An Introduction to

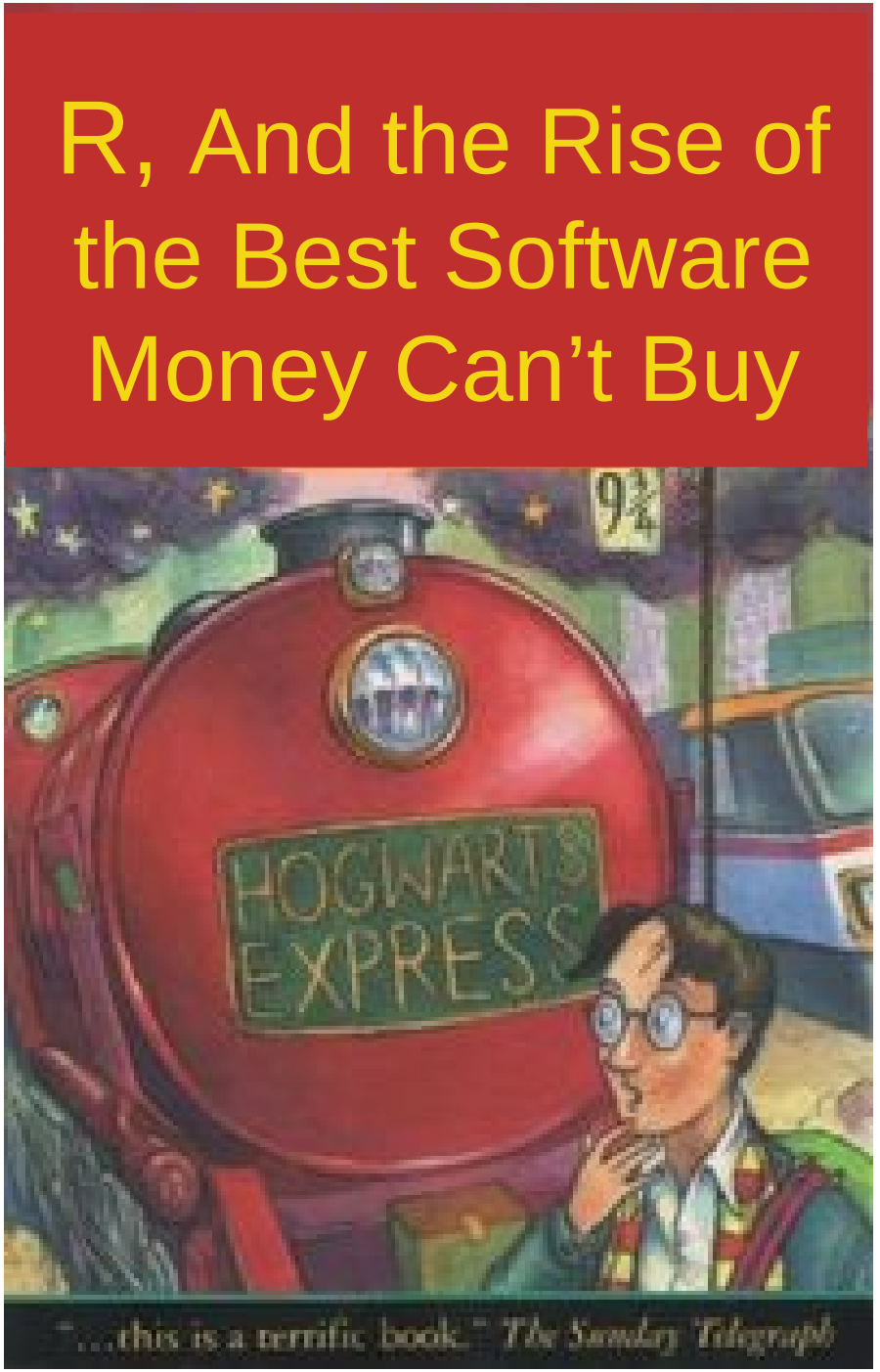
Richard Ngamita

Tech lead (Refugees United)

