

Getting Started with R

Why



?

Why



?

- Some of you may have used statistical software with a GUI, like Minitab or SPSS. You may also be familiar with other programming languages, like C, Java, Python, etc.
- We will use the R programming language and environment as our “home base” for performing many data analytic tasks.

Why



?

- Allows custom analysis
- High-level scripting language
- Statistical programming language
- Interactive exploratory data analysis

Why



?

- Easy to replicate analysis
- Sound numerical methods
- Large Community of contributors
- It's Free!

Let's Install **R**

Let's Install R

- Open a Web browser and go to Google:
<http://www.google.com/>
- Search for R (that's right – the letter R).
- One of the top links will be to
[The R Project for Statistical Computing](http://www.r-project.org/),
which takes you to [http:// www.r-project.org/](http://www.r-project.org/)
- At <http://www.r-project.org> click on CRAN (left menu)
- Select a mirror site near us, i.e. there is a mirror site at <http://cran.stat.ucla.edu>

Let's Install RStudio

- Open a Web browser and go to Google:
<http://www.google.com/>
- Search for RStudio
- One of the top links will be
<http://www.rstudio.org> click on Download tab
- Download the version for running R on your desktop

Let's Try It - RStudio

The image shows the RStudio interface with four main panels and several annotations:

- Source Code:** Contains R code for summarizing and histogramming birth weight data. A blue arrow points from the selected code to the Console panel.
- Console:** Shows the execution of the code, including an error for an undefined function and the output of a histogram and summary statistics. A green arrow points from the console output to the Plots panel.
- Session History:** Lists the commands entered in the console. A green arrow points from the history to the Source Code panel.
- Plots:** Displays a histogram titled "Birth weight" showing the frequency distribution of birth weights.

Annotations:

- A circle around the **Run** button in the Source Code panel.
- A circle around the **To Source** button in the Session History panel.
- Text: "Select Source Code and Run it in Console" with a blue arrow pointing to the Console panel.
- Text: "Code in History can be saved as Source Code" with a green arrow pointing from the Session History panel to the Source Code panel.
- Text: "Code Automatically placed in History" with a green arrow pointing from the Console panel to the Session History panel.

Source Code:

```
1 summary(infants$bwt)
2 hist(infants$bwt, breaks = 50, xlab = "Weight (ounces)", main = "Birth weight")
3
```

Console:

```
> clar()
Error: could not find function "clar"
>
> hist(infants$bwt, breaks = 50)
> hist(infants$bwt, breaks = 50, xlab = "Weight (ounces)", main = "Birth weight")
> hist(infants$bwt, breaks = 50, xlab = "Weight (ounces)", main = "Birth weight")
> summary(infants$bwt)
  Min. 1st Qu.  Median    Mean 3rd Qu.   Max.
  55.0   108.8   120.0   119.6   131.0   176.0
> hist(infants$bwt, breaks = 50, xlab = "Weight (ounces)", main = "Birth weight")
>
>
```

Session History:

```
dev.off()
q()
?quantile
data()
names(Obesity)
data(Obesity)
```

Plots:

Birth weight

Frequency

Weight (ounces)

Let's Try It - RStudio

Data Grid

	gestation	bwt	parity	age	ed	ht	wt	dage	ed	dht	dwt	m
1	284	120	1	27	College	62	100	31	College	65	110	Ma
2	282	113	2	33	College	64	135	38	College	70	140	Ma
3	279	128	1	28	High School	64	115	32	Some High School	NA	NA	Ma
4	NA	123	2	36	College	69	100	43	Some College	68	197	Ma
5	282	108	1	23	College	67	115	2	College	NA	NA	Ma
6	286	136	4	25	High School	62	93	28	High School	64	130	Ma
7	244	138	4	33	High School	62	178	37	Some College	NA	NA	Ma
8	245	132	2	23	Some High School	65	140	23	Some College	71	192	Ma
9	289	120	3	25	Some College	62	125	26	Some High School	70	180	Ma
10	299	143	3	30	College	66	136	34	College	NA	NA	Ma

Displayed 1000 rows of 1236 (236 omitted)

Work Space

Data	
grades	110 obs. of 7 variables
infants	1236 obs. of 15 variables
augtemp	numeric[31]
histInfants	histogram[7]
vbot	0

Console

```
>  
> hist(infants$bwt, breaks = 50)  
> hist(infants$bwt, breaks = 50, xlab = "Weight (ounces)", main =  
"Birth weight")  
> hist(infants$bwt, breaks = 50, xlab = "Weight (ounces)", main =  
"Birth weight")  
> summary(infants$bwt)  
  Min. 1st Qu.  Median    Mean 3rd Qu.    Max.   
  55.0   108.8   120.0   119.6   131.0   176.0   
> hist(infants$bwt, breaks = 50, xlab = "Weight (ounces)", main =  
"Birth weight")  
>  
> View(infants)  
> View(infants)  
>
```

Plots

Birth weight

A histogram showing the frequency distribution of birth weight in ounces. The x-axis is labeled 'Weight (ounces)' and ranges from 60 to 180. The y-axis is labeled 'Frequency' and ranges from 0 to 60. The distribution is roughly bell-shaped, centered around 120-130 ounces, with a peak frequency of approximately 60.

