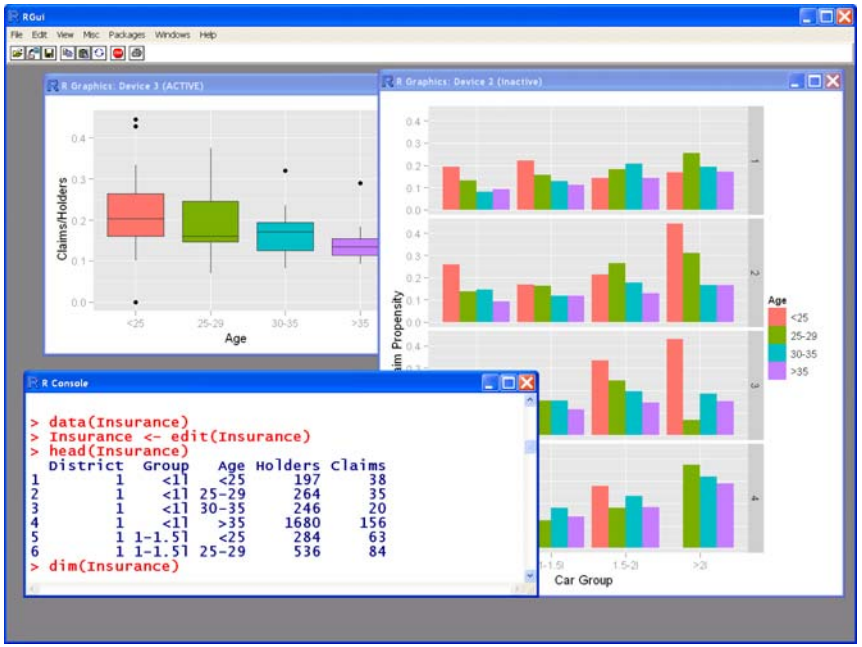
An Interactive Introduction to R for Actuaries

CAS Conference

November 2009

Boston, Massachusetts



Michael E. Driscoll, Ph.D.
med@dataspora.com

dataspora

Daniel Murphy FCAS, MAAA
dmurphy@trinostics.com

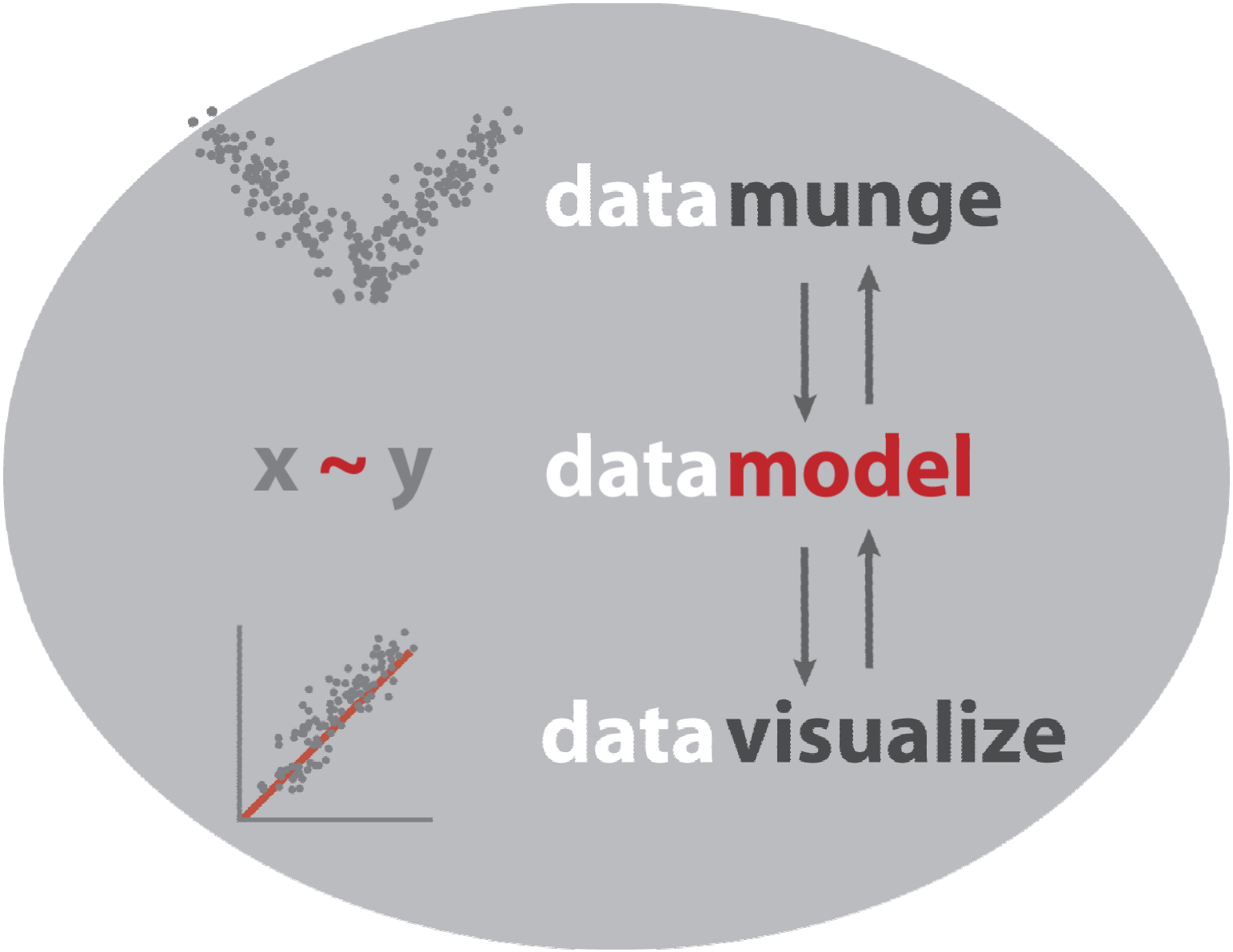
TRINOSTICS LLC

The New York Times
January 6, 2009
Business Computing

Data Analysts Captivated by R's Power

Stuart Isett for The New York Times

R first appeared in 1996, when the statistics professors Robert Gentleman, left, and Ross Ihaka released the code as a free


data munge
x ~ y
data model
data visualize

R is a tool for…

Data Manipulation

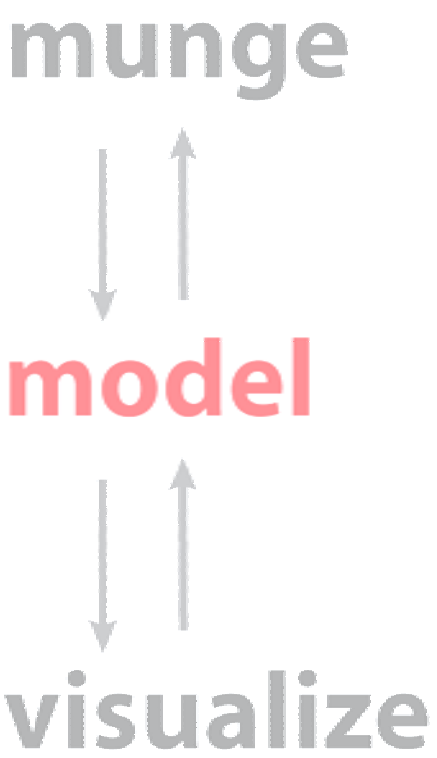
- connecting to data sources
- slicing & dicing data

Modeling & Computation

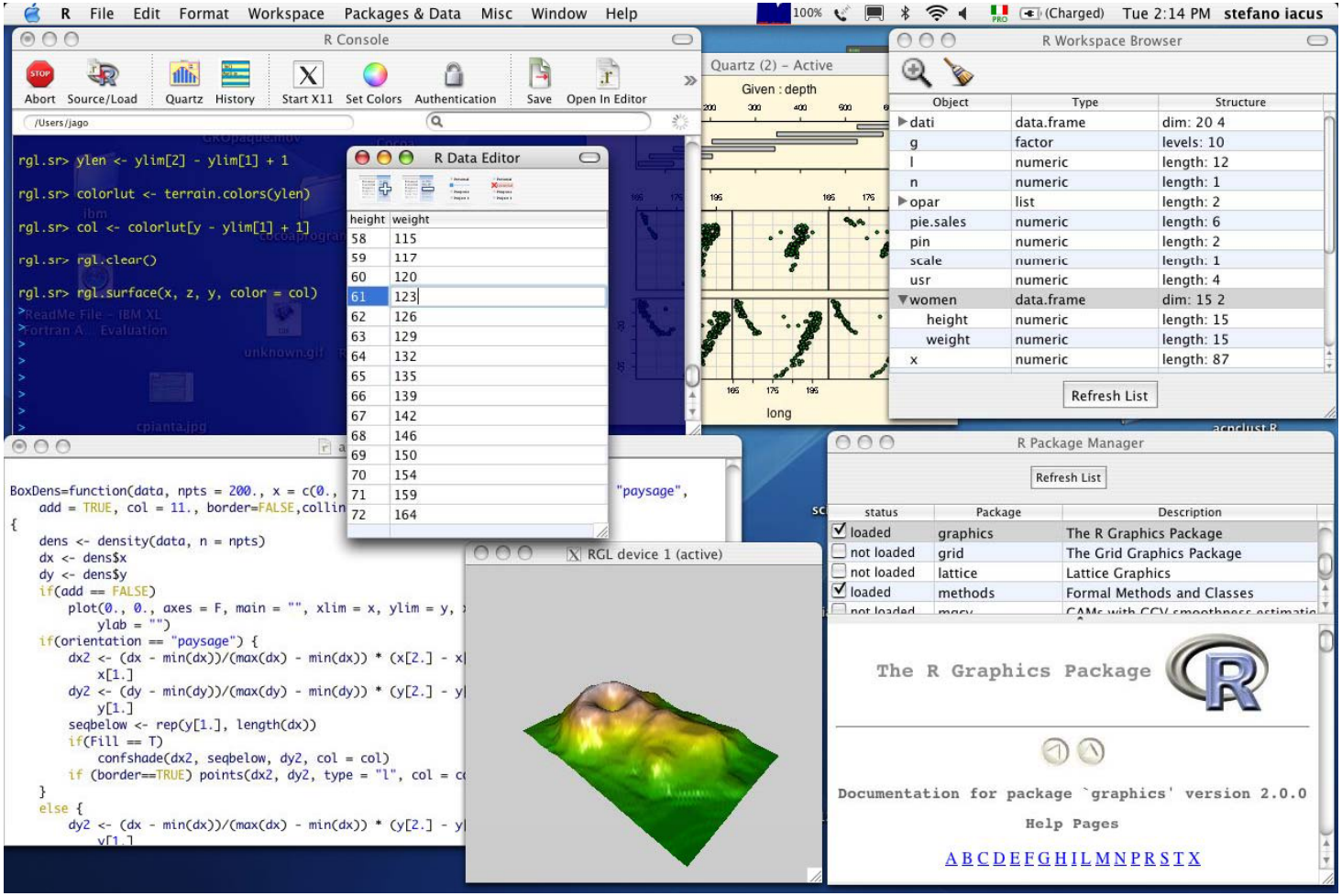
- statistical modeling
- numerical simulation