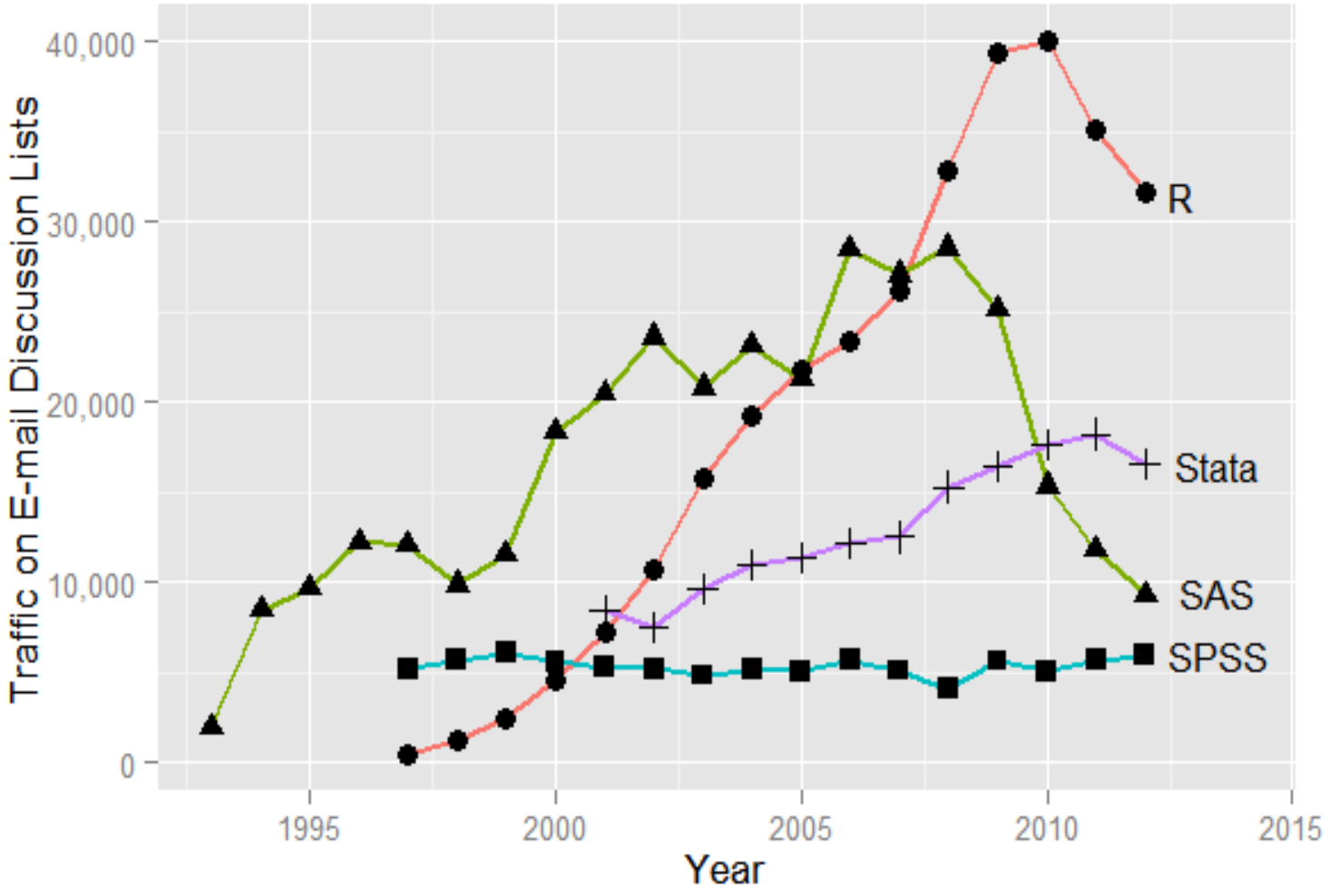
Web resources

- **R** home page: http://www.r-project.org/
- **R** Archive: http://cran.r-project.org/
- **R** FAQ (frequently asked questions about **R**): http://cran.r-project.org/doc/FAQ/R-FAQ.html
- **R** manuals: http://cran.r-project.org/manuals.html

R programming language is a lot like witchcraft... except instead of spells you have functions.



