Module 4
Graphics in R

Module 4: Graphics in R

No.	Topics
1	Creating Graphics Parameters Basic Graphics
2	Grammar of Graphics: ggplot()
3	Lattice Graphics
4	Save Graphics

1. Creating Graphs

- The oldest graphics systems in R is known as ‘base graphic’.
- Packages: grid; advance package: lattice, ggplot
- When plot is executed, a new graphic window pops up. If another plot is executed, the current graphic is replaced with a new graphic.
- Can be used to examine:
  - Marginal distributions
  - Relationships between variables
  - Summary of very large dat

Creating Graphs: Parameters cont…

- Some parameters:
  - ylim: set the range/scale of y axis; xlim: set the range of x axis
  - ylab: set the legend title of y axis; xlab: set tht legend title of x axis
  - main: set the main title of graphs
  - sub: set the sub title, with smaller font
  - Line type with lty = (see Fig. 1) and lwd to set the width (default = 1)
  - pch: set the symbol
  - type: set the line type for functions plot(), points() and lines()
  - mfrow: number of plots per row, column (plots are filled row-wise)
  - mfcol: number of plots per row, column (plots are filled column-wise)
  - oma: the outer margin size (default is 0)
  - las: orientation of the axis labels on the plot
  - beside=TRUE: values in each column to be plotted side-by-side
  - legend=TRUE/FALSE: to add legend in the top right

Figure 1: Line Type

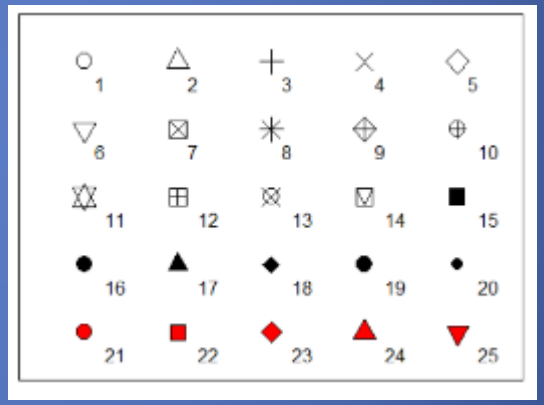


Figure 2: Symbol

Creating Graphs: Parameters cont…

- col: set the colours
  - Options for colors (see table)

option	description
col	Default plotting color. Some functions (e.g. lines) accept a vector of values that are recycled.
col.axis	color for axis annotation
col.lab	color for x and y labels
col.main	color for titles
col.sub	color for subtitles
fg	plot foreground color (axes, boxes - also sets col= to same)
bg	plot background color

- To check for colours, use this command:
  - colours()
  - palette() #standard colours
  - rainbow(n) #it gives you the colour hex code
  - hsv(): hue, saturation and value
  - Hex RGB: red = “#FF0000”
  - Must choice the colours carefully
    - Visible to everyone

Creating Graphs: Parameters cont…

– cex: to control text and symbol size

option	description
cex	number indicating the amount by which plotting text and symbols should be scaled relative to the default. 1=default, 1.5 is 50% larger, 0.5 is 50% smaller, etc.
cex.axis	magnification of axis annotation relative to cex
cex.lab	magnification of x and y labels relative to cex
cex.main	magnification of titles relative to cex
cex.sub	magnification of subtitles relative to cex

– Control the margin and graph size:

option	description
mar	numerical vector indicating margin size c(bottom, left, top, right) in lines. default = c(5, 4, 4, 2) + 0.1
mai	numerical vector indicating margin size c(bottom, left, top, right) in inches
pin	plot dimensions (width, height) in inches

Creating Graphs: Parameters cont…

– Font:

option	description
font	Integer specifying font to use for text. 1=plain, 2=bold, 3=italic, 4=bold italic, 5=symbol
font.axis	font for axis annotation
font.lab	font for x and y labels
font.main	font for titles
font.sub	font for subtitles
ps	font point size (roughly 1/72 inch) text size=ps*cex
family	font family for drawing text. Standard values are "serif", "sans", "mono", "symbol". Mapping is device dependent.

