

Introduction to R

Anindya Bhattacharya, PhD | abhattacha3@ucla.edu
CBI fellow | Xia Yang lab

Workshop 3: Introduction to R

▶ Day 2

▶ Statistical methods

- ▶ Probability distributions
- ▶ Random number generation
- ▶ Common statistical tests
 - Fisher's exact test, chi square, qvalue, etc...

▶ Visualization

- ▶ Plotting functions
- ▶ Customizing plots
- ▶ Saving plots for publication
- ▶ IGV
- ▶ CummeRbund



Statistics

- ▶ Many distributions and tests built in
- ▶ Others added as packages
- ▶ Interpreting output, names, pvalues



Basic stat functions

- ▶ `summary()`
- ▶ `var()`
- ▶ `sd()`
- ▶ `mean()`
- ▶ `median()`



Probability distributions

- ▶ For every statistical distribution in R, these functions are available (Four fundamental items for a statistical distribution)
 - ▶ Cumulative distribution function
 - ▶ Probability of “hitting” x or less in a distribution.
`>pnorm(160, mean=132, sd=13)`
`[1] 0.9843739`
`dnorm(0)`
 - ▶ Density or point probability
 - ▶ relative probability of getting a value close to x
`>x<-seq(-4,4,0.1)`
`>plot(x, dnorm(x), type="l")`
`#letter l=line curve`
 - ▶ Quantiles
 - ▶ Opposite to Cumulative distribution
`qnorm(0.01)`
 - ▶ Pseudo-random numbers
 - ▶ #10 random number
`rnorm(10)`
`rnorm(10,mean=7, sd=5)`



Example distributions

- ▶ Gaussian or normal
 - ▶ `dnorm`, `pnorm`, `qnorm`, and `rnorm`
- ▶ binomial
- ▶ poisson
- ▶ gamma
- ▶ beta
- ▶ t
- ▶ chi-square



Common statistical tests

- ▶ `cor.test()`
- ▶ `t.test()`
- ▶ `fisher.test()`
- ▶ `chisq.test()`
- ▶ `wilcox.test()`
- ▶ `ks.test()`
- ▶ `p.adjust()`
- ▶ regression: `lm(y ~ x)`
- ▶ anova: `aov()`



Try it out: ALL data

- ▶ `x = read.table("ALL_subset.txt", header=T)`
- ▶ Compare two samples by `t.test`
- ▶ `t.test(x[,1], x[,2])`
- ▶ Compute the correlation between two samples
- ▶ `cor.test(x[,1], x[,2])`
- ▶ Compute all pairwise correlations, returns matrix
- ▶ `cor(x)`



A chisquare example

```
> mat = matrix( c(1463, 48486, 7892, 334758), nrow=2, ncol=2)
```

```
> mat
```

```
      [,1] [,2]
```

```
[1,]  1463   7892
```

```
[2,] 48486 334758
```

```
> chisq.test(mat)
```

```
      Pearson's Chi-squared test with Yates' continuity  
correction
```

```
data:  mat
```

```
X-squared = 73.1195, df = 1, p-value < 2.2e-16
```

```
# Critical value. Quantile function from Chi-square distribution
```

```
>qchisq(0.95,1)
```



Simple Regression

```
> data(iris)
> head(iris)
> # attach(iris) - did not run
> names(iris)
[1] "Sepal.Length" "Sepal.Width"  "Petal.Length" "Petal.Width"
"Species"
> lm(iris$Sepal.Length ~ iris$Petal.Length)
```

Call:

```
lm(formula = Sepal.Length ~ Petal.Length)
```

Coefficients:

```
(Intercept)    Petal.Length
      4.3066         0.4089
# Sepal.Length = 4.3066 + 0.4089 * Petal.Length
```



Simple Regression

```
> a = lm(iris$Sepal.Length~ iris$Petal.Length)
> names(a)
[1] "coefficients" "residuals"      "effects"        "rank"           "fitted.values" "assign"         "qr"
"df.residual"   "xlevels"
[10] "call"         "terms"        "model"
> summary(a)
```

Call:

```
lm(formula = Sepal.Length ~ Petal.Length)
```

Residuals:

Min	1Q	Median	3Q	Max
-1.24675	-0.29657	-0.01515	0.27676	1.00269

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	4.30660	0.07839	54.94	<2e-16 ***
Petal.Length	0.40892	0.01889	21.65	<2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4071 on 148 degrees of freedom

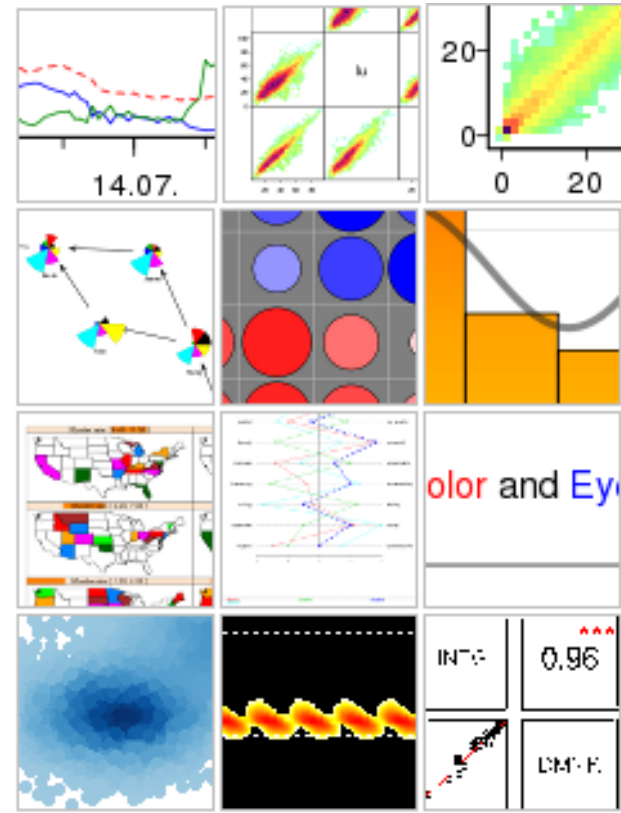
Multiple R-squared: 0.76, Adjusted R-squared: 0.7583