Data Visualization

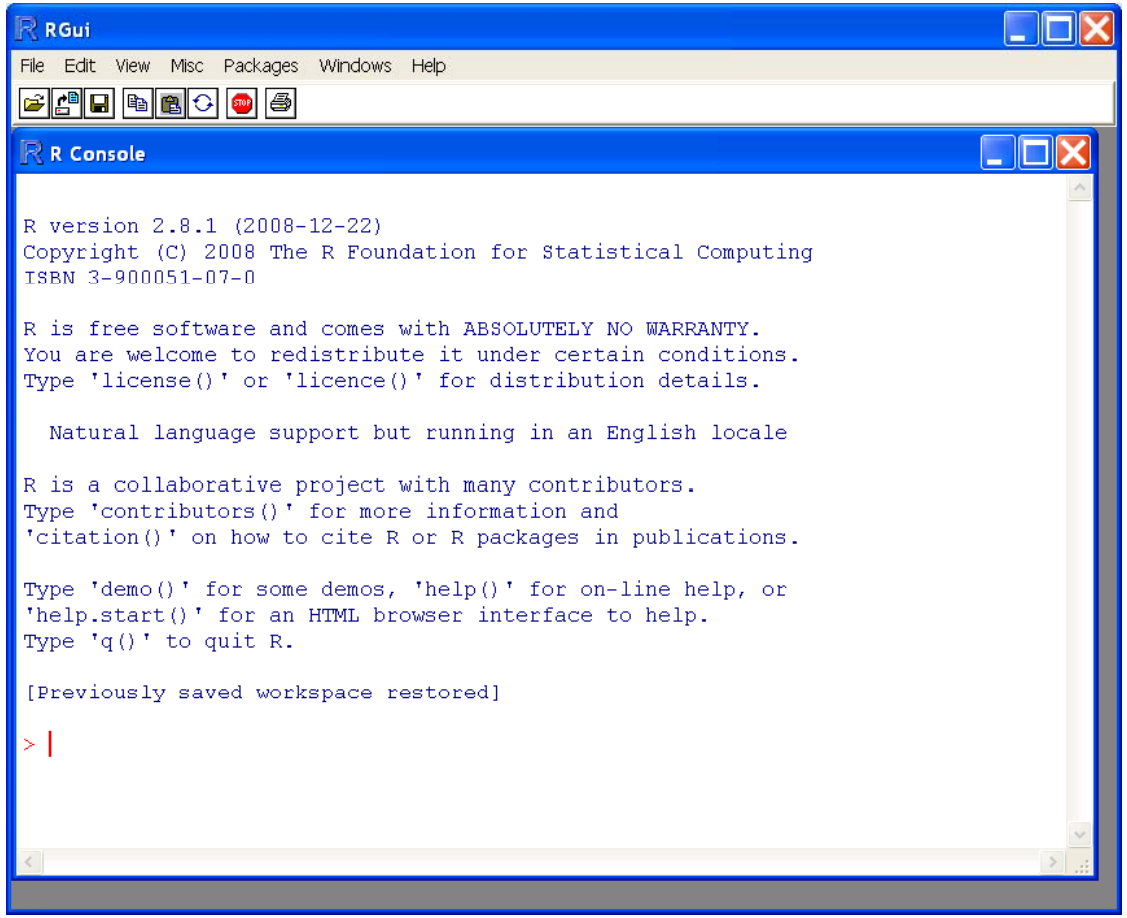
- visualizing fit of models
- composing statistical graphics



R is an environment



Its interface is plain



Let's take a tour of some claim data in R

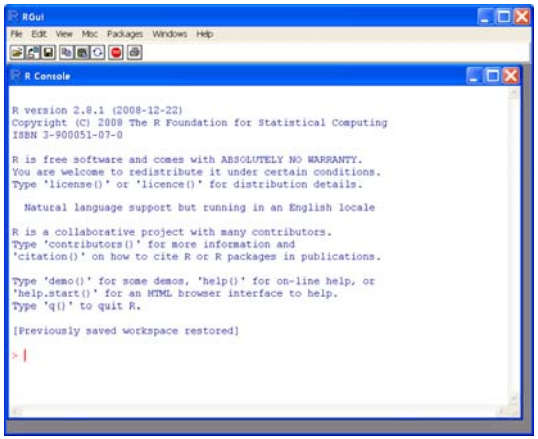
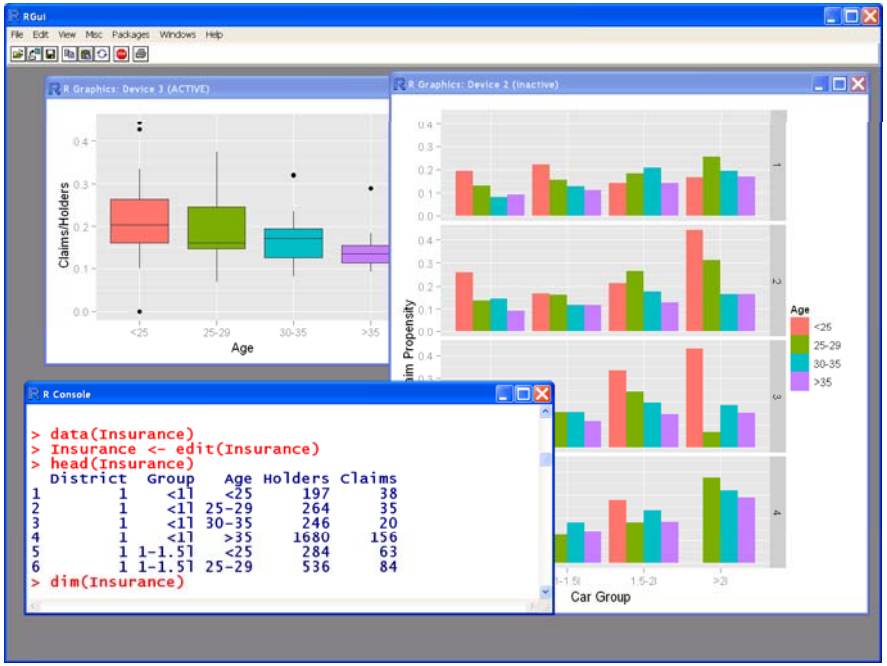


Table 8.1 *Average cost of claims for own damage (adjusted for inflation) for privately owned, comprehensively insured cars in 1975*

Policy-holder's age	*Car group*	*Vehicle age* 0–3 £	0–3 No.	4–7 £	4–7 No.	8–9 £	8–9 No.	10+ £	10+ No.
17–20	A	289	8	282	8	133	4	160	1
	B	372	10	249	28	288	1	11	1
	C	189	9	288	13	179	1	–	0
	D	763	3	850	2	–	0	–	0
21–24	A	302	18	194	31	135	10	166	4
	B	420	59	243	96	196	13	135	3
	C	268	44	343	39	293	7	104	2
	D	407	24	320	18	205	2	–	0
25–29	A	268	56	285	55	181	17	110	12
	B	275	125	243	172	179	36	264	10
	C	334	163	274	129	208	18	150	8
	D	383	72	305	50	116	6	636	1
30–34	A	236	43	270	53	160	15	110	12
	B	259	179	226	211	161	39	107	19
	C	340	197	260	125	189	30	104	9
	D	400	104	349	55	147	8	65	2
35–39	A	207	43	129	73	157	251	113	14
	B	208	191	214	219	149	46	137	23
	C	251	210	232	131	204	32	141	8
	D	233	119	325	43	207	4	–	0
40–49	A	254	90	213	98	149	35	98	22
	B	218	380	209	434	172	97	110	59
	C	239	401	250	253	174	50	129	15
	D	387	199	299	88	325	8	137	9
50–59	A	251	69	227	120	172	42	98	35
	B	196	366	229	353	164	95	132	45
	C	268	310	250	148	175	33	152	13
	D	391	105	228	46	346	10	167	1
60+	A	264	64	198	100	167	43	114	53
	B	224	228	193	233	178	73	101	44
	C	269	183	258	103	227	20	119	6
	D	385	62	324	22	192	6	123	6

Let's take a tour of some claim data in R



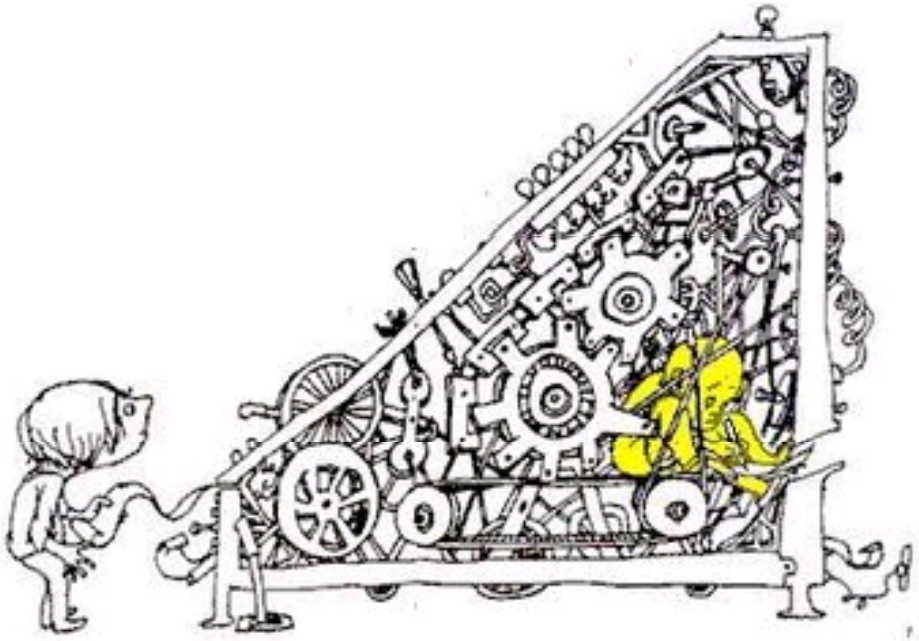

```
## load in some Insurance Claim data
library(MASS)
data(Insurance)
Insurance <- edit(Insurance)
head(Insurance)
dim(Insurance)

## plot it nicely using the ggplot2 package
library(ggplot2)
qplot(Group, Claims/Holders,
      data=Insurance,
      geom="bar",
      stat='identity',
      position="dodge",
      facets=District ~ .,
      fill=Age,
      ylab="Claim Propensity",
      xlab="Car Group")

## hypothesize a relationship between Age ~ Claim Propensity
## visualize this hypothesis with a boxplot
x11()
library(ggplot2)
qplot(Age, Claims/Holders,
      data=Insurance,
      geom="boxplot",
      fill=Age)

## quantify the hypothesis with linear model
m <- lm(Claims/Holders ~ Age + 0, data=Insurance)
summary(m)
```

R is “an overgrown calculator”



```
sum(rgamma(rpois(1,lambda=2),shape=49,scale=.2)))
```

R is “an overgrown calculator”

- simple math

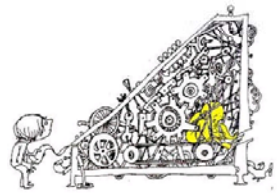
```
> 2+2
4
```

- storing results in variables

```
> x <- 2+2    ## ‘<-’ is R syntax for ‘=’ or assignment
> x^2
16
```

- vectorized math

```
> weight <- c(110, 180, 240)      ## three weights
> height <- c(5.5, 6.1, 6.2)      ## three heights
> bmi <- (weight*4.88)/height^2   ## divides element-wise
17.7  23.6  30.4
```



R is “an overgrown calculator”

- basic statistics

```
mean(weight)        sd(weight)          sqrt(var(weight))
176.6               65.0                65.0  # same as sd
```

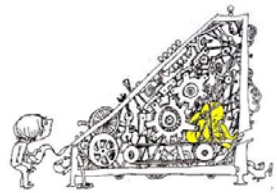
- set functions

```
union               intersect           setdiff
```

- advanced statistics

```
> pbinom(40, 100, 0.5)  ##  P that a coin tossed 100 times
0.028                   ##  that comes up 40 heads is ‘fair’
```

```
> pshare <- pbirthday(23, 365, coincident=2)
0.530  ## probability that among 23 people, two share a birthday
```



Try It! #1
Overgrown Calculator

- basic calculations

```
> 2 + 2          [Hit ENTER]
> log(100)       [Hit ENTER]
```

- calculate the value of $100 after 10 years at 5%

```
> 100 * exp(0.05*10) [Hit ENTER]
```

