Base Plotting System in R

## Default:

> hist(airquality$Ozone)

> with(airquality,plot(Wind,Ozone))

> airquality <- transform(airquality, Month = factor(Month))

> boxplot(Ozone~Month,airquality,xlab="Month",ylab="Ozone(ppb)")

Use ?par to look up the parameters.

Use par() to specify global graphics parameters that affect all plots in an R session.

## Base Plotting Functions:

* plot
* lines: add lines to a plot.
* points: add points to a plot
* text: add text labels to a plot.
* title: add annotations to x, y axis labels.
* mtext: add arbitrary text to the margins
* axis: adding axis ticks/labels.

## Examples:

> with(airquality, plot(Wind, Ozone))

> title(main="Ozone and Wind in NYC") #add a title.

> #Base Plot with Regression Line

> with(airquality,plot(Wind, Ozone,main="Ozone and Wind in NYC",pch=20))

> model <- lm(Ozone~Wind,airquality)

> abline(model,lwd=2)

## Multiple Base Plots:

par(mfrow=c(1,3),mar=c(4,4,2,1),oma=c(0,0,2,0))

with(airquality, {

plot(Wind,Ozone,main="Ozone and Wind")

plot(Solar.R, Ozone, main = "Ozone and Solar Radiation")

plot(Temp,Ozone,main="Ozone and Temperature")

mtext("Ozone and Weather in NYC", outer = T)

})

MARGINS:

1: down 2: left 3: up 4: right

## Many parameters:

> par(mar = c(4,4,2,2)) # set the margins

> plot(x, y, pch = 4) # change plotting symbols.

> title("Scatterplot") # Change title

> text(-2,-2, "Label") # Add a label to the specific position

> legend("topleft", legend = "data")

> legend("topleft", legend = "data", pch = 20) # add a little circle

> fit <- lm(y~x) #Regression

> abline(fit) # Add the line to the graph

> abline(fit,lwd=3) # Change line width

> abline(fit,lwd=3, col = "red")

> plot(x,y,xlab="Weight",ylab="Height",main="Scatterplot",pch=20)

> legend("topright", legend = "Data", pch = 20)

# If you want details, please go to “Add legend to a plot”:

http://www.r-bloggers.com/adding-a-legend-to-a-plot/

> par(mfrow = c(2,1)) # Draw graphs of 2 rows and 1 col.

> plot(x,y,pch=20)

> plot(x,z,pch=19)

# Different colors (subsetting the dots)

> y <- x + rnorm(100)

> g <- gl(2,50, labels = c("Male","Female"))

#gl : Generate factor levels

> str(g)

Factor w/ 2 levels "Male","Female": 1 1 1 1 1 1 1 1 1 1 ...

> plot(x,y)

> plot(x,y,type = "n")

> points(x[g=="Male"],y[g=="Male"], col = "green")

> points(x[g=="Female"], y[g=="Female"], col = "pink")

What is a Graphics Device?

## A graphics device is something where you can make a plot appear.

* A window on your computer (screen device)
* A PDF file (file device)
* A PNG or JPEG file (file device)
* A scalable vector graphics (SVG) file (file device)

## How to launch the screen device:

* Mac: quartz()
* Windows: windows()
* Unix/Linux: x11()

P.S. List of Graphic Device: **?Devices**;

## Plot to your screen device

> library(datasets)

> with(faithful, plot(eruptions,waiting))

> title(main = "Old Faithful Geyser data")

## Plot to file device

> pdf(file = "myplot.pdf") # will create the file in your working directory

> with(faithful, plot(eruptions,waiting))

> title(main = "Old Faithful Geyser data")

> dev.off() # Close the PDF file device

## Multiple open graphic devices

It is possible to open multiple graphics devices at one time, but plotting can only occur on one graphics device at a time. The currently active graphics device can be found by calling **dev.cur()**.

You can change the active graphics device with **dev.set(<integer>).**

The Lattice Plotting System

All plotting/annotation is done at once with a single function call.