F-statistic: 468.6 on 1 and 148 DF, p-value: < 2.2e-16



Plotting

- ▶ A great strength of R is visualization
- ▶ Many type of plots:
 - ▶ line
 - ▶ histogram
 - ▶ scatter plot
 - ▶ topographical
- ▶ A great display of R graphs at <http://gallery.r-enthusiasts.com/>

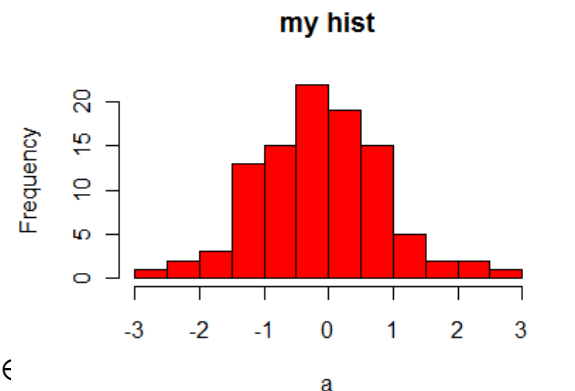


Histogram

► Very useful!

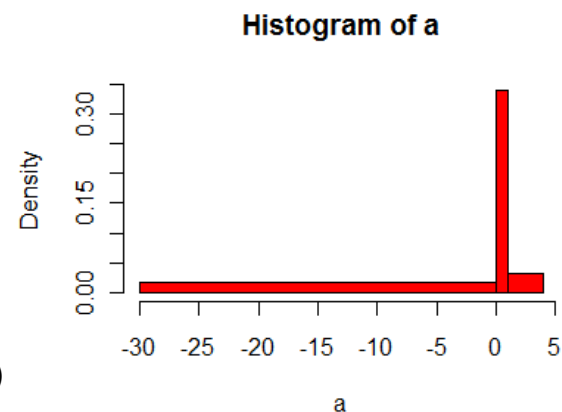
```
> a = rnorm(100)
> hist(a)
> hist(a, breaks=10, col="red", main="my hist")
```

```
hist(a, breaks=c(-30, 0, 1, 4), col="red")
```



► breaks can be customized

```
> br = c(-30, 0, 1, 4)
> br
[1] -30    0    1    4
> res = hist(a, breaks=br, col="red")
```



Histogram

- ▶ **returns an object of class "histogram"**

```
> result.a = hist(a)
```

```
> class(result.a)
```

```
[1] "histogram"
```

```
> names(result.a)
```

```
[1] "breaks"      "counts"      "intensities" "density"  
"mids"
```

```
[6] "xname"      "equidist"
```



Histogram

```
> result.a
$breaks
[1] -3 -2 -1  0  1  2  3
$counts
[1]  3 16 37 34  7  3
$intensities
[1] 0.03 0.16 0.37 0.34 0.07 0.03
$density
[1] 0.03 0.16 0.37 0.34 0.07 0.03
$mids
[1] -2.5 -1.5 -0.5  0.5  1.5  2.5
$xname
[1] "a"
$equidist
[1] TRUE
attr(,"class")
[1] "histogram"
>
```



Histogram

- ▶ `> help(hist)`
- ▶ `hist(x, breaks = "Sturges",
 freq = NULL,
 probability = !freq,
 include.lowest = TRUE,
 right = TRUE,
 density = NULL,
 angle = 45,
 col = NULL,
 border = NULL,
 main = paste("Histogram of" , xname),
 xlim = range(breaks),
 ylim = NULL,
 xlab = xname, ylab, axes = TRUE,
 plot = TRUE,
 labels = FALSE,
 nclass = NULL,
 warn.unused = TRUE, ...)`



Try it out

```
#Read in the ALL dataset
> x = read.table("ALL_subset.txt")
# how many expts are there? how many probes?
> dim(all)

# plot histogram
> hist(x[,1], xlim=c(0,25), xlab = "tryout workshop2")
> hist(x[,2], xlim=c(0,25), col="blue", add=T)
> hist(x[,3], xlim=c(0,25), col="red", add=T) # add=T: overlap the
plot.

# box plot of all expts; three expts
> boxplot(x)
➤ boxplot(x[, c(1,12,3)])
```



Plotting

▶ plots

- ▶ histogram
- ▶ boxplot
- ▶ barplot
- ▶ scatter plot
- ▶ heatmap
- ▶ cluster
- ▶ points, lines, segments
- ▶ abline
- ▶ qqplot
- ▶ & more

▶ customizations

- ▶ axes
- ▶ header text/title
- ▶ margin text
- ▶ identify
- ▶ display symbols
- ▶ layout
- ▶ colors



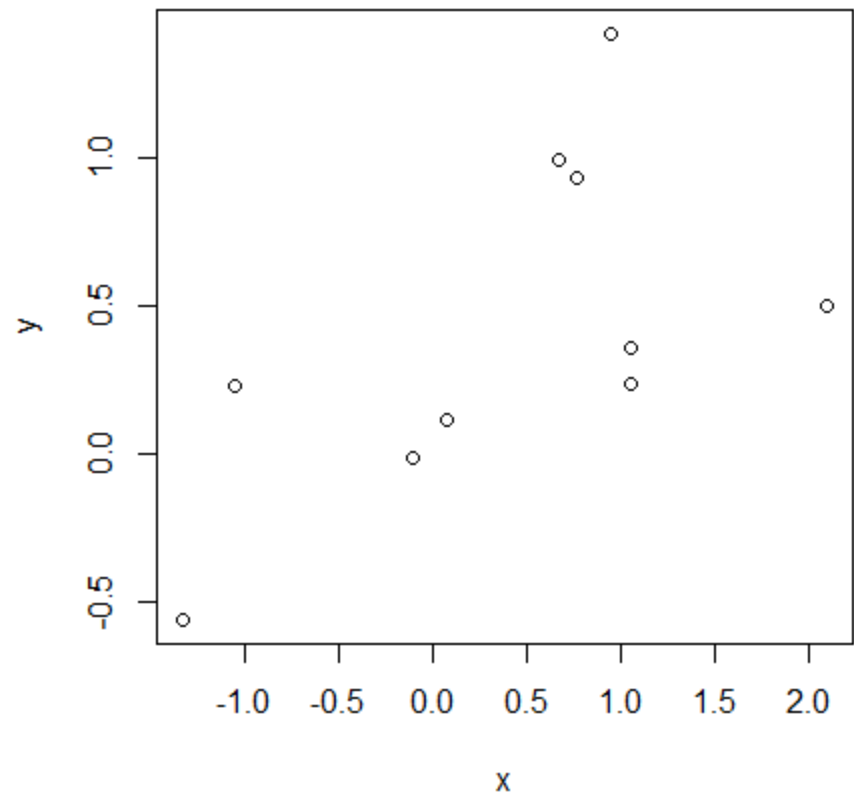
A plain scatter plot

```
x = rnorm(10)
```

```
y = rnorm(10)
```

```
plot(x, y)
```

```
plot(x, y, pch="&")
```