– Sample command:

```
# Type family examples - creating new mappings
plot(1:10,1:10,type="n")
windowsFonts(
  A=windowsFont("Arial Black"),
  B=windowsFont("Bookman Old Style"),
  C=windowsFont("Comic Sans MS"),
  D=windowsFont("Symbol")
)
text(3,3,"Hello World Default")
text(4,4,family="A","Hello World from Arial Black")
text(5,5,family="B","Hello World from Bookman Old Style")
text(6,6,family="C","Hello World from Comic Sans MS")
text(7,7,family="D", "H
```

Creating Graphs: Parameters cont…

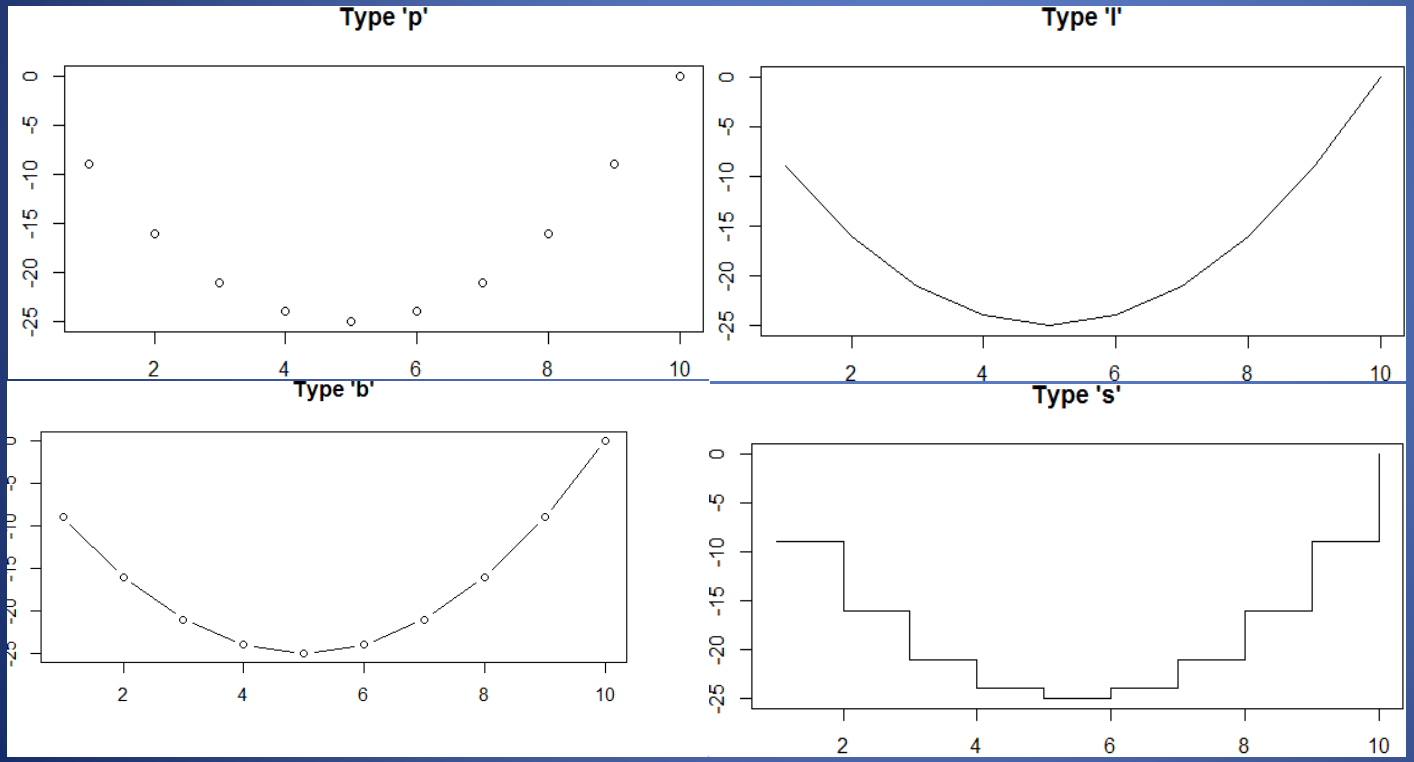
- Control the layout with par()
  - attach(mtcars)
    - When a data set is attach, we can access to the variables directly with its name. For e.g.: mpg instead of mtcars$mpg
  - par(mfrow=c(2,2))
    - let the display layout like a matrix with row = 2 and column = 2
  - plot(wt,mpg, main="Scatterplot of wt vs. mpg")
  - plot(wt,disp, main="Scatterplot of wt vs disp")

