

RStudio is good for you!

Julyan Arbel (CREST - INSEE) - **Statisfaction**

Semin-R

7 February 2012

Useful links



- RStudio (download, blog)
 - R-bloggers
 - Satisfaction on Google + and Facebook
- RStudio is good for you!
- Running R on an iPhone/iPad with RStudio

Multi platform IDE

The screenshot displays the RStudio IDE interface. The top-left pane shows an R script with the following code:

```
1 # Home Prices in Mid-West
2
3 homes <- read.csv("homePriceData.csv")
4 View(homes)
5 names(homes)
6 summary(homes$price)
7 summary(homes$age)
8
9 states <- levels(homes$state)
10 avePrice <- round(mean(homes$price),2)
11 aveAge <- round(mean(homes$age), 0)
12 |
13
14
```

The top-right pane shows the 'Workspace' and 'History' tabs. The 'Data' tab is active, displaying the following information:

Data	
homes	580 obs. of 7 variables
Values	
aveAge	36
avePrice	416485.55
states	character[11]

The bottom-right pane shows the 'Files', 'Plots', 'Packages', and 'Help' tabs. The 'Help' tab is active, displaying the documentation for the 'subset' function. The title is 'Subsetting Vectors, Matrices and Data Frames'. The description states: 'Return subsets of vectors, matrices or data frames which meet conditions.' The usage is shown as 'subset(x, ...)'. A search bar at the top of the help pane contains the text 'subset'.

The bottom-left pane shows the 'Console' output:

```
> homes <- read.csv("homePriceData.csv")
> View(homes)
> names(homes)
[1] "city"      "state"     "price"     "age"
[5] "condition" "remodeling" "neighborhood"
> summary(homes$price)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 75290 152900 252900 252900 352900 452900
> summary(homes$age)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  0.00  5.00  10.00  15.00  20.00  30.00
> states
[1] "IL" "IN" "MI" "OH" "WI"
> avePrice
[1] 416485.55
> aveAge
[1] 36
> old <- sub
```

A tooltip is visible over the 'subset' function name in the console, showing the following details:

```
subset(x, ...)
Return subsets of vectors, matrices or data frames which meet
conditions.
```

Multi platform IDE

The screenshot displays the RStudio IDE interface on a Linux system. The main window is titled "analysis.R*" and contains the following R code:

```
1 # User Analysis
2
3 setwd("~/analysis")
4 source("prep.R")
5
6 library(plyr)
7 library(lattice)
8 library(ggplot2)
9
10 # Import data set
11 rawdata <- read.csv("stats.csv")
12 totalUsers <- dim(rawdata)[1]
13
14 # Clean data set
15 clean <- prepareStats(rawdata)
16
17 # Subset of active users
18 active <- subset(clean, active == 1)
19 count(active, "daysSinceAccountCreated < 30")[2,2]
20 mean(active$age)
21
22
23
24
```

A tooltip is visible over the "Run" button (a green play icon) with the text: "Run the current line or selection (Ctrl+Enter)".

The console at the bottom shows the execution of the code:

```
> # User Analysis
> setwd("~/analysis")
> source("prep.R")
> library(plyr)
> library(lattice)
> library(ggplot2)
Loading required package: grid
Loading required package: proto
> # Import data set
> rawdata <- read.csv("stats.csv")
> totalUsers <- dim(rawdata)[1]
>
> # Clean data set
> clean <- prepareStats(rawdata)
>
```

The right-hand side of the interface features a "Workspace" pane showing the current environment:

- Data:** clean (360404 obs. of 35 variables), rawdata (530750 obs. of 35 variables)
- Values:** totalUsers (530750L)
- Functions:** evalPercentage(expression, index), formatInteger(object, ...), prepareStats(data, sampleSize = NA)

Below the workspace is a "Files" pane with a search bar and a list of installed and available packages:

Files	Plots	Packages	Help
<input type="checkbox"/>		Install Packages	<input checked="" type="checkbox"/> Check for Updates
<input type="checkbox"/>		codetools	Code Analysis Tools for R
<input type="checkbox"/>		colorspace	Color Space Manipulation
<input type="checkbox"/>		compiler	The R Compiler Package
<input checked="" type="checkbox"/>		datasets	The R Datasets Package
<input type="checkbox"/>		digest	Create cryptographic hash digests of R objects
<input type="checkbox"/>		foreign	Read Data Stored by Minitab, S, SAS, SPSS, Stata, Systat, dBase, ...
<input type="checkbox"/>		ggplot2	An implementation of the Grammar of Graphics
<input checked="" type="checkbox"/>		graphics	The R Graphics Package
<input checked="" type="checkbox"/>		qrDevices	The R Graphics Devices and Support for Colours and Fonts
<input type="checkbox"/>		grid	The Grid Graphics Package
<input type="checkbox"/>		iterators	Iterator construct for R

Multi platform IDE

The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains R code for loading data, summarizing it, and creating a faceted plot.
- Workspace:** Shows the loaded 'diamonds' dataset with 53940 observations and 10 variables.
- Console:** Shows the execution output of the R code, including summary statistics for 'x', 'y', and 'z' (likely 'carat', 'price', and 'clarity').
- Plots:** Displays a scatter plot titled 'Diamond Pricing' showing Price vs. Carat, faceted by Clarity.

```
1 library(ggplot2)
2
3 View(diamonds)
4 summary(diamonds)
5
6 summary(diamonds$price)
7 aveSize <- round(mean(diamonds$carat), 4)
8 clarity <- levels(diamonds$clarity)
9
10 p <- qplot(carat, price,
11            data=diamonds, color=clarity,
12            xlab="Carat", ylab="Price",
13            main="Diamond Pricing")
14
```

Workspace Summary:

Data	Summary
diamonds	53940 obs. of 10 variables
Values	
aveSize	0.7979
clarity	character [8]
p	ggplot [8]
Functions	
format.plot(plot, size)	

Console Output:

```
Min.   x : 0.000   Min.   y : 0.000   Min.   z : 0.000
1st Qu.: 4.710  1st Qu.: 4.720  1st Qu.: 2.910
Median : 5.700  Median : 5.710  Median : 3.530
Mean   : 5.731  Mean   : 5.735  Mean   : 3.539
3rd Qu.: 6.540  3rd Qu.: 6.540  3rd Qu.: 4.040
Max.   :10.740  Max.   :58.900  Max.   :31.800
> summary(diamonds$price)
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  326   950   2401   3933   5324   18820
> aveSize <- round(mean(diamonds$carat), 4)
> clarity <- levels(diamonds$clarity)
> p <- qplot(carat, price,
+           data=diamonds, color=clarity,
+           xlab="Carat", ylab="Price",
+           main="Diamond Pricing")
>
> format.plot(plot=p, size=23)
>
```

Scatter Plot: Diamond Pricing

The plot shows Price (Y-axis, 0 to 15000) versus Carat (X-axis, 0 to 3). The data points are colored by Clarity, with a legend on the right:

- I1 (Red)
- SI2 (Orange)
- SI1 (Yellow)
- VS2 (Green)
- VS1 (Cyan)
- VVS2 (Blue)
- VVS1 (Purple)
- IF (Pink)

RStudio over the web

The screenshot displays the RStudio web interface in a browser window. The address bar shows `http://rstudio.example.com`. The interface includes a menu bar (File, Edit, View, Project, Workspace, Plots, Tools, Help), a toolbar with icons for source, run, and source, and a workspace area with tabs for `portfolio.R`, `bs.option.R`, and `Q1.Report.Rnw`.

The code editor shows the following R code:

```
1 # Black Scholes
2 # Option Pricing Model
3
4 # s = current stock price
5 # x = strike price
6 # r = risk free rate
7 # sigma = volatility
8 # t.exp = expiration time
9 # t = current time
10
11 # price of call option
12 callprice.bs <- function (s, x, r, sigma, t.exp, t) {
13   d.pos <- log(s/x) + (r + 0.5 * sigma^2) * (t.exp - t)
14   d.pos <- d.pos / (sigma * (t.exp - t)^0.5)
15   d.neg <- d.pos - sigma * (t.exp - t)^0.5
16   s * pnorm(d.pos) - x * exp(- r * (t.exp - t)) * pnorm(d.neg)
17 }
18
19 # price of put option
20
21 c <- callprice.bs(s, x, t.exp, t, r, sigma)
22 c - s + x * exp(- r * (t.exp - t))
23
```

A context menu is open over the code, showing options: `Extract Function` (⌘U), `Comment/Uncomment Lines` (⌘/), and `Reindent Lines` (⌘I).