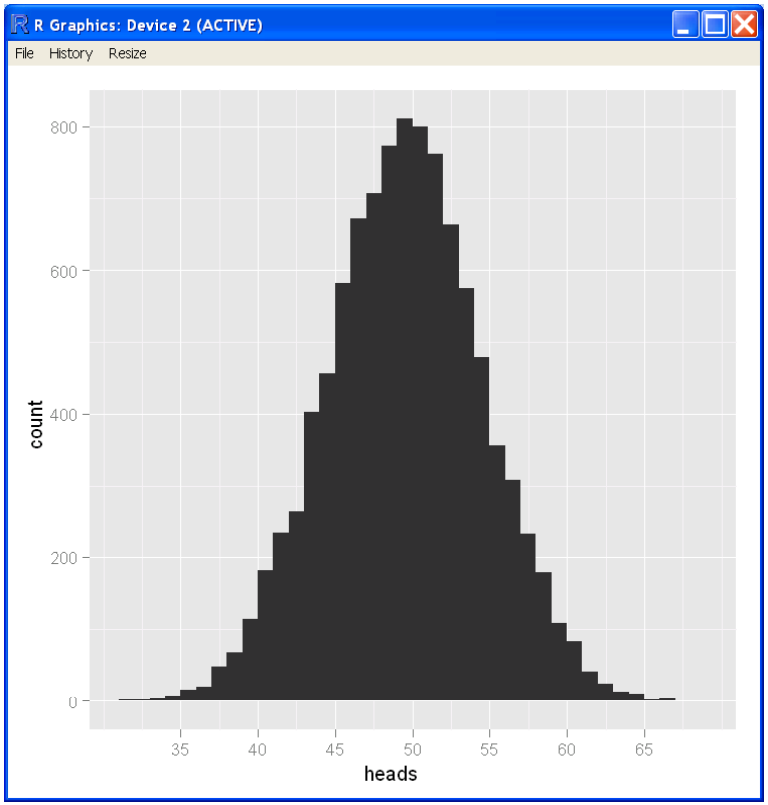
- construct a vector & do a vectorized calculation

```
> year <- (1,2,5,10,25)   [Hit ENTER]  this returns an error.  why?
> year <- c(1,2,5,10,25)  [Hit ENTER]
> 100 * exp(0.05*year)    [Hit ENTER]
```

R is a numerical simulator

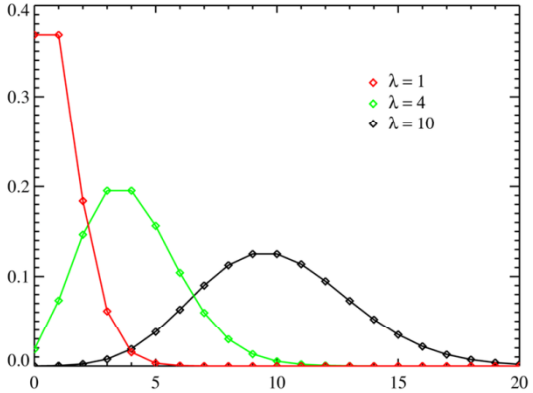
- built-in functions for classical probability distributions

- let's simulate 10,000 trials of 100 coin flips. what's the distribution of heads?




```
> heads <- rbinom(10^5,100,0.50)
> hist(heads)
```

Functions for Probability Distributions



d*dist*( )	density function (pdf)
p*dist*( )	cumulative density function
q*dist*( )	quantile function
r*dist*( )	random deviates

Examples

Normal	**d**norm, **p**norm, **q**norm, **r**norm
Binomial	**d**binom, **p**binom, …
Poisson	**d**pois, …

> pnorm(0)	0.05
> qnorm(0.9)	1.28
> rnorm(100)	vector of length 100

Functions for Probability Distributions

How to find the functions for lognormal distribution?

1) Use the double question mark '??' to search

```
> ??lognormal
```

2) Then identify the package

```
> ?Lognormal
```

3) Discover the dist functions

`dlnorm, plnorm, qlnorm, rlnorm`

distribution	*dist suffix in R*
Beta	-beta
Binomial	-binom
Cauchy	-cauchy
Chisquare	-chisq
Exponential	-exp
F	-f
Gamma	-gamma
Geometric	-geom
Hypergeometric	-hyper
Logistic	-logis
Lognormal	-lnorm
Negative Binomial	-nbinom
Normal	-norm
Poisson	-pois
Student t	-t
Uniform	-unif
Tukey	-tukey
Weibull	-weib
Wilcoxon	-wilcox

Try It! #2
Numerical Simulation

- simulate 1m policy holders from which we expect 4 claims

```
> numclaims <- rpois(n, lambda)
```

(hint: use `?rpois` to understand the parameters)

- verify the mean & variance are reasonable

```
> mean(numclaims)
> var(numclaims)
```

- visualize the distribution of claim counts

```
> hist(numclaims)
```

Getting Data In

from Files

```
> Insurance <- read.csv(“Insurance.csv”,header=TRUE)
```

from Databases

```
> con <- dbConnect(driver,user,password,host,dbname)
> Insurance <- dbSendQuery(con, “SELECT * FROM
  claims”)
```

from the Web

```
> con <- url('http://labs.dataspora.com/test.txt')
> Insurance <- read.csv(con, header=TRUE)
```

from R objects

```
> load(‘Insurance.RData’)
```

Getting Data Out

to Files

```
write.csv(Insurance,file="Insurance.csv")
```

to Databases

```
con <- dbConnect(dbdriver,user,password,host,dbname)
dbWriteTable(con, "Insurance", Insurance)
```

to R Objects

```
save(Insurance, file="Insurance.RData")
```

Navigating within the R environment

- listing all variables

```
> ls()
```

- examining a variable ‘x’

```
> str(x)
> head(x)
> tail(x)
> class(x)
```

- removing variables

```
> rm(x)
> rm(list=ls())    # remove everything
```

Try It! #3
Data Processing

- load data & view it

```
library(MASS)
head(Insurance)  ## the first 7 rows
dim(Insurance)   ## number of rows & columns
```

- write it out

```
write.csv(Insurance,file="Insurance.csv", row.names=FALSE)
getwd()  ## where am I?
```

- view it in Excel, make a change, save it

```
remove the first district
```

- load it back in to R & plot it

```
Insurance <- read.csv(file="Insurance.csv")
plot(Claims/Holders ~ Age, data=Insurance)
```

A Swiss-Army Knife for Data

A Swiss-Army Knife for Data

- Indexing
- Three ways to index into a data frame
  - array of integer indices
  - array of character names
  - array of logical Booleans
- Examples:

```
df[1:3,]
df[c(“New York”, “Chicago”),]
df[c(TRUE,FALSE,TRUE,TRUE),]
df[city == “New York”,]
```

A Swiss-Army Knife for Data

- subset – extract subsets meeting some criteria

```
subset(Insurance, District==1)
subset(Insurance, Claims < 20)
```

- transform – add or alter a column of a data frame

```
transform(Insurance, Propensity=Claims/Holders)
```

- cut – cut a continuous value into groups

```
cut(Insurance$Claims, breaks=c(-1,100,Inf),
labels=c('lo','hi'))
```

- Put it all together: create a new, transformed data frame

```
transform(subset(Insurance, District==1),
   ClaimLevel=cut(Claims, breaks=c(-1,100,Inf),
   labels=c('lo','hi')))
```

A Statistical Modeler

- R's has a powerful modeling syntax
- Models are specified with formulae, like

```
y ~ x
growth ~ sun + water
```

  model relationships between continuous and categorical variables.

- Models are also guide the visualization of relationships in a graphical form

A Statistical Modeler

- Linear model

```
m <- lm(Claims/Holders ~ Age, data=Insurance)
```

- Examine it

```
summary(m)
```

- Plot it

```
plot(m)
```

A Statistical Modeler

- Logistic model

```
m <- glm(Age ~ Claims/Holders, data=Insurance,
       family=binomial("logit"))
)
```

