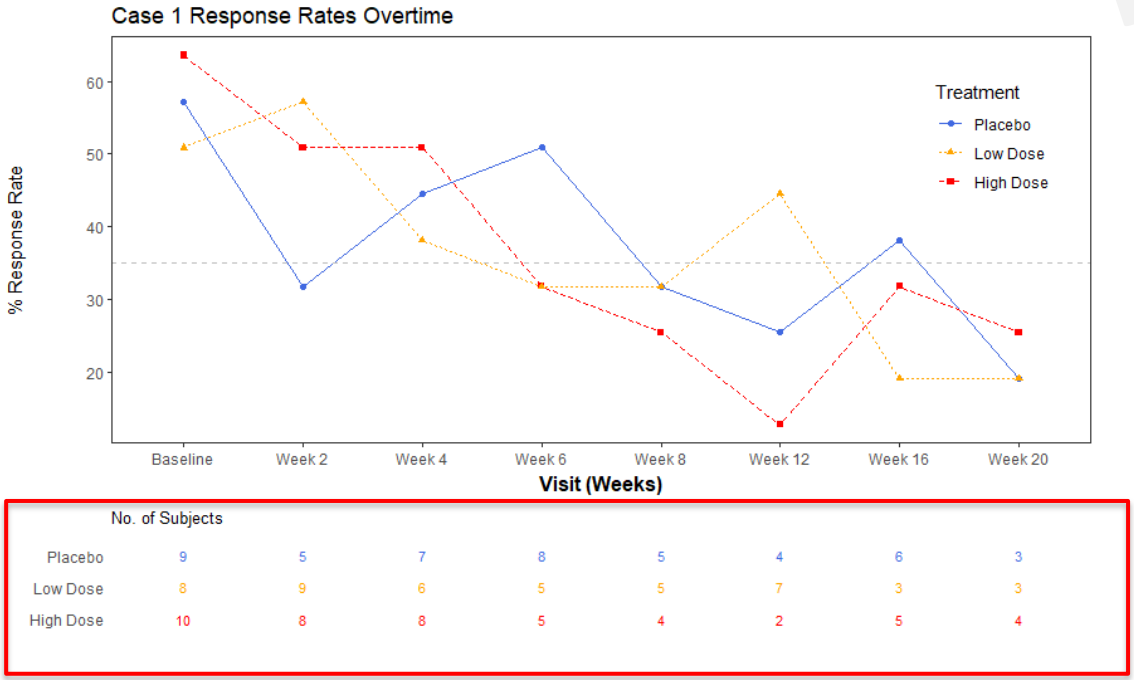


2.2 Text

I tried many times just to put text in a suitable position.
Is there a smart way?

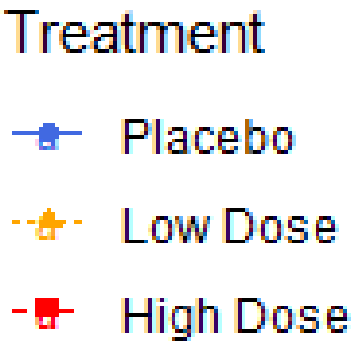
We have another solution to add text table under a plot.



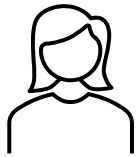


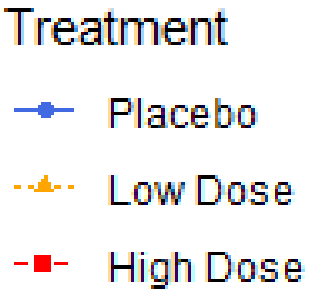
2.3 Legend

Something wrong with the legend.




```
p + geom_text(..., show.legend=F, ...)
```





2022 PharmaSU
Pre-conference Semi

2.3 Legend

I want to put the legend inside plot.

```
p + theme(legend.position=c(0.9,0.85))
```

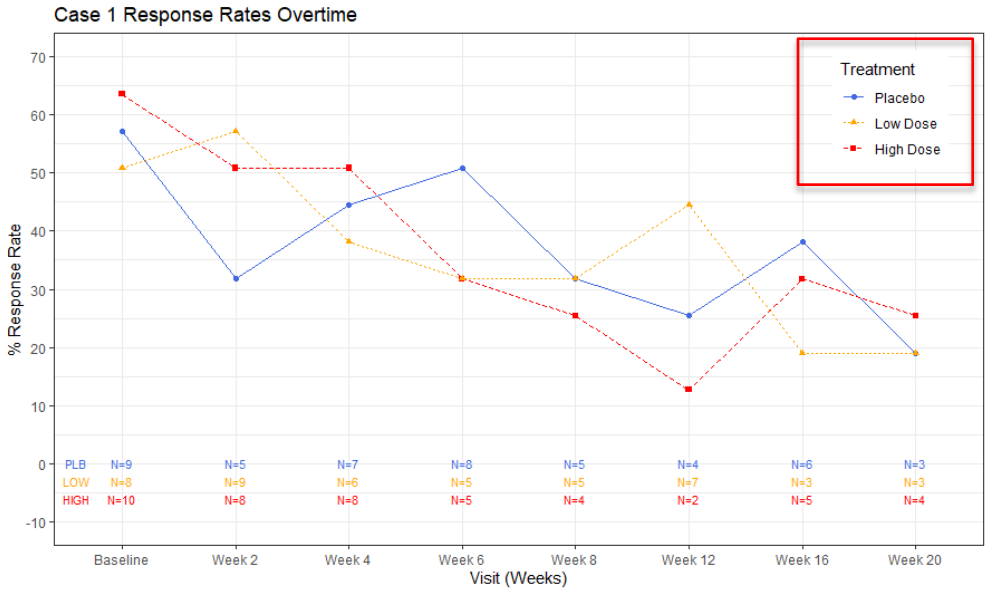
2022 PharmaSUG Pre-conference Seminar

The position of legends can be

- “none”
- “left”
- “right”
- “bottom”
- “top”

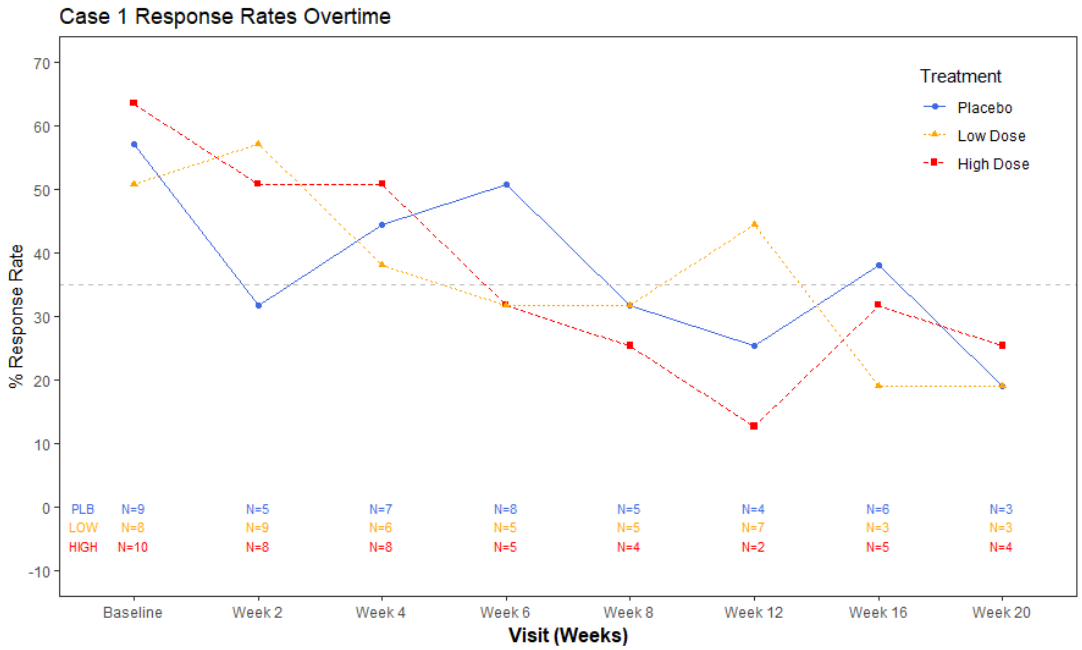
or

- two-element numeric vector



Finally …

```
p + theme(panel.grid.major=element_blank(),
          panel.grid.minor=element_blank(),
          axis.title.x = element_text(size=12, face="bold")) +
    geom_hline(yintercept=35,color="grey",linetype=2)
```