Graphical parameters

adj	controls text justification with respect to the left border of the text so that 0 is left-justified, 0.5 is centred, 1 is right-justified, values > 1 move the text further to the left, and negative values further to the right; if two values are given (e.g., <code>c(0, 0)</code>) the second one controls vertical justification with respect to the text baseline
bg	specifies the colour of the background (e.g., <code>bg="red"</code> , <code>bg="blue"</code> ; the list of the 657 available colours is displayed with <code>colors()</code>)
bty	controls the type of box drawn around the plot, allowed values are: "o", "l", "7", "c", "u" ou "]" (the box looks like the corresponding character); if <code>bty="n"</code> the box is not drawn
cex	a value controlling the size of texts and symbols with respect to the default; the following parameters have the same control for numbers on the axes, <code>cex.axis</code> , the axis labels, <code>cex.lab</code> , the title, <code>cex.main</code> , and the sub-title, <code>cex.sub</code>
col	controls the colour of symbols; as for <code>cex</code> there are: <code>col.axis</code> , <code>col.lab</code> , <code>col.main</code> , <code>col.sub</code>
font	an integer which controls the style of text (1: normal, 2: italics, 3: bold, 4: bold italics); as for <code>cex</code> there are: <code>font.axis</code> , <code>font.lab</code> , <code>font.main</code> , <code>font.sub</code>
las	an integer which controls the orientation of the axis labels (0: parallel to the axes, 1: horizontal, 2: perpendicular to the axes, 3: vertical)
lty	controls the type of lines, can be an integer (1: solid, 2: dashed, 3: dotted, 4: dotdash, 5: longdash, 6: twodash), or a string of up to eight characters (between "0" and "9") which specifies alternatively the length, in points or pixels, of the drawn elements and the blanks, for example <code>lty="44"</code> will have the same effet than <code>lty=2</code>
lwd	a numeric which controls the width of lines
mar	a vector of 4 numeric values which control the space between the axes and the border of the graph of the form <code>c(bottom, left, top, right)</code> , the default values are <code>c(5.1, 4.1, 4.1, 2.1)</code>
mfc	a vector of the form <code>c(nr,nc)</code> which partitions the graphic window as a matrix of <code>nr</code> lines and <code>nc</code> columns, the plots are then drawn in columns (see section 4.1.2)
mfrow	id. but the plots are then drawn in line (see section 4.1.2)
pch	controls the type of symbol, either an integer between 1 and 25, or any single character within "" (Fig. 2)
ps	an integer which controls the size in points of texts and symbols

pty	a character which specifies the type of the plotting region, "s": square, "m": maximal
tck	a value which specifies the length of tick-marks on the axes as a fraction of the smallest of the width or height of the plot; if <code>tck=1</code> a grid is drawn
tcl	id. but as a fraction of the height of a line of text (by default <code>tcl=-0.5</code>)
xaxt	if <code>xaxt="n"</code> the <i>x</i> -axis is set but not drawn (useful in conjunction with <code>axis(side=1, ...)</code>)
yaxt	if <code>yaxt="n"</code> the <i>y</i> -axis is set but not drawn (useful in conjunction with <code>axis(side=2, ...)</code>)

R for Beginners
E. Paradis

Printing character

- ▶ you can specify the pch parameter
 - ▶ a number (pch=1 gives a circle)
 - ▶ a text character
(pch="v" uses the letter "v")

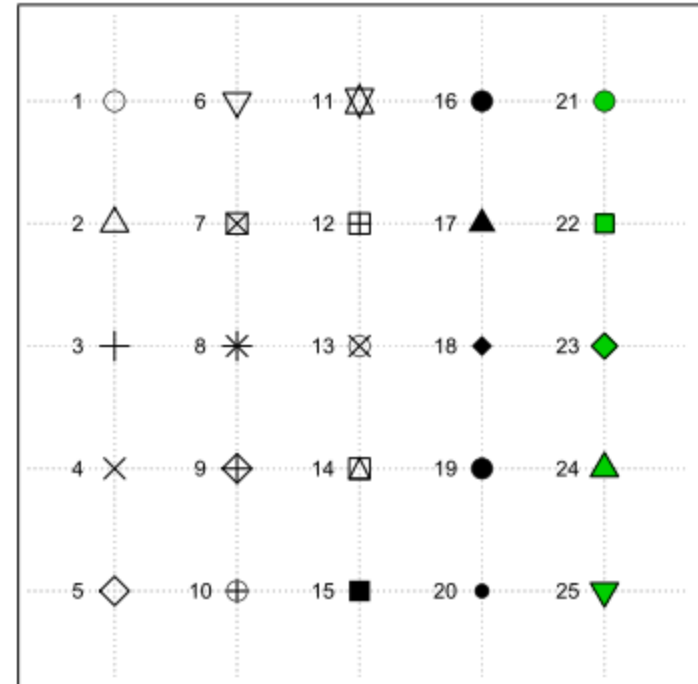


Figure 5: Plotting symbols

Try it out

```
#Read in the ALL dataset
> x = read.table("ALL_subset.txt")
# how many expts are there? how many probes?
> dim(all)
# plot the first three expts against the first on the same page
> par(mfrow=c(1,3)); # specify the space for three plot.
> plot(x[,1], x[,2], pch=8);
> plot(x[,1], x[,3]);
> plot(x[,1], x[,4], col="red", pch=4)
#close the window
> dev.off()
# plot one graph again, what's different?
plot(x[,1], x[,2])
# change some plotting options
> plot(x[,1], x[,2], pch=".", col="blue", xlab="expt1", ylab="expt2")
# add a title with the correlation in it
> x.cor = cor.test(x[,1], x[,2])
> title(paste("Expt 1 versus Expt 2\n cor=", round(x.cor$estimate, 2))) # 2\n mean new line.
# log transform
> plot(log2(x[,1]), log2(x[,2]), pch=".", col="blue", xlab="expt1", ylab="expt2", main="Expt1 versus Expt2")
> X=1:10
> Y=11:20
> Plot(x,y)
> Plot(x,y, type="l")
> Plot(x,y, type=" b")
> Plot(x,y, type="h")
```