- Examine it

```
summary(m)
```

- Plot it

```
plot(m)
```

Try It! #4
Statistical Modeling

- fit a linear model

```
m <- lm(Claims/Holders ~ Age + 0, data=Insurance)
```

- examine it

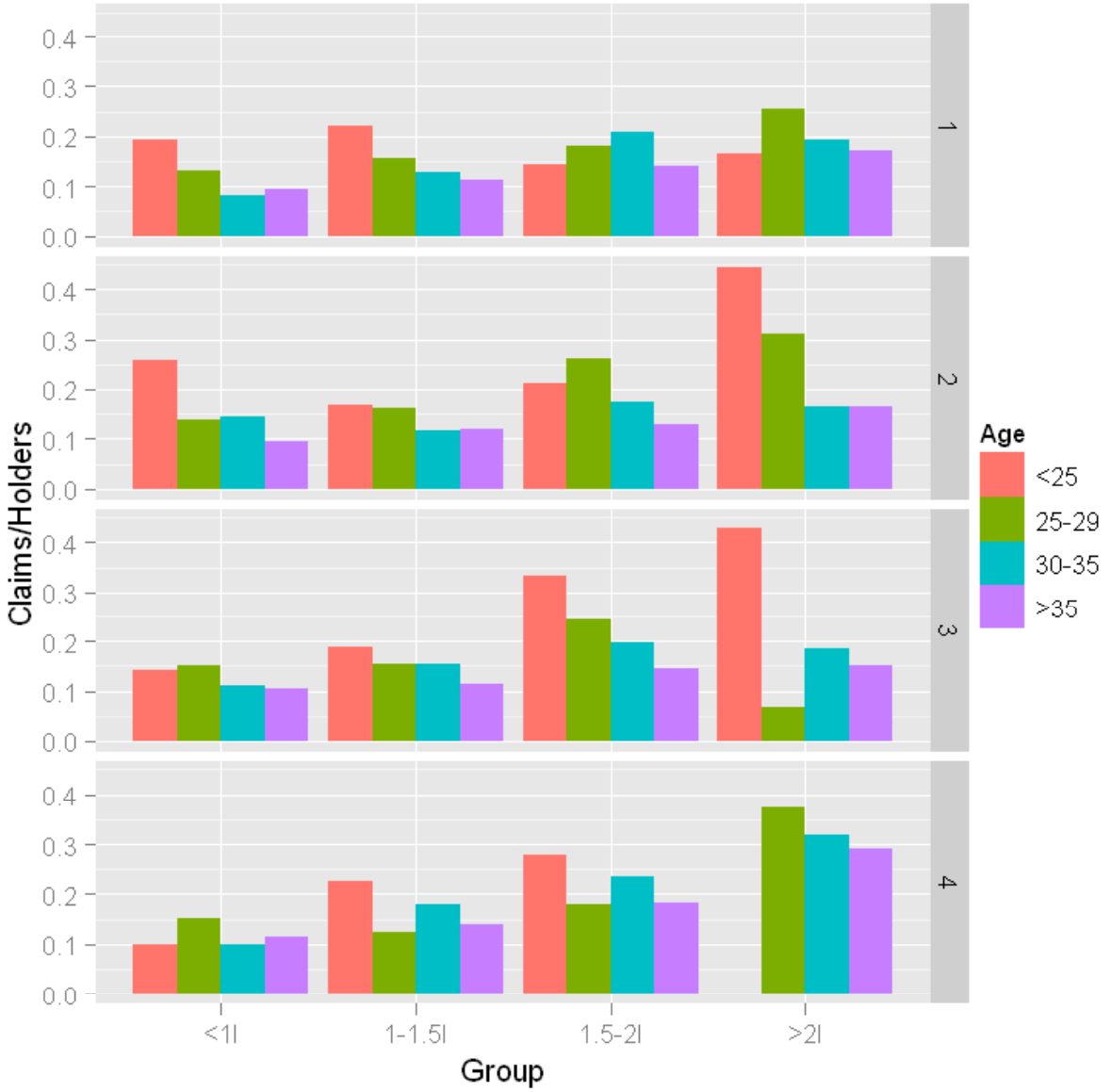
```
summary(m)
```

- plot it

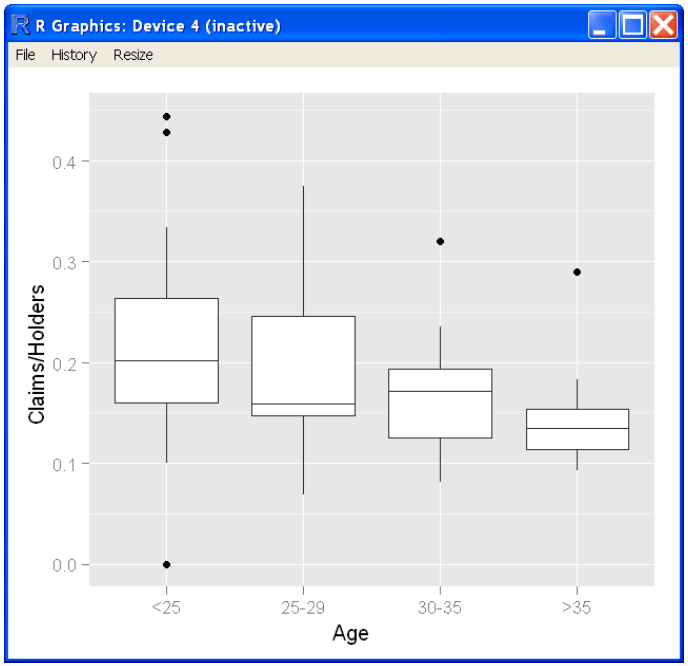
```
plot(m)
```

Visualization: Multivariate Barplot

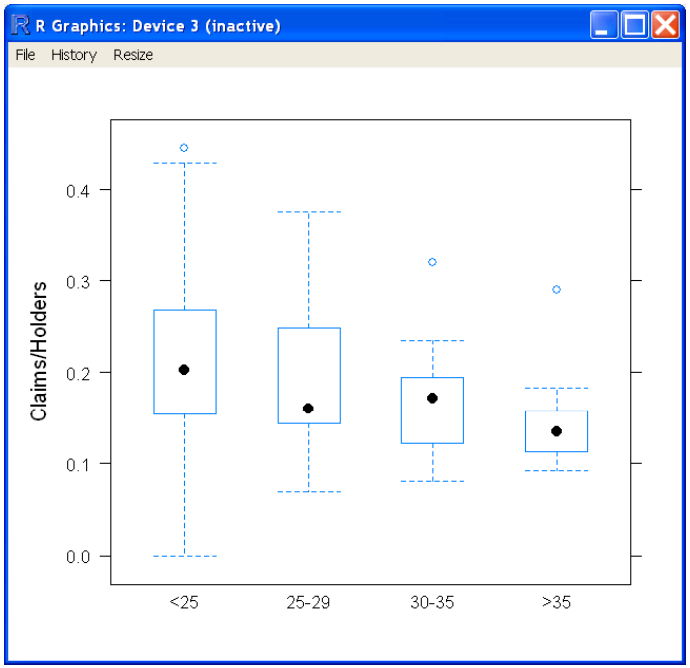
```
library(ggplot2)
qplot(Group, Claims/Holders,
      data=Insurance,
      geom="bar",
      stat='identity',
      position="dodge",
      facets=District ~ .,
      fill=Age)
```



Visualization: Boxplots

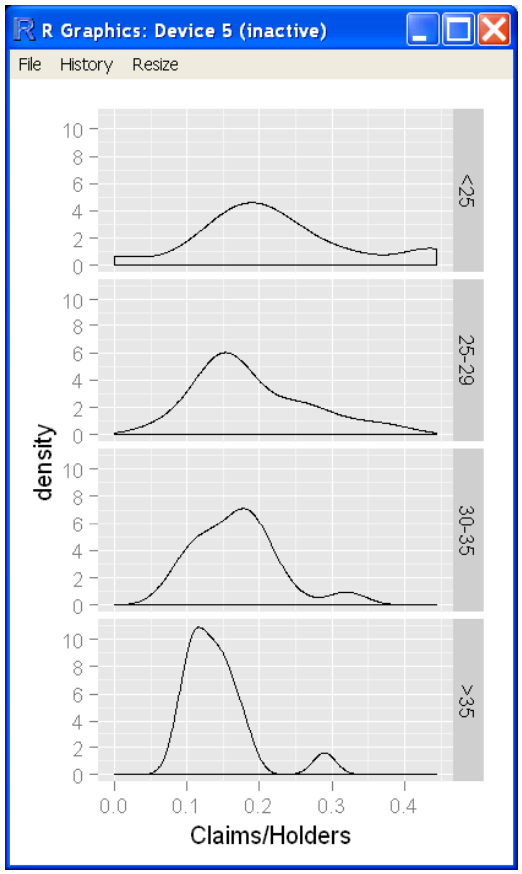



```
library(ggplot2)
qplot(Age, Claims/Holders,
  data=Insurance,
  geom="boxplot“)
```

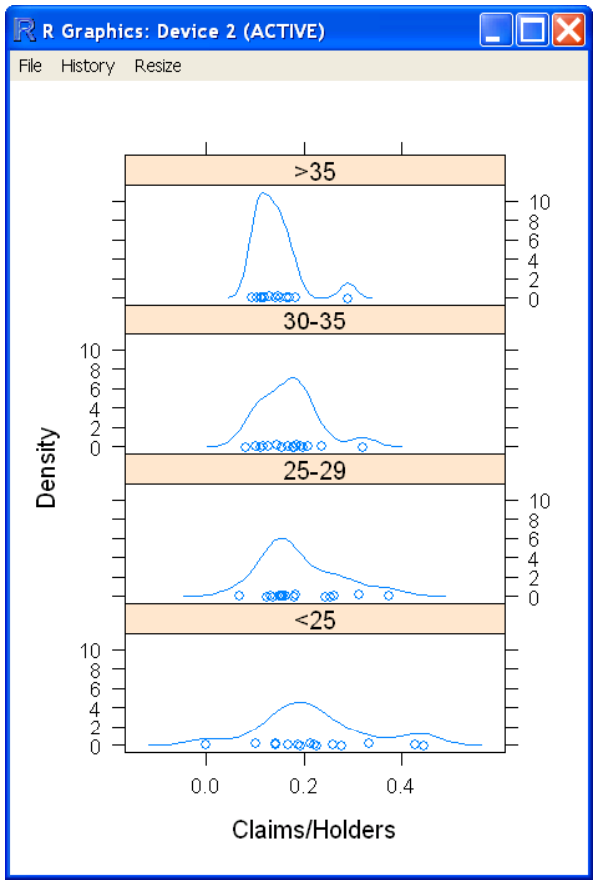



```
library(lattice)
bwplot(Claims/Holders ~ Age,
  data=Insurance)
```

Visualization: Histograms




```
library(ggplot2)
qplot(Claims/Holders,
 data=Insurance,
 facets=Age ~ ., geom="density")
```




```
library(lattice)
densityplot(~ Claims/Holders | Age,
data=Insurance, layout=c(4,1)
```

Try It! #5
Data Visualization

- simple line chart

```
> x <- 1:10
> y <- x^2
> plot(y ~ x)
```

- box plot

```
> library(lattice)
> boxplot(Claims/Holders ~ Age, data=Insurance)
```

- visualize a linear fit

```
> abline()
```

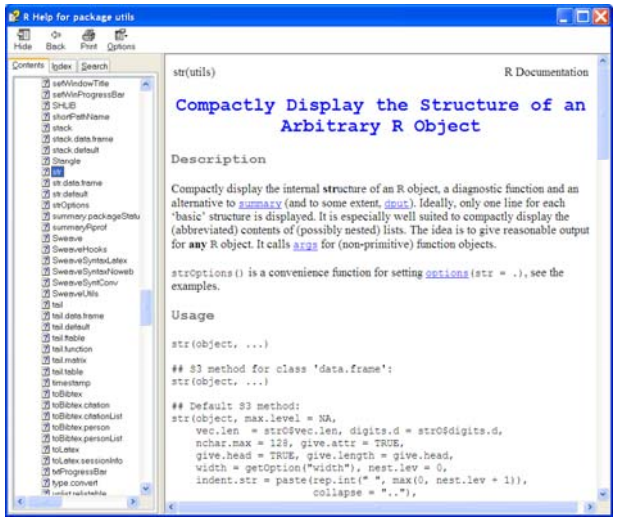
Getting Help with R

Help within R itself for a function

```
> help(func)
> ?func
```

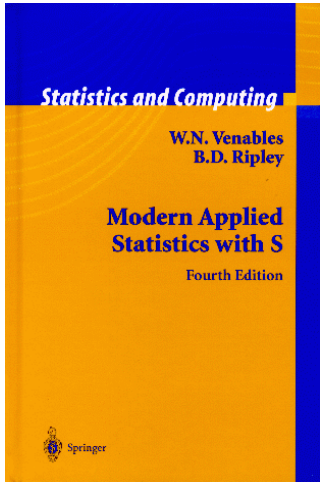
For a topic

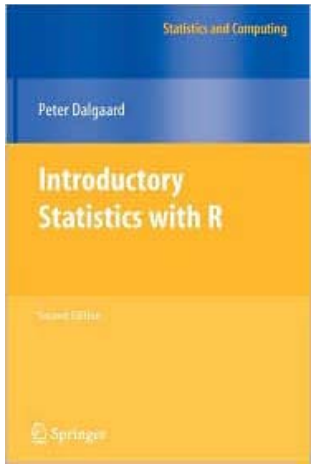
```
> help.search(topic)
> ??topic
```



- **search.r-project.org**
- **Google Code Search** www.google.com/codesearch
- **Stack Overflow** http://stackoverflow.com/tags/R
- **R-help list** http://www.r-project.org/posting-guide.html

Six Indispensable Books on R



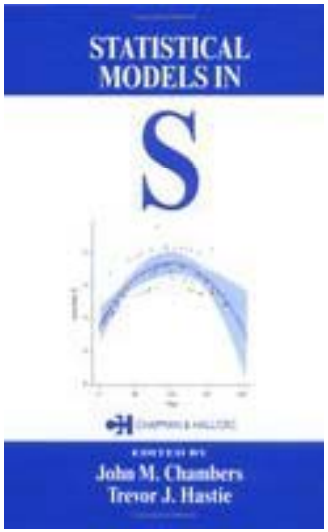


Learning R

Data Manipulation

Visualization

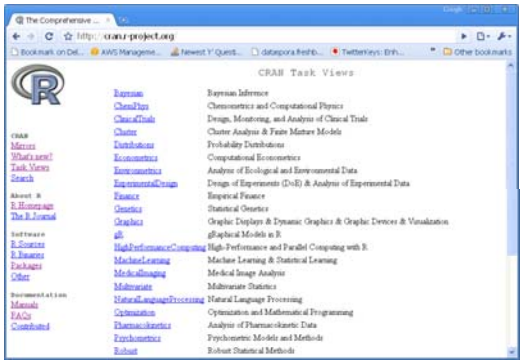
Statistical Modeling



Extending R with Packages

Over one thousand user-contributed packages are available on CRAN – the Comprehensive R Archive Network

http://cran.r-project.org



Install a package from the command-line

```
> install.packages('actuar')
```

Install a package from the GUI menu

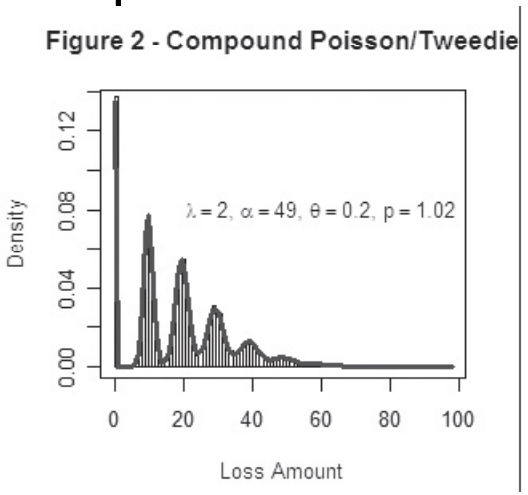
"Packages"--> "Install packages(s)"

Final Try It!
Simulate a Tweedie

- Simulate the number of claims from a Poisson distribution with $\lambda=2$ (NB: mean poisson = λ, variance poisson = λ)
- For as many claims as were randomly simulated, simulate a severity from a gamma distribution with shape $\alpha=49$ and scale $\theta=0.2$ (NB: mean gamma = $\alpha\theta$, variance gamma = $\alpha\theta^2$)
- Is the total simulated claim amount close to expected?
- Calculate usual parameterization (μ,p,ϕ) of this Tweedie distribution

$$\mu = \lambda\alpha\theta, p = \frac{\alpha+2}{\alpha+1}, \phi = \frac{\lambda^{1-p}(\alpha\theta)^{2-p}}{2-p}$$