## Packages

* lattice: includes functions like xyplot, bwplot, levelplot
* grid

## Lattice Functions

* xyplot: This is the main function for creating scatterplots
* bwplot: box-and-whiskers plots (“boxplots”)
* histogram
* stripplot: like a boxplot but with actual points
* dotplot: plot dots on “violin strings”
* splom: scatterplot matrix; like pairs in base plotting system
* levelplot, contourplot: for plotting “image” data

## xyplot

Form: **xyplot(y ~ x | f \* g, data)**

* f and g are conditioning variables – they are optional
* the \* indicates an interaction between two variables

## A Simple Lattice Plot

> library(lattice)

> library(datasets)

> ## Simple scatter plot

> xyplot(Ozone ~ Wind, data = airquality)

> ## Convert 'Month' to a factor variable

> airquality <- transform(airquality, Month = factor(Month))

> xyplot(Ozone ~ Wind | Month, data = airquality, layout = c(5,1))

## Lattice Panel Functions (I cannot understand….)

Panel functions can be specified/customized to modify what is plotted in each of the plot panels.

> set.seed(10)

> x <- rnorm(100)

> f <- rep(0:1, each = 50)

> y <- x + f - f \* x + rnorm(100,sd=0.5)

> f <- factor(f,labels=c("Group 1","Group 2"))

> xyplot(y~x | f, layout = c(2,1))

## Custom panel function (abline)

xyplot(y ~ x | f, panel = function(x,y, ...) {

panel.xyplot(x,y,...) # First call the default panel function for 'xyplot'

panel.**abline**(h = median(y), lty = 2) # Add a horizontal line at the median

})

## Regression line

xyplot(y ~ x | f, panel = function(x,y, ...) {

panel.xyplot(x,y,...) # First call the default panel function for 'xyplot'

panel.**lmline**(x,y,col=2) #Overlay a simple linear regression line

})

The ggplot2 Plotting System

The package ggplot2 is an implementation of the Grammar of Graphics.

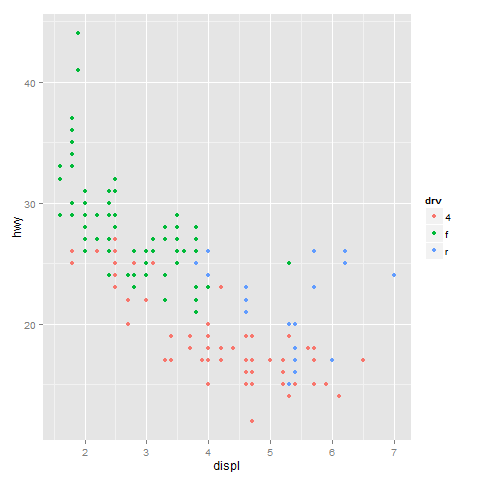
## Simple Plotting with qplot()

### 1. Scatter graph

> library(ggplot2)

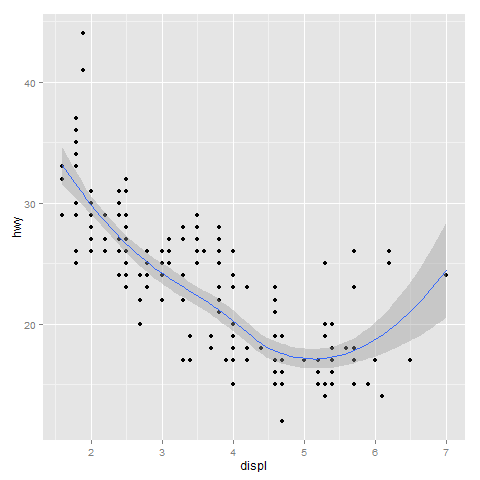
> qplot(displ,hwy,data=mpg)

> qplot(displ,hwy,data=mpg,color=drv) #Auto legend placement when specified subsetting by drv;



Adding a geom

qplot(displ,hwy,data=mpg,geom=c("point","smooth"))



### 2. Histograms

qplot(hwy,data=mpg,fill=drv)

### 3. Facets

> qplot(displ,hwy,data=mpg,facets=.~drv)

> qplot(hwy,data=mpg,facets=drv~.,binwidth=2)

Notice:

Facets = x ~ y where x means separate rows by variable x, y means separate columns by variable y.