Scatter plots, line plots

- ▶ the `type` parameter controls the scatter plot type
 - ▶ options include `points`, `lines`, `points+lines` and `histogram`

```
Id=sample(1:100, 25)
```

```
> somedata = rnorm(25)
> plot(somedata, type="p")
> plot(somedata, type="l")
> plot(somedata, type="b")
> plot(somedata, type="h")
▶ x1=rnorm(100000)
▶ y1=rnorm(100000)
▶ x1=rnorm(100000, mean=2, sd=1.5)
▶ x2=rnorm(100000, mean=2, sd=1.5)
▶ y2=rnorm(100000, mean=2, sd=1.5)
▶ x1=rnorm(100000)
▶ Xy=rbind(cbind(x1,y1), cbind(x2,y2))
▶ smoothScatter(xy)
```



A few tips

- ▶ If you have many points (100-1000's), the plot may become obscured
- ▶ Potential remedies:
 - ▶ downsample your data: suppose you have 1000 rows of data in a data frame, you can pick some indices to display

```
# this will pick 25 numbers between 1 and 100  
ids = sample(1:100, 25)
```

- ▶ try a smaller plotting symbol: `pch="."` will use a dot
- ```
plot(x[ids, 1], x[ids, 2], pch=".")
```





---

## ► smoothScatter

```
x1 = rnorm(100000)
y1 = rnorm(100000)
x2 = rnorm(100000, mean=2, sd=1.5)
y2 = rnorm(100000, mean=3, sd=1.5)
xy = rbind(cbind(x1, y1), cbind(x2, y2))
smoothScatter(xy)
```

## ► Alpha values

```
colors are given as amount of
Red, Green, Blue and Alpha with range [0,1].
you can specify maxColorValue=255 if you want to use [0,255]
plot(x,y, col=rgb(0,0.5,0,0.1))
```



- 
- ▶ `lines()`
  - ▶ `abline(lm (y ~x))`
  - ▶ `points()`
  - ▶ `legend()`
  - ▶ `qqnorm`
  - ▶ `identify()`
  - ▶ `barplot()`, stacked barplot
  - ▶ `pie()`
  - ▶ `boxplot()`



# Adding additional points to a plot

---

- ▶ **points()** lets you add points to an existing plot

```
x = rnorm(10); y=rnorm(10)
```

```
xx = rnorm(10); yy = rnorm(10)
```

```
plot(x, y)
```

```
points(xx, yy, col="green", pch="v")
```

- ▶ if nothing shows up, your axes might not contain the data area



# More plots

---

## ► Hierarchical clustering

```
all = read.table("ALL_subset.txt")
hier1 = hclust(dist(1-cor(all)))
plot(hier1)
```

## ► Heatmap

```
heatmap(cor(all), symm=T)
```



# Multiple plots per page

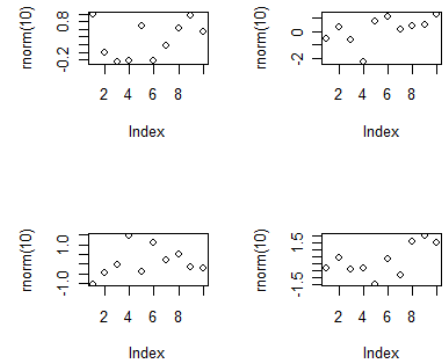
---

## ► Use par

```
par(mfrow=c(2,1))
```

## ► Another method is to designate matrix of slots

```
layout(matrix(c(1,2,3,4), ncol=2, nrow=2))
for(i in 1:4){
 plot(rnorm(10))
}
compare with
layout(matrix(c(1,1,2,3), 2, 2))
```



## ► Once you close a window, settings disappear



# Graphics devices

---

- ▶ R will "write" plots to a graphics device
- ▶ If you don't have one, for example you ssh into a remote machine, it will write to a file called Rplots.pdf in your current directory.

# **show your current graphic device**

```
dev.cur()
```

# show all devices

```
dev.list()
```

# select a device (from already open)

```
dev.set()
```

# ***create a new window***

```
x11()
```

---



# Output formats

---

- ▶ `pdf()`, `png()`, `bmp()`, `tiff()`, `jpeg()`, formats all supported
- ▶ `png(filename="file.png", width=800, height=800)`
- ▶ `dev.cur()`, `dev.off()`, `dev.set()`
  
- ▶ To create a graphic, think of it as writing an image to a file

```
pdf(file="foo.pdf")
```

```
plot(rnorm(100))
```

```
dev.off()
```

- ▶ Don't forget to `dev.off()` your device, otherwise your figure will get corrupted!

- ▶ pl



# Fancier graphics

---

- ▶ There are more sophisticated plotting paradigms for multipanel, multidimensional data
  - ▶ lattice
  - ▶ trellis
  - ▶ ggplot





# Extending R

---

- ▶ Many distributions and tests built in
- ▶ Others added as packages
  - ▶ available from CRAN
  - ▶ bioconductor is specialized for life sciences
- ▶ Packages essentially add new functionality to R



# Installing package from CRAN

---


- ▶ installing packages is usually easy
- ▶ biggest problems are due to incompatibility of versions of R with versions of a package
  - ▶ eg MASS ver 2 requires R version  $\geq 2.13$
- ▶ source code is always available
- ▶ `install.packages()`
- ▶ Where do files get stored?

```
> .libPaths()
```

```
[1] "/Library/Frameworks/R.framework/Resources/library"
```



# Finding packages



The screenshot shows the CRAN website with the following content:

- Contributed Packages**
- Available Packages**

Currently, the CRAN package repository features 4076 available packages.

[Table of available packages, sorted by date of publication](#)

[Table of available packages, sorted by name](#)
- Installation of Packages**

Please type `help("INSTALL")` or `help("install.packages")` in R for information on how to install packages from this repository. The manual [R Installation and Administration \[PDF\]](#) (also contained in the R base sources) explains the process in detail.