- Extra credit:
  - Repeat the above 10000 times.
  - Does your histogram look like Glenn Meyers'?
    http://www.casact.org/newsletter/index.cfm?fa=viewart&id=5756



Final Try It!
Simulate a Tweedie- ANSWERS

- Simulate the number of claims from a Poisson distribution with λ=2 (NB: mean poisson = λ, variance poisson = λ)

```
rpois(1,lambda=2)
```

- For as many claims as were randomly simulated, simulate a severity from a gamma distribution with shape α=49 and scale θ=0.2

```
rgamma(rpois(1,lambda=2),shape=49,scale=.2)
```

- Is the total simulated claim amount close to expected?

```
sum(rgamma(rpois(1,lambda=2),shape=49,scale=.2))
```

- Repeat the above 10000 times

```
replicate(10000,
  sum(rgamma(rpois(1,lambda=2),shape=49,scale=.2)))
```

- Visualize the distribution

```
hist(replicate(10000,
  sum(rgamma(rpois(1,lambda=2),shape=49,scale=.2))),
  breaks=200, freq=FALSE)
```

Contact Us

dataspora

From Data to Decision
Big Data • Analytics • Visualization
www.dataspora.com

Michael E. Driscoll, Ph.D.
med@dataspora.com
415.860.4347

TRINOSTICS LLC

P&C Actuarial Models
Design • Construction
Collaboration • Education
Valuable • Transparent

Daniel Murphy, FCAS, MAAA
dmurphy@trinostics.com
925.381.9869

Appendices

- R as a Programming Language
- Advanced Visualization
- Embedding R in a Server Environment

R as a Programming Language

Image from cover of Abelson & Sussman's text*The Structure and Interpretation of Computer Languages*

```
fibonacci <- function(n) {
  fib <- numeric(n)
  fib [1:2] <- 1
  for (i in 3:n) {
      fib[i] <- fib[i-1] + fib[i-2]
  }
  return(fib[n])
}
```

Assignment

```
x <- c(1,2,6)
```

`x` a variable *x*

`<-` R's assignment operator, equivalent to '='

`c(` a function *c* which **combines** its arguments into a vector

```
y <- c('apples','oranges')
z <- c(TRUE,FALSE)
c(TRUE,FALSE) -> z
```

These are also valid assignment statements.

Function Calls

- There are ~ 1100 built-in commands in the R "base" package, which can be executed on the command-line. The basic structure of a call is thus:

```
output <- function(arg1, arg2, …)
```

- Arithmetic Operations

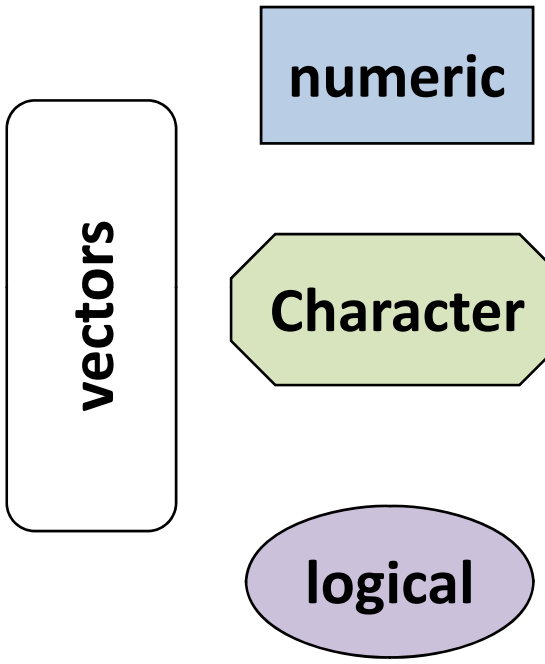
```
+  -  *  /  ^
```

- R functions are typically *vectorized*

```
x <- x/3
```

works whether **x** is a one or many-valued vector

Data Structures in R



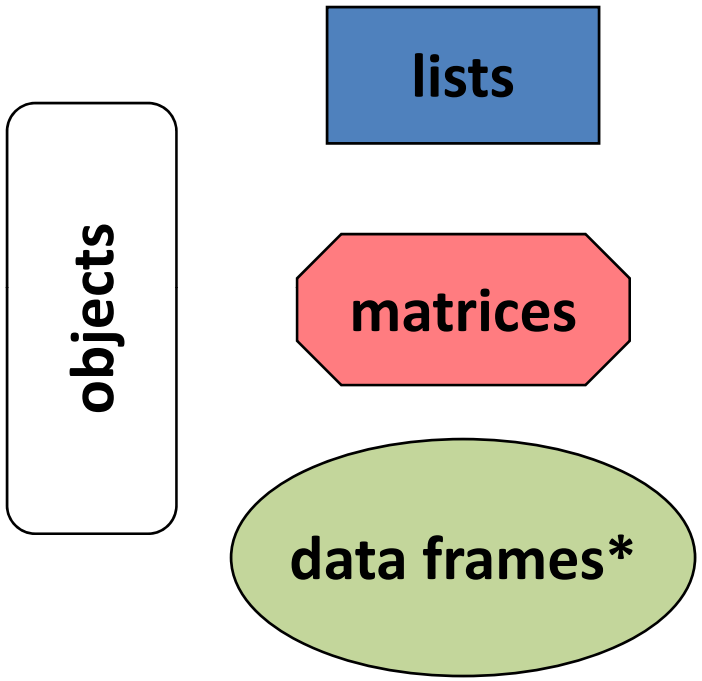

```
x <- c(0,2:4)
```

```
y <- c("alpha", "b", "c3", "4")
```

```
z <- c(1, 0, TRUE, FALSE)
```

```
> class(x)
[1] "numeric"
> x2 <- as.logical(x)
> class(x2)
[1] "logical"
```

Data Structures in R




```
lst <- list(x,y,z)
```

```
M <- matrix(rep(x,3),ncol=3)
```

```
df <- data.frame(x,y,z)
```

```
> class(df)
[1] “data.frame"
```