The workspace area on the right shows a table of values:

Variable	Value
r	0.07
s	36.12
sigma	0.05
t	0.384615384615385
t.exp	0.5
x	37

Below the workspace, there is a file browser showing a directory structure:

- Home
- stocks
 - ..
 - bs.option.R (585 bytes, Jan 7, 2011, 9:11 AM)
 - Q1 Report.Rnw (113 bytes, Jan 7, 2011, 9:07 AM)
 - stockData.csv (86.5 MB, Jan 7, 2011, 9:09 AM)

The console at the bottom shows the execution of the code:

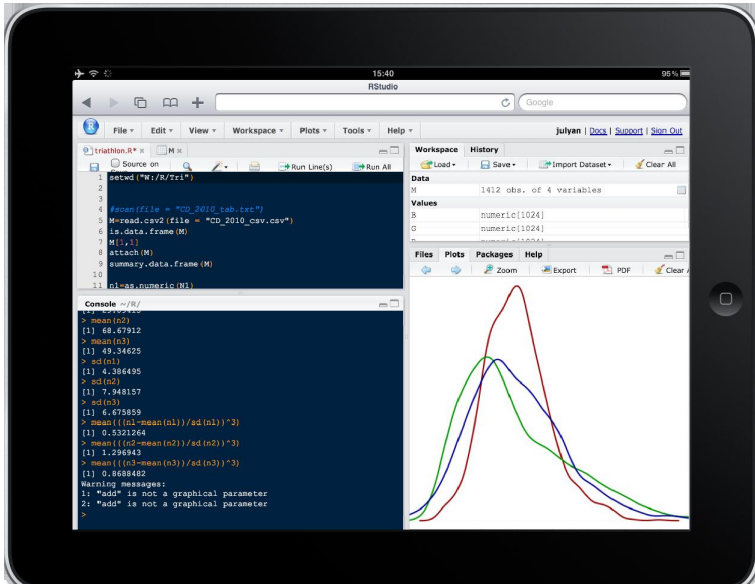
```
> # price of call option
> callprice.bs <- function (s, x, r, sigma, t.exp, t) {
+   d.pos <- log(s/x) + (r + 0.5 * sigma^2) * (t.exp - t)
+   d.pos <- d.pos / (sigma * (t.exp - t)^0.5)
+   d.neg <- d.pos - sigma * (t.exp - t)^0.5
+   s * pnorm(d.pos) - x * exp(- r * (t.exp - t)) * pnorm(d.neg)
+ }
>
>
```

