

An Introduction to *R*

Part 1: Basics

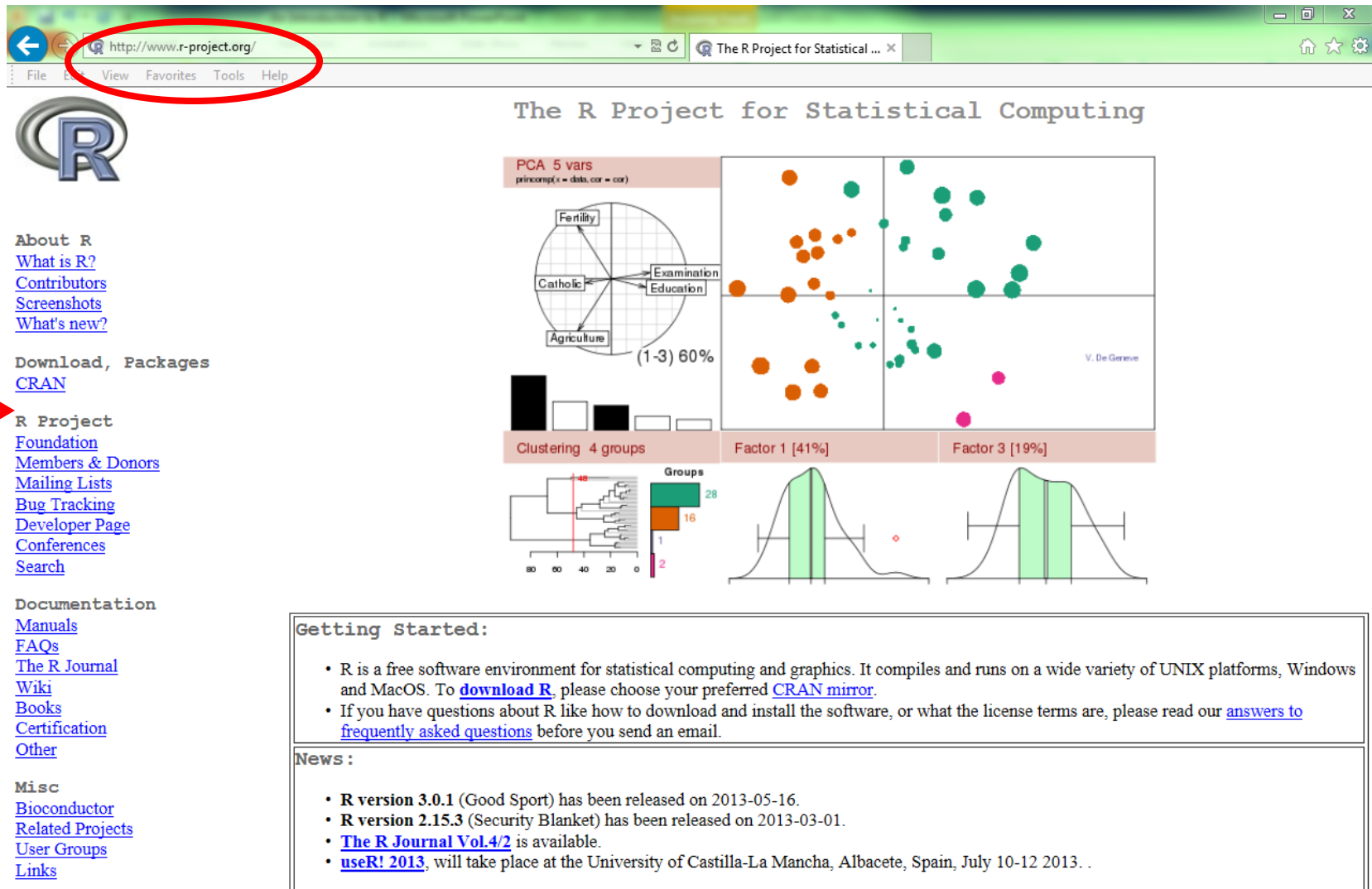
Kjell Johnson
Arbor Analytics, LLC



Overview

- Installing R
- Editors
- Data types
- Subsetting data
- Row and column information
- Importing data
- Installing packages
- Basics of visualizing data

Installing R from **r-project.org**



The screenshot shows the homepage of the R Project for Statistical Computing. The browser's address bar is circled in red, showing the URL <http://www.r-project.org/>. The page features the R logo, a navigation menu on the left, and a main content area with various statistical plots and a 'Getting Started' section.

About R

- [What is R?](#)
- [Contributors](#)
- [Screenshots](#)
- [What's new?](#)

Download, Packages

- [CRAN](#)

R Project

- [Foundation](#)
- [Members & Donors](#)
- [Mailing Lists](#)
- [Bug Tracking](#)
- [Developer Page](#)
- [Conferences](#)
- [Search](#)

Documentation

- [Manuals](#)
- [FAQs](#)
- [The R Journal](#)
- [Wiki](#)
- [Books](#)
- [Certification](#)
- [Other](#)

Misc

- [Bioconductor](#)
- [Related Projects](#)
- [User Groups](#)
- [Links](#)

The R Project for Statistical Computing

PCA 5 vars
prcomp(x = data, cor = cor)

Fertility
Catholic
Agriculture
Examination
Education
(1-3) 60%

Clustering 4 groups

Factor 1 [41%]
Factor 3 [19%]

Groups
28
16
1
2

V. De Geneuse

Getting Started:

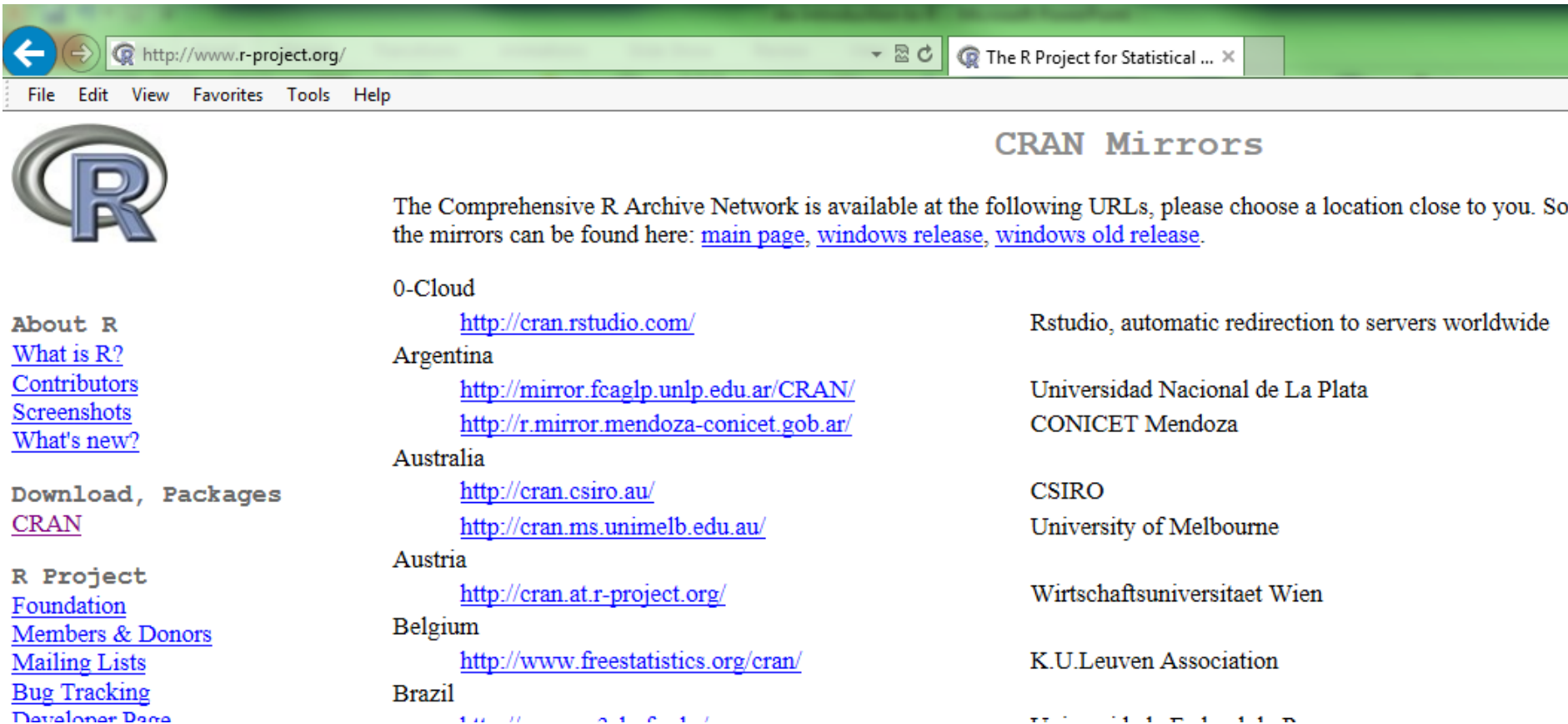
- R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS. To [download R](#), please choose your preferred [CRAN mirror](#).
- If you have questions about R like how to download and install the software, or what the license terms are, please read our [answers to frequently asked questions](#) before you send an email.