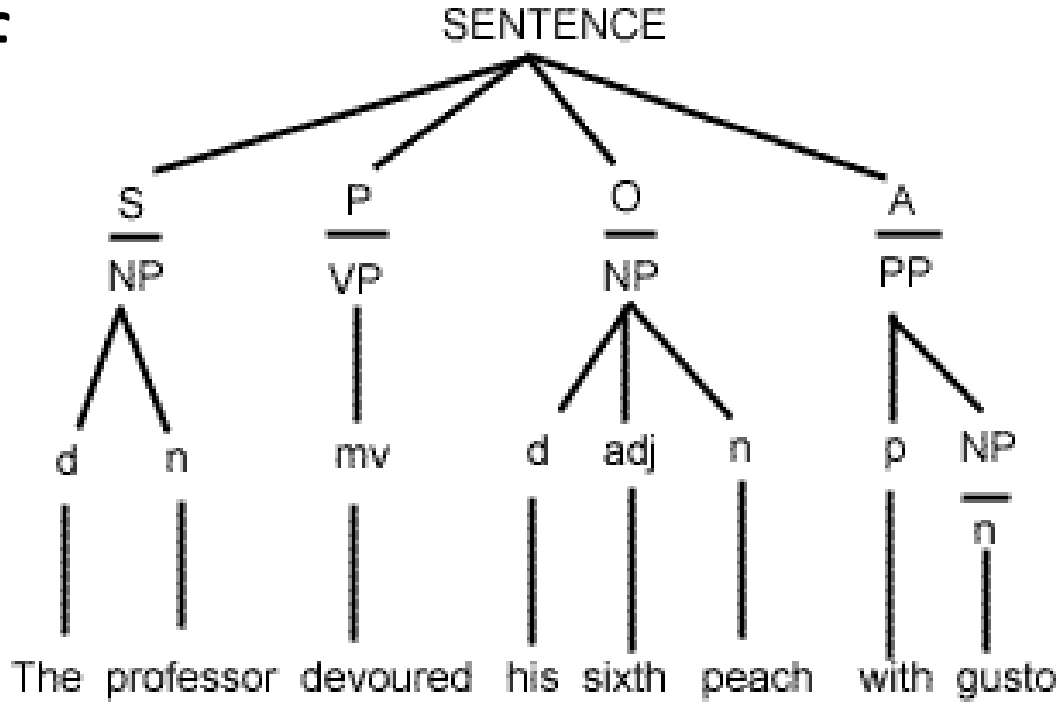
Summary of Data Structures

	Linear	Rectangular
Homogeneous	vectors	matrices
Heterogeneous	lists	data frames*

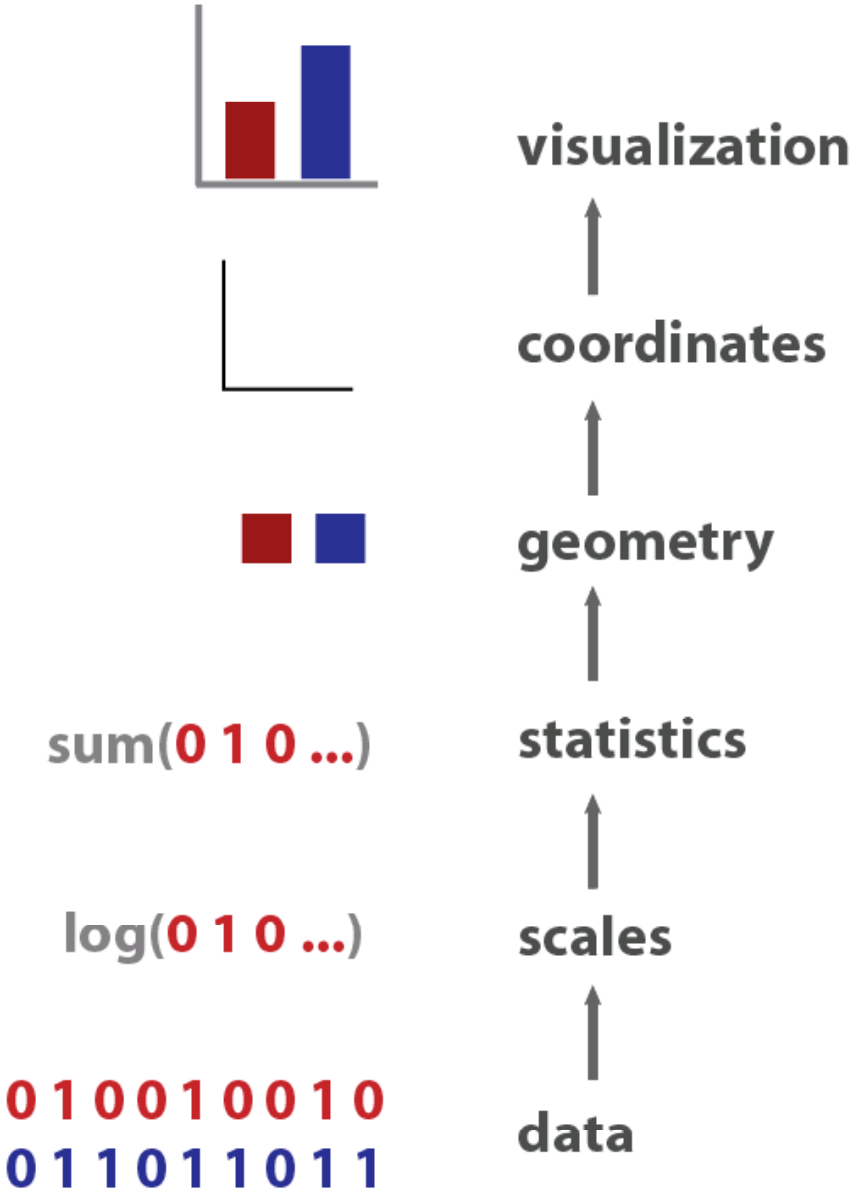
Advanced Visualization

lattice, ggplot2, and colorspace

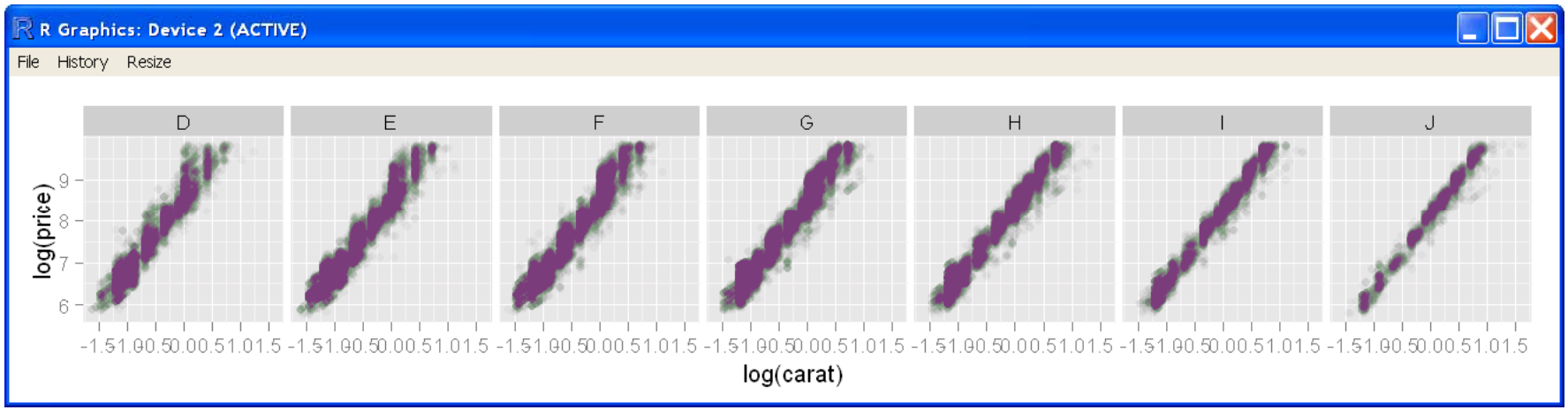
ggplot2 =
grammar of
graphics



ggplot2 =
grammar of
graphics

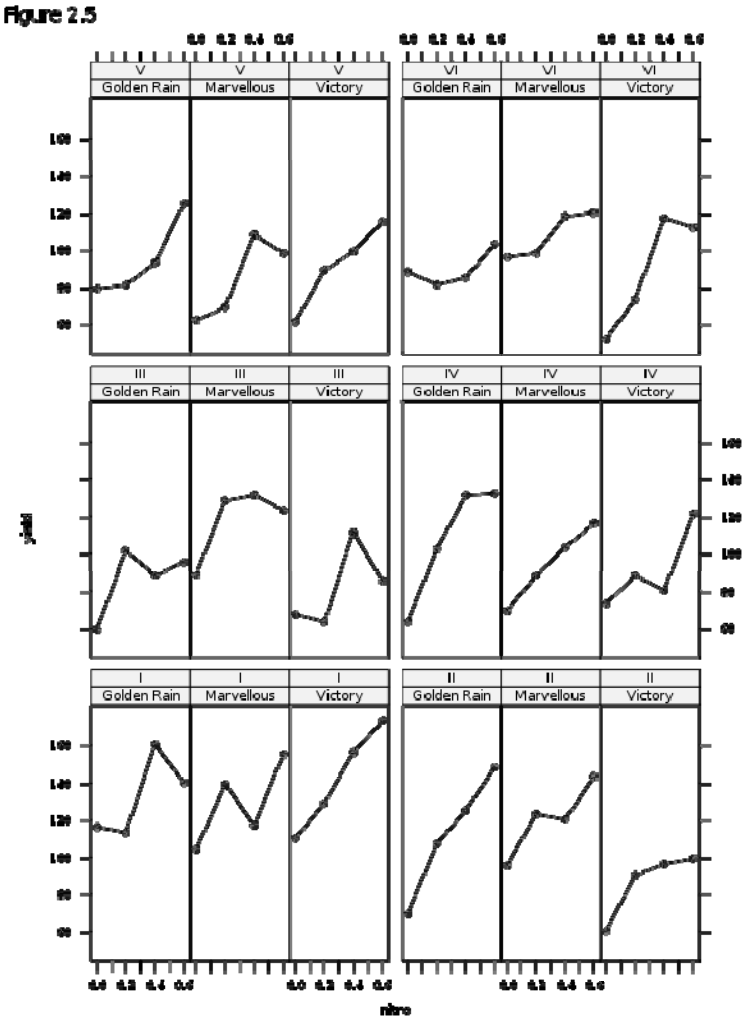


Achieving small multiples with “facets”




```
qplot(log(carat), log(price), data = diamonds,
alpha=I(1/20)) + facet_grid(. ~ color)
```

lattice = trellis

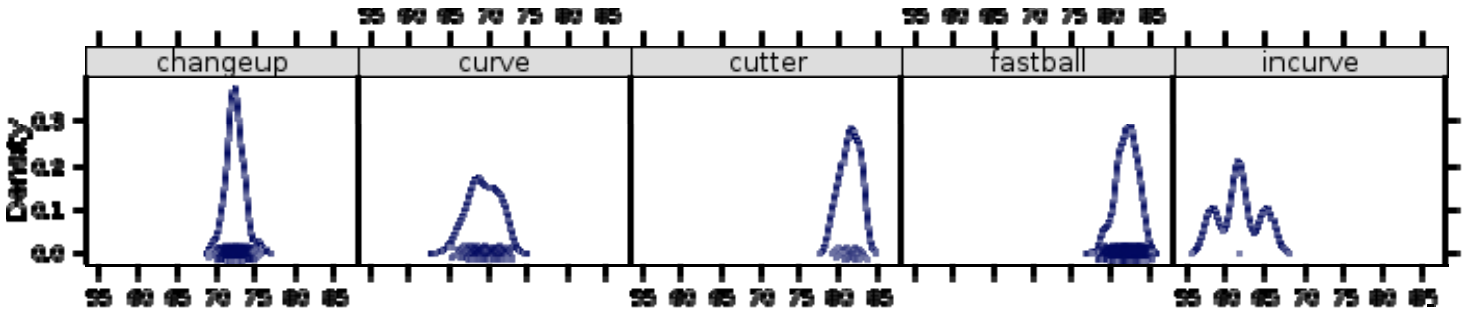


(source: http://lmdvr.r-forge.r-project.org)

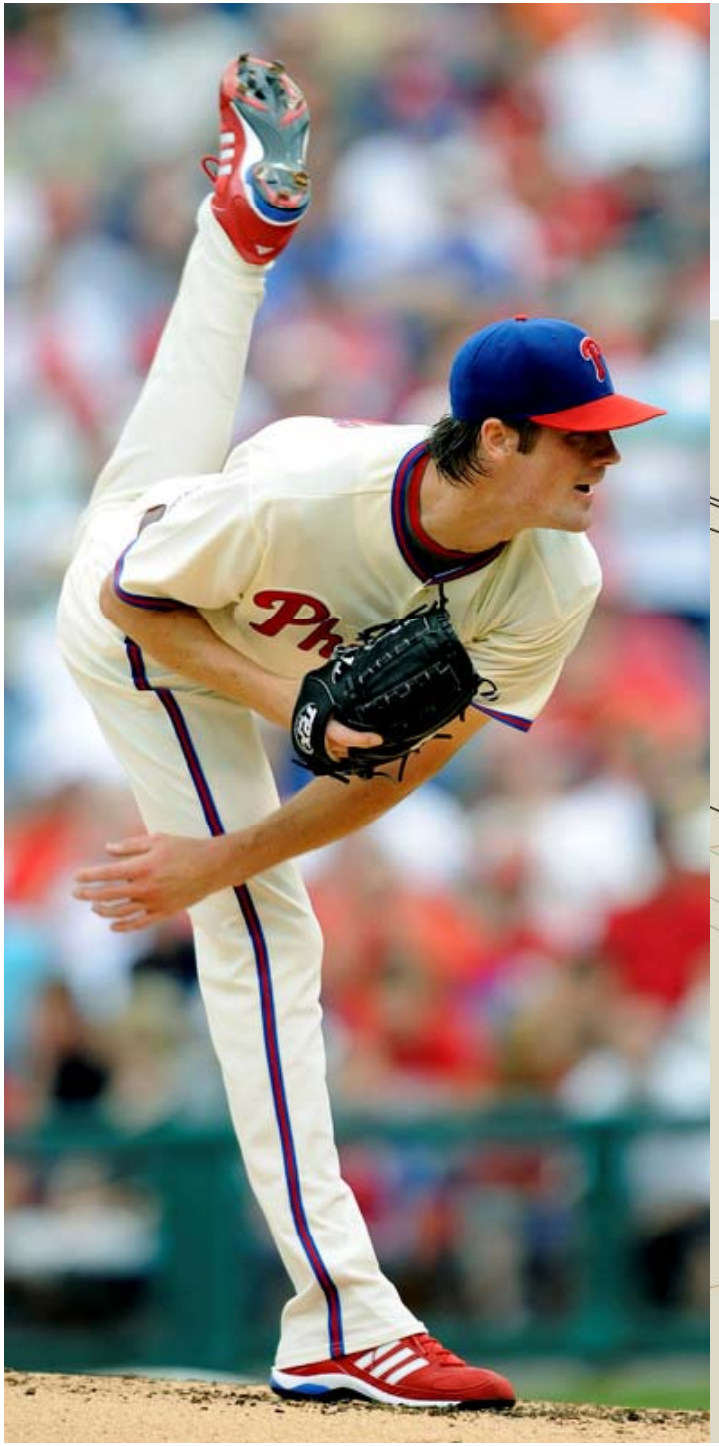
list of lattice functions

Function	Default Display
`histogram()`	Histogram
`densityplot()`	Kernel Density Plot
`qqmath()`	Theoretical Quantile Plot
`qq()`	Two-sample Quantile Plot
`stripplot()`	Stripchart (Comparative 1-D Scatter Plots)
`bwplot()`	Comparative Box-and-Whisker Plots
`dotplot()`	Cleveland Dot Plot
`barchart()`	Bar Plot
`xyplot()`	Scatter Plot
`splom()`	Scatter-Plot Matrix
`contourplot()`	Contour Plot of Surfaces
`levelplot()`	False Color Level Plot of Surfaces
`wireframe()`	Three-dimensional Perspective Plot of Surfaces
`cloud()`	Three-dimensional Scatter Plot
`parallel()`	Parallel Coordinates Plot

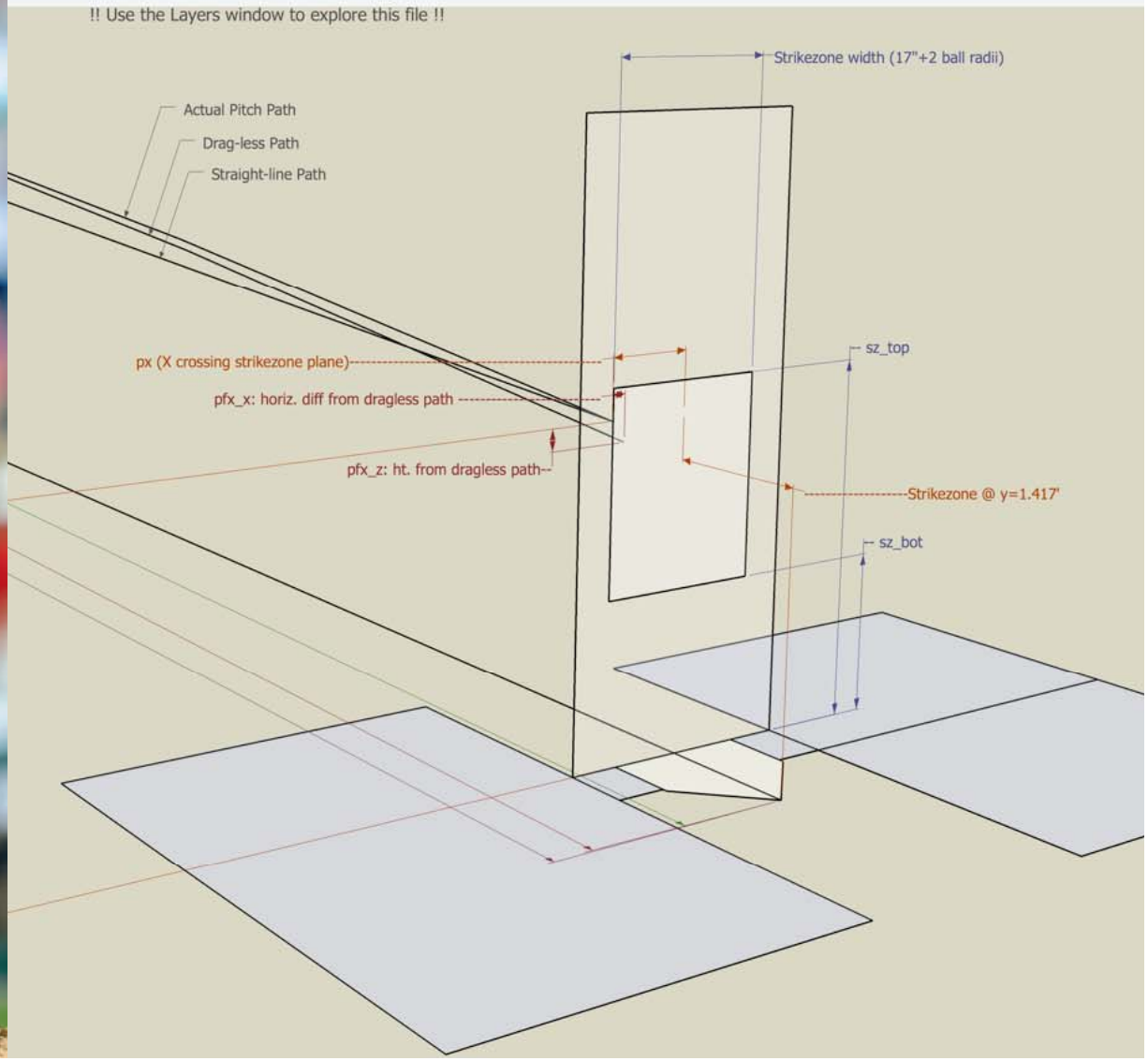
Table 1.1. High-level functions in the lattice package and their default displays.