Creating Graphs: Parameters cont…

- More about layout:
  - eglayout<-matrix(1:4, 2, 2)
  - layout(eglayout)
  - layout.show(4)
  - layout(matrix(1:6, 3, 2))
  - layout.show(4)
  - layout(matrix(1:6, 2, 3))
  - layout.show(6)
  - layout(matrix(c(1:3, 3), 2, 2))
  - layout.show(3)
  - layout(mat, widths=c(1, 3), heights=c(3, 1))
    - We can set the height and width
  - layout.show(4)

Creating Graphs: Parameters cont…

– Type – p, l, b, o, h, s:



Creating Graphs: Basic Graphics

- plot() is the main graphing function
  - Can automatically produce simple plot for vector, data frames or functions
  - x<-seq(1,10)
  - y<-x^2-10*x
  - plot(x,y)
  - plot(women$height)
  - plot(women$height, col="red1") #put in colours
  - plot(women$height, women$weight, xlab="height", ylab="weight", col="tomato1") #label the axis
  - plot(women$height, women$weight, xlab="height", ylab="weight", main="Average Height & Weights for American WOmen",col="tomato1") #put in main title
  - plot(women$height, women$weight, xlab="height", ylab="weight", main="Average Height & Weights for American WOmen",pch = 2,col="tomato1") #change the plot symbol through pch

```
> methods("plot")
 [1] plot.acf*              plot.data.frame*     plot.decomposed.ts*
 [4] plot.default           plot.dendrogram*     plot.density*
 [7] plot.ecdf              plot.factor*         plot.formula*
[10] plot.function          plot.hclust*         plot.histogram*
[13] plot.HoltWinters*      plot.isoreg*         plot.lm*
[16] plot.medpolish*        plot.mlm*            plot.ppr*
[19] plot.prcomp*           plot.princomp*       plot.profile.nls*
[22] plot.spec*             plot.stepfun         plot.stl*
[25] plot.table*            plot.ts              plot.tskernel*
[28] plot.TukeyHSD*

   Non-visible functions are asterisked
```

Creating Graphs: Basic Graphics cont…

- Add a line with abline() function
  - x<-c(1,2,3)
  - y<-c(1,3,8)
  - plot(x,y)
  - lmout<-lm(y~x) #lm contains the slope and intercept of the fitted line
  - abline(lmout) # the graph will show the dots and line together
- To add more lines by using lines()
  - lines(c(1.5,2.5),c(3,3))
  - #connect the dots but do not want the dots:
  - plot(x,y,type="l")
- Use the density function with plot() and lines()
  - summary(women) #let us check the data
  - #observe the minimum and maximum values
  - d1=density(women$height, from=0, to=200)
  - d2=density(women$weight, from=0, to=200)
  - plot(d1,main="",xlab="")
  - lines(d2)
  - #the graphs is meaningful if we have another men data sets for comparison
  - text(49,0.06,"height") # to label the line
  - Trick to obtain the coordinate:
    - Use locator()
    - locator(1) #tell R you will click in one place in the graph
    - #The coordinate is generated and now you can put in the label with text()

Creating Graphs: Basic Graphics cont…

- Plot a histogram by hist()
  - The parameter breaks is key
    - Specifies the no of categories to plot or
    - Specifies the breakpoints for each category
  - hist(VADeaths, ylim=c(0,10), ylab="Deaths per 1000", main="Death rates in Virgina", col=c("red1","blue1","brown1","seagreen1"))
- Plot a bar chart
  - barplot(VADeaths, beside=TRUE, legend=TRUE, ylim=c(0,90), ylab="Deaths per 1000", main="Death rates in Virginia", col=rainbow(6))
- Dot chart:
  - dotchart(VADeaths, xlim=c(0,75),xlab="Deaths per 1000",main="Death rates in Virginia")
- Pie charts
  - groupsizes<-c(18,30,32,10,10)
  - labels<-c("A","B","C","D", "E")
  - pie(groupsizes,labels, col=rainbow(5))

Creating Graphs: Basic Graphics cont…

- Box plots
  - boxplot(Sepal.Length~Species, data=iris, ylab="Sepal length(cm)", main="Iris measurements",boxwex=0.5)