News:

- R version 3.0.1** (Good Sport) has been released on 2013-05-16.
- R version 2.15.3** (Security Blanket) has been released on 2013-03-01.
- [The R Journal Vol.4/2](#) is available.
- [useR! 2013](#), will take place at the University of Castilla-La Mancha, Albacete, Spain, July 10-12 2013. .

This server is hosted by the [Institute for Statistics and Mathematics](#) of [WU \(Wirtschaftsuniversität Wien\)](#).

Pick a Mirror Site



The screenshot shows a web browser window with the address bar displaying <http://www.r-project.org/>. The browser's menu bar includes File, Edit, View, Favorites, Tools, and Help. The page title is "The R Project for Statistical ...". The main content area features the R logo on the left and the heading "CRAN Mirrors" on the right. Below the heading, a paragraph states: "The Comprehensive R Archive Network is available at the following URLs, please choose a location close to you. So the mirrors can be found here: [main page](#), [windows release](#), [windows old release](#)." A table of mirrors follows, with columns for the mirror name, URL, and location. The table lists mirrors for 0-Cloud, Argentina, Australia, Austria, Belgium, and Brazil. On the left side of the page, there is a sidebar with links: "About R", "What is R?", "Contributors", "Screenshots", "What's new?", "Download, Packages", "CRAN", "R Project", "Foundation", "Members & Donors", "Mailing Lists", "Bug Tracking", and "Developer Page".

CRAN Mirrors

The Comprehensive R Archive Network is available at the following URLs, please choose a location close to you. So the mirrors can be found here: [main page](#), [windows release](#), [windows old release](#).

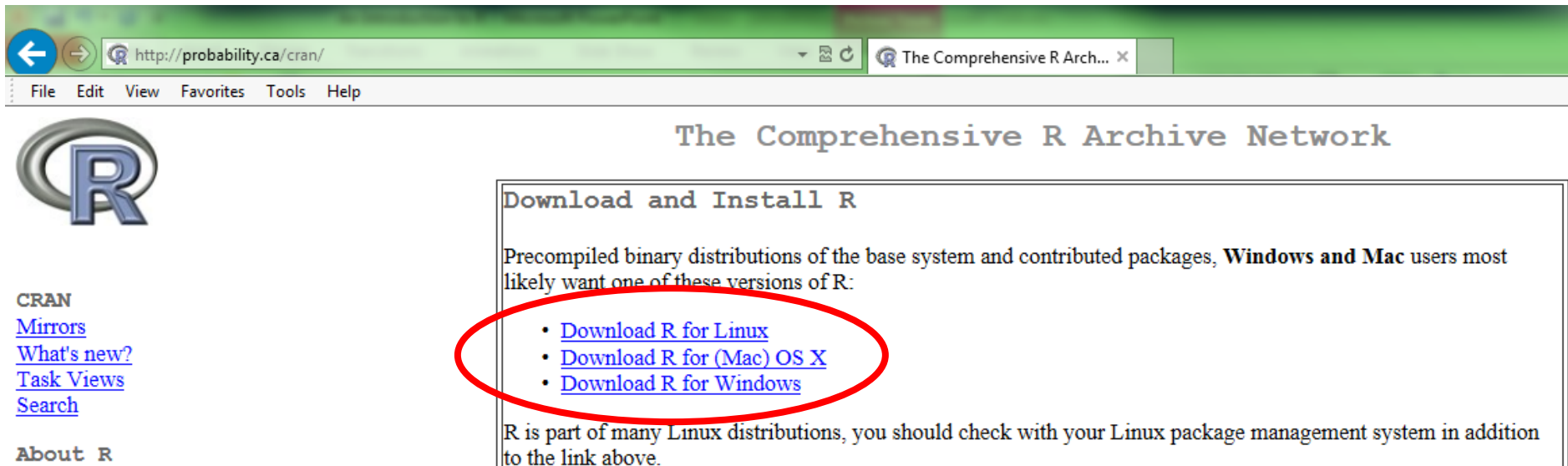
0-Cloud	http://cran.rstudio.com/	Rstudio, automatic redirection to servers worldwide
Argentina	http://mirror.fcaglp.unlp.edu.ar/CRAN/ http://r.mirror.mendoza-conicet.gob.ar/	Universidad Nacional de La Plata CONICET Mendoza
Australia	http://cran.csiro.au/ http://cran.ms.unimelb.edu.au/	CSIRO University of Melbourne
Austria	http://cran.at.r-project.org/	Wirtschaftsuniversitaet Wien
Belgium	http://www.freeststatistics.org/cran/	K.U.Leuven Association
Brazil	http://cran.usp.br/	USP

About R
[What is R?](#)
[Contributors](#)
[Screenshots](#)
[What's new?](#)

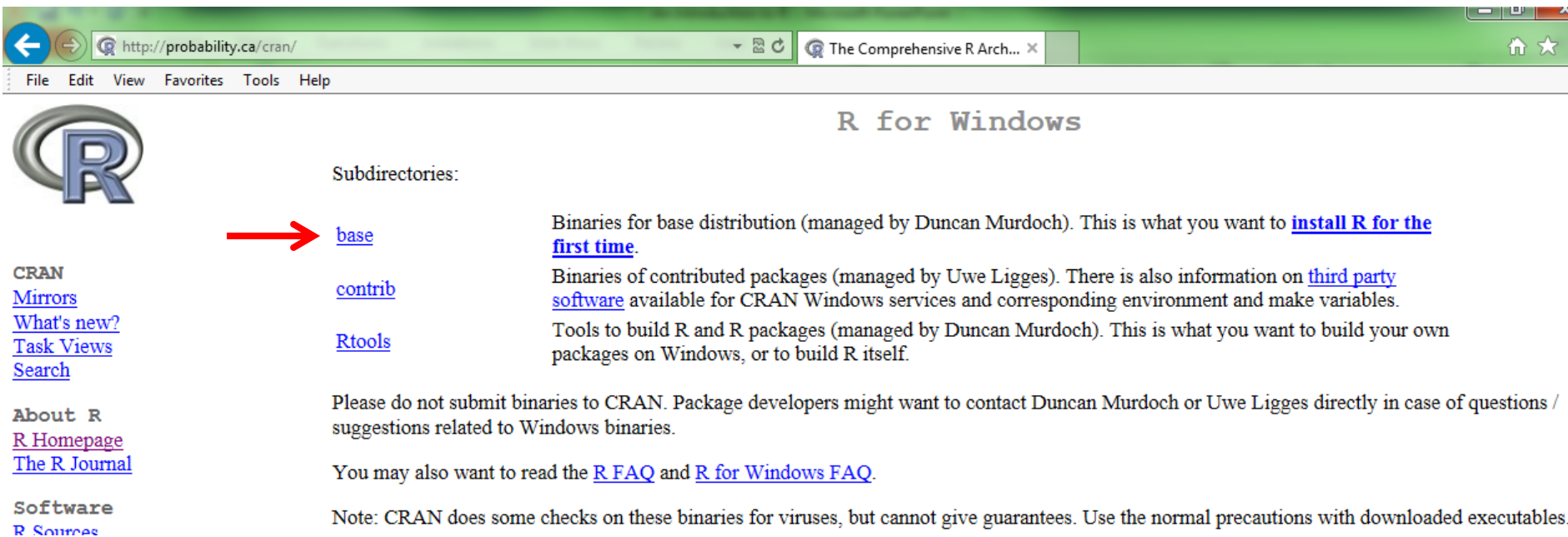
Download, Packages
[CRAN](#)

R Project
[Foundation](#)
[Members & Donors](#)
[Mailing Lists](#)
[Bug Tracking](#)
[Developer Page](#)

Select Your Operating System



Select “base”



The screenshot shows a web browser window with the address bar displaying <http://probability.ca/cran/>. The page title is "R for Windows". On the left side, there is a navigation menu with links: "CRAN", "Mirrors", "What's new?", "Task Views", "Search", "About R", "R Homepage", "The R Journal", "Software", and "R Sources". The main content area is titled "Subdirectories:" and lists three options: "base", "contrib", and "Rtools". A red arrow points to the "base" link. To the right of each subdirectory link is a brief description of its contents. Below the subdirectories, there is a note about submitting binaries to CRAN and a link to the "R FAQ" and "R for Windows FAQ". At the bottom, there is a note about CRAN's virus checks.