Why R?


Traffic on E-mail Discussion Lists
40,000
30,000
20,000
10,000
0
1995
2000
2005
2010
2015
Year
R
Stata
SAS
SPSS

Installation

Mac Install

http://cran.r-project.org/bin/macosx/

Windows Install

http://cran.r-project.org/bin/windows/base/

Linux Install

http://cran.r-project.org/bin/linux/ubuntu/README

Next, choose the appropriate package for RStudio :

http://www.rstudio.com/ide/download/desktop

The R environment

- **R** command window (console) or Graphical User Interface (GUI)
  - Used for entering commands, data manipulations, analyses, graphing
  - Output: results of analyses, queries, etc. are written here
  - Toggle through previous commands by using the up and down arrow keys

The R environment

- The **R** workspace
  - Current working environment
  - Comprised primarily of variables, datasets, functions

The R environment

- **R** scripts
  - A text file containing commands that you would enter on the command line of **R**
  - To place a comment in a **R** script, use a hash mark (#) at the beginning of the line

Executing simple commands

- The assignment operator **<-**
- **X <- 1** assigns the value of 1 to the variable x
- **y <- 5*x** assigns the value of 5 times x (5 in this case) to the variable y
- **r <- 2**
- **area.circle <- pi*r^2**
- **NOTE: R is case-sensitive (y ≠ Y)**
- 

Some tips for getting started in R

1. Create a new folder on your hard drive for your current **R** session
2. Open **R** and set the working directory to that folder
3. Save the workspace with a descriptive name and date
4. Open a new script and save the script with a descriptive name and date (Google's R style guide).

R object types

- Vector
- Matrix
- Array
- Data frame
- Function
- List

Vectors and arrays

- **Vector**: a one-dimensional array, all elements of a vector must be of the same type (numerical, character, etc)

- **Matrix**: a two-dimensional array with rows and columns

- **Array**: as a matrix, but of arbitrary dimension

Entering data into R

- **Vectors**:
- The “c” command
  - combine or concatenate data

- Data can be character or numeric

```
v1 <- c(1, 15, 26, 4, 4)

v2 <- c("Mukene", "Engege", "Angara")
```

Entering data into R

- **Vectors**:
- Sequences of numbers

***c*()**

***seq*()**

```
years<-c(2006:2014)

x<-seq(0,100,10)

x<-seq(0, 200, length=100)
```

Entering data into R

- **Arrays:**

  ***array*()**

  ***matrix*()**

```
m1<-array(1:20, dim=c(4,5))

m2<-matrix(1:20, ncol=5, nrow=4)
```

Entering data into R

- **Arrays:**
- Combine vectors as columns or rows

  ***cbind*()**

  ***rbind*()**

Matrix1 <- cbind(v1, v2)

Matrix2 <- rbind(v1, v2)

Data frames

- A data frame is a list of variables of the same length with unique row names
- A collection of variables which share many of the properties of matrices and of lists
- Used as the fundamental data structure by most of **R**'s modeling software

Data frames

- Convert vectors or matrices into a data frame

  ***data.frame*()**

  df1<-data.frame(v1, v2)

  df2<-data.frame(matrix1)

Data frames

- Editing data frames in spreadsheet-like view

  ***edit*()**

  df2<-edit(df1)

Placing variables in the R search path

- When variables in a data frame are used in **R**, the data frame name followed by a **$** sign and then the variable name is required

query1<-df1**$**v3 > 20

Placing variables in the R search path

- Alternatively, the ***attach*()** function can be used

```
attach(df1)
query1 <- v3 > 20
detach(df1)
```

Accessing data from an array, vector, or data frame

- *Subscripts* are used to extract data from objects in **R**

- Subscripts appear in square brackets and reference rows and columns, respectively

Subscripts

df[3,5]

df[,3]

df[5,]

df[2:5,]

df1

	C1	C2	C3	C4	C5
R1	25	Mon	56	45	Cat
R2	2	Tues	84	2	Dog
R3	24	Wed	7	15	Dog
R4	15	Thurs	56	236	Cat
R5	26	Fri	89	6	Cat
R6	25	Sat	23	58	Dog
R7	2	Sun	11	8	Dog