Expressions in **R**

Expressions in R

- The R prompt is: `>`
- At the prompt, type an ***expression***
- Hit the return/enter key
- R ***evaluates*** the expression (performs a ***computation***)
- R returns a value

What do expressions look like?

2 + 3

9 - 8

4 * 5

10 / 3

7 ^ 2

9 % / % 2

11 %% 7

These are simple
arithmetic expressions

Similar to what you
have with a calculator

Parsing Expressions

- How does R know what computation to perform?
- It breaks down an expression into parts, called tokens
- From these pieces it can figure out what computation to perform

Parsing English

hatheads...

- The above letters are the beginning of something that I am writing.
- Can you figure out what it is?
- What would make it easier for you to do this?

How do we parse English?

- Punctuation: . , ! ? : ;
- Capitalization
- Blank spaces

So What Does hatheads... mean?

How does **R** parse expressions?

- White space: **22** vs **2 2**
- Atomic Tokens
+ **-** ***** **/** **^**
; end of line
comment
- Quotation Marks
"Hi" or **'Bye'** not **"My"**
- Naming Conventions
x2 not **2x**
- New Line

How does **R** parse expressions?

- New Line

Order of Operations

Order of operations is what you expect:

- exponentiation first, followed by multiplication and division, then addition and subtraction;
- left to right;
- parentheses override order

Try It

- Write the following as an R expression:

`power(10, subtract(divide(15,3),2))`

- Write the following as an R expression:

$$\frac{\sqrt{6-2}}{3^2}$$

- Circle the tokens in the following R expression

`cat = (1 + x2)^24`

We all make mistakes in writing
code

Understanding how R parses
expressions will help you fix them

Variables in **R**

Output and Assignment

When we evaluate an expression, R prints the results to the screen as output

- How do we save the result?
- How do we use the output as input to another expression?

Output and Assignment

- We can assign the result of the computation to a variable, e.g., named `x`:

```
> x = 10^(15 / 3 - 2)
```

- We can use `x` as an input in another expression

```
> sqrt(x)
```

- To see the value of a variable type the variable name at the prompt and hit return

```
> x
```


Output and Assignment

- `=` and `<-` are both valid assignment operators

`x <- 10^(15 / 3 - 2)`

- Choose one and use it consistently
- As we'll see `==` means something completely different.

Variables

- Variables have a name and a value
- To access the value we use the name
- Variables allow us to:
 - Store a value without needing to recompute it
 - Write a general expression,
e.g. **`sqrt(a^2 + b^2)`**
 - Reduce redundancy (and mistakes)

Rules for Variable Names

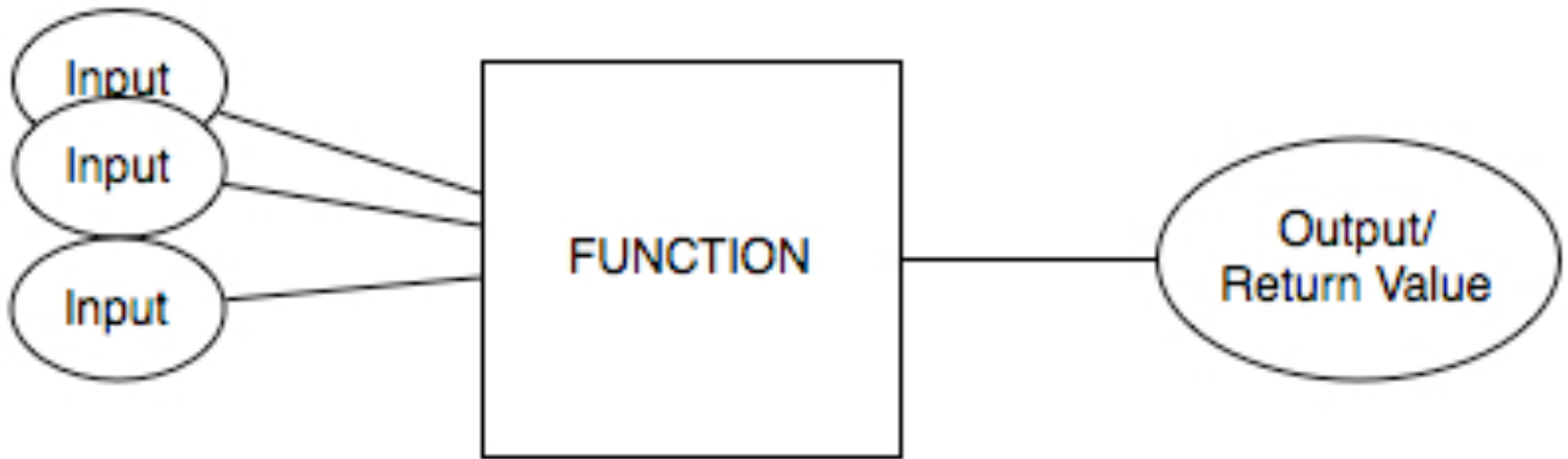
- Variable names must follow some rules
 - May not start with a digit or underscore (`_`)
 - May contain numbers, characters, and some punctuation - period and underscore are ok, but most others are not
 - Case-sensitive, so `x` and `X` are different
- Advice on variable names:
 - Use meaningful names
 - Avoid names that have meaning in R, e.g., function names. If in doubt, check:

```
> exists("pi")  
[1] TRUE
```