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

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

Software
[R Sources](#)

R for Windows

Subdirectories:

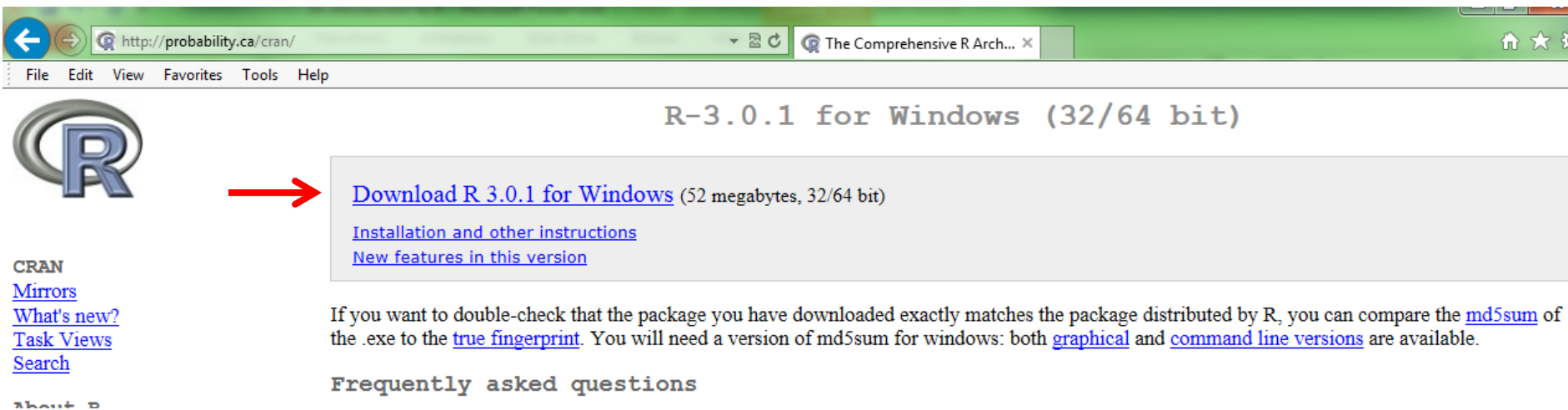
- [base](#): Binaries for base distribution (managed by Duncan Murdoch). This is what you want to [install R for the first time](#).
- [contrib](#): Binaries of contributed packages (managed by Uwe Ligges). There is also information on [third party software](#) available for CRAN Windows services and corresponding environment and make variables.
- [Rtools](#): Tools to build R and R packages (managed by Duncan Murdoch). This is what you want to build your own packages on Windows, or to build R itself.

Please do not submit binaries to CRAN. Package developers might want to contact Duncan Murdoch or Uwe Ligges directly in case of questions / suggestions related to Windows binaries.

You may also want to read the [R FAQ](#) and [R for Windows FAQ](#).

Note: CRAN does some checks on these binaries for viruses, but cannot give guarantees. Use the normal precautions with downloaded executables.

Download R and Follow Prompts



The screenshot shows a web browser window with the address bar displaying <http://probability.ca/cran/>. The browser has a menu bar with File, Edit, View, Favorites, Tools, and Help. The main content area features the R logo on the left and the heading "R-3.0.1 for Windows (32/64 bit)" in the center. Below the heading, there is a grey box containing the link "Download R 3.0.1 for Windows (52 megabytes, 32/64 bit)" and two sub-links: "Installation and other instructions" and "New features in this version". A red arrow points to the main download link. To the left of the R logo, there is a list of links: "CRAN", "Mirrors", "What's new?", "Task Views", "Search", and "About R". Below the grey box, there is a paragraph of text and a link to "Frequently asked questions".

R-3.0.1 for Windows (32/64 bit)

[Download R 3.0.1 for Windows](#) (52 megabytes, 32/64 bit)

[Installation and other instructions](#)

[New features in this version](#)

If you want to double-check that the package you have downloaded exactly matches the package distributed by R, you can compare the [md5sum](#) of the .exe to the [true fingerprint](#). You will need a version of md5sum for windows: both [graphical](#) and [command line versions](#) are available.

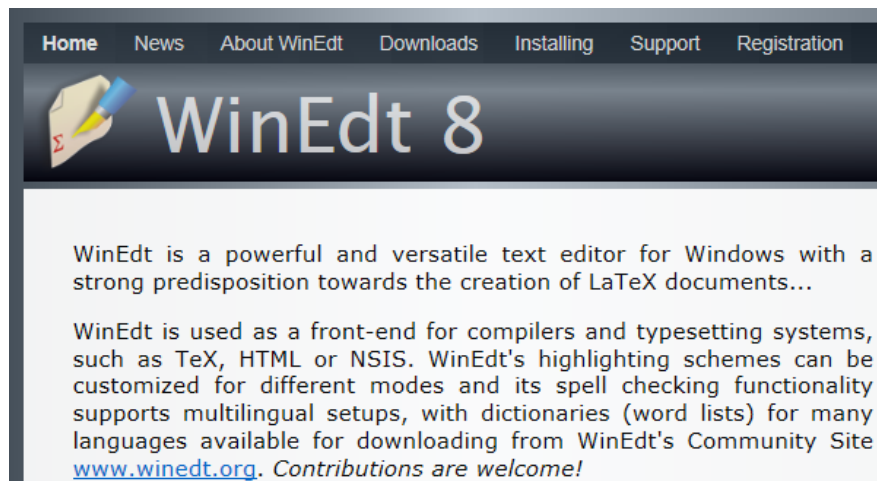
[Frequently asked questions](#)

Editors for R

- Any text editor will work:
 - Notepad, Wordpad, Word, vi, etc.
- More sophisticated editing can be done with:

winedt.com

rstudio.com



Welcome to RStudio

Software, education, and services for the R community



Powerful IDE for R

RStudio IDE is a powerful and productive user interface for R. It's free and open source, and works great on Windows, Mac, and Linux.

[Download now](#)

[Learn more](#)

R training and education

We've got hands-on courses for beginners and even R experts. Customize an on-site training or enroll in one of our public workshops.

[Request on-site](#)

[View courses](#)

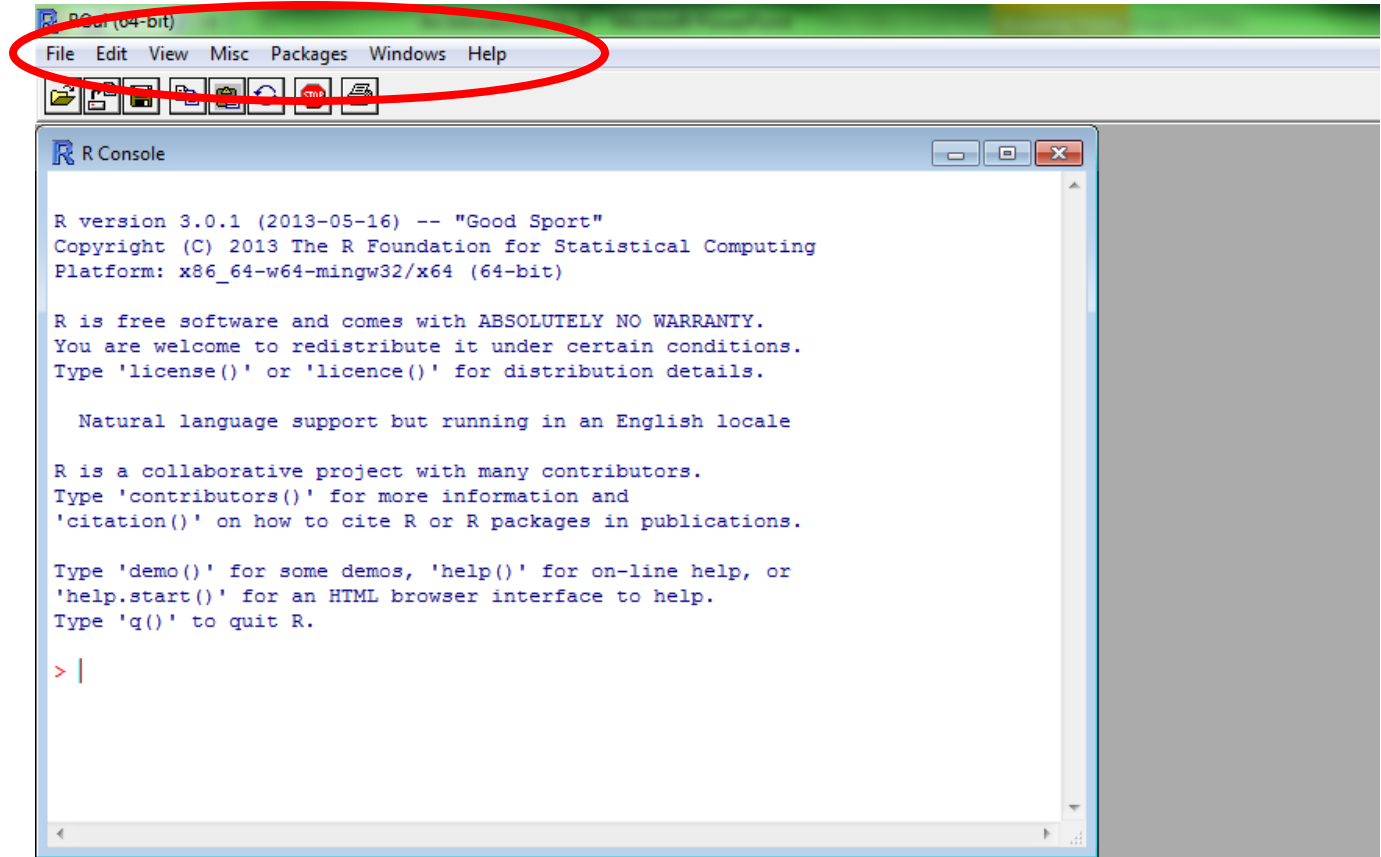
Open source R packages

Our developers and expert trainers are the authors of several popular R packages, including ggplot2, plyr, lubridate, and others.