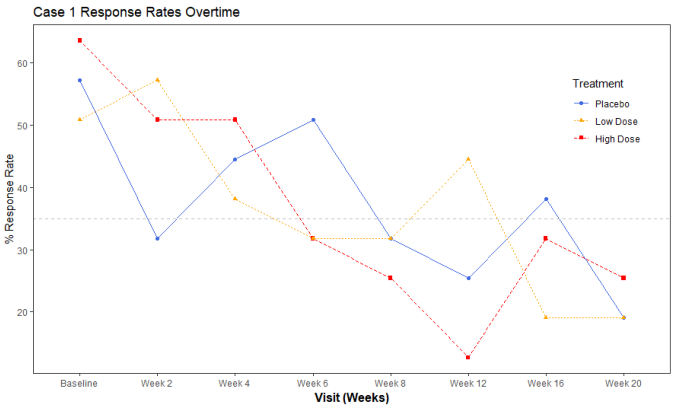


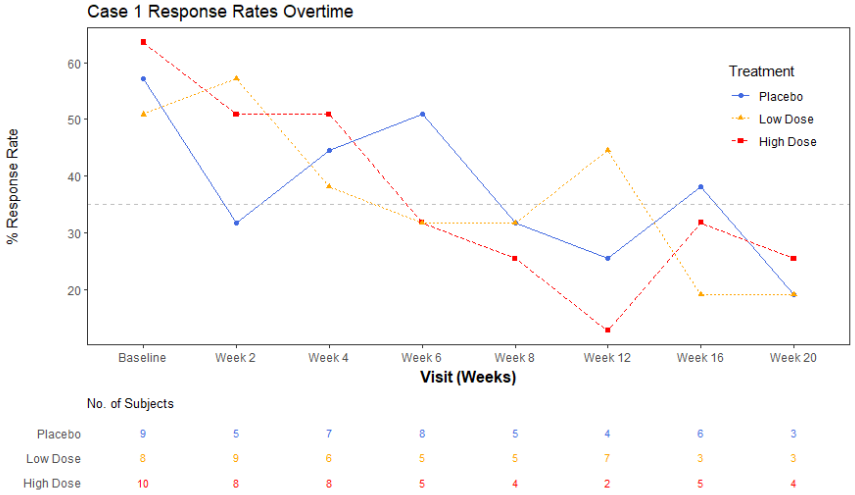
2022 PharmaSUG Pre-conference minar

AND

- We have another solution for plot with text table...

```
library(patchwork)

p1 <- ggplot(...)
p2 <- ggplot(...)

p1 / p2 +  plot_layout(heights = c(8, 2))
```





2. Color, Legend and Text Modification - Summary

In this case, we've continued to explore ggplot2 syntax in a customized plot with modification in:

- Color
  - Change default colors
  - Use palette
- Legend
  - Not show legend
  - Change legend position
- Text table
  - geom_text()
  - annotate()
  - Text table

2022 PharmaSUG Pre-conference Seminar

KM Plot

Aug 18, 2022

3. KM Plot

3.1 Meet New Packages

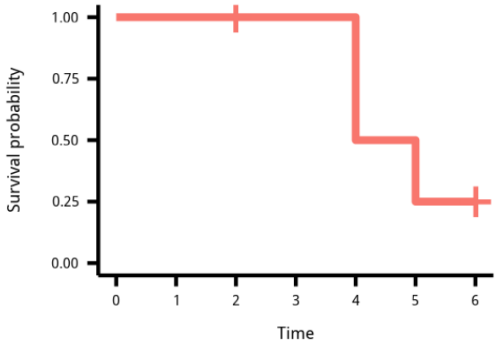
3.2 Create a Survival Object

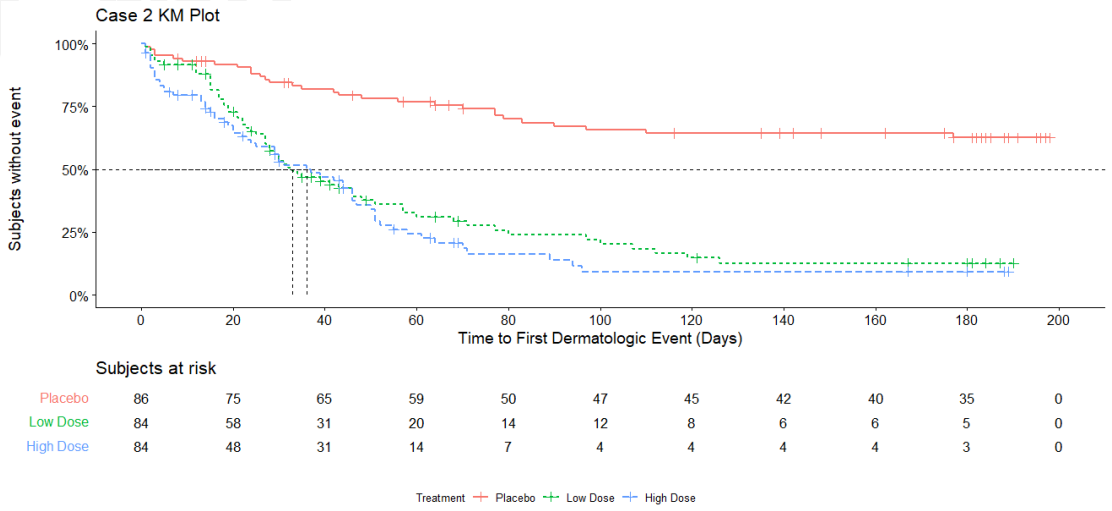
3.3 Create Survival Curves

3.4 Draw Survival Curves

3.5 KM Plot – Case 2

- One basic KM plot
- Other modifications





2022 Pharm
Pre-conference Se
r

3.1 Meet New Packages

In order to create KM plot, two new packages are need.

```
library(survival)
```

“survival” contains the core survival analysis routines, including definition of Surv objects, Kaplan-Meier curves, Cox models, etc.

```
library(survminer)
```

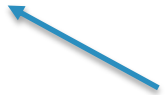
“survminer” contains the function 'ggsurvplot()' for drawing easily beautiful and 'ready-to-publish' survival curves with the 'number at risk' table, etc.

2022 PharmaSUG Pre-conference Seminar

3.2 Create a Survival Object

Survival Object, or Surv Object, is an object combined the variables time and event together.

```
Surv(time, event)
```



"event": The status indicator

0=alive, **1=dead**

TRUE or FALSE (TRUE = death)
1 or **2** (2=death)

Example:

```
time <- c(5, 6, 2, 4, 4)
event <- c(1, 0, 0, 1, 1)

Surv(time, event)
```

```
> Surv(time, event)
[1] 5  6+ 2+ 4  4
```

Survival Object is usually used as a response variable in a model formula.

2021 SUG Pre-conference Seminar

3.3 Create Survival Curves

We use survfit() function to compute the Kaplan-Meier curve.

```
survfit(formula, ...)
```

This function creates survival curves from a formula (e.g. the Kaplan-Meier).

```
survfit(Surv(time, event) ~ 1)
```

It returns an object of class survfit which contains one or more survival curves.

```
> fit <- survfit(Surv(time,event)~1)
> class(fit)
[1] "survfit"
> str(fit)
List of 13
 $ n         : int 5
 $ time      : num [1:4] 2 4 5 6
 $ n.risk    : num [1:4] 5 4 2 1
 $ n.event   : num [1:4] 0 2 1 0
 $ n.censor  : num [1:4] 1 0 0 1
 $ surv      : num [1:4] 1 0.5 0.25 0.25
 $ type      : chr "right"
 $ std.err   : num [1:4] 0 0.5 0.866 0.866
 $ lower     : num [1:4] 1 0.1877 0.0458 0.0458
 $ upper     : num [1:4] 1 1 1 1
 $ conf.type: chr "log"
 $ conf.int : num 0.95
 $ call      : language survfit(formula = Surv(time, event) ~ 1)
 - attr(*, "class")= chr "survfit"
```

3.4 Draw Survival Curves

We use ggsurvplot() function to plot survival curves.

```
ggsurvplot(fit,
           data = NULL,
           fun = NULL,
           palette = NULL,
           linetype = 1,
           conf.int = FALSE,
           pval = FALSE,
           surv.median.line = "none",
           risk.table = FALSE,
           tables.height = 0.25,
           ggtheme = theme_survminer(),
           tables.theme = ggtheme,
           ...)
```

allowed values include:

- **a survfit object**
- a list of survfit objects. Passed to ggsurvplot_list()
- a data frame containing survival curves summary. Passed to ggsurvplot_df().