[CRAN Task Views](#) allow you to browse packages by topic and provide tools to automatically install all packages for special areas of interest. Currently, 29 views are available.
- Package Check Results**

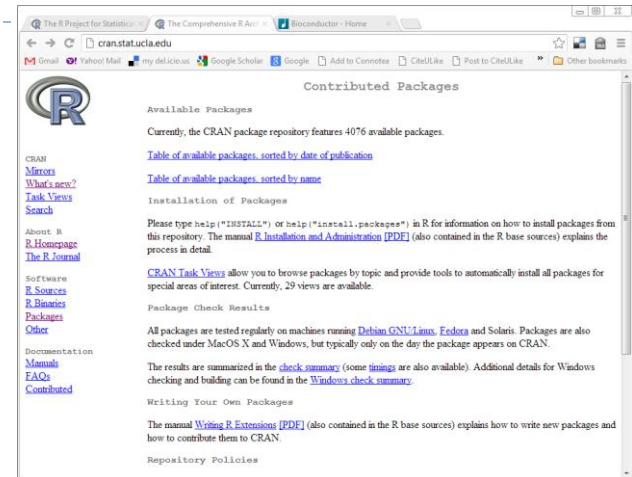
All packages are tested regularly on machines running [Debian GNU/Linux](#), [Fedora](#) and Solaris. Packages are also checked under MacOS X and Windows, but typically only on the day the package appears on CRAN.

The results are summarized in the [check summary](#) (some [timings](#) are also available). Additional details for Windows checking and building can be found in the [Windows check summary](#).
- Writing Your Own Packages**

The manual [Writing R Extensions \[PDF\]](#) (also contained in the R base sources) explains how to write new packages and how to contribute them to CRAN.
- Repository Policies**

**Left Sidebar:**

- CRAN
- [Mirrors](#)
- [What's new?](#)
- [Task Views](#)
- [Search](#)
- About R
- [R Homepage](#)
- [The R Journal](#)
- Software
- [R Sources](#)
- [R Binaries Packages](#) (circled in red)
- [Other](#)
- Documentation
- [Manuals](#)
- [FAQs](#)
- [Contributed](#)



The screenshot shows the CRAN website with the following content:

- Contributed Packages**
- Available Packages**

Currently, the CRAN package repository features 4076 available packages.

[Table of available packages, sorted by date of publication](#)

[Table of available packages, sorted by name](#)
- Installation of Packages**

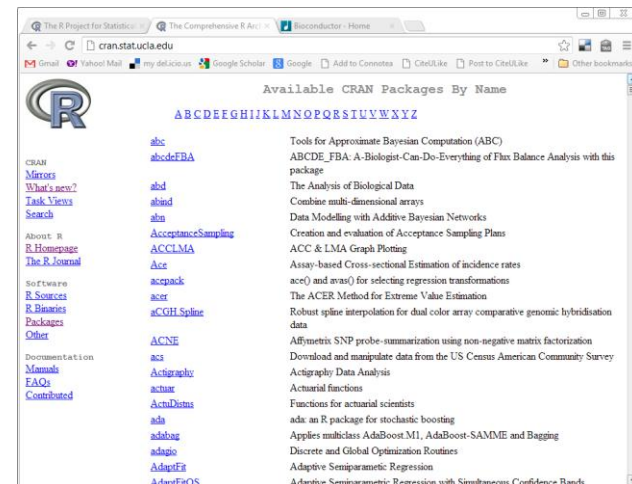
Please type `help("INSTALL")` or `help("install.packages")` in R for information on how to install packages from this repository. The manual [R Installation and Administration \[PDF\]](#) (also contained in the R base sources) explains the process in detail.

[CRAN Task Views](#) allow you to browse packages by topic and provide tools to automatically install all packages for special areas of interest. Currently, 29 views are available.
- Package Check Results**

All packages are tested regularly on machines running [Debian GNU/Linux](#), [Fedora](#) and Solaris. Packages are also checked under MacOS X and Windows, but typically only on the day the package appears on CRAN.

The results are summarized in the [check summary](#) (some [timings](#) are also available). Additional details for Windows checking and building can be found in the [Windows check summary](#).
- Writing Your Own Packages**

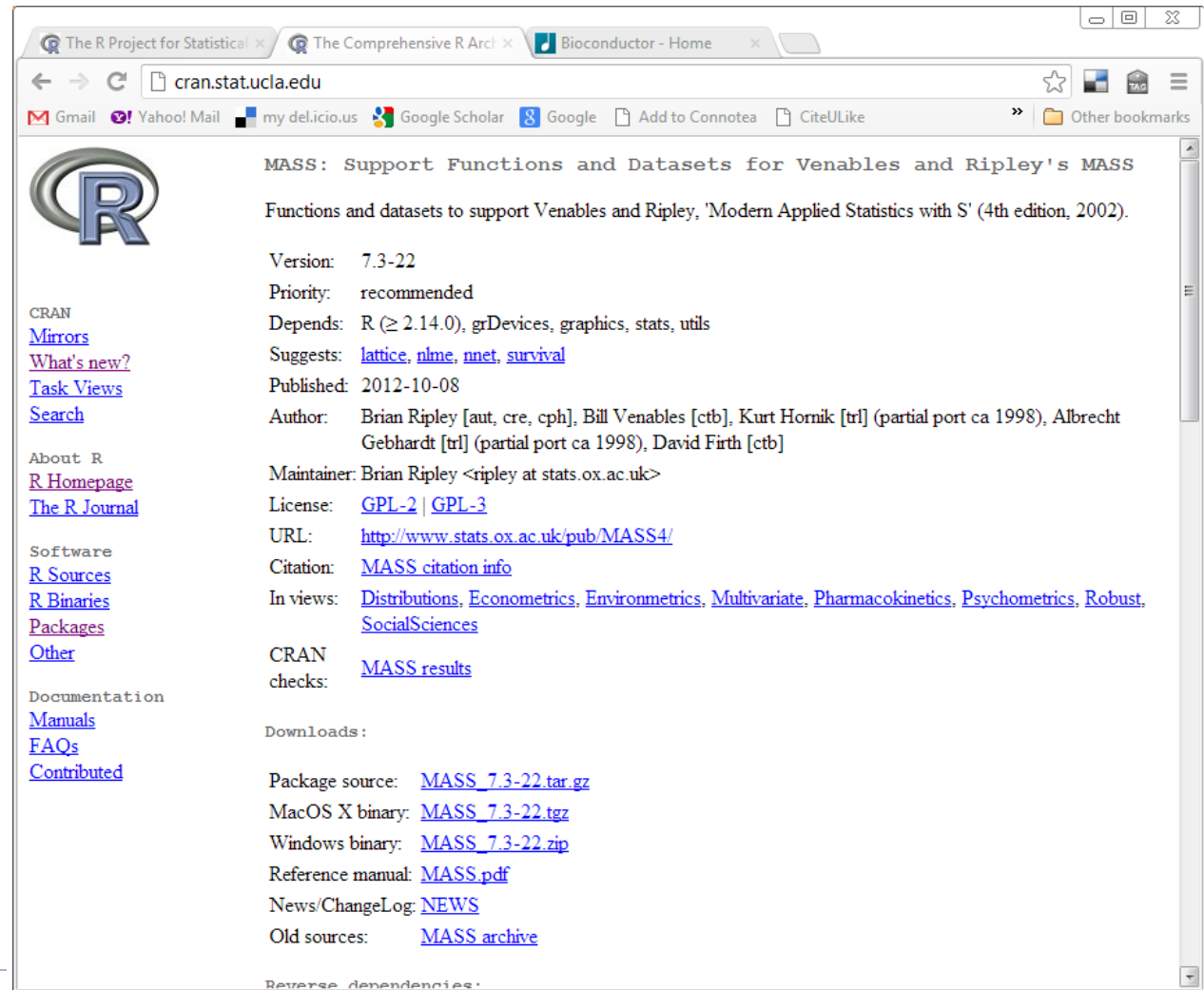
The manual [Writing R Extensions \[PDF\]](#) (also contained in the R base sources) explains how to write new packages and how to contribute them to CRAN.
- Repository Policies**



