BECOMING 0 TO 1 IN R

SESSION OUTLINE :-

Session 1 (10/3): Introduction to R and RStudio

Pre-requisites for this class

1. Install R - http://www.r-project.org/
2. Install RStudio - http://www.rstudio.com/products/rstudio/download/

Topics Covered

- RStudio IDE features
- Customizing the RStudio Environment
- Creating and Managing a RStudio Project
- Data Analysis Value Chain
- Installing and Loading `packages`

Session 2 (10/10): Graphics and Visualization in R

Topics Covered

- Base plotting
- ggplot2

Session 3 (10/17): Data Structures in R

Topics Covered

- Data Structures: Vectors, Lists, Matrices, Data frames
- Creation
- Inspection
- Adding and Removing
- Indexing

- Type casting and coercion

Session 4 (10/24): Data Munging in R - Part I

Topics Covered

- readxl
- haven
- jsonlite
- tidyr

Session 5 (10/31): Data Munging in R - Part II

Topics Covered

- dplyr

Session 6 (11/7): Functions and `apply` family

Topics Covered

- Function syntax
- Control structures
- apply family of functions

Session 7 (11/14): Reporting and Business App Development

Topics Covered

- Working with RMarkdown
- Shiny

SESSION 2:-

Histogram

```
library(datasets)

hist(mtcars$disp)
hist(mtcars$disp, breaks=100, col="Green")

## density instead of frequency

hist(mtcars$disp, breaks=100, col="Green", freq=FALSE)

## density plot
d <- density(mtcars$mpg) ## saving the density output in a variable
plot(d) # intelligently plots the results

## filled density plot

d <- density(mtcars$mpg)
plot(d, main="Kernel Density of Miles Per Gallon")
polygon(d, col="red", border="blue")
```

2. Scatterplot

```
## population vs income
plot(state.x77[,1], state.x77[,2])

## adding plot title
plot(state.x77[,1], state.x77[,2], main = "Population vs Income")

#### adding x and y labels - xlab and ylab
plot(state.x77[,1], state.x77[,2], xlab="Population", ylab="Income")

#### adding color - color number
plot(state.x77[,1], state.x77[,2], xlab="Population", ylab="Income", col=2)

#### adding color - with name
plot(state.x77[,1], state.x77[,2], xlab="Population", ylab="Income", col="blue")

#### pch

## changing type of point using pch
plot(state.x77[,1], state.x77[,2], xlab="Population", ylab="Income", col=3, pch=20)

#### cex

## controlling size of symbols using cex

### cex = 0.8
plot(state.x77[,1], state.x77[,2], xlab="Population", ylab="Income", col=3, pch=20,
cex = 0.8)

### cex = 1.8
```

```
plot(state.x77[,1], state.x77[,2], xlab="Population", ylab="Income", col=3, pch=20,
cex = 1.8)
```

3. Line graphs

```
## line plots
plot(state.x77[,1], state.x77[,2], xlab="Population", ylab="Income", col=3, type="l")

## points and lines
plot(state.x77[,1], state.x77[,2], xlab="Population", ylab="Income", col=3, type="b")

## line type
plot(state.x77[,1], state.x77[,2], xlab="Population", ylab="Income", col=3, type="b",
lty=2)

## different line type
plot(state.x77[,1], state.x77[,2], xlab="Population", ylab="Income", col=3, type="b",
lty=4)

## line width
plot(state.x77[,1], state.x77[,2], xlab="Population", ylab="Income", col=3, type="b",
lty=4, lwd=2)

## abline
plot(state.x77[,1], state.x77[,2], xlab="Population", ylab="Income", col=3)
abline(h=4000,col="red")
abline(v=7000,col="blue", lty=3, lwd=4)

plot(state.x77[,1], state.x77[,2], xlab="Population", ylab="Income", col=3)
model <- lm(state.x77[,2] ~ state.x77[,1])
abline(model, lwd=2, lty=3)
```

labelling points

```
## Example of labeling points

plot(mtcars$wt, mtcars$mpg, main="Mileage vs. Car Weight",
    xlab="Weight", ylab="Mileage", pch=18, col="blue")
text(mtcars$wt, mtcars$mpg, row.names(mtcars), cex=0.6, pos=4, col="red")
```

Illustrating all type= values

```
x <- c(1:5); y <- x # create some data
par(pch=22, col="red") # plotting symbol and color
par(mfrow=c(2,4)) # all plots on one page
opts = c("p","l","o","b","c","s","S","h")
for(i in 1:length(opts)){
  heading = paste("type=",opts[i])
  plot(x, y, type="n", main=heading)
  lines(x, y, type=opts[i])
}
```

4. Boxplot

```
library(datasets)
boxplot(state.x77)
boxplot(scale(state.x77))

## population
boxplot(state.x77[,1], ylab="Population")
title("Boxplot of State Populations")

# Boxplot of MPG by Car Cylinders
boxplot(mpg~cyl,data=mtcars, main="Car Milage Data",
    xlab="Number of Cylinders", ylab="Miles Per Gallon")
```

---

5. Multiple plots on screen

```
## Row-wise
par(mfrow=c(2,1))
plot(state.x77[,1], state.x77[,2], xlab="Population", ylab="Income", col=3)

plot(state.x77[,1], ylab="Population")
## Column-wise
par(mfcol=c(1,2))
plot(state.x77[,1], state.x77[,2], xlab="Population", ylab="Income", col=3)

plot(state.x77[,1], ylab="Population")

plot(state.x77[,2], ylab="Income")

par(mfcol=c(1,1))
```