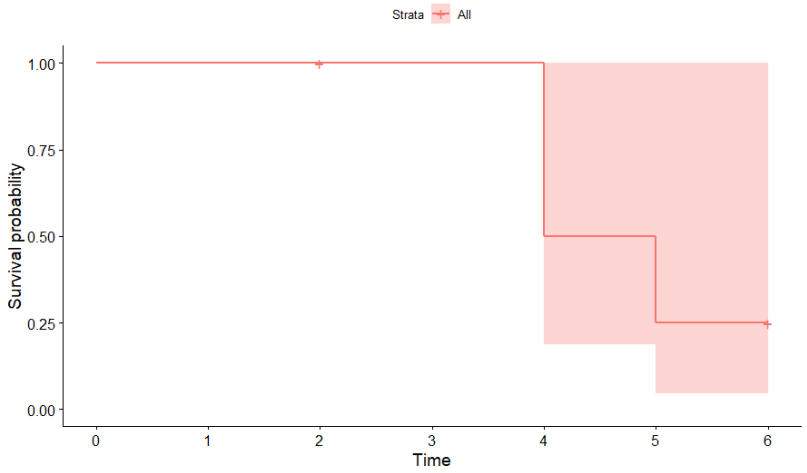
Example：

```
indata <- data.frame(time=c(5, 6, 2, 4, 4),event=c(1, 0, 0, 1, 1))
fit <- survfit(Surv(time,event)~1,data=indata)

ggsurvplot(fit)
```

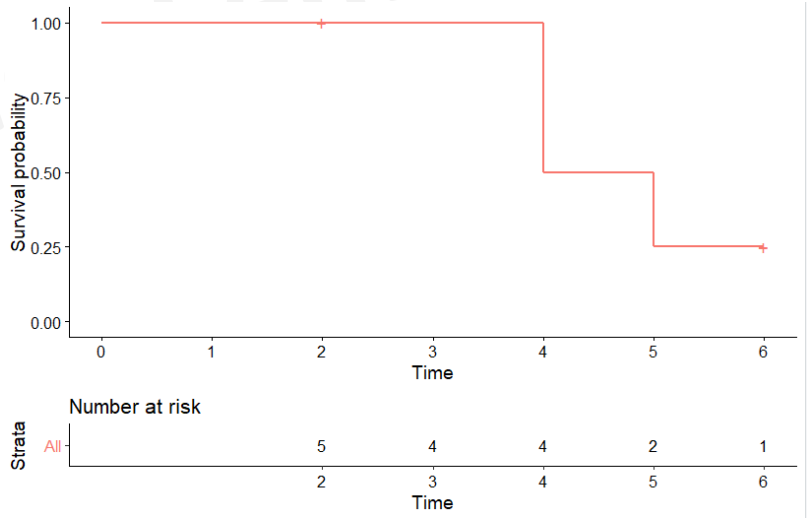
After Survminer 0.4.1, in ggsurvplot() the data argument should be strictly provided, either in ggsurvfit() or survfit().

2022 Pharm Pre-conference Sem...

```
ggsurvplot(fit,
           indata)
```




```
ggsurvplot(fit,
           data        = indata,
           conf.int    = FALSE,
           risk.table  = TRUE,
           legend      = "none")
```



2022 Pharm Preconferen Semin

3. KM Plot - Quick Summary

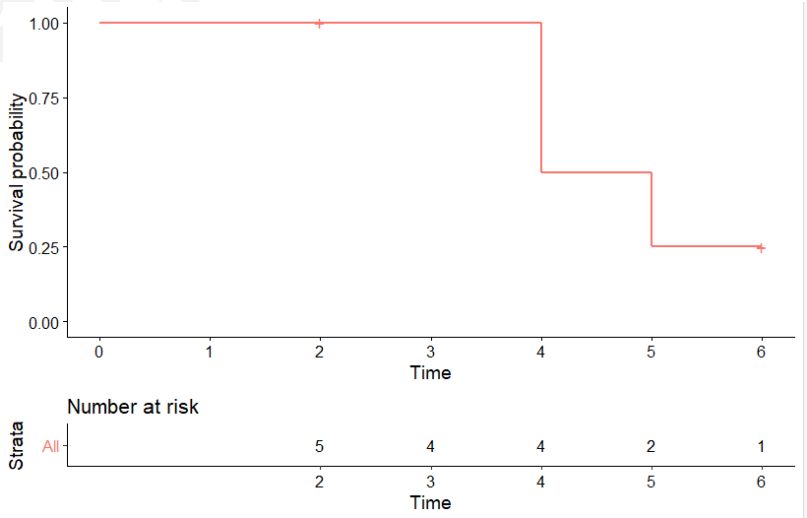
For all kinds of analyses:

library(survival) → Surv(), survfit()

For pretty visualization:

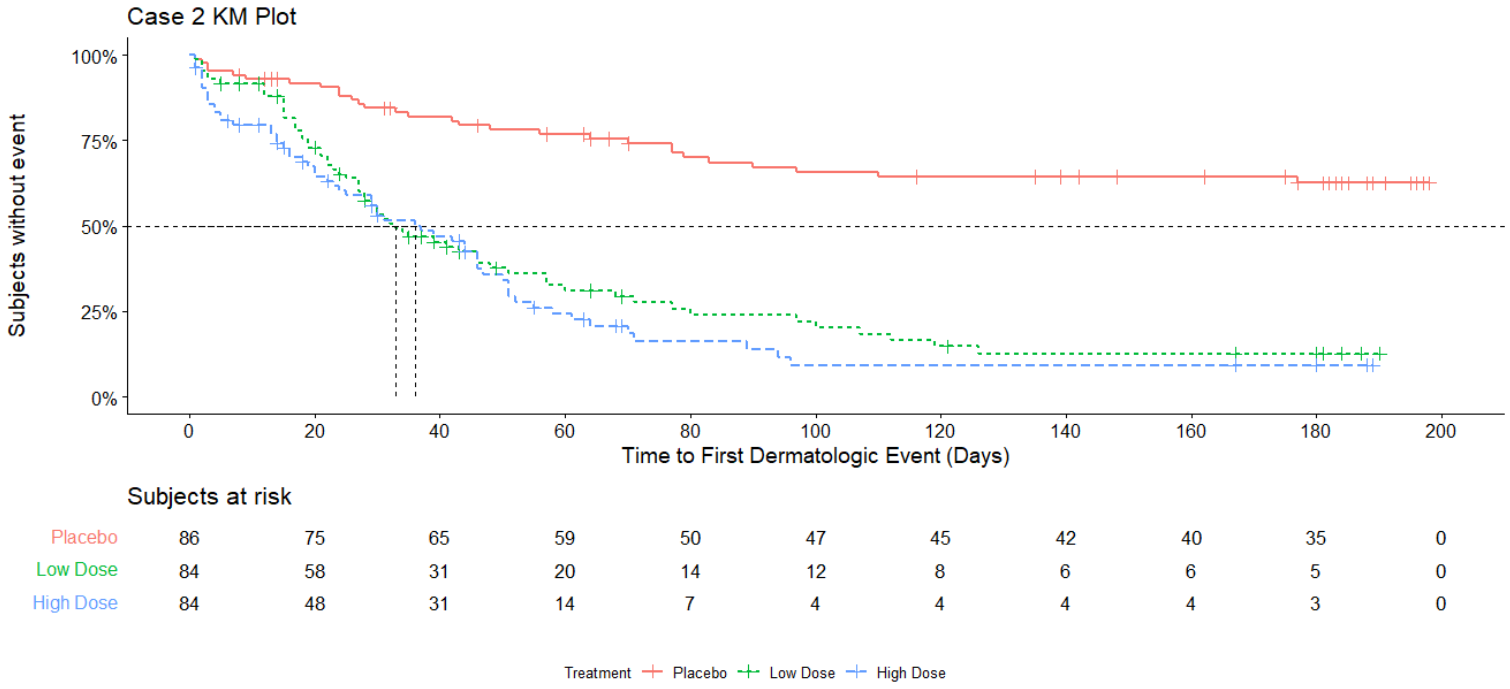
library(survminer) → ggsurvplot()