The screenshot shows the CRAN website with the following content:

- Available CRAN Packages By Name**
- ABCDEF GHIJKLMNOPQRSTUVWXYZ**
- abc**
  - [abceFBA](#)
- abd**
  - [abind](#)
  - [abn](#)
- abnd**
  - [AcceptanceSampling](#)
  - [ACCLMA](#)
  - [Ace](#)
- acp**
  - [acpack](#)
  - [acer](#)
  - [acGH.Spline](#)
- acne**
  - [ACNE](#)
  - [ac2](#)
  - [Ac3graph](#)
  - [actuar](#)
  - [AcadDists](#)
  - [ada](#)
  - [adaboos](#)
  - [adagio](#)
  - [AdaptFit](#)
  - [AdaptFitOS](#)

# Finding packages



The screenshot shows a web browser window with the address bar displaying `cran.stat.ucla.edu`. The page title is "MASS: Support Functions and Datasets for Venables and Ripley's MASS". The page content includes a sidebar with links to CRAN resources, a main section with package details, and a bottom section for downloads and reverse dependencies.

**CRAN**  
[Mirrors](#)  
[What's new?](#)  
[Task Views](#)  
[Search](#)

**About R**  
[R Homepage](#)  
[The R Journal](#)

**Software**  
[R Sources](#)  
[R Binaries](#)  
[Packages](#)  
[Other](#)

**Documentation**  
[Manuals](#)  
[FAQs](#)  
[Contributed](#)

**MASS: Support Functions and Datasets for Venables and Ripley's MASS**  
Functions and datasets to support Venables and Ripley, 'Modern Applied Statistics with S' (4th edition, 2002).

Version: 7.3-22  
Priority: recommended  
Depends: R ( $\geq 2.14.0$ ), grDevices, graphics, stats, utils  
Suggests: [lattice](#), [nlme](#), [nnet](#), [survival](#)  
Published: 2012-10-08  
Author: Brian Ripley [aut, cre, cph], Bill Venables [ctb], Kurt Hornik [trl] (partial port ca 1998), Albrecht Gebhardt [trl] (partial port ca 1998), David Firth [ctb]  
Maintainer: Brian Ripley <[ripley@stats.ox.ac.uk](mailto:ripley@stats.ox.ac.uk)>  
License: [GPL-2](#) | [GPL-3](#)  
URL: <http://www.stats.ox.ac.uk/pub/MASS4/>  
Citation: [MASS citation info](#)  
In views: [Distributions](#), [Econometrics](#), [Environmetrics](#), [Multivariate](#), [Pharmacokinetics](#), [Psychometrics](#), [Robust](#), [SocialSciences](#)  
CRAN checks: [MASS results](#)

Downloads:

Package source: [MASS\\_7.3-22.tar.gz](#)  
MacOS X binary: [MASS\\_7.3-22.tgz](#)  
Windows binary: [MASS\\_7.3-22.zip](#)  
Reference manual: [MASS.pdf](#)  
News/ChangeLog: [NEWS](#)  
Old sources: [MASS archive](#)

Reverse dependencies:

# Using the GUI

---

- ▶ Use "Package" menu
- ▶ select a mirror (aka geographically close repository)
- ▶ Repo should include CRAN
- ▶ Install packages allows you to select
- ▶ Update packages allows you to select which packages to update



# Troubleshooting install problems

---

- ▶ Usually a dependency is missing
  - ▶ missing dev package
  - ▶ out of date package
  - ▶ sometimes R needs access to a C or Fortran compiler
  - ▶ need administrator access



# Installing from bioconductor

---


- ▶ set of packages tailored to life sciences
- ▶ requires its own installation method but relies on many tools from CRAN
- ▶ navigating the bioconductor website
- ▶ `source("http://bioconductor.org/biocLite.R")`
- ▶ example datasets



The R Project for Statistical  
The Comprehensive R Arch  
Bioconductor - Home

www.bioconductor.org

Google Scholar Google Add to Connotea CiteULike Post to CiteULike Other bookmarks


**Bioconductor**  
OPEN SOURCE SOFTWARE FOR BIOINFORMATICS

Search:

Home Install Help Developers About

## About Bioconductor

Bioconductor provides tools for the analysis and comprehension of high-throughput genomic data. Bioconductor uses the R statistical programming language, and is open source and open development. It has two releases each year, [6.10 software packages](#), and an active user community. Bioconductor is also available as an [Amazon Machine Image \(AMI\)](#).



## Use Bioconductor for...


➔ **Microarrays**  
Import Affymetrix, Illumina, Nimblegen, Agilent, and other platforms. Perform quality assessment, normalization, differential expression, clustering, classification, gene set enrichment, genetical genomics and other workflows for expression, exon, copy number, SNP, methylation and other assays. Access GEO, ArrayExpress, Biomart, UCSC, and other community resources.