[See projects](#)

The R Console

Let's briefly
explore the
menu items



We enter code
at the prompt:

Beginning Notes

- Beware: R is case sensitive!
- Help documentation can be found by typing `help(function name)` or `?(function name)`
 - If you only remember part of the name, then using `??(function name)` will search for any help documentation with that name
- The up arrow at the command prompt gives previous command prompt entries

Telling R Where to Work

The `setwd` function sets the working directory for R:

```
setwd("c:/Part1")
```

Note: R requires forward slashes between levels (or double backslashes):

```
setwd("c:\\Part1")
```

Alternatively we can define an object with the file location and then set the working directory:

```
myLocation <- "c:/Part1"  
setwd(myLocation)
```

Notes:

`myLocation` is an object in the current R session

We can see what objects are in the current session with the `ls()` command

We can see what directory we are in with the `getwd()` command

Creating Data in R

A vector is an object that contains elements. Elements can be numeric:

```
scores <- c(89, 102, 73, 54, 92, 27)
```

Or elements can be character:

```
sample <- c("a", "a", "b", "b", "c", "c")
```

To check to see if a vector is numeric or character use the following functions:

```
is.numeric(scores)
```

```
is.character(sample)
```

To see the contents of scores or sample, type `scores` or `sample` at the R prompt

Functions like `head()` or `tail()` give the first 5 or last 5 elements of a data set.

Factor

- A *factor* is an important data structure in R
- This data structure contains the original values of the variable as well as labels corresponding to each “level” of the factor
 - Levels are often necessary for graphing and analyzing data

To convert an object to a factor:

```
sample.f <- factor(sample)
is.factor(sample.f)
sample.f
```

Matrices

- A *matrix* is a two dimensional object that contains data of the same type (i.e. only numeric or only character)
- Matrices are more efficient for some operations, but not as convenient as data frames...

Create a matrix:

```
prepTime <- c(9,10,7,4,9,3)  
myMatrix <- cbind(prepTime,scores)
```

The function `dim()` tells us the number of rows and columns in the matrix. See also `nrow()` and `ncol()`

Data Frame

- A *data frame* is another important data structure in R. A data frame is often rectangular with rows (samples) and columns (variables)
 - Data frames can contain a mix of character, numeric, and factor variables

Create a data frame:

```
myData <- data.frame(sampleID = sample.f, Score = scores,  
                     Prep = prepTime)
```

`sampleID`, `Score`, and `Prep` are the column names that contain the data from `sample.f`, `scores`, and `prepTime`.

The function `colnames()` returns the column names of the data frame

The function `rownames()` returns the row names of the data frame

Accessing Information in a Data Frame

To access data in an individual column of a data frame, we use the `$` operator:

```
myData$Score
```

```
myData$sampleID
```

```
myData$Prep
```

We can also access data in a data frame using row and column location identification:

All of the data in the first column:

```
myData[,1]
```

All of the data in the second row:

```
myData[2,]
```

Rows 3 through 5 of the second column:

```
myData[3:5,2]
```

All of the data in rows 2, 4, and 6:

```
myData[c(2,4,6),]
```


Other Ways to Subset or Select Data

The subset function:

```
?subset
```

Select samples with scores greater than 50:

```
subset(myData, Score > 50)
```

Select samples with scores greater than 50 and less than or equal to 89:

```
subset(myData, Score > 50 & Score <= 89)
```

Select samples with sampleID's of "a":

```
subset(myData, sampleID == "a")
```

Select samples with sampleID's of "a" or "b":

```
subset(myData, sampleID %in% c("a", "b"))
```

Select samples with scores greater than 50 and keep the sampleID and Score variables:

```
subset(myData, Score > 50, select=c("sampleID", "Score"))
```

Other Ways to Subset or Select Data

Logical constraints with row and column referencing:

Select samples with scores greater than 50:

```
myData[myData$Score > 50,]
```

Select samples with scores greater than 50 and less than or equal to 89:

```
myData[myData$Score > 50 & myData$Score <= 89,]
```

Select samples with sampleID's of "a":

```
myData[myData$sampleID == "a",]
```

Select samples with sampleID's of "a" or "b":

```
myData[myData$sampleID %in% c("a", "b"),]
```

Select samples with scores greater than 50 and keep the sampleID and Score variables:

```
myData[myData$Score > 50, c("sampleID", "Score")]
```

Renaming Rows and Columns

A few options:

1) Rename all rows or columns

```
rownames(myData) <- c("Kermit", "Miss Piggy", "Rizzo",  
                      "Gonzo", "Animal", "Fozzy")  
colnames(myData) <- c("Column1", "Column2")
```

2) Renaming one row or column:

```
rownames(myData)[rownames(myData) == "Kermit"] <- "Ralph"  
colnames(myData)[colnames(myData) == "Column2"] <- "C2"
```

3) Renaming rows or columns by index numbers:

```
rownames(myData)[1:2] <- c("Ralph", "Swedish Chef")  
colnames(myData)[1] <- "C1"
```

Missing Data

Suppose that the last score was missing. That entry needs to be represented with a value of NA:

```
scores <- c(89,102,73,54,92,NA)
is.numeric(scores)
```

Using another value to represent missing will change the object's type:

```
scores <- c(89,102,73,54,92,".")
is.numeric(scores)
```

What can we do with missing values?

Omit them with the `na.omit` function:

```
myData <- data.frame(sampleID = sample.f, Score = scores,
                     Prep = prepTime)
myData.omit <- na.omit(myData)
```

Replace them using logical referencing:

```
myData.replace <- myData
myData.replace$Score[is.na(myData.replace$Score)] <- 0
```

The Problem with Missing Data; Other Useful Functions

Let's start with:

```
scores <- c(89, 102, 73, 54, 92, NA)
```

Find the mean of the vector:

```
mean(scores)
```

Oops...

Omit them:

```
mean(scores, na.rm = TRUE)
```

Other useful functions:

```
mean()
```

```
sd()
```

```
summary()
```

```
str()
```

What Objects are in the R Session?

The `ls()` command list all of the objects in the current session:

```
ls()
```

We may want to eliminate an object that we've created. The `rm()` function does that:

```
rm(sample)
```

If we have been working in R for awhile, objects may need to be cleared from the session:

```
rm(list=ls())
```

Or we may want to clear all but a few objects:

```
rm(list= ls()[!(ls() %in% c('myData','sample','scores'))])
```

Reading External Data

- Example data:
 - Forced expiratory volume from 654 children
 - Several measurements were collected from each child:

	A	B	C	D	E
1	age	FEV	height	sex	smoke
2	9	1.708	57	female	nonsmoker
3	8	1.724	67.5	female	nonsmoker
4	7	1.72	54.5	female	nonsmoker
5	9	1.558	53	male	nonsmoker
6	9	1.895	57	male	nonsmoker
7	8	2.336	61	female	nonsmoker
8	6	1.919	58	female	nonsmoker
9	6	1.415	56	female	nonsmoker
10	8	1.987	58.5	female	nonsmoker
11	9	1.942	60	female	nonsmoker
12	6	1.602	53	female	nonsmoker
13	8	1.735	54	male	nonsmoker

Reading External Data

- We usually want to read in an external file
 - There are several functions to do this

First set the working directory:

```
myLocation <- "c:/Part1"  
setwd(myLocation)
```

For .csv files, we can use the `read.csv` function:

```
fev <- read.csv("fev_dat.csv", header = TRUE)
```

Or:

```
fev <- read.table("fev_dat.csv", header = TRUE, sep = ",")
```

`fev` is now a data frame. We can check the contents with the `str()` function

Tab delimited text files can be read with the `read.table` function:

```
fev <- read.table("fev_dat.txt", header = TRUE, sep = "\t")
```


Other Important Options with the Read Function

- `as.is = TRUE or FALSE`
 - FALSE: converts character variables to factors
 - TRUE: does not convert character variables to factors
- `na.strings`
 - A character vector of strings that are to be interpreted as missing values. For example if “.” and “miss” represent missing values, then we would define:
 - `na.strings = c(".", "miss")`
- `Skip`
 - The number of rows to skip before reading the file