Queries in R:
Common logical arguments

>	Greater than
<	Less than
==	Equals
!x	! Indicates logical negation (not), not x
x & y	Logical and, x and y
x | y	Logical or, x or y

Queries in R

- The use of logical tests

query1<-df1$v3 > 20

df1[query1,]

query2<-df1$v3 > 20 & df1$v4 < 30 (&=and)

query2<-df1$v3 > 20 | df1$v4 < 30 (| = or)

Introduction to R functions

- **R** has many built-in functions and many more that can be downloaded from CRAN sites (Comprehensive **R** Archive Network)

- User-defined functions can also be created

Introduction to R functions

- **Common functions**

  ***names*()**: obtain variable names of a df

  ***summary*()**: summary of all variables in a df

  ***mean*()**: Mean

  ***var*()**: Variance

  ***sd*()**: standard deviation

Introduction to R functions, cont

***head*()**: print first few rows of data frame

***sapply*()** and **tapply()**: column-wise summaries

***levels*()**: obtain levels of a character variable

***by*()**: produce summaries *by* group

Introduction to R functions, cont

tapply(variable, list(group1, group2), mean)

Applies function to each element in ragged arrays

sapply(variable, FUN=)

Applies a function to elements in a list

by(data, INDICES, FUN)

Importing data from Excel

- ***read.table*()**

  data.frame.name <- read.table(“file path”, na.strings=”NA”, header=TRUE)

  df1<-read.table("C:\\R\\Example\\datafile1.txt", na.strings="NA", header=TRUE)

read.xlsx(), read.xlsx2(), write.xlsx()

Slow,file, sheetindex etc

Note the use of **** instead of **** in path name

Data Munging 70%

- ***read.table*()**

  tolower(), toupper()

  strsplit() e.g on period “\\.”

  sapply()

  sub(), gsub()

  as.Date(), lubridate

  merge() like joins in SQL.

  Sort() , order() and REGex is KING.

Introduction to R loops

Basic syntax:

```
for (i in 1:n){
   some code
}
```

Introduction to R loops

Basic syntax:

```
for (i in 1:n){
   some code
}
```

User-defined functions

```
Function name <- function(x){
        argument }
```

R functions part 2: subset data

- *subset*() function

sub<- subset(data frame, criteria)

sub1<-subset(cars, mpg == 21)

sub2<-subset(cars, cyl >50 & gear == 4)

R functions part 2: subset data

- Select specific columns

```
sub3<-subset(cars, select=c(mpg,disp , hp))
```

Introduction to basic graphing

- ***plot*(): ** A generic function that produces a type of plot that is dependent on the type of the first arguement
- ***Hist*() or density():** Creates a histogram of frequencies
- ***barplot*():** Creates a histogram of values
- ***boxplot*():** Creates a boxplot
- ***map():*** *Creates maps*

Common high-level functions

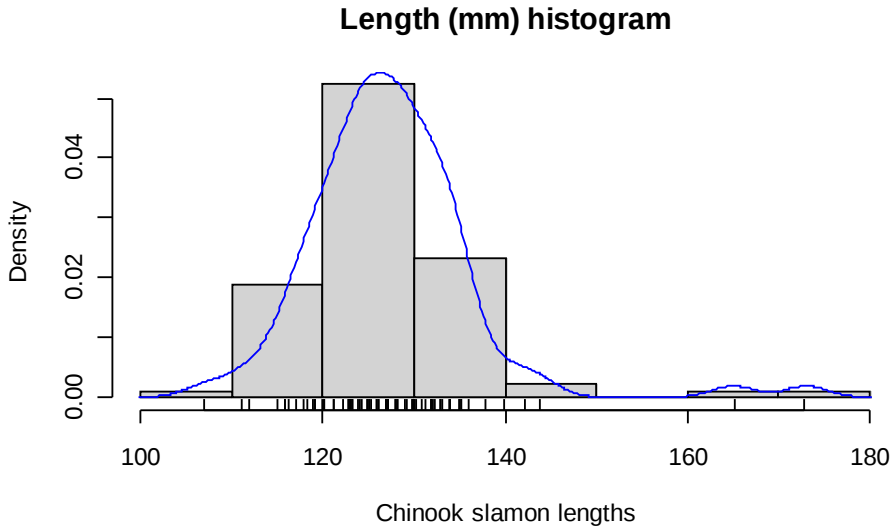
plot()