2. Grammar of Graphics: ggplot()

- ggplot2(): the grammar of graphics, i.e. gg which is used to create graph in a creative way
- Applicable to data frame only
  - To initialise ggplot():
    - ggplot(dataframe, aes(x, y,))
    - ggplot(dataframe)
    - ggplot()
  - Fundamental parameters of ggplot():
    - aes: aesthetics, to define how your data are presented visually. A variable may control where points appear, the colour or shape of a point, the height of a bar and so on.
    - geom – these are the geometric objects. Do you need bars, points, lines?
    - Statistics – these are the functions like linear regression
    - Scales – To modify axes and colors
      - **scale_y_continuous()** Set name, breaks, labels, limits of y-axis
      - **scale_x_log10()** log transform the x-axis
      - **scale_colour_manual()** Specify colors for geoms
      - **scale_fill_discrete()** Specify colors for geoms
    - facets – these are the groups in your data. Faceting by gender would cause the graph to repeat for the two genders.
    - theme – to have more precise control

Grammar of Graphics: ggplot() cont…

- Try the following commands:
  - install.packages("ggplot2")
  - library(ggplot2)
  - Iris<-iris
  - ggplot(data=Iris,aes(x=Sepal.Width, y=Sepal.Length)) + geom_point() + theme_minimal()
    - check the relation between Sepal length and Speal width: theme_minimal() is used to adjust the appearance of the plots
  - ggplot(data=Iris,aes(x=Sepal.Width, y=Sepal.Length,color=Species)) + geom_point() + theme_minimal()
    - marked the Species by different colours
  - ggplot(data=Iris,aes(x=Sepal.Width, y=Sepal.Length,color=Species)) + geom_point() +geom_smooth() + theme_minimal()
    - Add a trend line to visualize the general trend through geom_smooth()
  - options(repr.plot.width = 10, repr.plot.height = 6)# to adjust size of plots
  - ggplot(data=Iris,aes(x=Sepal.Width, y=Sepal.Length,color=Species)) + geom_point() + geom_smooth(se=FALSE) +facet_wrap(~Species) +theme_minimal()
    - Off the grey trend using se=FALSE, and split the diplay of graph according to species

Grammar of Graphics: ggplot() cont…

- Box plot:
  - options(repr.plot.width = 5, repr.plot.height = 4)
  - ggplot(data=Iris,aes(x=Species, y=Petal.Length,color=Species)) + geom_boxplot() +theme_minimal()+ theme(legend.position="none")
- Histogram: best way to show frequency
  - ggplot(data=Iris,aes(x=Sepal.Length)) + geom_histogram() +theme_minimal()
  - ggplot(data=Iris,aes(x=Sepal.Length)) + geom_histogram(binwidth=0.3) +theme_minimal()
    - Set the bin width
  - ggplot(data=Iris,aes(x=Sepal.Length,fill=Species)) + geom_histogram() +theme_minimal()
    - Put in colours based on Species
  - ggplot(data=Iris,aes(x=Sepal.Length,fill=Species)) + geom_histogram() +theme_minimal() +facet_wrap(~Species)
    - Split the histogram according to species
- Density curve:
  - ggplot(data=Iris,aes(x=Sepal.Length,color=Species)) + geom_density() +theme_minimal()
  - ggplot(data=Iris,aes(x=Petal.Width,y=Petal.Length,color=Species)) +geom_density2d()+ theme_minimal()
- Add in statistic:
  - ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width, color=Species)) + geom_point() + stat_smooth(method="lm")
- With scale:
  - ggplot(data=iris, aes(x=Sepal.Length, y=Sepal.Width, color=Species)) + geom_point(size=3) + scale_colour_manual(values=c("red","blue","yellow"))

Grammar of Graphics: ggplot() cont…

- Large data set is a challenge to visualise them
- Type the following commands:
  - pretest2 <- round(rnorm(n=5000, mean=80, sd=5))
  - posttest2 <- round(pretest2 + rnorm(n=5000, mean=3, sd=3))
  - pretest2 [pretest2 > 100] <- 100
  - posttest2[posttest2 > 100] <- 100
  - temp <- data.frame(pretest2,posttest2)
    - Create a data set with 5000 points
  - ggplot(temp, aes(pretest2, posttest2), size=2, position = position_jitter(x = 2,y = 2)) + geom_jitter(colour=alpha("black",0.15))
    - Colouring with transparency call alpha under geom_jitter
  - ggplot(temp, aes(x=pretest2, y=posttest2)) + geom_point(size=1) + geom_density2d()
    - Clearer view of points

4. Save Graphs

- To save a graph with basic save function in R:
  - png(file="bowling.png", width=400, height=400)
  - x <- 1:10 >
  - y <- (x - 5)^2 >
  - plot(y ~ x, type="b", xlab="Frame", ylab="Bowling score")
  - dev.off() #find the file in your working directory