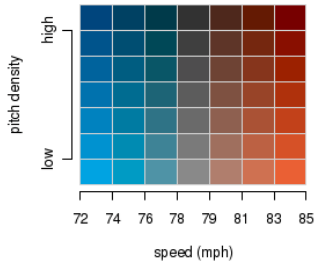


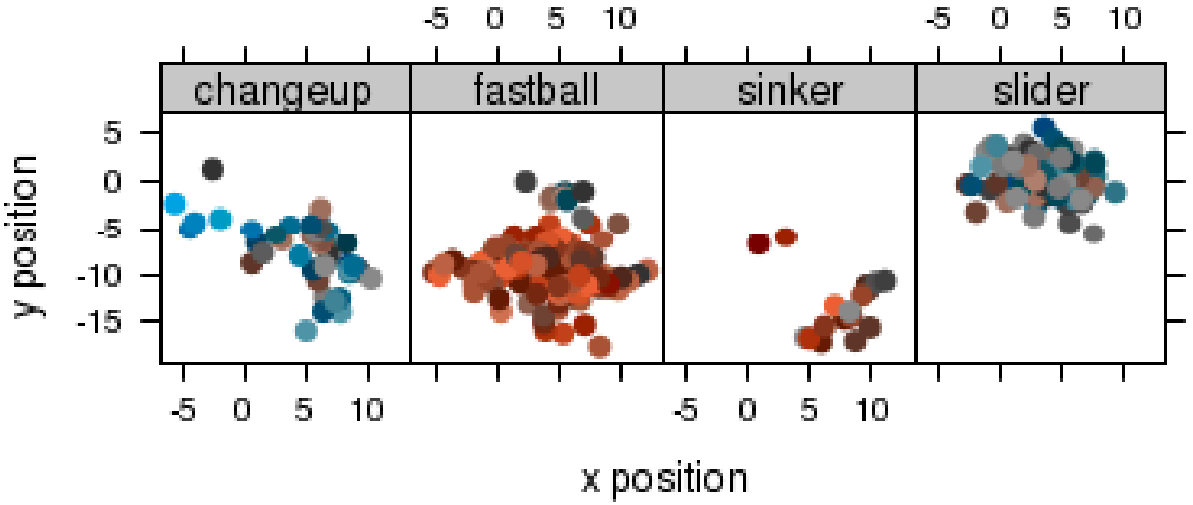

```
densityplot(~ speed | type, data=pitch)
```

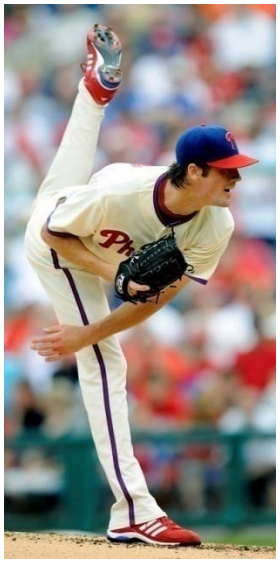


visualizing six dimensions of MLB pitches with lattice





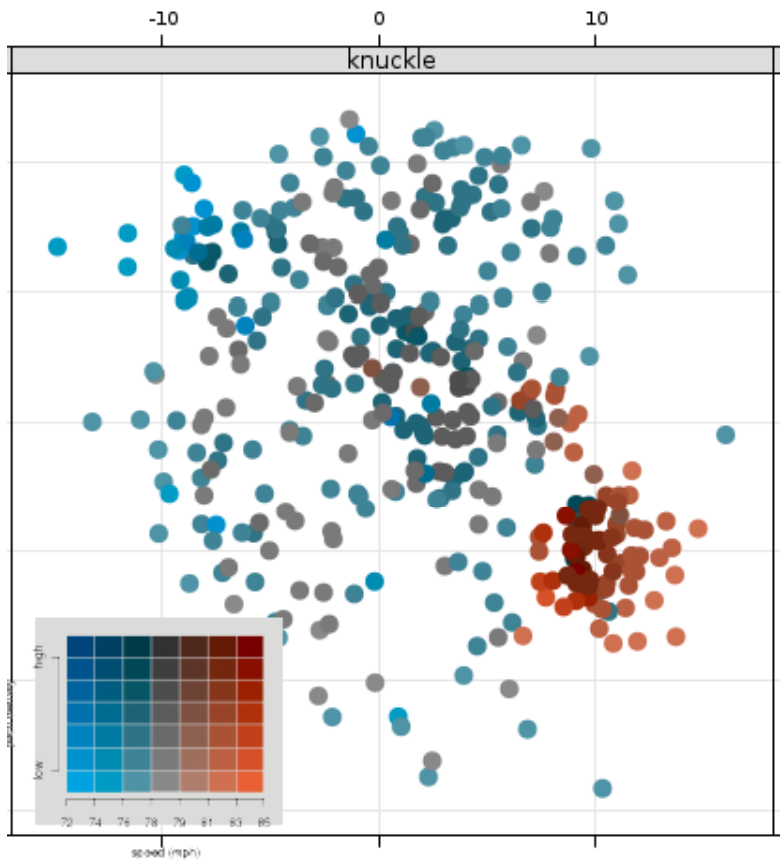




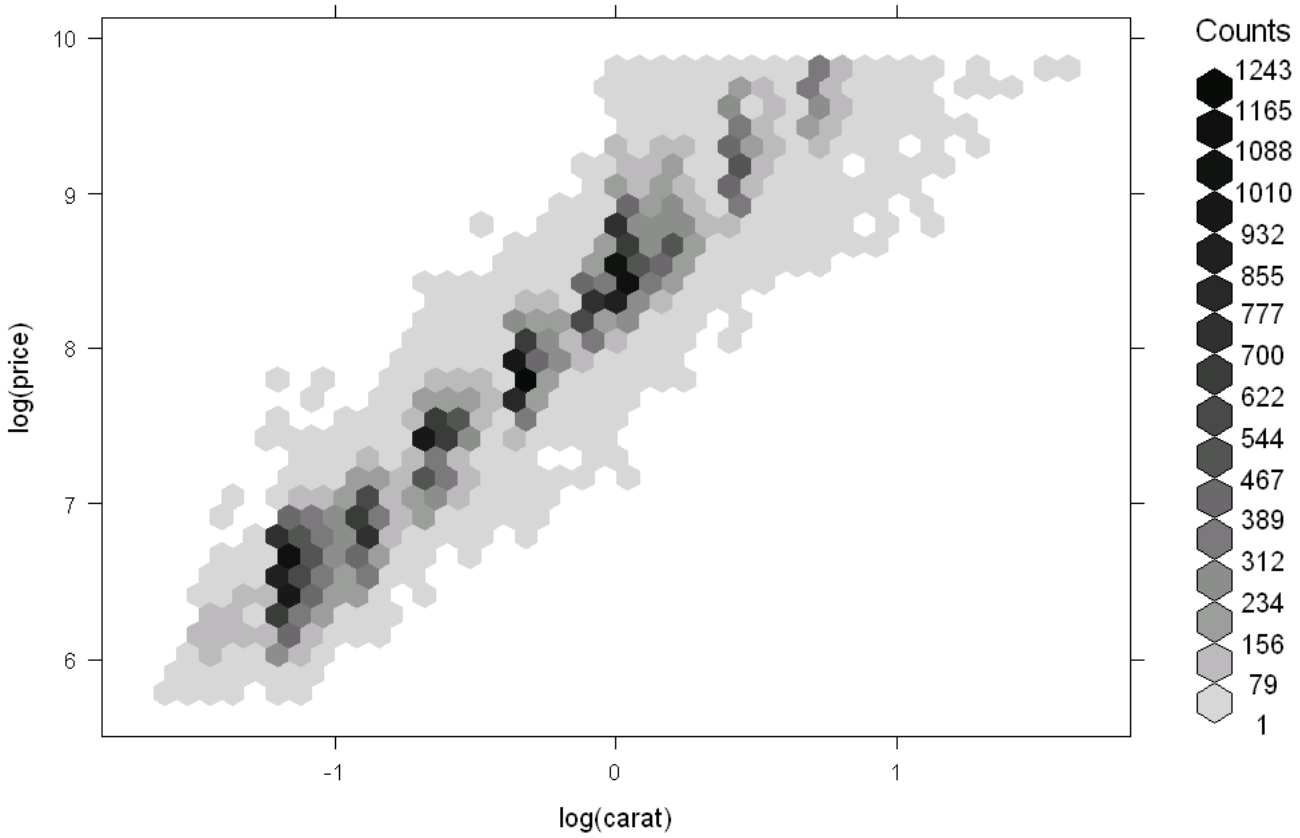
```
xyplot(x ~ y | type, data=pitch,
fill.color = pitch$color,
panel = function(x,y, fill.color, …, subscripts) {
  fill <- fill.color[subscripts]
  panel.xyplot(x, y, fill= fill, …) })
```

Beautiful Colors with Colorspace

```
library(“Colorspace”)
red <- LAB(50,64,64)
blue <- LAB(50,-48,-48)
mixcolor(10, red, blue)
```