RStudio over the web

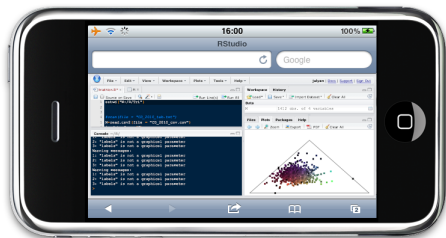
Need to create a server

- on Linux
- or on a Mac

R accessed from a mobile/tablet



R accessed from a mobile/tablet



Source, Console, Workspace, and Plots

The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains R code for loading data, creating factors, and plotting. The code includes comments and function calls like `library(plyr)`, `library(ggplot2)`, `read.csv`, `subset`, `as.factor`, `levels`, `names`, `count`, `View`, `summary`, and `qplot`.
- Console:** Shows the execution output, including the frequency of active and non-active users, the state abbreviations for the active subset, and the execution of the `qplot` command.
- Workspace:** Lists the loaded data (`userData`), values for `active` and `states`, and the `split` function.
- Plots:** Displays a plot titled "Breakdown of Users by Age and State". The plot is a dot plot where the y-axis represents age (0-70) and the x-axis represents state abbreviations (IA, IL, IN, KS, MI, MN, MO, ND, NE, OH, SD). Red dots represent non-active users (active == 0) and blue dots represent active users (active == 1).

```
1 # User Trend Analysis
2 # Breakdown of active and non-active users
3
4 library(plyr)
5 library(ggplot2)
6
7 userData <- read.csv("userDataTrends.csv")
8 userData <- subset(userData, select = -c(id, group))
9 userData$active <- as.factor(userData[,1])
10
11 states <- levels(userData$state)
12
13 names(userData)
14 count(userData, "active == 1")
15 View(userData)
16
17 summary(subset(userData, active == 1)$state)
18 summary(subset(userData, active == 0)$state)
19
20 qplot(state, age, color = active, data = userData,
21       main = "Breakdown of Users by Age and State") +
22       opts(plot.title = theme_text(size = 19))
23
```

```
active...1 freq
1 FALSE 310
2 TRUE  270
> View(userData)
> summary(subset(userData, active == 1)$state)
IA IL IN KS MI MN MO ND NE OH SD
19 26 21 21 27 49 22 26 19 16 24
> summary(subset(userData, active == 0)$state)
IA IL IN KS MI MN MO ND NE OH SD
26 27 18 31 27 49 22 32 19 33 26
> qplot(state, age, color = active, data = userData,
+       main = "Breakdown of Users by Age and State") +
+       opts(plot.title = theme_text(size = 19))
>
```

Breakdown of Users by Age and State

age

state

active

- 0
- 1

Custom Theme and Layout

~/customize - RStudio

```
binomialLattice.R* | priceMat | probMat |
Source on Save | Run | Source
1 priceMat <- matrix(NA, ncol=nsteps+1, nrow=2*nsteps+1,
2 dimnames=list(as.character(nsteps:-nsteps),
3 as.character(0:nsteps)))
4 priceMat["0", "0"] <- S0
5 for(column in 1:nsteps) {
6   for(rownum in seq(column, -column, by=-2)) {
7     if(rownum=0) {
8       priceMat[paste(rownum),
9         paste(column)] <- priceMat[paste(rownum-1),
10        paste(column-1)]*u
11     }
12     else {
13       priceMat[paste(rownum),
14         paste(column)] <- priceMat[paste(rownum+1),
15        paste(column-1)]*d
16     }
17   }
18 }
19 probMat <- matrix(0, ncol=nsteps+1, nrow=2*nsteps + 1,
20 dimnames=list(as.character(nsteps:-nsteps),
21 as.character(0:nsteps)))
22 probMat["0", "0"] <- 1
23 for(column in 1:nsteps) {
24   for(rownum in seq(column, -column, by=-2)) {
25     if(! (rownum == -column) ) {
26       probMat[paste(rownum),
27         paste(column)] <- probMat[paste(rownum-1),
28        paste(column-1)]*p + probMat[paste(rownum),
29        paste(column)]
30     }
31     if(! (rownum == column) ) {
32       probMat[paste(rownum),
33         paste(column)] <- probMat[paste(rownum+1),
34        paste(column-1)]*(1-p) + probMat[paste(rownum),
35        paste(column)]
36     }
37   }
38 }
11:15 | (Top Level) | R Script
```

Console ~/binomial/

```
> sigma <- 0.20
> nsteps <- 4
> source("~/binomial/binomialLattice.R")
> priceMat
      0      1      2      3      4
4 NA      NA      NA      NA 149.18247
3 NA      NA      NA 134.98588      NA
2 NA      NA 122.14028      NA 122.14028
1 NA 110.51709      NA 110.51709      NA
0 100      NA 100.00000      NA 100.00000
-1 NA 90.48374      NA 90.48374      NA
-2 NA      NA 81.87308      NA 81.87308
-3 NA      NA      NA 74.08182      NA
-4 NA      NA      NA      NA 67.03200
> plotLattice(priceMat, "Price Lattice", digits=2)
```