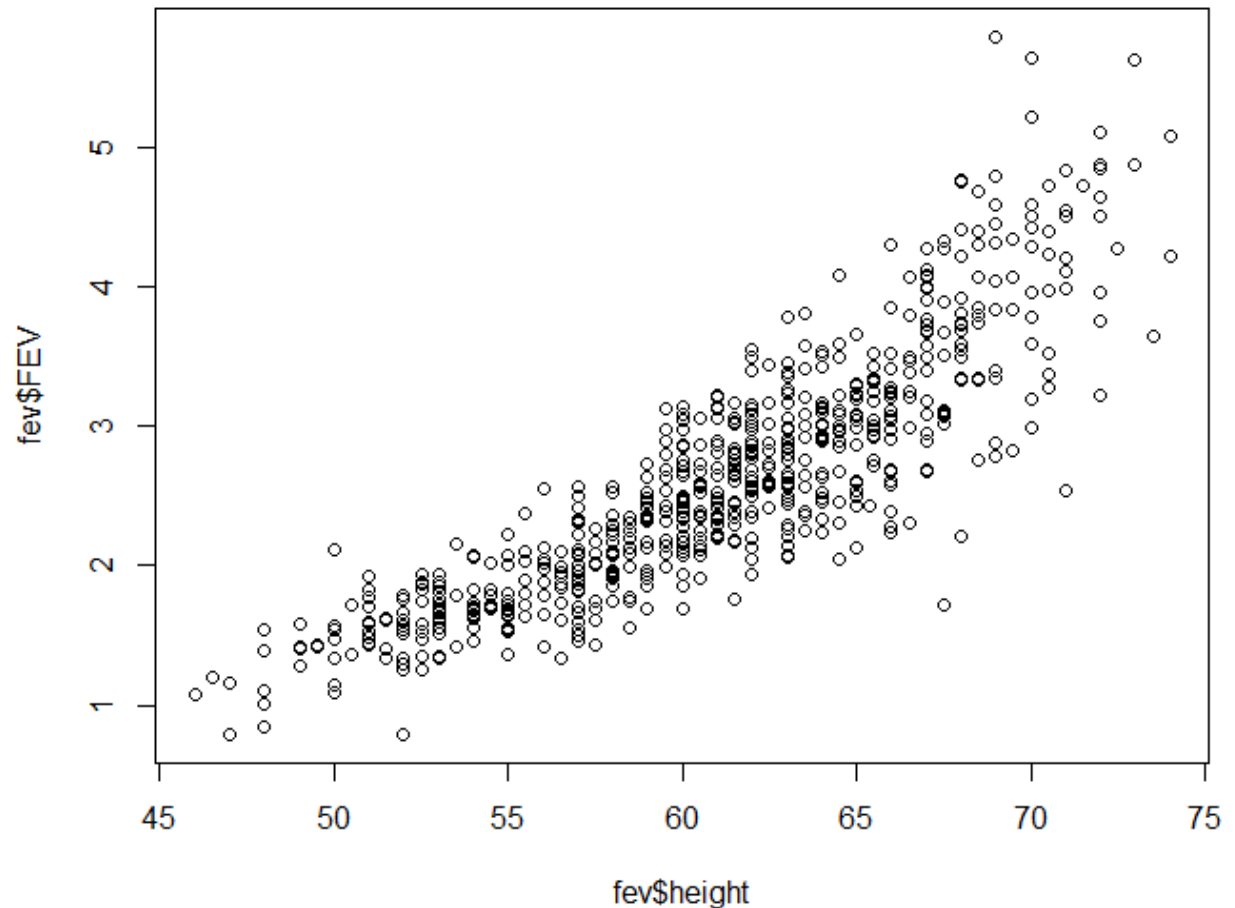
Visualizing Data

- Once the data is in R, we should begin to understand relationships between variables. Visualization is a good way to do this
- R has LOTS of ways to visualize data. Some are simple and straightforward...others are NOT.
- Let's examine the relationship between height and fev

The `plot()` function takes as input the x-axis variable and y-axis variable:
`plot(fev$height, fev$FEV)`

Output

Decent output for rough investigation, but we can customize the output further with a title, axis labels, different symbols, colors, font sizes, etc...



R Symbols

○ 1	△ 2	⊕ 3	× 4	◇ 5
▽ 6	⊠ 7	⋆ 8	⬡ 9	⊕ 10
⬠ 11	⊞ 12	⊗ 13	⊞ 14	■ 15
● 16	▲ 17	◆ 18	● 19	● 20
○ 21	□ 22	◇ 23	△ 24	▽ 25

R Colors

(and there are many more!)

R colors

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75
76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100
101	102	103	104	105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125
126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146	147	148	149	150
151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192	193	194	195	196	197	198	199	200
201	202	203	204	205	206	207	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225
226	227	228	229	230	231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250
251	252	253	254	255	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275
276	277	278	279	280	281	282	283	284	285	286	287	288	289	290	291	292	293	294	295	296	297	298	299	300
301	302	303	304	305	306	307	308	309	310	311	312	313	314	315	316	317	318	319	320	321	322	323	324	325
326	327	328	329	330	331	332	333	334	335	336	337	338	339	340	341	342	343	344	345	346	347	348	349	350
351	352	353	354	355	356	357	358	359	360	361	362	363	364	365	366	367	368	369	370	371	372	373	374	375
376	377	378	379	380	381	382	383	384	385	386	387	388	389	390	391	392	393	394	395	396	397	398	399	400
401	402	403	404	405	406	407	408	409	410	411	412	413	414	415	416	417	418	419	420	421	422	423	424	425
426	427	428	429	430	431	432	433	434	435	436	437	438	439	440	441	442	443	444	445	446	447	448	449	450
451	452	453	454	455	456	457	458	459	460	461	462	463	464	465	466	467	468	469	470	471	472	473	474	475
476	477	478	479	480	481	482	483	484	485	486	487	488	489	490	491	492	493	494	495	496	497	498	499	500
501	502	503	504	505	506	507	508	509	510	511	512	513	514	515	516	517	518	519	520	521	522	523	524	525
526	527	528	529	530	531	532	533	534	535	536	537	538	539	540	541	542	543	544	545	546	547	548	549	550
551	552	553	554	555	556	557	558	559	560	561	562	563	564	565	566	567	568	569	570	571	572	573	574	575
576	577	578	579	580	581	582	583	584	585	586	587	588	589	590	591	592	593	594	595	596	597	598	599	600
601	602	603	604	605	606	607	608	609	610	611	612	613	614	615	616	617	618	619	620	621	622	623	624	625
626	627	628	629	630	631	632	633	634	635	636	637	638	639	640	641	642	643	644	645	646	647	648	649	650
651	652	653	654	655	656	657																		

1	white	#FFFFFF	255	255	255
2	aliceblue	#F0F8FF	240	248	255
3	antiquewhite	#FAEBD7	250	235	215
4	antiquewhite1	#FPEFDB	255	239	219
5	antiquewhite2	#EEDFCC	238	223	204
6	antiquewhite3	#CDC0B0	205	192	176
7	antiquewhite4	#B8B788	139	131	120
8	aquamarine	#7FFFD4	127	255	212
9	aquamarine1	#7FFFD4	127	255	212
10	aquamarine2	#76E8C6	118	238	198
11	aquamarine3	#66CDAA	102	205	170
12	aquamarine4	#458B74	69	139	116
13	azure	#F0FFFF	240	255	255
14	azure1	#F0FFFF	240	255	255
15	azure2	#E0EEEE	224	238	238
16	azure3	#C1CDCD	193	205	205
17	azure4	#838B8B	131	139	139
18	beige	#F5F5DC	245	245	220
19	bisque	#FFE4C4	255	228	196
20	bisque1	#FFE4C4	255	228	196
21	bisque2	#EED5B7	238	213	183
22	bisque3	#CDB79E	205	183	158
23	bisque4	#8B7D6B	139	125	107
24	black	#000000	0	0	0
25	blanchedalmond	#FFEB3D	255	235	205
26	blue	#0000FF	0	0	255
27	blue1	#0000FF	0	0	255
28	blue2	#0000EE	0	0	238
29	blue3	#0000CD	0	0	205
30	blue4	#00008B	0	0	139
31	blueviolet	#8A2BE2	138	43	226
32	brown	#A52A2A	165	42	42
33	brown1	#FF4040	255	64	64
34	brown2	#8B3838	238	59	59
35	brown3	#CD3333	205	51	51
36	brown4	#8B2323	139	35	35
37	burlywood	#DEB887	222	184	135
38	burlywood1	#FFD39B	255	211	155
39	burlywood2	#EBC991	238	197	145
40	burlywood3	#CDA07D	205	170	125
41	burlywood4	#8B7355	139	115	85
42	cadetblue	#5F9EA0	95	158	160
43	cadetblue1	#98F5FF	152	245	255
44	cadetblue2	#8EE5EE	142	229	238
45	cadetblue3	#7AC5CD	122	197	205
46	cadetblue4	#53868B	83	134	139
47	chartreuse	#7FFF00	127	255	0
48	chartreuse1	#7FFF00	127	255	0
49	chartreuse2	#76EE00	118	238	0
50	chartreuse3	#66CD00	102	205	0
51	chartreuse4	#458B00	69	139	0
52	chocolate	#D2691E	210	105	30
53	chocolate1	#FF7F24	255	127	36
54	chocolate2	#EE7621	238	118	33
55	chocolate3	#CD661D	205	102	29
56	chocolate4	#8B4513	139	69	19
57	coral	#FF7F50	255	127	80
58	coral1	#FF7256	255	114	86
59	coral2	#EE6A50	238	106	80
60	coral3	#CD5B45	205	91	69
61	coral4	#8B3E2F	139	62	47
62	cornflowerblue	#6495ED	100	149	237
63	comsilk	#FFF9DC	255	248	220
64	comsilk1	#FFF9DC	255	248	220
65	comsilk2	#EEE9CD	238	232	205
66	comsilk3	#CDC9B1	205	200	177
67	comsilk4	#8B8778	139	136	120
68	cyan	#00FFFF	0	255	255
69	cyan1	#00FFFF	0	255	255
70	cyan2	#00EEEE	0	238	238
71	cyan3	#00CDCD	0	205	205
72	cyan4	#008B8B	0	139	139
73	darkblue	#00008B	0	0	139
74	darkcyan	#008B8B	0	139	139
75	darkgoldenrod	#8B690B	184	134	11
76	darkgoldenrod1	#FFB90F	255	185	15
77	darkgoldenrod2	#EAD03E	238	173	14
78	darkgoldenrod3	#CD950C	205	149	12
79	darkgoldenrod4	#8B6508	139	101	8
80	darkgray	#A9A9A9	169	169	169
81	darkgreen	#006400	0	100	0
82	darkgrey	#A9A9A9	169	169	169
83	darkkhaki	#8B764B	189	183	107
84	darkmagenta	#8B008B	139	0	139
85	darkolivegreen	#556B2F	85	107	47
86	darkolivegreen1	#C8FF70	202	255	112
87	darkolivegreen2	#BCE660	188	238	104
88	darkolivegreen3	#A2CD5A	162	205	90
89	darkolivegreen4	#6E8B3D	110	139	61
90	darkorange	#FF8C00	255	140	0
91	darkorange1	#FF7F00	255	127	0
92	darkorange2	#EE7600	238	118	0
93	darkorange3	#CD6600	205	102	0
94	darkorange4	#8B4500	139	69	0
95	darkorchid	#9932CC	153	50	204
96	darkorchid1	#FF7F00	191	62	255
97	darkorchid2	#B23A8B	178	58	238
98	darkorchid3	#9A32CD	154	50	205
99	darkorchid4	#68228B	104	34	139
100	darkred	#8B0000	139	0	0