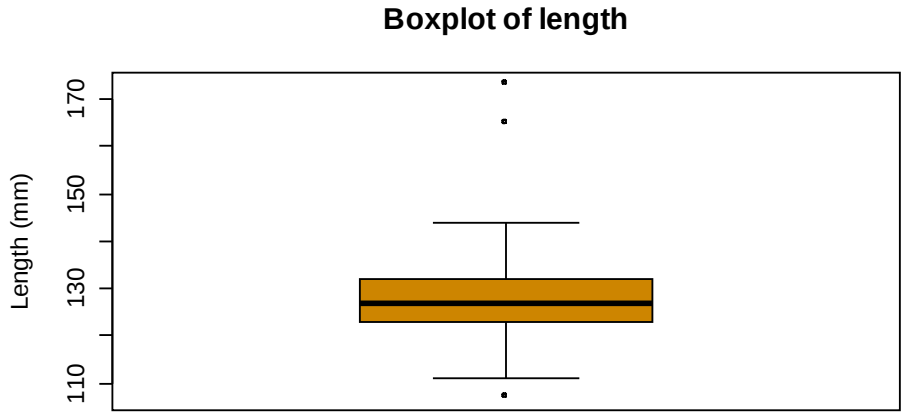
plot(x)

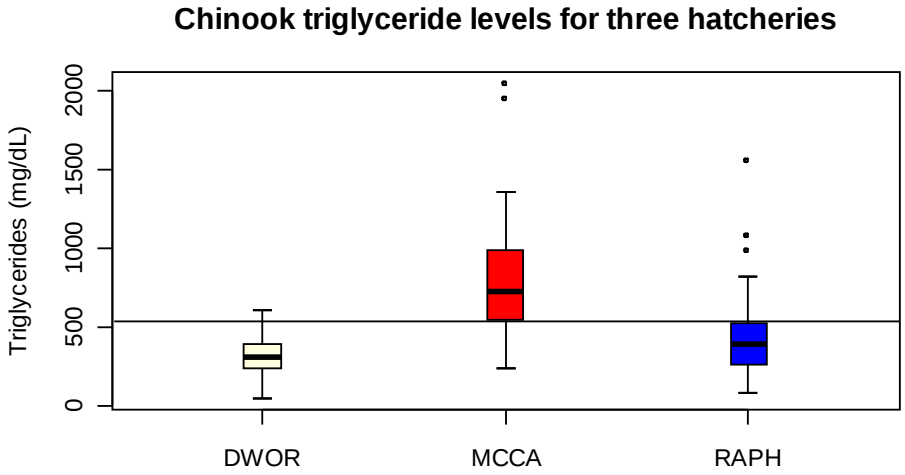
plot(x,y) : scatter plot

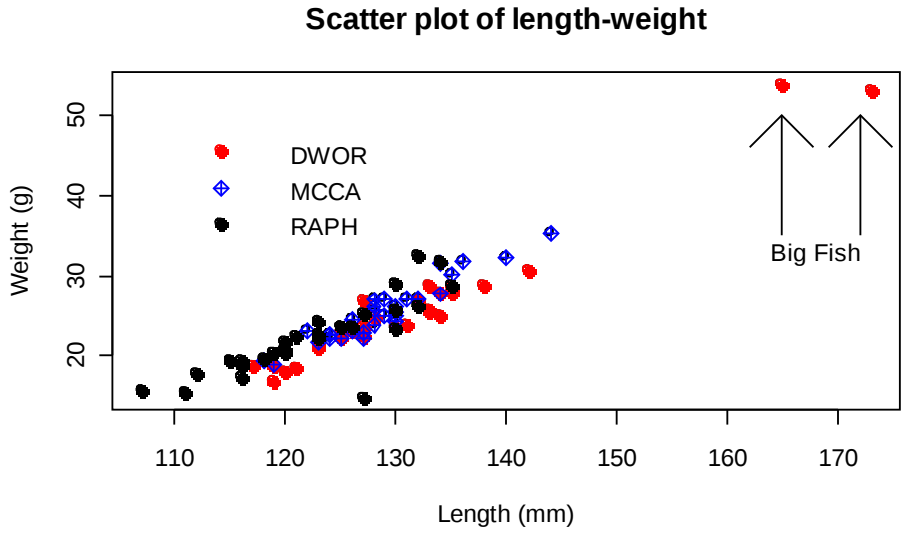
plot(y~x) : scatter plot