Files | Plots | Packages

Zoom | Export | Clear All

Price Lattice

Time Step	Step -4	Step -3	Step -2	Step -1	Step 0	Step 1	Step 2	Step 3	Step 4
0					100				
1			90.48		110.52				
2		81.87		100		122.14			
3		74.08	90.48		110.52		134.99		
4		67.03	81.87	100		122.14		149.18	

up/down steps

time steps

Workspace | History | Help

Code Completion

~/customize - RStudio

```
binomialLattice.R* | priceMat | probMat |
Source on Save | Run | Source
1 priceMat <- matrix(NA, ncol=nsteps+1, nrow=2*nsteps+1,
2 dimnames=list(as.character(nsteps:-nsteps),
3 as.character(0:nsteps)))
4 priceMat["0", "0"] <- S0
5 for(column in 1:nsteps) {
6   for(rownum in seq(column, -column, by=-2)) {
7     if(rownum=0) {
8       priceMat[paste(rownum),
9         paste(column)] <- priceMat[paste(rownum-1),
10        paste(column-1)]*u
11     }
12     else {
13       priceMat[paste(rownum),
14         paste(column)] <- priceMat[paste(rownum+1),
15        paste(column-1)]*d
16     }
17   }
18 }
19 probMat <- matrix(0, ncol=nsteps+1, nrow=2*nsteps + 1,
20 dimnames=list(as.character(nsteps:-nsteps),
21 as.character(0:nsteps)))
22 probMat["0", "0"] <- 1
23 for(column in 1:nsteps) {
24   for(rownum in seq(column, -column, by=-2)) {
25     if( !(rownum == -column) ) {
26       probMat[paste(rownum),
27         paste(column)] <- probMat[paste(rownum-1),
28        paste(column-1)]*p + probMat[paste(rownum),
29        paste(column)]
30     }
31     if( !(rownum == column) ) {
32       probMat[paste(rownum),
33         paste(column)] <- probMat[paste(rownum+1),
34        paste(column-1)]*(1-p) + probMat[paste(rownum),
35        paste(column)]
36     }
37   }
38 }
39
```

```
Console ~/binomial/
> sigma <- 0.20
> nsteps <- 4
> source("~/binomial/binomialLattice.R")
> priceMat
      0      1      2      3      4
4 NA      NA      NA      NA 149.18247
3 NA      NA      NA 134.98588      NA
2 NA      NA 122.14028      NA 122.14028
1 NA 110.51709      NA 110.51709      NA
0 100      NA 100.00000      NA 100.00000
-1 NA 90.48374      NA 90.48374      NA
-2 NA      NA 81.87308      NA 81.87308
-3 NA      NA      NA 74.08182      NA
-4 NA      NA      NA      NA 67.03200
> plotLattice(priceMat, "Price Lattice", digits=2)
|

```

Files | Plots | Packages

Zoom | Export | Clear All

Price Lattice

Time Step	Step -4	Step -3	Step -2	Step -1	Step 0	Step 1	Step 2	Step 3	Step 4
0					100				
1				90.48	100	110.52			
2			81.87	100	122.14				
3		74.08	90.48	110.52	134.99				
4	67.03	81.87	100	122.14	149.18				

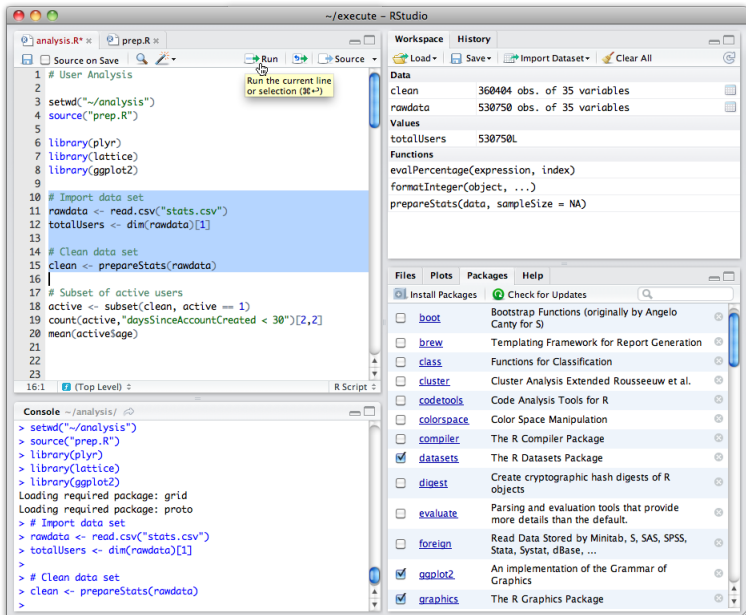
up/down steps

time steps

11:15 | (Top Level) | R Script

Workspace | History | Help

Execute From Source



The screenshot shows the RStudio interface with the following components:

- Source Editor:** Contains an R script with the following code:

```
1 # User Analysis
2
3 setwd("~/analysis")
4 source("prep.R")
5
6 library(plyr)
7 library(lattice)
8 library(ggplot2)
9
10 # Import data set
11 rawdata <- read.csv("stats.csv")
12 totalUsers <- dim(rawdata)[1]
13
14 # Clean data set
15 clean <- prepareStats(rawdata)
16
17 # Subset of active users
18 active <- subset(clean, active == 1)
19 count(active, "daysSinceAccountCreated < 30")[2,2]
20 mean(active$age)
21
22
23
```
- Run Button:** A tooltip over the 'Run' button reads: "Run the current line or selection (Ctrl+R)".
- Console:** Shows the output of the script execution:

```
> setwd("~/analysis")
> source("prep.R")
> library(plyr)
> library(lattice)
> library(ggplot2)
Loading required package: grid
Loading required package: proto
> # Import data set
> rawdata <- read.csv("stats.csv")
> totalUsers <- dim(rawdata)[1]
>
> # Clean data set
> clean <- prepareStats(rawdata)
>
```
- Environment Pane:** Shows the objects created in the workspace:

Object	Description
clean	360404 obs. of 35 variables
rawdata	530750 obs. of 35 variables
totalUsers	530750L
- Files Pane:** Shows a list of installed and available packages:

Package	Description
<input type="checkbox"/> boot	Bootstrap Functions (originally by Angelo Canty for S)
<input type="checkbox"/> brew	Templating Framework for Report Generation
<input type="checkbox"/> class	Functions for Classification
<input type="checkbox"/> cluster	Cluster Analysis Extended Rousseeuw et al.
<input type="checkbox"/> codetools	Code Analysis Tools for R
<input type="checkbox"/> colorspace	Color Space Manipulation
<input type="checkbox"/> compiler	The R Compiler Package
<input checked="" type="checkbox"/> datasets	The R Datasets Package
<input type="checkbox"/> digest	Create cryptographic hash digests of R objects
<input type="checkbox"/> evaluate	Parsing and evaluation tools that provide more details than the default.
<input type="checkbox"/> foreign	Read Data Stored by Minitab, S, SAS, SPSS, Stata, Systat, dBase, ...
<input checked="" type="checkbox"/> ggplot2	An implementation of the Grammar of Graphics
<input checked="" type="checkbox"/> graphics	The R Graphics Package

Go to File/Function

The screenshot shows the RStudio interface with the 'Go to File/Function' dialog box open. The dialog box is displaying a search for 'read.' and a list of files and functions. The code editor in the background shows the definition of the 'mean' function.

```
13 #  
14 # A copy of the GNU General Public License is available at  
15 # http://www.r-project.org/licenses/  
16  
17 mean <- function(x, ...) UseMethod("mean")  
18  
19 mean.default <- function(x, trim = 0, na.rm = TRUE, ...) {  
20 {  
21   if(!is.numeric(x) && !is.complex(x) && !is.logical(x))  
22     warning("argument is not numeric or logical: cannot  
23     return(NA_real_)"  
24   }  
25   if (na.rm) {  
26     x <- x[!is.na(x)]  
27     if(!is.numeric(trim) || length(trim) != 1) {  
28       stop("'trim' must be numeric of length 1")  
29     }  
30     n <- length(x)  
31     if(trim > 0 && n) {  
32       stop("'trimmed means are not defined for this data type")  
33       if(any(is.na(x))) return(NA_real_)  
34     }  
35     lo <- floor(n*trim)+1  
36     hi <- n+1-lo  
37     x <- sort.int(x, partial=unique(c(lo, hi)))  
38     }  
39     .Internal(mean(x))  
40 }  
41  
42 mean.data.frame <- function(x, ...) {  
43   ## cannot use .Deprecated("sapply(<data>, mean)")  
44   msg <- "mean(<data.frame>) is deprecated.\n Use colMeans"  
45   warning(paste(msg, collapse = ""), call. = FALSE, domain = "R")  
46   sapply(X = x, FUN = mean, ...)  
47 }  
48
```

The 'Go to File/Function' dialog box shows the following search results:

- read.00Index.R
- read.DIF.R
- read.fortran.R
- read.fwf.R
- read.00Index (src/library/tools/R/read.00Index.R)
- read.DIF (src/library/utills/R/read.DIF.R)
- read.csv (src/library/utills/R/readtable.R)
- read.csv2 (src/library/utills/R/readtable.R)
- read.dcf (src/library/base/R/dcf.R)
- read.delim (src/library/utills/R/readtable.R)
- read.delim2 (src/library/utills/R/readtable.R)
- read.fortran (src/library/utills/R/read.fortran.R)
- read.ftable (src/library/stats/R/ftable.R)
- read.fwf (src/library/utills/R/read.fwf.R)
- read.socket (src/library/utills/R/sock.R)
- read.table (src/library/utills/R/readtable.R)
- read.table.url (src/library/base/R/Defunct.R)

The background code editor shows the definition of the 'mean' function, which is partially obscured by the dialog box. The function 'mean' is defined with a default value of 'trim = 0' and 'na.rm = TRUE'. It uses 'UseMethod' to dispatch to the appropriate method. The 'mean.default' function handles the core logic, including handling of 'na.rm' and 'trim' parameters. The 'mean.data.frame' function is a wrapper that uses 'sapply' to apply 'mean' to each column of a data frame.

Versioning

The screenshot shows the RStudio interface with the 'Review Changes' window open. The window title is '~ / portfolioAnalysis - RStudio'. The 'Changes' tab is active, showing a list of commits for the 'master' branch. The selected commit is 'HEAD master Year-End Report - Draft' by Josh Paulson, dated 2012-01-11, with SHA 7ccae2f0. Below the commit list, the commit details for the selected commit are shown, including the SHA (923f02e5), Author (Josh Paulson), Date (2012-01-11 15:20), Subject (add call price function, move strike function), and Parent (9de73223). The diff view shows changes to the file 'stocks/bs.option.R'. The diff is color-coded: lines 4-12 are highlighted in red, indicating deletions, and lines 13-17 are highlighted in green, indicating additions. The console at the bottom is empty.

Subject	Author	Date	SHA
HEAD master Year-End Report - Draft	Josh Paulson	2012-01-11	7ccae2f0
update project notes	Josh Paulson	2012-01-11	44ad4809
add call price function, move strike function	Josh Paulson	2012-01-11	923f02e5
update .gitignore	Josh Paulson	2012-01-11	9de73223
origin/master temp function for strike price range	Josh Paulson	2012-01-11	3c4d8063

```
SHA 923f02e5
Author Josh Paulson
Date 2012-01-11 15:20
Subject add call price function, move strike function
Parent 9de73223
stocks/bs.option.R
stocks/portfolio.R
stocks/bs.option.R View file @ 923f02e5
@@ -1,15 +1,13 @@
1 1 # Black Scholes
2 2 # Option Pricing Model
3 3
4 strike <- function(data)
5 {
6     m <- dim(data)[1]
7     n <- m/2
8     callRange <- range(data[1:n,]$strike)
9     putRange <- range(data[(n+1):m,]$strike)
10    cat("Call Strike Price Range:", callRange, "\n")
11    cat("Put Strike Price Range:", putRange)
12 }
13 # price of call option
14 callprice.bs <- function(s, x, r, sigma, t.exp, t) {
15     d1 <- log(x) + (r + 0.5 * sigma^2) * t.exp + sigma * sqrt(t.exp)
16     d2 <- d1 - sigma * sqrt(t.exp)
17     call <- exp(-r * t.exp) * (x * exp(d1 * sigma * sqrt(t.exp)) - K * exp(d2 * sigma * sqrt(t.exp)))
18 }
```