2022 PharmaSUG Pre-conference Seminar



3.5 KM Plot – Case 2

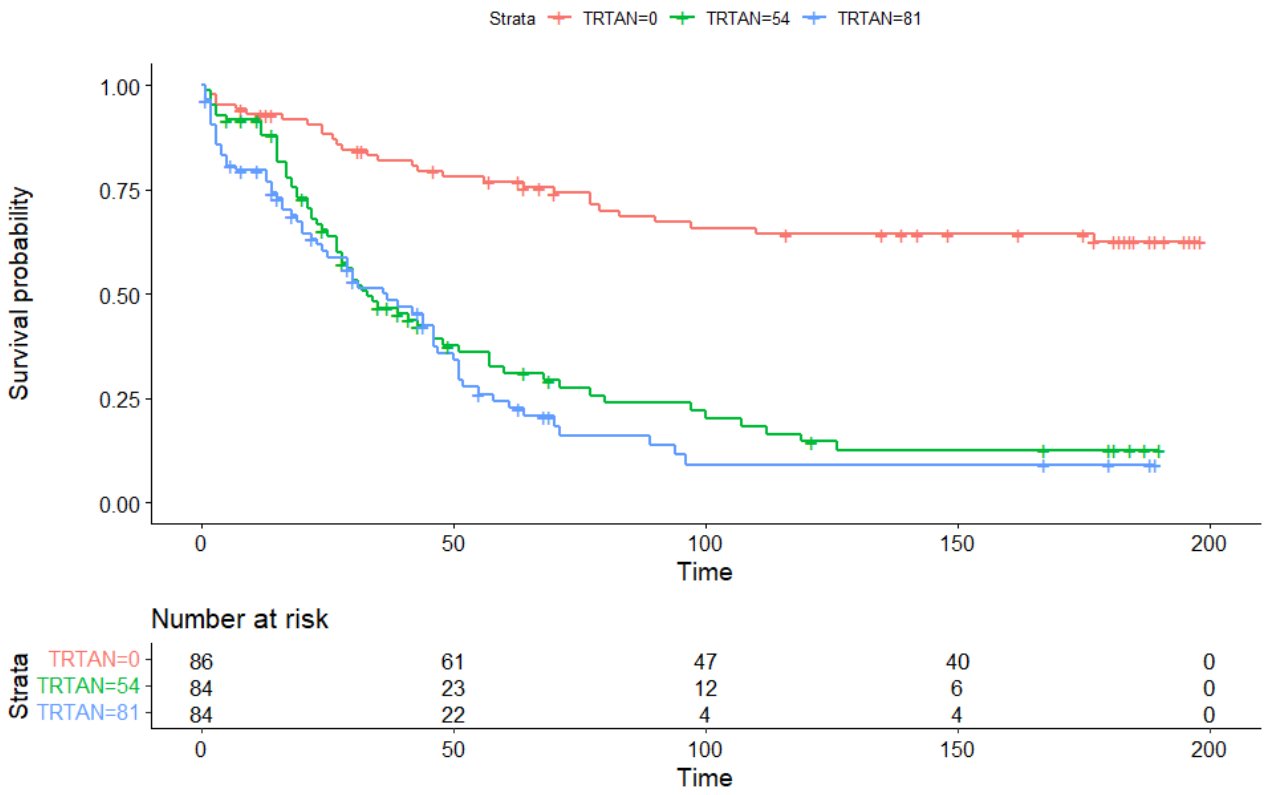
- Data: ADTTE
  - Parameter: Time to First Dermatologic Event (TTDE)
  - Time info: AVAL
  - Event info: CNSR (0=event, 1=censor)
  - Strata info: TRTAN (1=Placebo, 54=Xanomeline Low Dose, 81=Xanomeline High Dose)



3.5 KM Plot – Case 2

- One basic forest plot

```
fit <- survfit(Surv(AVAL, 1-CNSR) ~ TRTAN, data=adtte)

ggsurvplot(fit, risk.table=T)
```



3.5 KM Plot – Case 2

We use ggsurvplot() function to plot survival curves.

```
ggsurvplot(fit,
           data = NULL,
           fun = NULL,
           palette = NULL,
           linetype = 1,
           conf.int = FALSE,
           pval = FALSE,
           surv.median.line = "none",
           risk.table = FALSE,
           tables.height = 0.25,
           ggtheme = theme_survminer(),
           tables.theme = ggtheme,
           ...)
```

2022 PharmaSUG Pre-conference Seminar

It returns an object of class ggsurvplot which is list containing the following components:

- plot: the survival plot (ggplot object)
- table: the number of subjects at risk table per time (ggplot object).
- cumevents: the cumulative number of events table (ggplot object).
- ncensor.plot: the number of censoring (ggplot object).
- data.survplot: the data used to plot the survival curves (data.frame).
- data.survtable: the data used to plot the tables under the main survival curves (data.frame).

We can also use ggplot2 syntax to modify the plot component of ggsurvplot

3.5 KM Plot – Case 2

- Other modifications 1 – Censor and Curve line

You can force the censor shape to a certain shape use *censor.shape* argument:

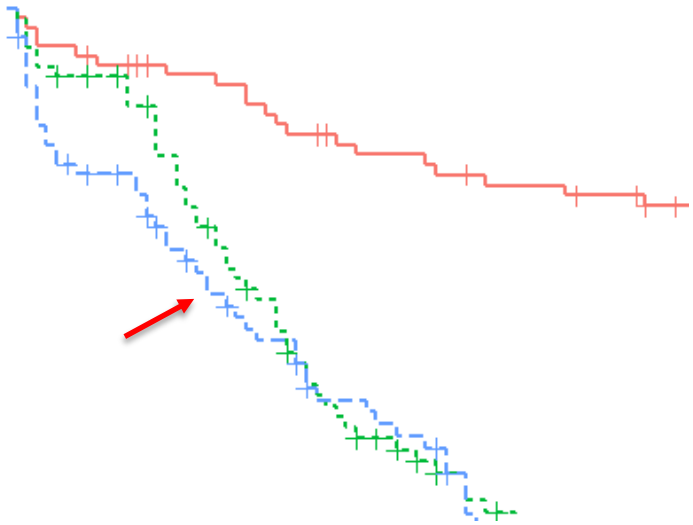
c(3) is "+", c(124) is "|", or change censor shape by strata (i.e. groups)

censor.size argument defines the point size of censors.

linetype argument allows changing linetypes:

i) "strata"; ii) a numeric vector - c(1, 2); iii) a character vector - c("solid", "dashed").

```
ggsurvplot(…,
           censor.shape = c(3),
           censor.size = 2.5,
           linetype= "strata",
           …)
```



2022 PharmaSUG Pre-conference Seminar

3.5 KM Plot – Case 2

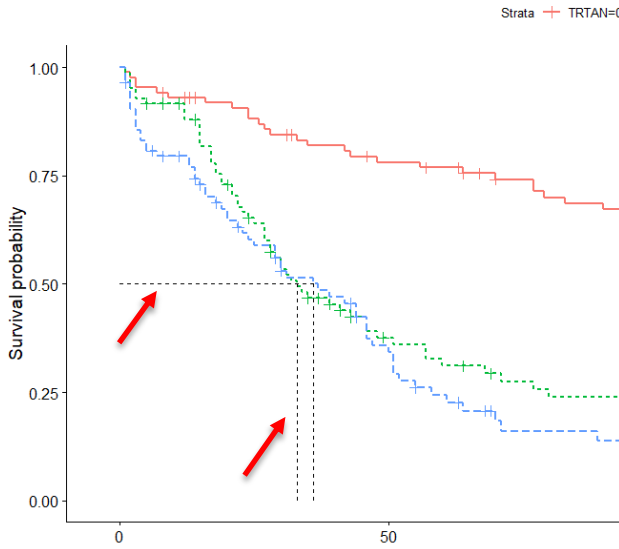
- Other modifications 2 – surv.median.line

surv.median.line argument can draw a horizontal/vertical line at median survival.

Allowed values include one of c("none", "hv", "h", "v"). v: vertical, h:horizontal.