Lower-level graphing functions









Lower-level graphing functions

- **Axis scales and labels**

  xlim=c(0,50)

  ylim=c(0,100)

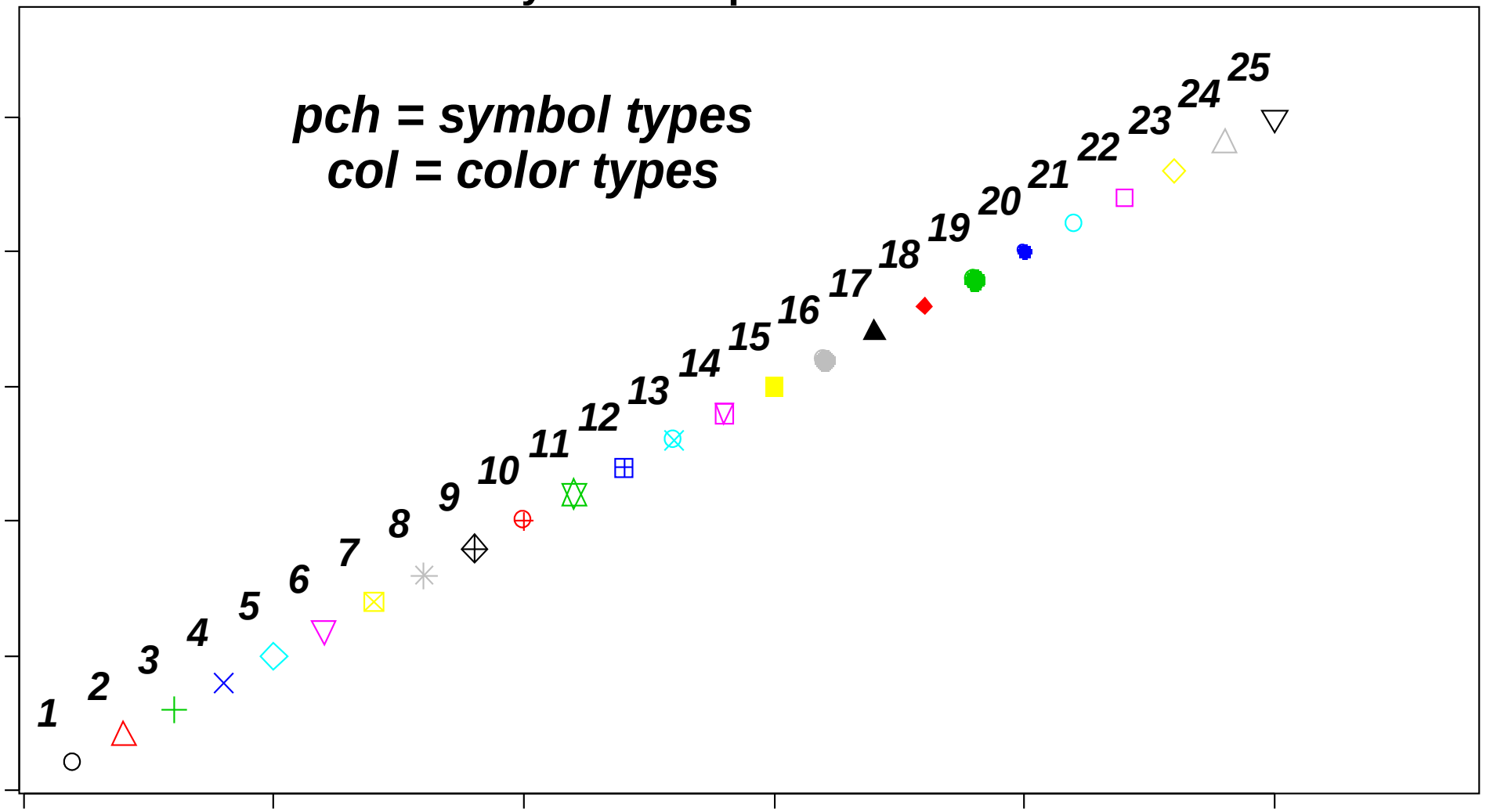
  xlab="text"