## Customizable ggplot2

### 1. Basic Components of a ggplot2 Plot

* A data frame
* Aesthetic mappings: how data are mapped to color, size
* Geoms: geometric objects like points, lines, sapes.
* Facets: for conditional plots
* Stats: statistical transformations like binning, quantiles, smoothing.
* Scales: what scale an aesthetic map uses (examples: male=red, female=blue).
* Coordinate system

### 2. Building Plots with ggplot2

Plots are built up in layers

* Plot the data
* Overlay a summary
* Metadata and annotation

### 3. Basic Examples

> g <- ggplot(mpg,aes(displ,hwy))

> print(g)

Error: No layers in plot

> p <- g+geom\_point()

> p #Auto print (The same as print(p))

### 4. Adding More Layers: Smooth

g + geom\_point() + geom\_smooth(method="lm")

### 5. Adding More Layers: Facets

g + geom\_point() + facet\_grid(.~drv)+geom\_smooth(method="lm")

### 6. Annotation

* Labels: xlab(), ylab(), labs(), ggtitle()
* For things that only make sense globally, use theme()
  + Example: theme(legend.position = “none”)
* Two standard appearance themes are included
  + Theme\_gray(): The default theme (gray background)
  + Theme\_bw(): More stark/plain

### 7. Modifying aesthetics

g + geom\_point(color = "steelblue", size=4, alpha=1/2)

g + geom\_point(aes(color=drv),size=4,alpha=1/2)

# alpha specifies the transparency ?

### 8. Modifying Labels

g + geom\_point(aes(color=drv)) + labs(title="MAACS Cohort",x=expression("log "\*PM[2.5]),y="Nocturnal Symptoms")

### 9. A Note about Axis Limits

> x <- 1:100

> y <- rnorm(100)

> testdat <- data.frame(x,y)

> testdat[50,2] <- 100 ## Outlier!

> plot(testdat$x,testdat$y,type="l",ylim = c(-3,3))

> g <- ggplot(testdat, aes(x=x,y=y)) ## ggplot

> g + geom\_line()

# 1. Outlier missing

> g + geom\_line() + ylim(-3,3)

# 2. Outlier included

> g + geom\_line() + coord\_cartesian(ylim=c(-3,3))

### 10. Categorize on Continuous Variable

## Calculate the deciles of the data (Making NO2 (Nitrogen dioxide) tertiles)

cutpoints <- quantile(maacs$logno2\_new, seq(0, 1, length=4), na.rm = TRUE)

## Cut the data at the deciles and create a new factor variable

maacs$no2dec <- **cut**(maacs$logno2\_new, cutpoints)

## See the levels of the newly created factor variable

levels(maacs$no2dec)

[result]: [1] “(0.378, 1.2]” “(1.2, 1.42]” “(1.42, 2.55]”

**facet\_wrap**(bmicat ~ no2dec, nrow=2, ncol=4

Working with Color in R Plots

## Built-in Package: grDevices

This package has two functions:

* colorRamp
* colorRampPalette
* The function colors() lists the names of colors you can use in any plotting function.

## colorRamp {grDevices}

> pal <- colorRamp(c("red","blue"))

> pal(0)

[,1] [,2] [,3] #[1 is Red] [2 is Green] [3 is Blue]

[1,] 255 0 0

> pal(0.5)

[,1] [,2] [,3]

[1,] 127.5 0 127.5

> pal(1)

[,1] [,2] [,3]

[1,] 0 0 255

## colorRampPalette {grDevices}

> pal <- colorRampPalette(c("red","yellow"))

> pal(2)

[1] "#FF0000" "#FFFF00"

# For example in the first element “FF0000”:

# First pair (FF): Red

# Second pair (00): Green

# Third pair (00): Blue

> pal(10)

[1] "#FF0000" "#FF1C00" "#FF3800" "#FF5500" "#FF7100" "#FF8D00" "#FFAA00" "#FFC600" "#FFE200" "#FFFF00"

## Package RColorBrewer

There are 3 types of palettes

* Sequential
* Diverging
* Qualitative

The color information can be used conjunctively with the two functions in package grDevices.

## Use RColorBrewer and colorRampPalette Together

> library(RColorBrewer)

> cols <- brewer.pal(3,"BuGn") # I want 3 colors

> ?brewer.pal # To look up the names of the palettes.

> cols

[1] "#E5F5F9" "#99D8C9" "#2CA25F"

> pal <- colorRampPalette(cols)

> image(volcano, col=pal(20))

## The smoothScatter Function

> x <- rnorm(10000)

> y <- rnorm(10000)

> smoothScatter(x,y) # When you have many points to plot

## The rgb Function

> ?rgb

* The rgb function can be used to produce any color via red, green, blue proportions
* Color transparency can be added via the alpha parameter to rgb

## Scatterplot with Transparency

> plot(x,y,pch=19)

# Versus:

> plot(x,y,col = rgb(0,0,0,0.2),pch=19)