We can also use ggplot2 geom_hline() to add horizontal/vertical line at median survival.

```
P <- ggsurvplot(…,
                surv.median.line = "hv",
                …)


p$plot <- p$plot+
          geom_hline(yintercept=0.5,linetype="dashed")
```

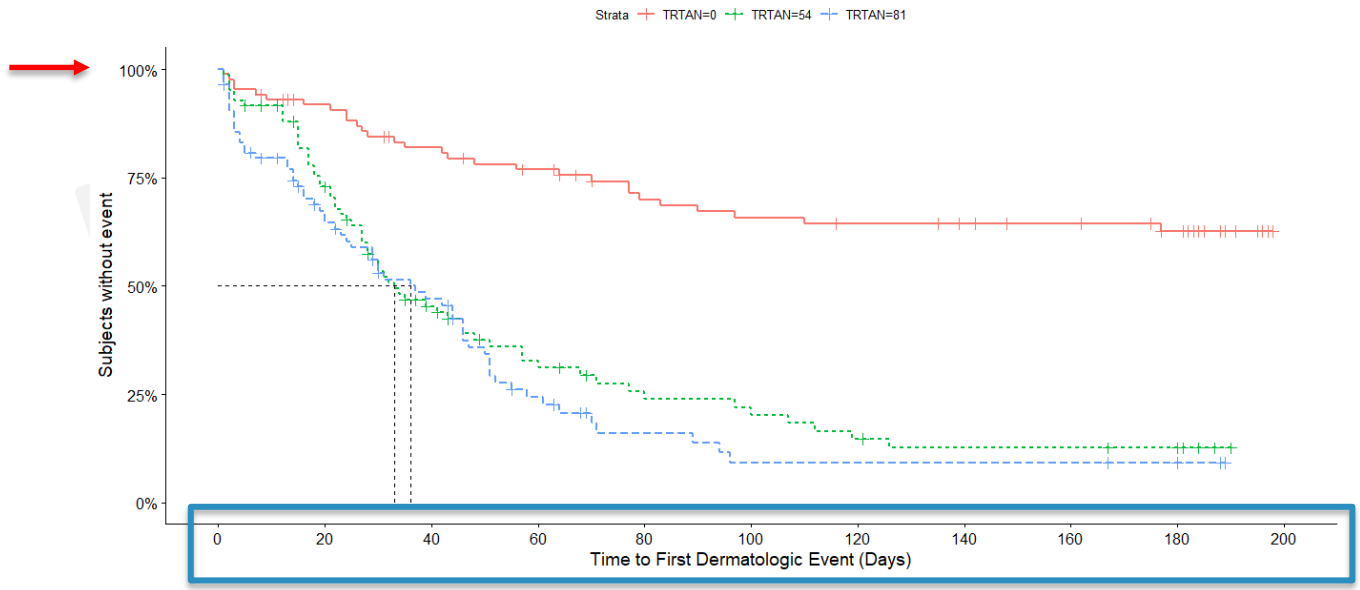


2022 PharmaSUG Pre-conference Seminar

3.5 KM Plot – Case 2

- Other modifications 3 – x and y axis

```
ggsurvplot(…,

           xlim = c(0, 200),
           break.time.by = 20,
           surv.scale = "percent",

           xlab = "Time to First Dermatologic Event (Days)",
           ylab = "Subjects without event",
```



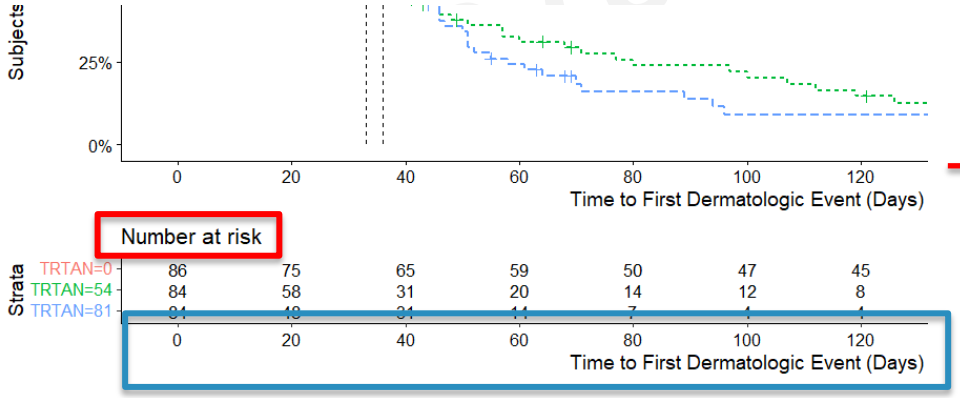
PharmaSUG 2022 Conference Seminar

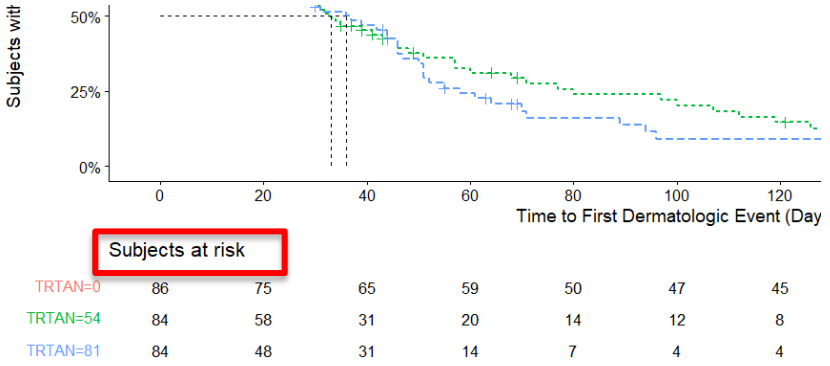
3.5 KM Plot – Case 2

- Other modifications 4 – risk.table

```
ggsurvplot(…,

           risk.table = TRUE,

           risk.table.title = "Subjects at risk",
           tables.theme = theme_cleantable(),

           …)
```

2022 PharmaSUG conference Seminar

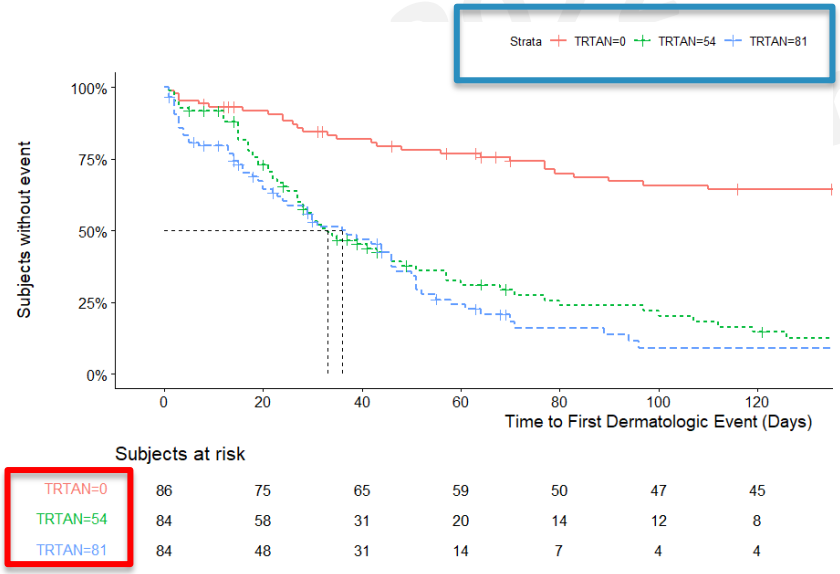


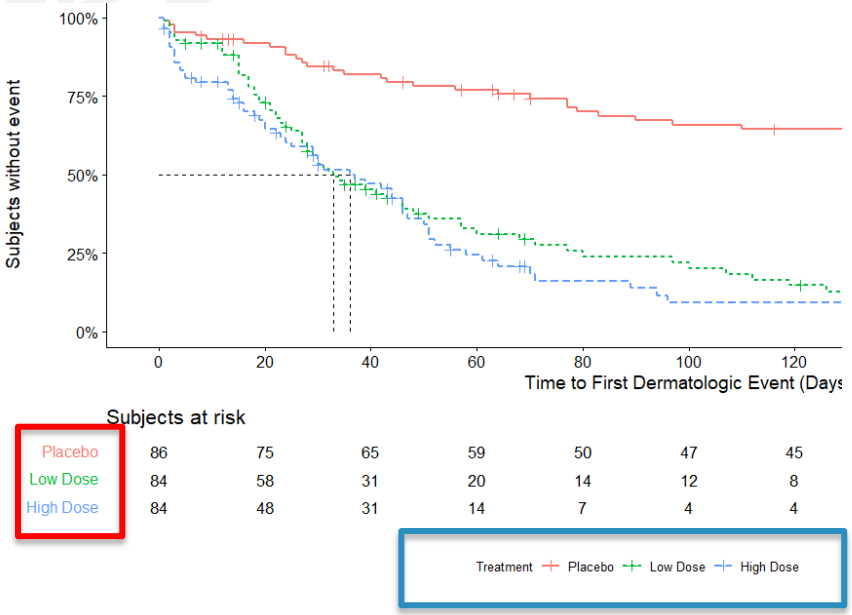


3.5 KM Plot – Case 2

- Other modifications 5 – legend

```
ggsurvplot(…,

           legend.labs = c("Placebo", "Low Dose", "High Dose"),
           legend = 'bottom',
           legend.title = 'Treatment',

           …)
```