  ylab="text"

  main="text"

  cex= <1 will make font smaller than default, >1 will increase font size

Lower-level graphing functions



High-level graphing (ggplot2)

- **Sneak peak at more mature graphics in R.**

Examples : ggplot2 and lattice.

http://docs.ggplot2.org/current/

Plotting 2 lines.

Get/Set your W/D

- **Importantly get/set your w/d**

  getwd()

  setwd()

- Differences Windows machines.

setwd("C:\\Users\\Ngamita\\Downloads")

- Relative vs Absolute paths.

Load data

- **Disclaimer: data-set only for this training purpose, delete after this training.**

file<-
'https://www.dropbox.com/s/vcnpqzl87e9jhi0/Eng_queries_01_15.csv'

download.file()

why wget not curl?

read.table(), read.csv(), read.csv2()

date()

RAM?

Load data

** Exploratory summarize data functions.*

dim()

Summary() *attach() and detach()*

Subset() *edit(df) or edit(data.frame())*

head() *quantile()*

tail()

class()

str() *nrow()* *ncols()*

Missing Values

Missing values and NAs

complete.cases() # dont forget coma.

is.na()

na.omit() # remove all raws or cases with NA occurrence

any(), all()

UseNA = 'ifany'

Data from Questions.

data from questions?

 # How many sms'es received monthly/quarterly?

 # Breakdown of statuses?

 # Which months recorded more sms'es than others?

 # Which shortcode received more messages than the other?

 # Under which short code did we have lots of no matches?

 # % duplicates more than 1?

 # Under no_match status % duplicates?

 # Query word length averages?

 # % of queries - non english?

 # classification of queries accordingly agric/news/sports/health/nonsense etc?

 # -- > supervised/non-supervised text classification

 # Most common keywords used? Wordcloud!!

 # What is the lexical diversity per message?

Packages(plyr)

- **Installing Packages**

  install.packages()

  installed.packages()

  old.packages()

  update.packages()

- Load Package

  library(plyr)

  require(plyr)

Packages(plyr)

- **Examples.**

- How many times per day do we receive messages? frequency messages per day?
- How many times did each status appear on specific shortcode?
- What total did each shortcode gain in total?
- What was the mean duplicate count in relation to shortcode?

Packages(RmySQL)

- **Examples.**

- library(RMySQL)
- Use db: world database

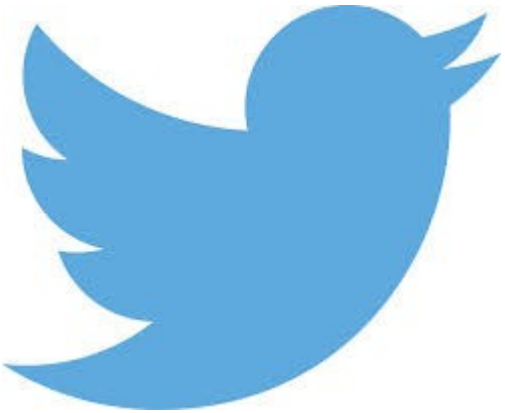
Using sqldf? Runs sql like queries in R.

Packages(RGoogleAnalytics)

- **Examples.**

- Analyzing web traffic with Google Analytics API.

Who's using R?



Recommended Resources.