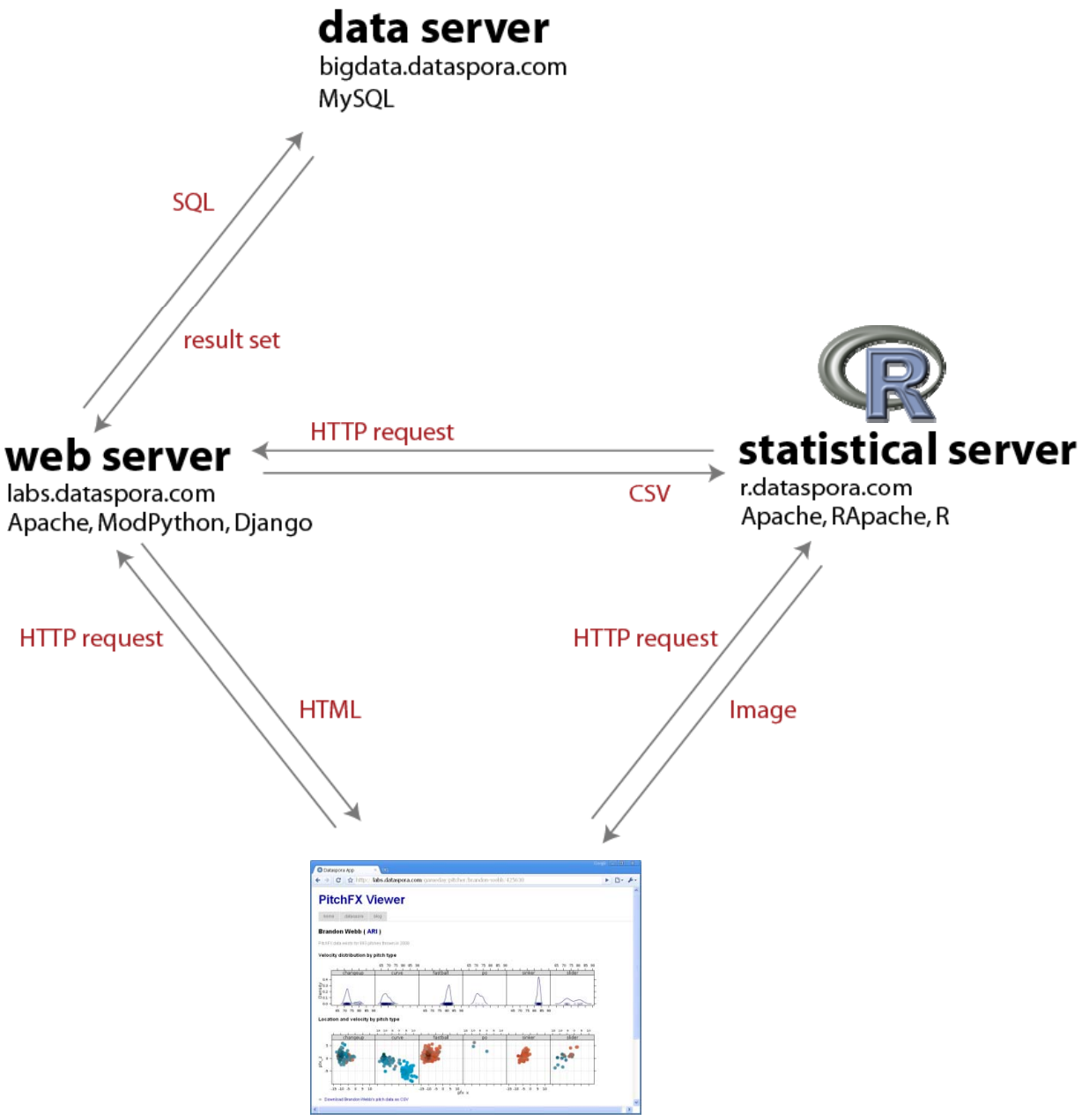
efficient plotting with `hexbinplot`

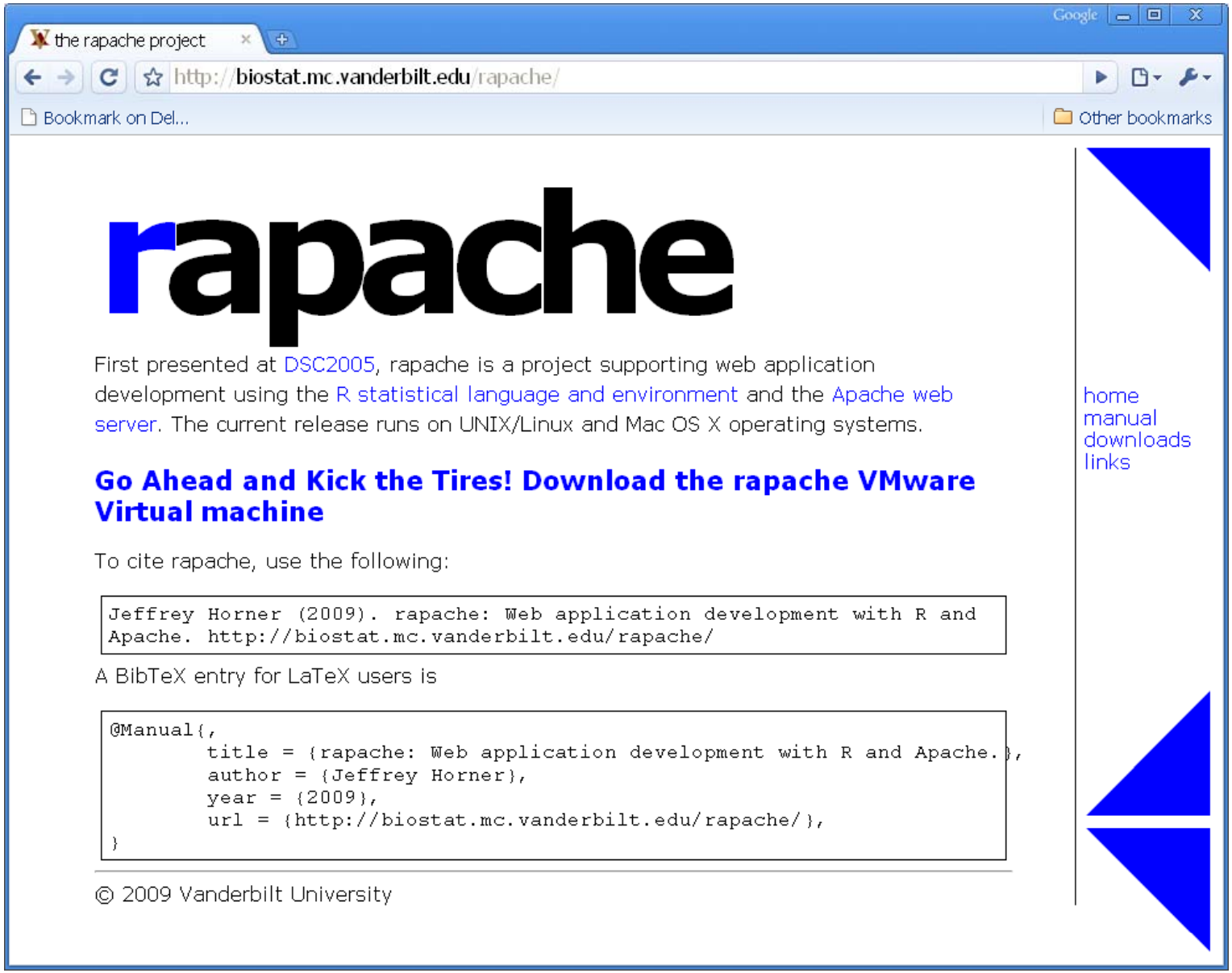



```
hexbinplot(log(price)~log(carat),data=diamonds,xbins=40)
```

Embedding R in a Web Server

Using Packages & R in a Server Environment


data server
bigdata.dataspora.com
MySQL
SQL
result set
web server
labs.dataspora.com
Apache, ModPython, Django
HTTP request
CSV
statistical server
r.dataspora.com
Apache, RApache, R
HTTP request
HTML
HTTP request
Image
PitchFX Viewer
Brandon Webb (ARI)
Velocity distribution by pitch type
Location and velocity by pitch type

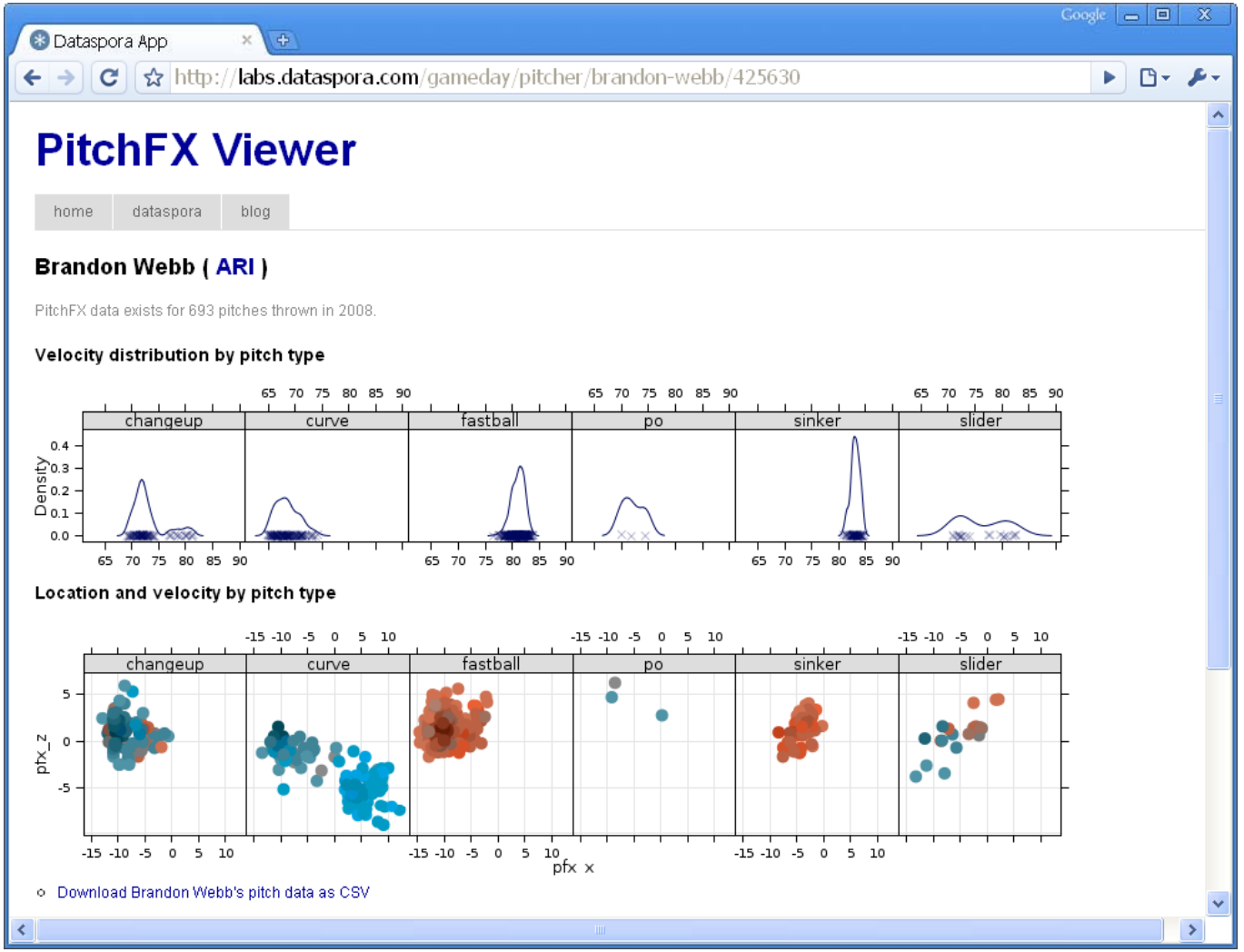

the rapache project
http://biostat.mc.vanderbilt.edu/rapache/
Bookmark on Del...
Other bookmarks
rapache
First presented at DSC2005, rapache is a project supporting web application development using the R statistical language and environment and the Apache web server. The current release runs on UNIX/Linux and Mac OS X operating systems.
Go Ahead and Kick the Tires! Download the rapache VMware Virtual machine
To cite rapache, use the following:
Jeffrey Horner (2009). rapache: Web application development with R and Apache. http://biostat.mc.vanderbilt.edu/rapache/
A BibTeX entry for LaTeX users is
@Manual{,
title = {rapache: Web application development with R and Apache.},
author = {Jeffrey Horner},
year = {2009},
url = {http://biostat.mc.vanderbilt.edu/rapache/},
}
© 2009 Vanderbilt University
home
manual
downloads
links

Linux

Apache

MySQL

R



http://labs.dataspora.com/gameday

Coding *vs* Clicking

```
File Edit Options Buffers Tools Imen
##  additional wrapper
pitchplot <- function(std.in,
                      plot="xyplot",
                      height=200,
                      model="pfx_x ~
                      ...) {

  lightblue <- LAB(50,-48,-48)
  lightred <- LAB(50,48,48)
  C <- plot2d(lightblue,lightred,60,
ab='density')

  n_pitch_type <- length(levels(std.

  height <- as.numeric(height)
  width <- n_pitch_type * 0.70 * hei
-uu-:---F1  urlAPI.R        61% L228
```