➔ **Variants**  
Read and write VCF files. Identify structural location of variants and compute amino acid coding changes for non-synonymous variants. Use SIFT and PolyPhen database packages to predict consequence of amino acid coding changes.


➔ **Sequence Data**  
Import fasta, fastq, ELAND, MAQ, BWA, Bowtie, BAM, gff, bed, wig, and other sequence formats. Trim, transform, align, and manipulate sequences. Perform quality assessment, ChIP-seq, differential expression, RNA-seq, and other workflows. Access the Sequence Read Archive.


➔ **Annotation**  
Use microarray probe, gene, pathway, gene ontology, homology and other annotations. Access GO, KEGG, NCBI, Biomart, UCSC, vendor, and other sources.

➔ **High Throughput Assays**  
Import, transform, edit, analyze and visualize flow cytometric, mass spec, HTqPCR, cell-based, and other assays.

 [Mailing Lists](#)

 Subscribe »

 [Events](#)

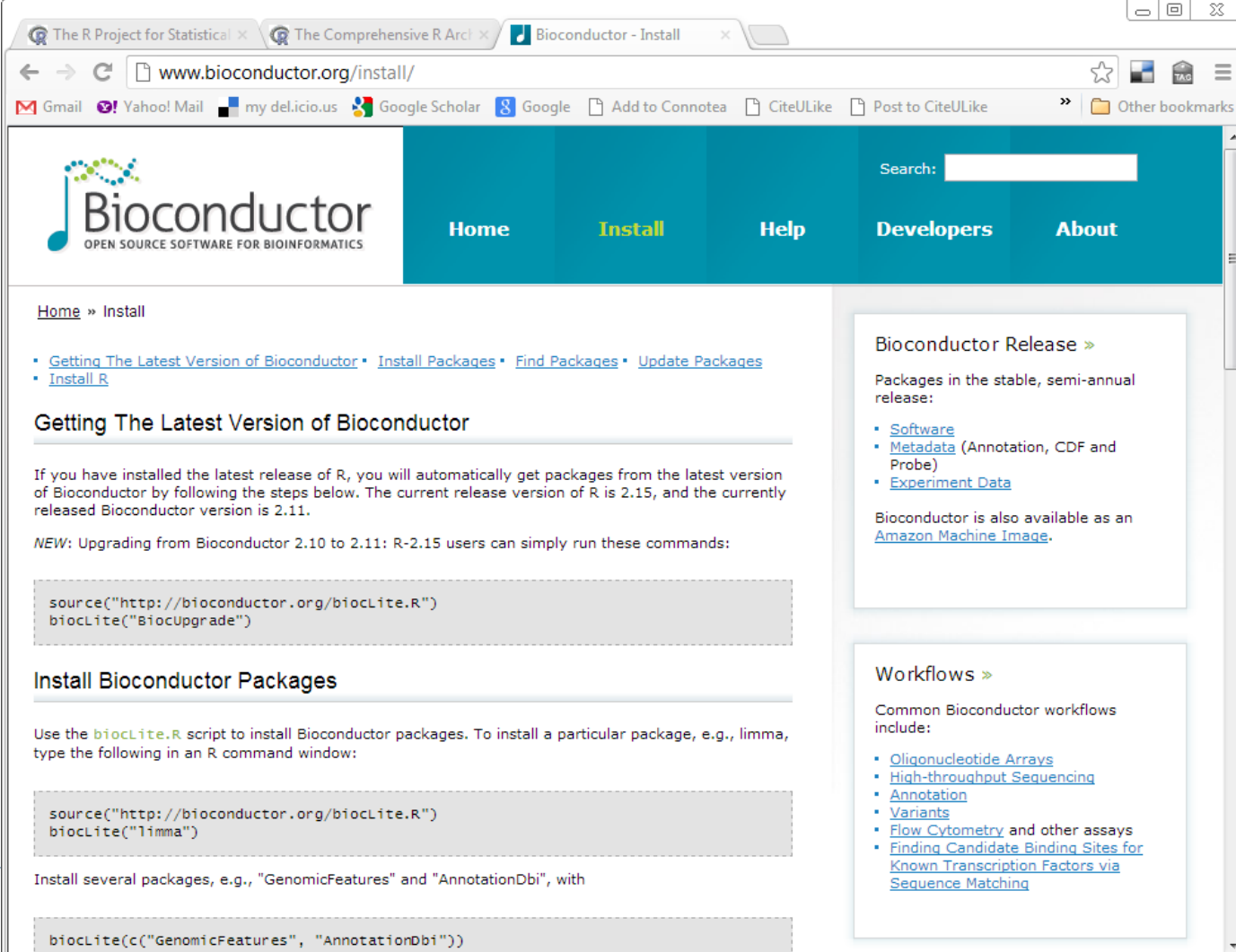
 [News](#)



C:\Users\KGong\AppData\Local\Temp\RtmpcIxmR\downloaded\_packages

## ► nloaded binary packages are in

►



The screenshot shows the Bioconductor website's installation page. The browser tabs include 'The R Project for Statistical Computing', 'The Comprehensive R Archive Network', and 'Bioconductor - Install'. The address bar shows 'www.bioconductor.org/install/'. The page has a teal header with the Bioconductor logo and navigation links: Home, Install, Help, Developers, and About. The main content area is titled 'Home » Install' and includes a list of links: 'Getting The Latest Version of Bioconductor', 'Install Packages', 'Find Packages', 'Update Packages', and 'Install R'. Below this is a section 'Getting The Latest Version of Bioconductor' with a paragraph explaining that the latest release of R (2.15) will automatically get the latest version of Bioconductor (2.11). It also provides a command to upgrade from Bioconductor 2.10 to 2.11: `source("http://bioconductor.org/biocLite.R")` and `biocLite("BiocUpgrade")`. The next section is 'Install Bioconductor Packages', which explains that the `biocLite.R` script is used to install packages. It provides a command to install the 'limma' package: `source("http://bioconductor.org/biocLite.R")` and `biocLite("limma")`. Finally, it shows a command to install multiple packages: `biocLite(c("GenomicFeatures", "AnnotationDbi"))`. On the right side, there are two sidebars. The first is 'Bioconductor Release' with links for 'Software', 'Metadata' (Annotation, CDF and Probe), and 'Experiment Data'. The second is 'Workflows' with links for 'Oligonucleotide Arrays', 'High-throughput Sequencing', 'Annotation', 'Variants', 'Flow Cytometry and other assays', and 'Finding Candidate Binding Sites for Known Transcription Factors via Sequence Matching'.