3. KM Plot – Summary

- In order to create KM plot, two new packages are need.

```
library(survival)
library(survminer)
```

- Survival Object, or Surv Object, is an object combined the variables time and event together

```
?Surv
```

- survfit() function is used to compute the Kaplan-Meier curve from a formula with Surv Object.

```
?survfit.object
```

- ggsurvplot() function is used to plot survival curves.

```
?ggsurvplot_arguments
```

```
fit <- survfit(Surv(AVAL, 1-CNSR) ~ TRTAN, data=adtte)

ggsurvplot(fit, risk.table=T)
```

2022 PharmaSUG Pre-conference Seminar

Forest Plot

Aug 18, 2022

4. Forest Plot – Case 3

4.1 Meet new package

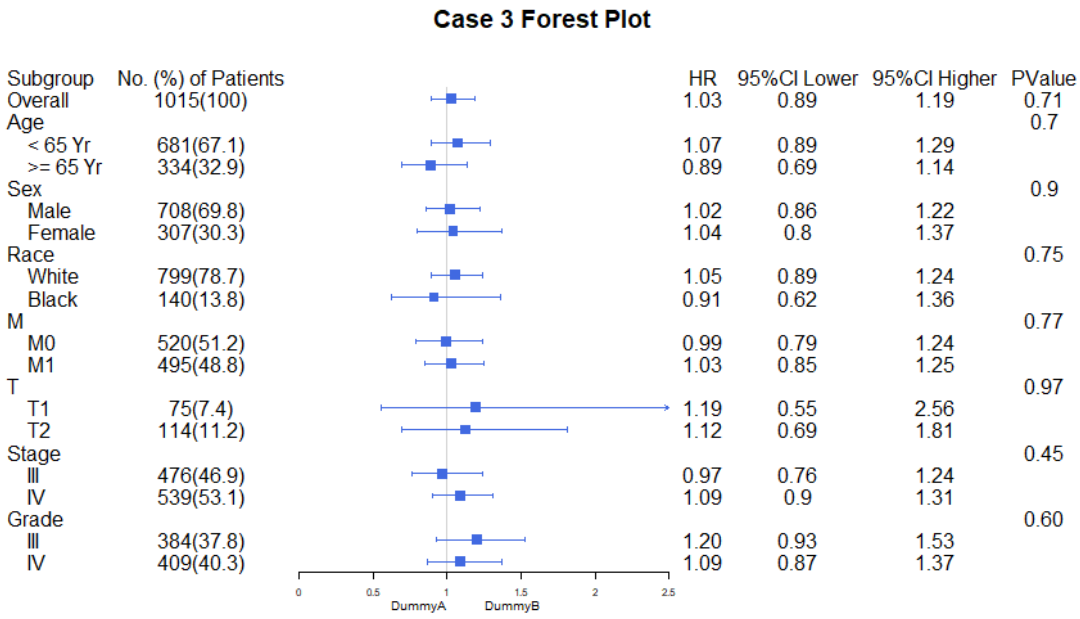
```
library(forestplot)
```

4.2 Prepare data for forest plot

4.3 Forest plot – Case 3

- One basic forest plot
- Other modifications

2022 PharmaSUG Pre-conference Seminar



4.1 Meet New Package

- Forest plots date back to 1970s and are most frequently seen in meta-analysis.
- A forest plot that allows for multiple confidence intervals per row, custom fonts for each text element, custom confidence intervals, text mixed with expressions, and more.
- The `forestplot` package is a more general version of the original '`rmeta`' package's forestplot function and relies heavily on the '`grid`' package.

```
library(forestplot)
```