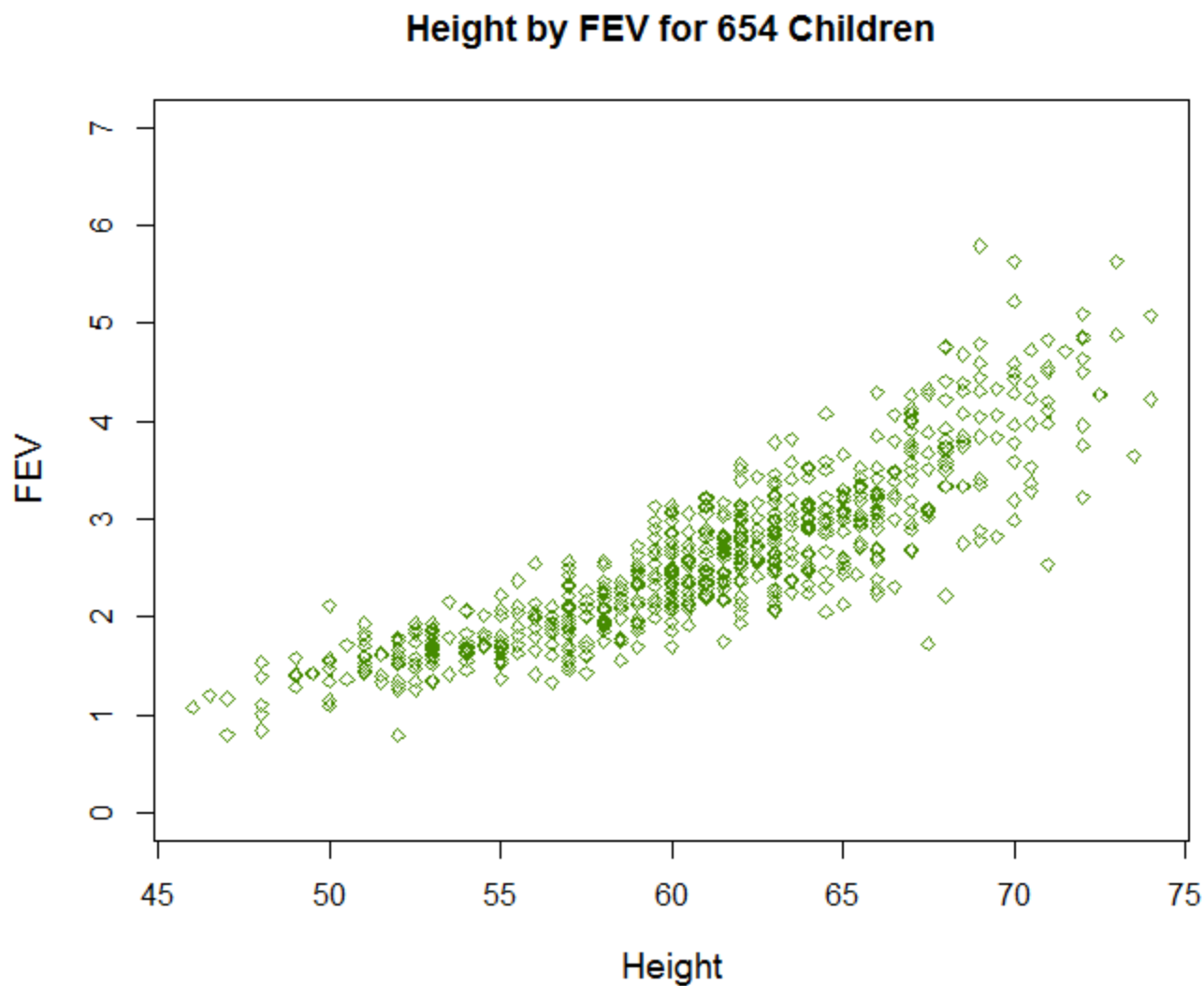
Customizing a Simple Plot

```
plot(fev$height, fev$FEV,  
     xlab = list("Height", cex=1.2),  
     ylab = list("FEV", cex=1.2),  
     ylim = c(0, 7),  
     main = "Height by FEV for 654 Children",  
     col = "chartreuse4",  
     pch = 5,  
     cex = 0.7)
```

This plot statement does the following:

- changes the x-axis label to Height and makes it 1.2 times larger than normal
- changes the y-axis label to FEV and makes it 1.2 times larger than normal
- changes the y-axis range to 0 to 7
- adds the title of Height by FEV for 654 Children
- changes the symbol color to chartreuse 4
- changes the symbol to an open diamond
- makes the symbols 70% of normal size

Modified Output



How to Save the Graph?

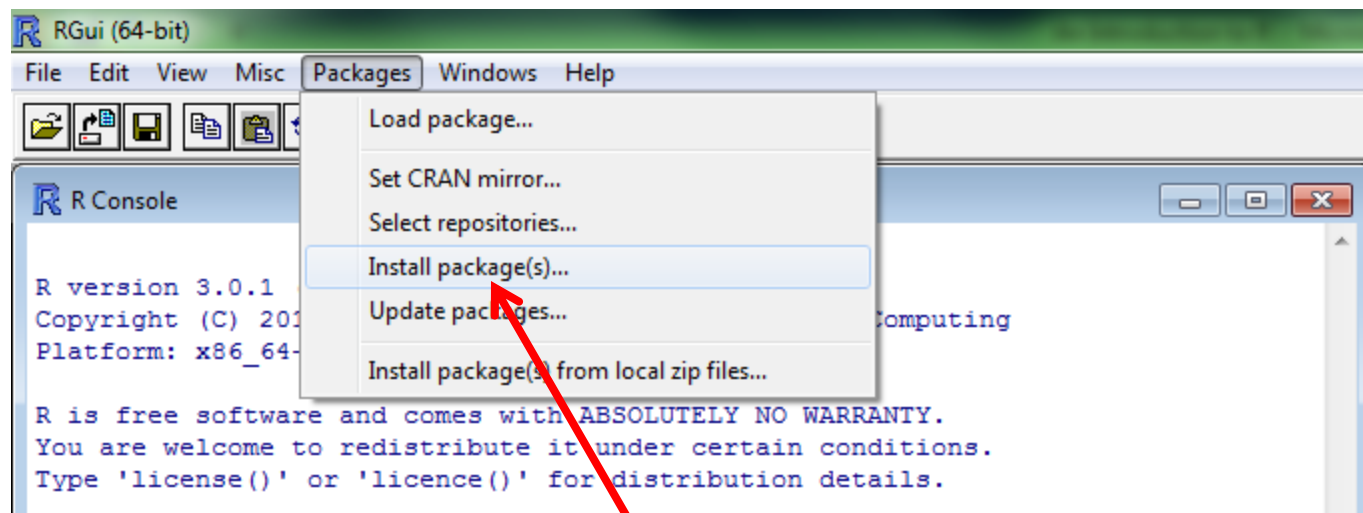
- A couple approaches
 - Right-click the graph and manually save it, OR...

```
png(filename = "FEVplot.png", width=600, height=600)
plot(fev$height, fev$FEV,
     xlab = list("Height", cex=1.2),
     ylab = list("FEV", cex=1.2),
     ylim = c(0, 7),
     main = "Height by FEV for 654 Children",
     col = "chartreuse4",
     pch = 5,
     cex = 0.7)
dev.off()
```