Home » Install

- [Getting The Latest Version of Bioconductor](#)
- [Install Packages](#)
- [Find Packages](#)
- [Update Packages](#)
- [Install R](#)

### Getting The Latest Version of Bioconductor

If you have installed the latest release of R, you will automatically get packages from the latest version of Bioconductor by following the steps below. The current release version of R is 2.15, and the currently released Bioconductor version is 2.11.

**NEW:** Upgrading from Bioconductor 2.10 to 2.11: R-2.15 users can simply run these commands:

```
source("http://bioconductor.org/biocLite.R")
biocLite("BiocUpgrade")
```

### Install Bioconductor Packages

Use the `biocLite.R` script to install Bioconductor packages. To install a particular package, e.g., `limma`, type the following in an R command window:

```
source("http://bioconductor.org/biocLite.R")
biocLite("limma")
```

Install several packages, e.g., "GenomicFeatures" and "AnnotationDbi", with

```
biocLite(c("GenomicFeatures", "AnnotationDbi"))
```

### Bioconductor Release »

Packages in the stable, semi-annual release:

- [Software](#)
- [Metadata](#) (Annotation, CDF and Probe)
- [Experiment Data](#)

Bioconductor is also available as an [Amazon Machine Image](#).

### Workflows »

Common Bioconductor workflows include:

- [Oligonucleotide Arrays](#)
- [High-throughput Sequencing](#)
- [Annotation](#)
- [Variants](#)
- [Flow Cytometry](#) and other assays
- [Finding Candidate Binding Sites for Known Transcription Factors via Sequence Matching](#)

# Exercise

---

- ▶ Try and install bioconductor packages
  - ▶ GenomicFeatures
  - ▶ cummeRbund
  - ▶ DEseq
  - ▶ Both will require several (many?) additional packages, so let's them install

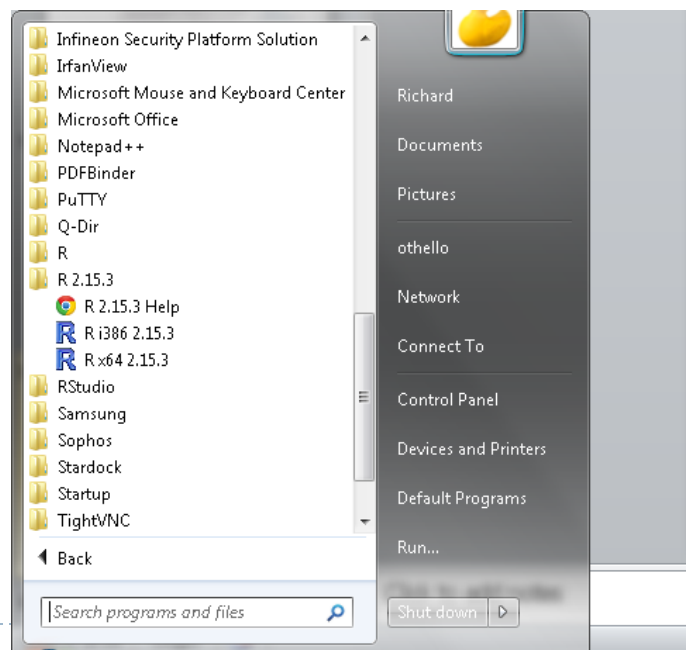
```
source("http://bioconductor.org/biocLite.R")
this installs core bioconductor packages
biocLite()
install Rsamtools
biocLite("Rsamtools")
GenomicFeatures and gene positions
biocLite("GenomicFeatures")
biocLite("TxDb.Hsapiens.UCSC.hg19.knownGene")
this installs DEseq2
biocLite("DESeq2")
other useful data sets
biocLite(c("pasilla", "parathyroidSE"))
this would install cummeRbund
biocLite("cummeRbund")
```



# Exercises

---

- ▶ If you run into trouble installing packages
  - ▶ On a PC running Windows
    - ▶ Try running R in “administrator mode”
    - ▶ From the start menu, right click on the R program and select “Run as administrator”



# Exercises

---

- ▶ If you run into trouble installing packages
  - ▶ On a Mac
    - ▶ You may need administrator privileges
  - ▶ On Unix/Linux
    - ▶ You may need to become root to install to the correct directory
    - ▶ Try running sudo either
      - > `sudo R`
    - ▶ OR
      - > `sudo su`
      - > `R`
  - And then installing packages



# Additional resources

---

- ▶ Many resources but sometimes difficult to access
  - ▶ R-bloggers.com
  - ▶ R-project.org – FAQ and manuals
  - ▶ bioconductor.org
  - ▶ UCLA ATS (<http://www.ats.ucla.edu/stat/r/>)
    - ▶ learning modules
  - ▶ Quick R <http://www.statmethods.net/index.html>
  - ▶ <http://onertipaday.blogspot.com/>
  - ▶ <http://wiki.stdout.org/rcookbook/>
  - ▶ <http://gettinggeneticsdone.blogspot.com/>
  - ▶ [manuals.bioinformatics.ucr.edu/workshops](http://manuals.bioinformatics.ucr.edu/workshops)
- ▶ Good free books
  - ▶ An Introduction to R (on r-project.org) – Venables, Smith, R Core
  - ▶ Using R
  - ▶ R for Beginners by Emmanuel Paradis



# Some R books

---

- ▶ R in Action, Robert I. Kabacoff
- ▶ R Cookbook, Paul Teetor
- ▶ The R book, Michael Crawley
- ▶ Introduction to the R Project for Statistical Computing for Use at the ITC, Rossiter



# More Resources

---

- ▶ [software-carpentry.org](http://software-carpentry.org)
- ▶ Google developer R lectures on Youtube
  - ▶ <http://www.youtube.com/watch?v=iffR3fWv4xw&list=PLOU2XLYxmslK9qQfztXeybpHvru-TrqAP>



# ggplot2

---

- ▶ `library(ggplot2)`
- > `install.packages("ggplot2")`