2022 PharmaSUG
Pre-conference Seminar

4.2 Prepare Data for Forest Plot

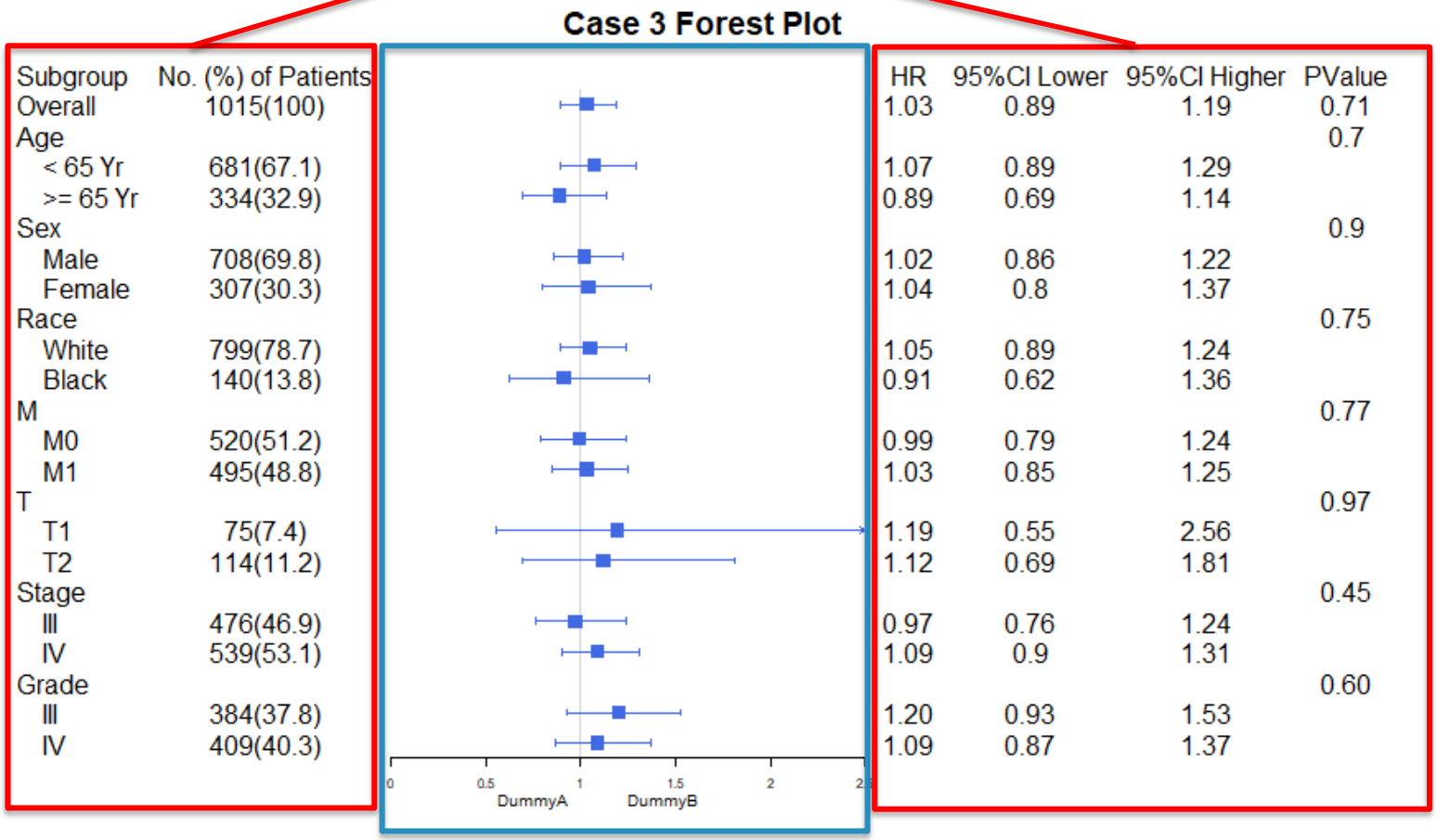
labeltext

row_text	count	hr_text	lower_text	upper_text	pvalue_text	hr	lower	upper
Subgroup	No. (%) of Patients	HR	95%CI Lower	95%CI Higher	PValue	NA	NA	NA
Overall	1015(100)	1.03	0.89	1.19	0.71	1.03	0.89	1.19
Age					0.7	NA	NA	NA
< 65 Yr	681(67.1)	1.07	0.89	1.29		1.07	0.89	1.29
>= 65 Yr	334(32.9)	0.89	0.69	1.14		0.89	0.69	1.14
Sex					0.9	NA	NA	NA
Male	708(69.8)	1.02	0.86	1.22		1.02	0.86	1.22
Female	307(30.3)	1.04	0.8	1.37		1.04	0.80	1.37
Race					0.75	NA	NA	NA
White	799(78.7)	1.05	0.89	1.24		1.05	0.89	1.24
Black	140(13.8)	0.91	0.62	1.36		0.91	0.62	1.36
M					0.77	NA	NA	NA
M0	520(51.2)	0.99	0.79	1.24		0.99	0.79	1.24
M1	495(48.8)	1.03	0.85	1.25		1.03	0.85	1.25
T					0.97	NA	NA	NA
T1	75(7.4)	1.19	0.55	2.56		1.19	0.55	2.56
T2	114(11.2)	1.12	0.69	1.81		1.12	0.69	1.81
Stage					0.45	NA	NA	NA
III	476(46.9)	0.97	0.76	1.24		0.97	0.76	1.24
IV	539(53.1)	1.09	0.9	1.31		1.09	0.90	1.31
Grade					0.60	NA	NA	NA
III	384(37.8)	1.20	0.93	1.53		1.20	0.93	1.53
IV	409(40.3)	1.09	0.87	1.37		1.09	0.87	1.37

confidence interval graph

4.2 Prepare Data for Forest Plot

Table of text



confidence interval graph

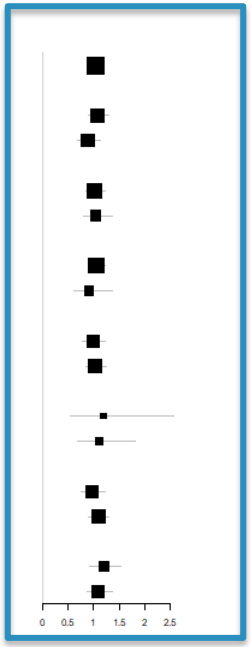
4.3 Forest Plot – Case 3

- One basic forest plot

```
forestplot(data      = df_for_plot,
           labeltext = labeltext,
           mean      = hr,
           lower     = lower,
           upper     = upper)
```

2022 PharmaSUG Pre-conf Seminar

Subgroup	No. (%) of Patients	HR	95%CI Lower	95%CI Higher	PValue
Overall	1015(100)	1.03	0.89	1.19	0.71
Age					0.7
< 65 Yr	681(67.1)	1.07	0.89	1.29	
>= 65 Yr	334(32.9)	0.89	0.69	1.14	
Sex					0.9
Male	708(69.8)	1.02	0.86	1.22	
Female	307(30.3)	1.04	0.8	1.37	
Race					0.75
White	799(78.7)	1.05	0.89	1.24	
Black	140(13.8)	0.91	0.62	1.36	
M					0.77
M0	520(51.2)	0.99	0.79	1.24	
M1	495(48.8)	1.03	0.85	1.25	
T					0.97
T1	75(7.4)	1.19	0.55	2.56	
T2	114(11.2)	1.12	0.69	1.81	
Stage					0.45
III	476(46.9)	0.97	0.76	1.24	
IV	539(53.1)	1.09	0.9	1.31	
Grade					0.60
III	384(37.8)	1.20	0.93	1.53	
IV	409(40.3)	1.09	0.87	1.37	



4.3 Forest Plot – Case 3

- Other Modifications 1 - The position of confidence interval graph

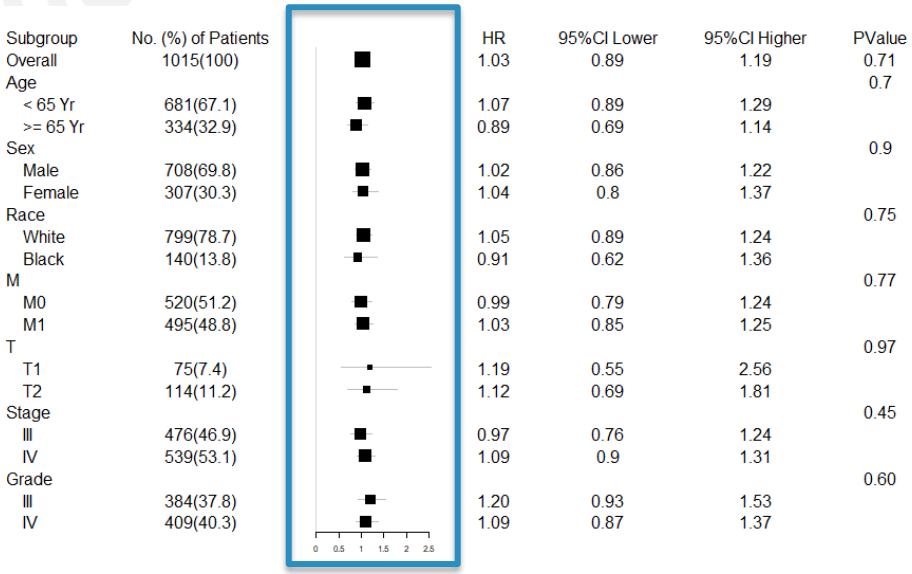
  By default, the confidence interval graph is created at right side.

  We can use *graph.pos* argument to change position of graph element within the table of text.

2022 PharmaSUG Pre-conference Seminar

```
forestplot(…,
           graph.pos='left',
           …)

or
forestplot(…,
           graph.pos=3,
           …)
```

The position can be from 1 to (ncol(labeltext) + 1).



Subgroup	No. (%) of Patients	HR	95%CI Lower	95%CI Higher	PValue
Overall	1015(100)	1.03	0.89	1.19	0.71
Age					0.7
< 65 Yr	681(67.1)	1.07	0.89	1.29	
>= 65 Yr	334(32.9)	0.89	0.69	1.14	
Sex					0.9
Male	708(69.8)	1.02	0.86	1.22	
Female	307(30.3)	1.04	0.8	1.37	
Race					0.75
White	799(78.7)	1.05	0.89	1.24	
Black	140(13.8)	0.91	0.62	1.36	
M					0.77
M0	520(51.2)	0.99	0.79	1.24	
M1	495(48.8)	1.03	0.85	1.25	
T					0.97
T1	75(7.4)	1.19	0.55	2.56	
T2	114(11.2)	1.12	0.69	1.81	
Stage					0.45
III	476(46.9)	0.97	0.76	1.24	
IV	539(53.1)	1.09	0.9	1.31	
Grade					0.60
III	384(37.8)	1.20	0.93	1.53	
IV	409(40.3)	1.09	0.87	1.37	

4.3 Forest Plot – Case 3

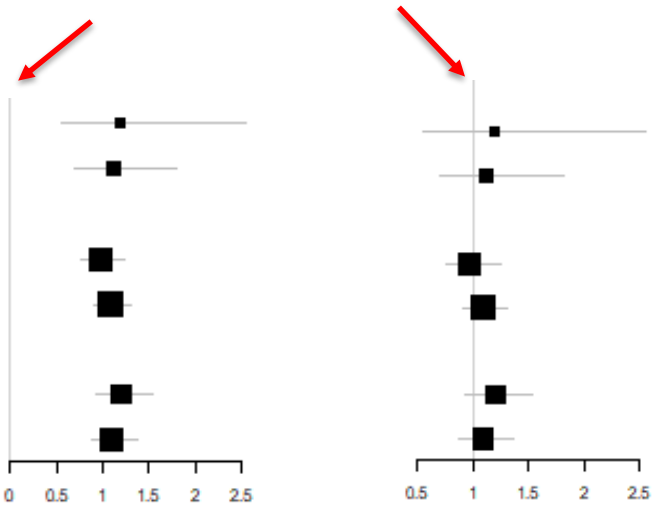
- Other Modifications 2 – Zero Line

zero argument defines x-axis coordinate for zero line.

If you provide a vector of length 2 it will print a rectangle instead of just a line.