Using Packages

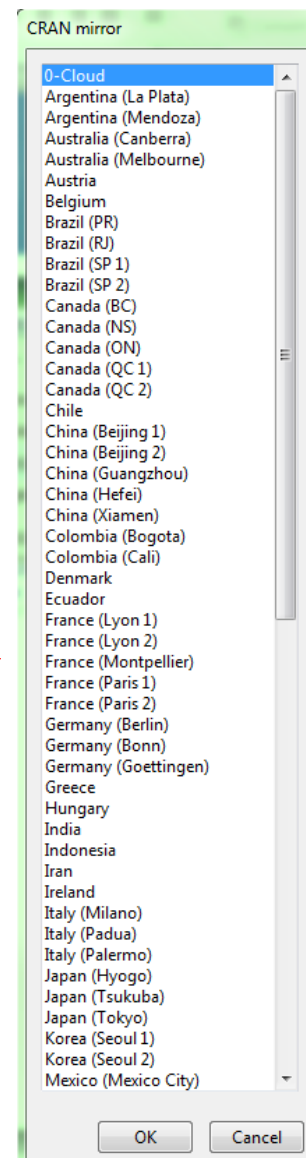
- Packages are pre-packaged code to implement specific types of analyses, visualizations, etc.
- Anyone can develop a package, and submit it for posting on the CRAN site
 - User beware!
- How do you obtain packages?
 - A couple of ways
- Let's download the `ggplot2` package, which generates more elegant figures

Installing Packages Through the Menu

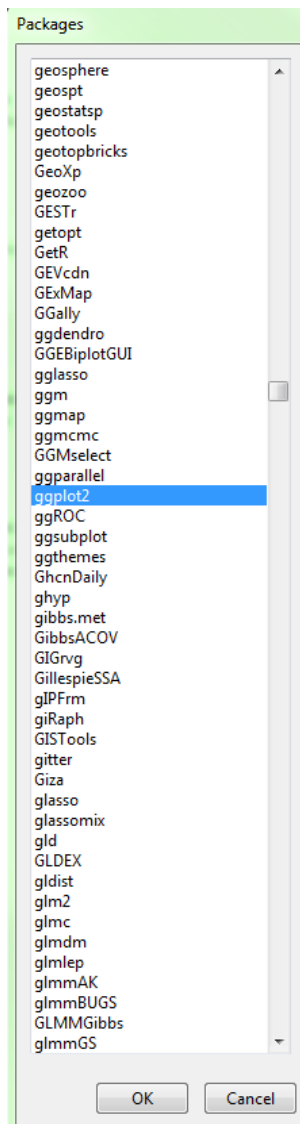


Select “Install
package(s)”,

then select a CRAN
mirror



Select a Package



R will install the package and any other dependent packages if not already installed.

Command Prompt Approach for Installing and Updating Packages

- Another convenient approach is to install a package through the command prompt

We can install one package:

```
install.packages("ggplot2", dependencies=TRUE)
```

Or we can install multiple packages at once:

```
graphicPackages = c("lattice", "ggplot2")  
install.packages(graphicPackages, dependencies=TRUE)
```

Note: the base R code is frequently updated. If you install a new version of R, you will also need to update the packages you have already installed in the new version of R. This is easy to do:

```
update.packages()
```

Loading a Package

- Once a package is installed, it must be loaded into the current R session

Loading a package:

```
library(ggplot2)
```

Or we can install multiple packages at once:

```
graphicPackages = c("lattice", "ggplot2")  
install.packages(graphicPackages, dependencies=TRUE)
```

Note: the base R code is frequently updated. If you install a new version of R, you will also need to update the packages you have already installed in the new version of R. This is easy to do:

```
update.packages()
```

Understanding What's in a Package

- A package usually contain more than just code to implement analyses or graphs. They can also contain data set, help documentation, and vignettes (short descriptions of the package)

Understanding what's in the package:

```
library(help=ggplot2)
```

Loading a data set from the package:

```
data(diamonds)
```

Get help for the `ggplot` function

```
?ggplot
```

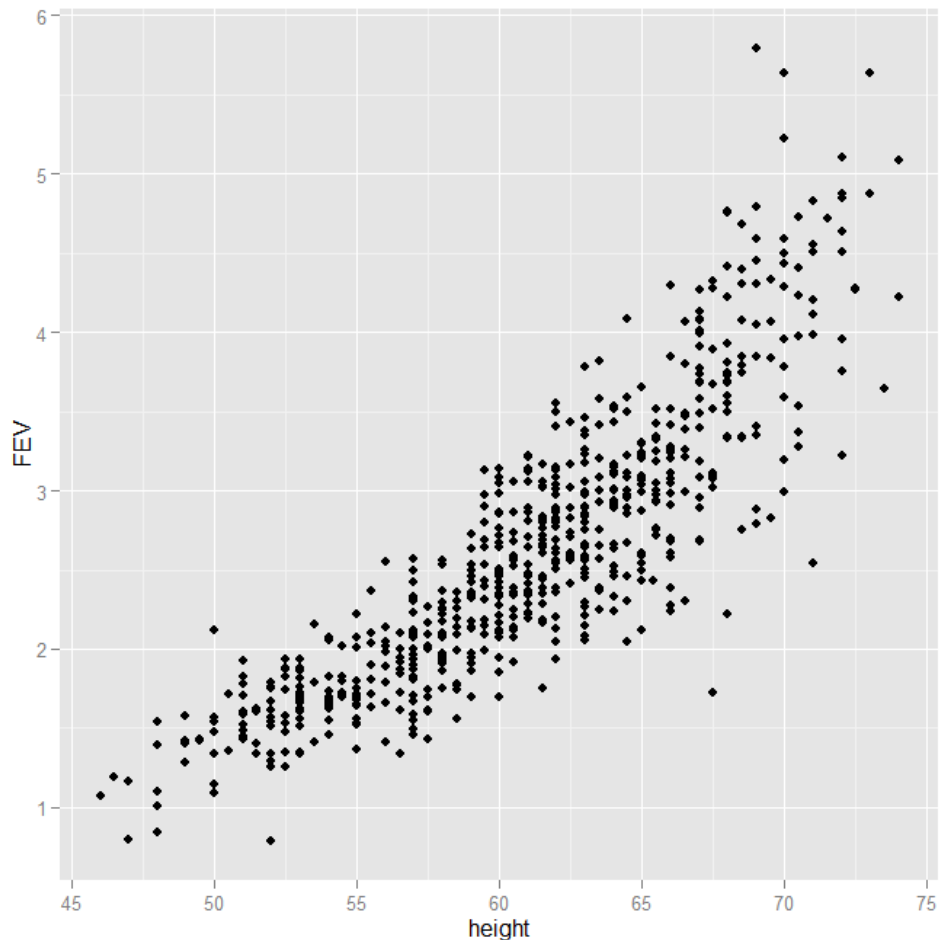
An example vignette for the `caret` package

```
vignette("caret")
```

Creating a Scatterplot with ggplot

We can create a simple scatterplot as follows:

```
ggplot(fev, aes(height, FEV)) + geom_point()
```