6. Matrix plots using `matplot`

```
JohnsonJohnson

class(JohnsonJohnson)

m <- matrix(JohnsonJohnson, ncol=4, byrow = TRUE)
m

matplot(m, type="l")
```

7. Q-Q Plots

```
# Q-Q plots
x1 <- rnorm(100)

qqnorm(x1)
qqline(x1)

# Comparing 2 distributions
par(mfrow=c(1,2))
```

```
x <- rt(100, df=3)
# normal fit
qqnorm(x); qqline(x)

# t(3Df) fit
qqplot(rt(1000,df=3), x, main="t(3) Q-Q Plot",
   ylab="Sample Quantiles")
abline(0,1)
```

Interpreting QQ Plots: http://stats.stackexchange.com/a/101290/21450

Key Parameters

Many base plotting functions share a set of parameters. Here are a few key ones:

- **pch**: the plotting symbol (default is open circle)
- **lty**: the line type (default is solid line)
- **lwd**: the line width, specified as an integer multiple
- **col**: plotting color
- **main**: main plot title
- **xlab**: x-axis label
- **ylab**: y-axis label

References:

1. pch: 0 to 25 (refer: http://www.endmemo.com/program/R/pchsymbols.php)
2. lty: 1 to 6
3. lwd: 1 to 8

Intro to ggplot2

```
library(ggplot2)

ggplot(data=mtcars, aes(x=wt, y=mpg)) +
    geom_point() +
    labs(title="Automobile Data", x="Weight", y="Miles Per Gallon")
```

- ggplot

    - Based on Graphics of Grammar
        - data (in data frame format)
        - geometry of one or multiple aesthetics

- geom
    - short for `Geometric objects`
        - includes
            - points
            - lines
            - bars
            - boxplots
            - density plots
- aes
    - how the information is represented visually
    - options in aes() - specifies what role each variable will play
- Optional annotations

Common options in geom functions

- color
- fill
- alpha
    - 0: transparent
    - 1: opaque
- linetype
    - 1 to 6
- size
- shape
- binwidth
- width
- position
    - dodge

- stacked
- fill
- jitter

Colors

```
ggplot(data=mtcars, aes(x=wt, y=mpg, color=cyl)) +
    geom_point(size=5) +
    labs(title="Automobile Data", x="Weight", y="Miles Per Gallon")

ggplot(data=mtcars, aes(x=wt, y=mpg, color=factor(cyl))) +
    geom_point() +
    labs(title="Automobile Data", x="Weight", y="Miles Per Gallon")

ggplot(data=mtcars, aes(x=wt, y=mpg, color=factor(cyl))) +
    geom_point(color="red") +
    labs(title="Automobile Data", x="Weight", y="Miles Per Gallon")
```

Histograms

```
library(lattice)

ggplot(singer, aes(x=height)) +
    geom_histogram()

ggplot(singer, aes(x=height)) +
    geom_histogram(binwidth = 2)
```

Box plots

```
ggplot(singer, aes(x=voice.part, y=height)) +
    geom_boxplot()
```

Bar plots

```
data(Salaries, package="car")

ggplot(Salaries, aes(x=rank, fill=sex)) +
```

```
    geom_bar(position="stack") +
    labs(title='position="stack"')

ggplot(Salaries, aes(x=rank, fill=sex)) +
    geom_bar(position="dodge") +
    labs(title='position="dodge"')

ggplot(Salaries, aes(x=rank, fill=sex)) +
    geom_bar(position="fill") +
    labs(title='position="fill"')

ggplot(Salaries, aes(x=rank, fill=sex)) +
    geom_bar(position="fill") +
    labs(title='position="fill"') +
    coord_flip()

ggplot(Salaries, aes(x=rank, fill=sex)) +
    geom_bar(position="dodge") +
    labs(title='position="dodge"') +
    scale_fill_grey(start = 0, end = 1)
```

Scatter plots

```
ggplot(Salaries, aes(x=rank, y=salary, color=sex)) +
    geom_point()

ggplot(Salaries, aes(x=rank, y=salary, color=sex)) +
    geom_point(position="jitter", size = 3)

ggplot(Salaries, aes(x=rank, y=salary, color=sex)) +
    geom_jitter(size = 3)

ggplot(Salaries, aes(x=rank, y=salary, color=sex)) +
    geom_jitter(aes(shape = sex), size = 3)

ggplot(Salaries, aes(x=yrs.service, y=salary)) +
    geom_jitter(size = 3) +
    geom_smooth(method=lm)
```

Line plots

```
JohnsonJohnson

jj <- matrix(JohnsonJohnson, ncol = 4, byrow = TRUE)

jj <- cbind(matrix(1960:1980),jj)
```

```
colnames(jj) <- c("Year","Q1", "Q2", "Q3", "Q4")

jj <- data.frame(jj)

ggplot(jj,aes(x=Year,y=Q1)) +
    geom_line()


### illustrating example of reshaping data for ggplot plotting
library(reshape2)

melt_jj <- melt(jj, id.vars = "Year")

ggplot(melt_jj, aes(x=Year, y = value, color=variable)) +
    geom_line()
```

Grouping

```
data(Salaries, package="car")
library(ggplot2)

ggplot(Salaries, aes(x=salary)) +
    geom_density(alpha=0.3)

ggplot(Salaries, aes(x=salary, fill=rank)) +
    geom_density(alpha=0.3)

ggplot(Salaries, aes(x=yrs.since.phd, y=salary, shape=sex, color=rank )) +
    geom_point(size=3)
```

Saving plots to disk

```
ggplot(mtcars,aes(x=wt, y=hp)) +
    geom_point()
ggsave("myplot.pdf")
ggsave("myplot.png")
```