Save Graphs cont…

- Function for saving graphic output (see Table):

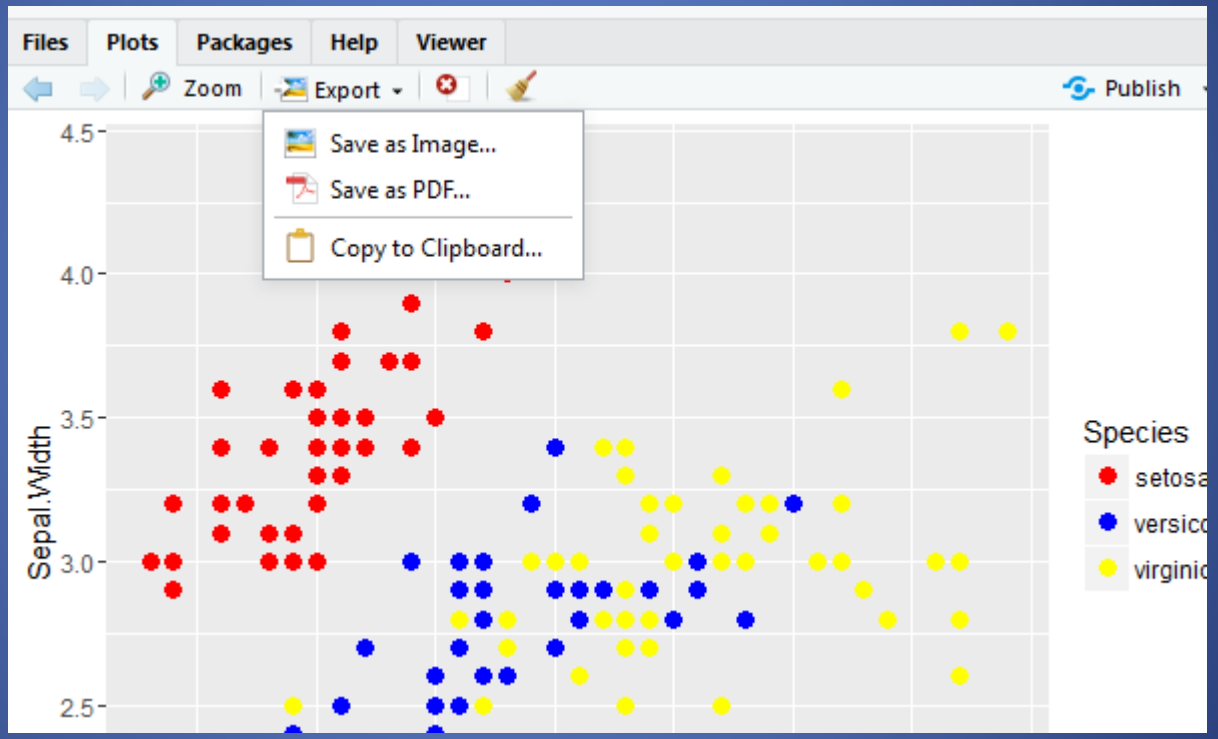
Function	Output
pdf("filename.pdf")	PDF file
win.metafile("filename.wmf")	Windows metafile
png("filename.png")	PBG file
jpeg("filename.jpg")	JPEG file
bmp("filename.bmp")	BMP file
postscript("filename.ps")	PostScript file

Save Graphs cont…

- Default graphic device in R is your computer screen
- Using ggplot2 package:
- my_plot<-ggplot(iris, aes(x=Sepal.Length, y=Sepal.Width, color=Species)) + geom_point() + stat_smooth(method="lm")
- ggsave("my_plot.jpg",my_plot,height=4,width=4, units="in")
  - You can specify the resolution, filename, dimensions etc
  - The above ggsave is save at the working directory (unless specified)

Save Graphs cont…

- Use the feature of Rstudio:
  - Need to set the format of image, detail of size and shape



Exercise 4.1 – 20 minutes

- Create a data frame which contain the data
- Compute the density of Exam1 and Exam2 separately.
- Then, plot the data of Exam1 and Exam2.
- Save the graph.

Exam 1	Exam 2	Quiz
2.0	3.3	4.0
3.3	2.0	3.7
4.3	4.0	4.0
2.3	0.0	3.3
2.3	1.0	3.3
3.3	3.7	4.0