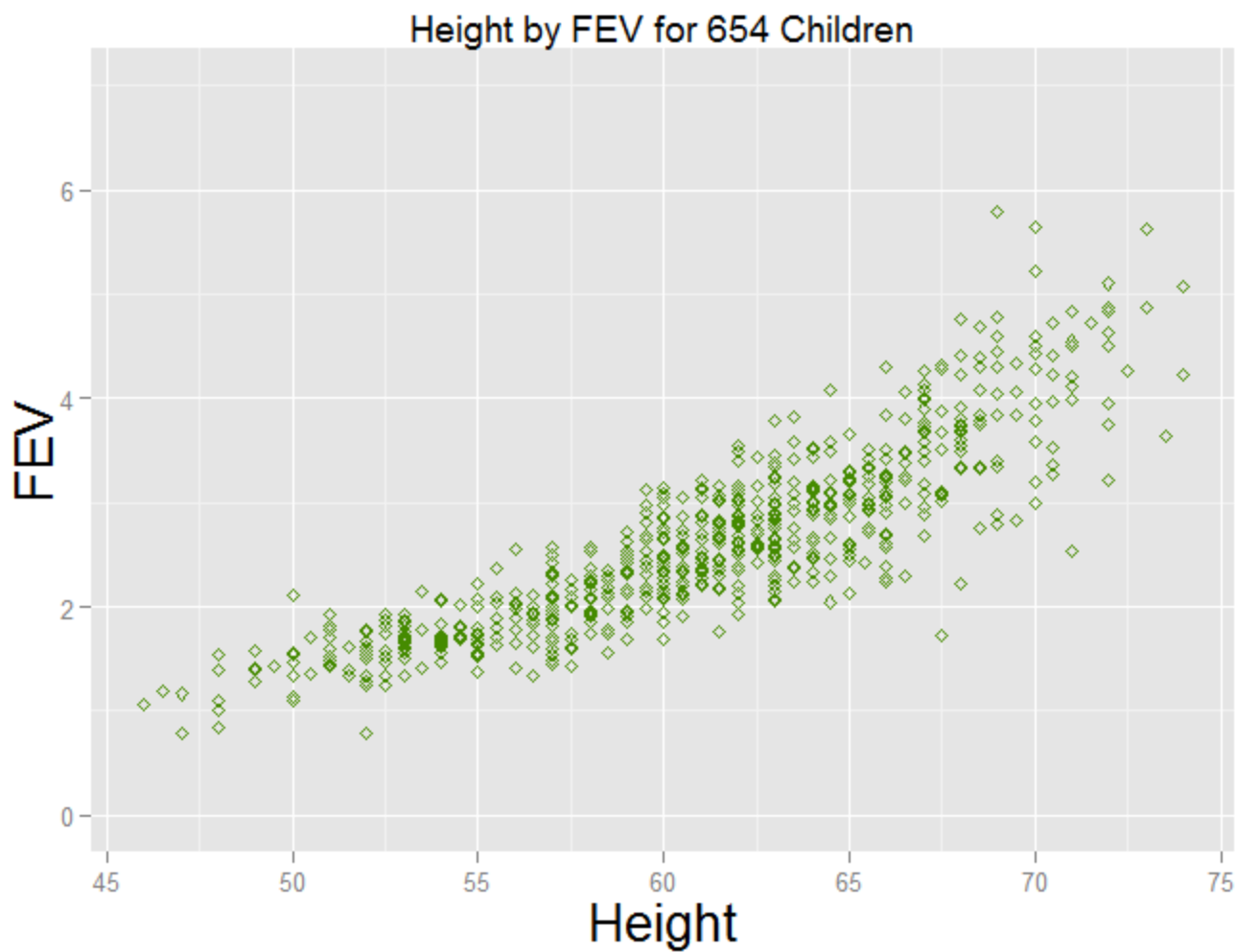
Customizing a ggplot

```
ggplot(fev, aes(height, FEV, ymin=0, ymax=7)) +  
  geom_point(colour = "chartreuse4",  
             shape = 5,  
             size = 1.5) +  
  ggtitle("Height by FEV for 654 Children") +  
  xlab("Height") +  
  ylab("FEV") +  
  theme(axis.title.x = element_text(size=20),  
        axis.title.y = element_text(size=20))
```

`ggplot` works with **layers**. It first creates the scatterplot, then we adjust the plot with subsequent commands. This plot statement does the following:

- changes the y-axis range to 0 to 7
- adds the title of Height by FEV for 654 Children
- changes the x-axis label to Height and makes it size 20
- changes the y-axis label to FEV and makes it size 20
- changes the symbol color to chartreuse 4
- changes the symbol to an open diamond
- makes the symbols size 1.5

Modified Output



Further Customization

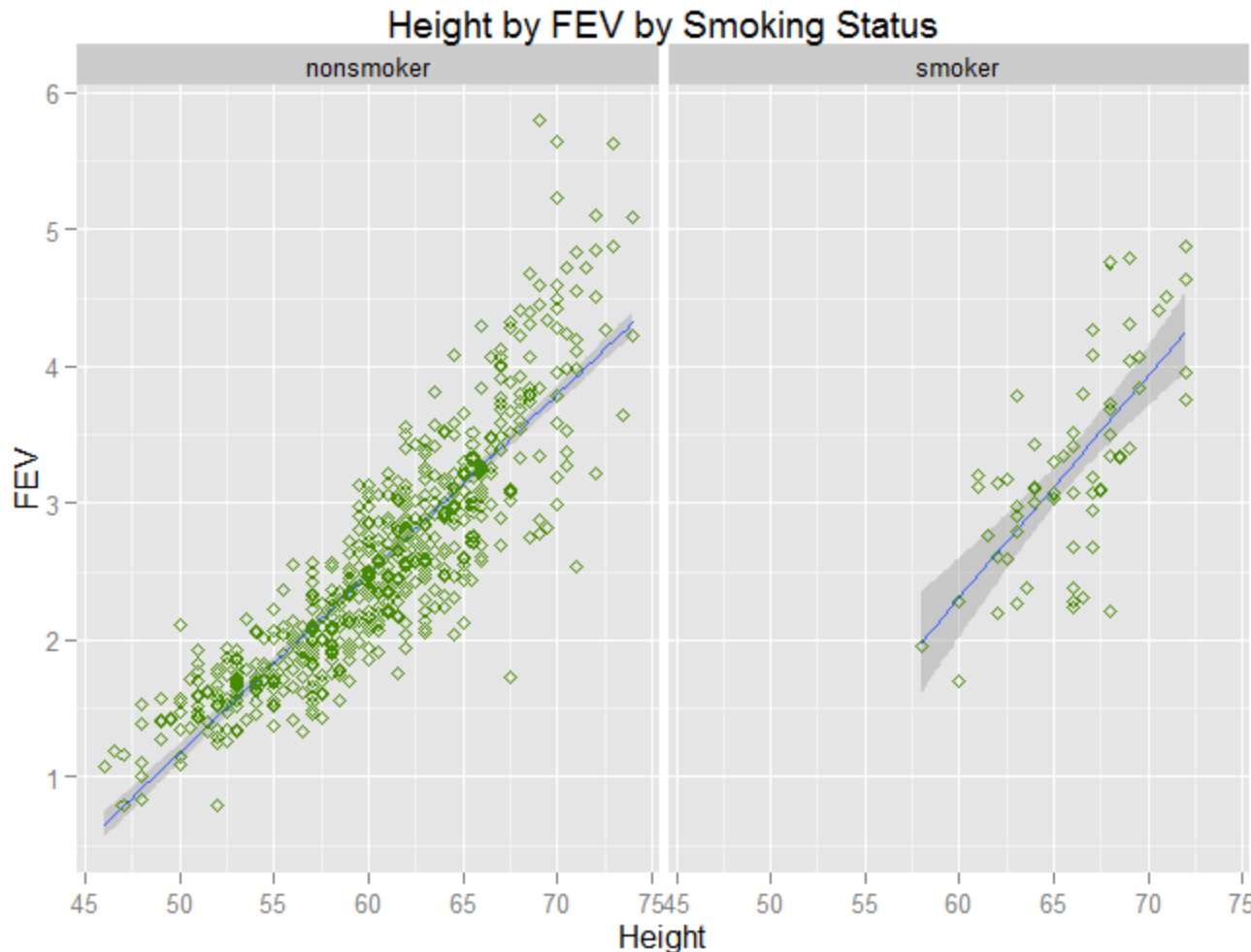
We can also create separate scatterplots by home smoking status and add a regression line to each:

```
ggplot(fev, aes(height, FEV)) +  
  geom_smooth(method = "lm") +  
  facet_grid(. ~ smoke) +  
  geom_point(colour = "chartreuse4",  
            shape = 5,  
            size = 1.5) +  
  ggtitle("Height by FEV by Smoking Status") +  
  xlab("Height") +  
  ylab("FEV")
```

The graph can be copied by right-clicking on the graph can copying. Or we can save the graph as follows:

```
ggsave(file = "fevBySmoke.png")
```

Modified Output



We will use ggplot to visualize data in the upcoming sessions.

<http://www.statmethods.net/> is a good location for help with ggplot2 and R in general.

Saving: History, Data, & Workspace

We have created a number of objects, and we may want to recall how some of those were created. We can do that with this history function:

```
history(100)
```

This gives the previous 100 command prompt entries.

We may also want to save data set(s) that were modified during the session. This can be done with the `write.csv` or `write.table` functions:

```
write.csv(myData, file = "myData.csv", row.names = FALSE)
```

The entire workspace can be saved as follows:

```
save.image("part1.RData")
```

This image can be reloaded for use at another time. All objects in the current session are saved:

```
fileLocation <- "C:/Part1"  
setwd(fileLocation)  
load.image("part1.RData")
```

Upcoming Sessions

- Part 2: Comparing Groups (1)
 - Data shaping, ANOVA, post-hoc test, two-way ANOVA, and visualization
- Part 3: Comparing Groups (2)
 - Fixed and random effects, how to model data with mixed (fixed and random) effects, repeated measures data, visualization
- Part 4: Covariance Structures in Mixed Models and Dimension Reduction and Classification
 - Principal component analysis (PCA), partial least squares (PLS), recursive partitioning (RPart), and random forests (RF)