If you provide NA the line is suppressed.

```
forestplot(…,
           zero=1,
           …)
```



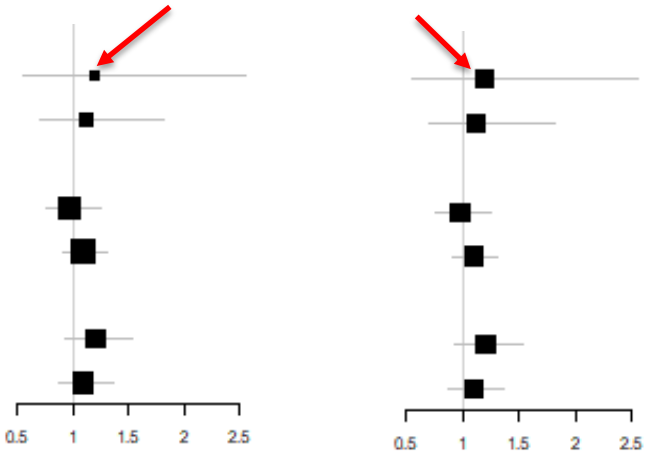
2022 PharmaSUG Pre-conference Seminar

4.3 Forest Plot – Case 3

- Other Modifications 3 – boxsize

You can force the box size to a certain size through the *boxsize* argument.

```
forestplot(…,
           boxsize=0.4,
           …)
```



2022 PharmaSUG Pre-conference Seminar

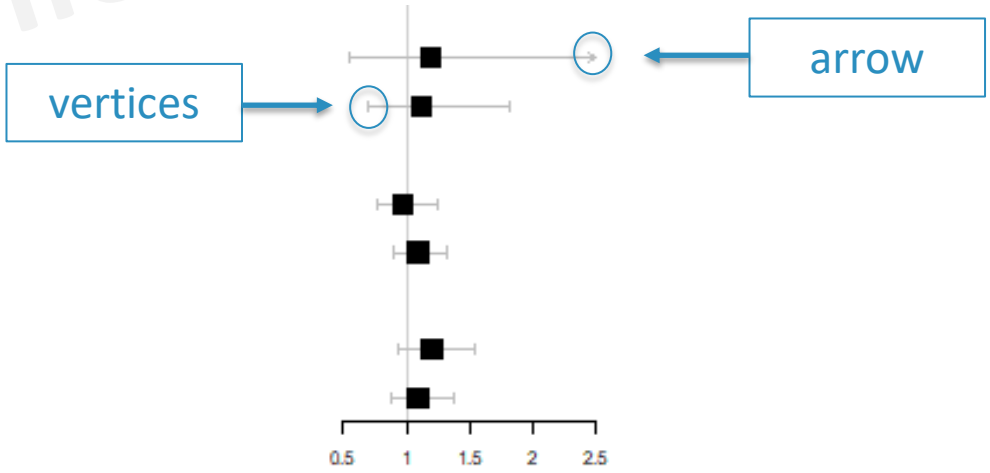
4.3 Forest Plot – Case 3

- Other Modifications 4 – arrow and vertices

```
forestplot(…,
           clip=c(0,2.5),
           ci.vertices=TRUE,
           …)
```

clip: Lower and upper limits for clipping confidence intervals to arrows

ci.vertices: Set this to TRUE if you want the ends of the confidence intervals to be shaped as a T.



2022 PharmaSUG Pre-conference Seminar

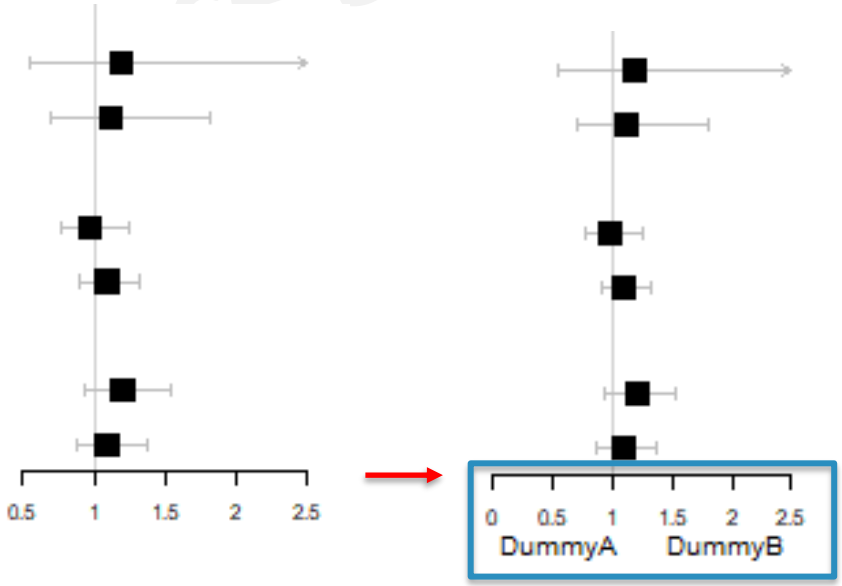
4.3 Forest Plot – Case 3

- Other Modifications 5 – x axis

```
forestplot(…,
           xticks= c(0,0.5,1,1.5,2.0,2.5),
           xlab="DummyA                DummyB ",
           …)
```

x axis ticks

x axis label

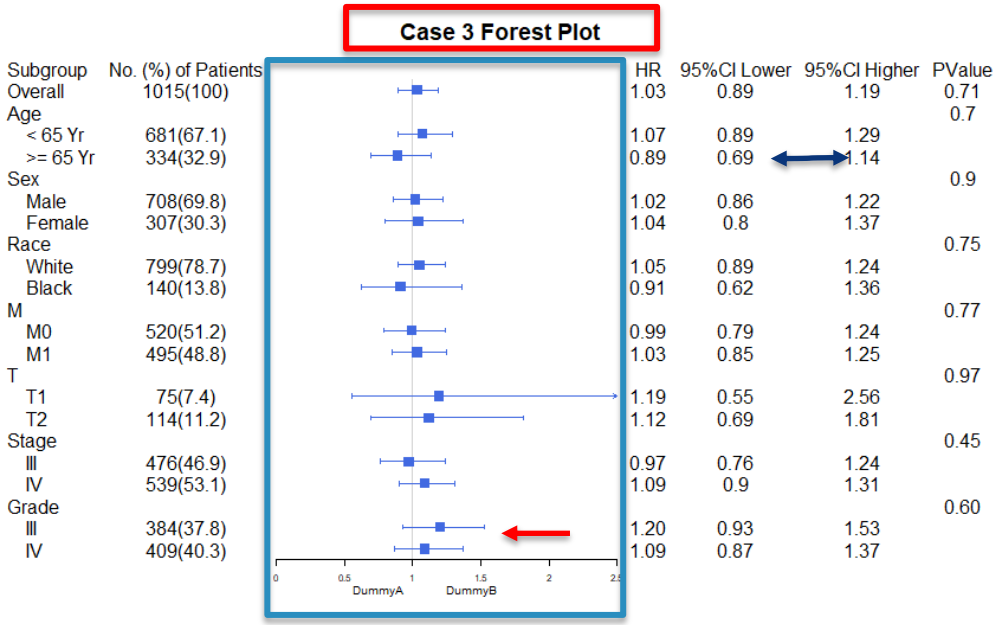


2022 PharmaSUG Pre-conference Seminar

4.3 Forest Plot – Case 3

- Other Modifications 6 – Graphical parameter settings (theme): color, graphwidth, colgap

```
forestplot(…,
           col=fpColors(box="royalblue",lines='royalblue')
           graph.width=unit(3,'inches'),
           colgap=unit(3,'mm'),
           title="Case 3 Forest Plot")
           …)
```



2022 PharmaSUG Pre-conference Seminar

4. Forest Plot - Summary

- `forestplot` package and `forestplot` function provide solutions to create a standard forest plot, with a few interesting features to make customized graph for more than just meta-analyses:

```
library(forestplot)
?forestplot
vignette("forestplot",  package="forestplot")


forestplot(data      =  ,
           labeltext =  ,
           mean      =  ,
           lower     =  ,
           upper     =   )
```

2022 PharmaSUG Pre-conference Seminar

5. Summary

- In this session, we've continued to more solutions to create customized graphs:
  - We can use ggplot2 and other packages and build basic plots with very little effort
  - Then, we can make the plots increasingly more sophisticated by simply adding additional layers/useful arguments to out plots.

- You now have the ability to create complex graphs, KM plot, forest plot.

- If you want to learn additional plots, remember…
  - a new plot is only a very basic geom/argument call away; start with the basic call, and build your layers from there,
  - be patient when exploring new packages, new functions and new arguments, and always use help documents smartly

2022 PharmaSUG Pre-Conference Seminar

Thanks!

2022 PharmaSUG
Pre-conference Seminar