Easy creation of Sweave documents

The screenshot shows the RStudio interface with a Sweave document being compiled. The main editor window displays the source code for a Sweave document named `binomialLattice.Rnw`. The code includes package loading, document title, and a function to plot a lattice. A tooltip is visible over the `Compile PDF` button, explaining its function. The console window shows the execution of `compilePdf`, which writes the document to `binomialLattice.tex` and processes code chunks with options like `term hide`.

```
2 \usepackage{amsmath}
3 \usepackage{amssymb}
4 \usepackage{epstopdf}
5 \usepackage{fullpage}
6 \usepackage[parfill]{parskip}
7 \usepackage{graphicx}
8
9 \title{Multiplicative Binomial Lattice}
10
11 \begin{document}
12
13 % Code Chunk 1
14 <<Code Chunk 1, fig=FALSE, include=FALSE, echo=FALSE,
15 | results=hide >>
16
17 # function to plot lattice
18 plotLattice <- function(mat,title="Price Lattice",digits=2)
19 | {
20 |   nstps <- ncol(mat) - 1
21 |   plot(NA,type="n",
22 |       xlim=c(0,nstps),
23 |
24 |       (mat[paste(rownum), paste(colnum)],digits),
25 |       pos=3,cex=0.75,col="darkgreen")
26 |       points(x=colnum,y=rownum,pch=18)
27 |   }
28 | }
29 | title(title)
30 | }
31 > compilePdf("~/binomial/binomialLattice.Rnw")
Writing to file binomialLattice.tex
Processing code chunks with options ...
1 : term hide (label = Code Chunk 1)
2 : term hide (label = Code Chunk 2)
3 : echo term hide (label = Code Chunk 3)
```

Workspace

Data

- priceMat 9x5 double matrix
- probMat 9x5 double matrix

Functions

- plotLattice(mat, title = "Price Lattice", digits = 2)

Files | **Plots** | **Packages** | **Help**

- New Folder
- Delete
- Rename
- More

Home | binomial

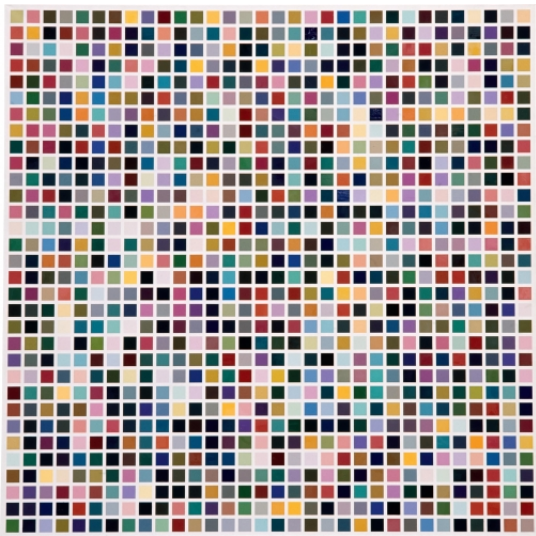
Name	Size	Modified
binomialLattice.pdf	237.3 KB	Jan 5, 2011, 2:27 PM
binomialLattice.Rnw	15.9 KB	Jan 5, 2011, 2:27 PM
binomialLattice.tex	10.6 KB	Jan 5, 2011, 2:27 PM
priceMatAprox1.png	27.1 KB	Jan 5, 2011, 2:27 PM
priceMatAprox2.png	26.2 KB	Jan 5, 2011, 2:27 PM
probabilityMatAprox1.png	27.6 KB	Jan 5, 2011, 2:27 PM
probabilityMatAprox2.png	28 KB	Jan 5, 2011, 2:27 PM

Interactive graphics via the manipulate package

The screenshot displays the RStudio interface with the following components:

- Source Editor:** Contains R code for creating an interactive plot using the `manipulate` function. The code defines a plot of 'Speed and Stopping Distances of Cars' with various controls for color, type, pch, axes, and labels.
- Console:** Shows the execution of the code, with the plot's title and axis labels being printed.
- Manipulate Panel:** A floating window that allows users to interactively change plot parameters. It includes:
 - A dropdown menu for 'color' set to 'blue'.
 - A dropdown menu for 'type' set to 'b'.
 - A slider for 'pch' set to 19.
 - Two checked checkboxes: 'Draw Axes' and 'Draw Labels'.
- Workspace/Plots Panel:** Displays the title 'Speed and Stopping Distances of Cars' and a scatter plot with blue points and lines. The y-axis is labeled 'Stopping Distance (feet)' and the x-axis is 'Car Speed (mph)'. The plot shows a positive correlation between speed and stopping distance.
- Documentation Panel:** Provides information about the `manipulate` function, including a description of its arguments and a search bar.

An example



RStudio is good for you!

Thank you for your attention!