Function Style Expressions

Function Style Expressions

- Functions contain code (expressions) that perform a specific task.



The Inputs are called ***arguments***

The output is the ***return value***

Function Style Expressions

- When you use a function with a particular set of arguments, you are said to be ***calling*** the function
- R ***evaluates*** the function call and returns the output
- For now, we will work with R's built-in functions

Examples Built-in functions

- `log()`
- `exp()`
- `sqrt()`
- `abs()`
- `sin()` `cos()`
- `mean()`
- `sd()`
- `median()`
- `min()` `max()` `range()`
- `sum()`

Example

Suppose we want to generate 3 random values from a normal curve with center 0 and spread 5.

- Is there a function in R that can do this?
- How do we find it?
- How do we call it?

How to find it?

- Use Google search
 - I find searches like the following helpful
“R random normal”
- R has built-in search
`help.search("topic")`
- Post a question on the class forum

r generate random normal – Google Search

+You Search Images Videos Maps News Shopping Gmail More -

Sign in

Google

R generate random normal

Search

About 336,000,000 results (0.27 seconds)

Everything

Images

Maps

Videos

News

Shopping

Discussions

Enable

Buzzdock

More

Berkeley, CA

Change location

All results

Related searches

More search tools

[R Tutorial: Basic Probability](#)
www.cyclismo.org/tutorial/R/probability.html
Jump to [The Normal Distribution](#): There are four functions that can be used to **generate** the ... probability that a **normally** distributed **random** number will be ...

[\[R\] Generate multivariate normal data with a random correlation matrix](#)
<https://stat.ethz.ch/pipermail/r-help/2011-February/268410.html>
Mar 4, 2011 – [\[R\] Generate](#) multivariate **normal** data with a **random** correlation matrix. Rick DeShon deshon at msu.edu. Thu Feb 10 18:43:48 CET 2011. Previous message: ...
[Generating random normal distribution with mean 0 and ...](#)
[generating normal numbers: GetRNGstate, PutRNGstate](#)
[Random samples from a multivariate normal distribution](#)
[generate random number](#)
[More results from stat.ethz.ch »](#)

[R help archive: Re: \[R\] Generating random normal distribution wi](#)
tolstoy.newcastle.edu.au/R/help/06/07/30915.html
Jul 14, 2006 – Re: [\[R\] Generating random normal](#) distribution with mean 0 and standard deviation 1. This message : [Message body] [More options]; Related ...

[R Learning Module: Probabilities and Distributions](#)
www.ats.ucla.edu > [Stat Computing](#) > [R](#) > [Learning Modules](#)
R Learning Module Probabilities and Distributions. 1. Generating random samples from a **normal** distribution. Even though we would like to think of our samples ...

[R Programming/Random Number Generation - Wikibooks, open ...](#)

Find: 230

Next Previous

☐ Highlight all

☐ Match case

Phrase not found

How to call a function

- We call a function as follows:

FunctionName(argument, ..., argument)

- Functions can have one or more inputs
- Some arguments are required.
- Other arguments are optional. They have default values so you don't have to specify them

How to call rnorm

- We can find out the arguments to rnorm:

```
> args(rnorm)
```

```
function (n, mean = 0, sd = 1)
```

- We see it has 3 arguments: `n`, `mean` and `sd`
- `mean` and `sd` are optional. – they have default values (0 and 1, respectively)
- `n` must be specified – it has no default
- We can learn more with the help function:

```
> ?rnorm
```

```
> help("rnorm")
```

Generate 3 random values from a normal with center 0 and spread 5

- Generate 3 normals with mean 0 and sd 1.

`rnorm(3)`

- Arguments can be identified by position:
- Arguments can be identified by name:
- Use position and name:

Simple and Compound Expressions

- Simple Expression: `rnorm(3, sd = 5)`
- Compound Expression: `mean(rnorm(3))`
- Ill-formed Expressions: `mean(rnorm(3))]`

Can you spot what's wrong?