

R Stats Bootcamp

R language

Megan Lewis

2024-11-14

How the R stats bootcamp works

- Practical, open instructional materials for learning R
- Traditional statistics in R
- Reproducible research and collaboration tools
- Self-guided learning and self-assessment
- Friendly community



R stats bootcamp - Module 1

Schedule:

- ~~Session 1: An introduction and script workflow~~
- Session 2: R language
- Session 3: R functions
- Session 4: Data objects
- Session 5: Data frames
- Session 6: Data subsetting



Session 2 objectives:

- Example script, comments, help and pseudocode
- Math operators
- Logical boolean operators
- Regarding “base R” and the Tidyverse
- Practice exercises

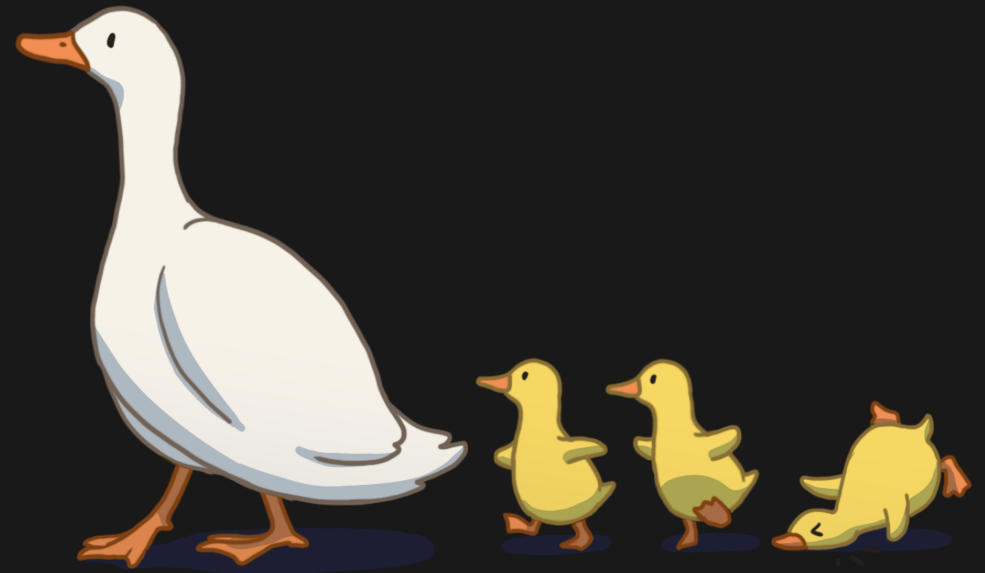
R Stats Bootcamp Session 2

- R as a passive aggressive butler



Example Script, comments, help and pseudocode

- Demo: The concept and purpose of a script
- Tips:
 - Work through bootcamp by coding
 - Type your own code
 - Document code with comments



Example script

Download and open the example script in R Studio:

[Example Script](#)

```
1  ## HEADER ####
2  ## Who: <Your name>
3  ## What: Bootcamp 1_2 R language
4  ## Last edited: <yyyy-mm-dd format>
5  ####
```

Contents > Organisation

```
1 ## CONTENTS ####  
2 ## 2 Example script, help, pseudocode  
3 ## 3 Math operators  
4 ## 4 Logical Boolean operators  
5 ## 5 Regarding base R and the Tidyverse  
6 ## 6 Practice exercises
```

Comment syntax

- All comments start with at least one hash tag

```
1 # This is a comment
2
3 ## Using two hash tags at the beginning to visually separate section title
4
5 ## Section heading ####
```

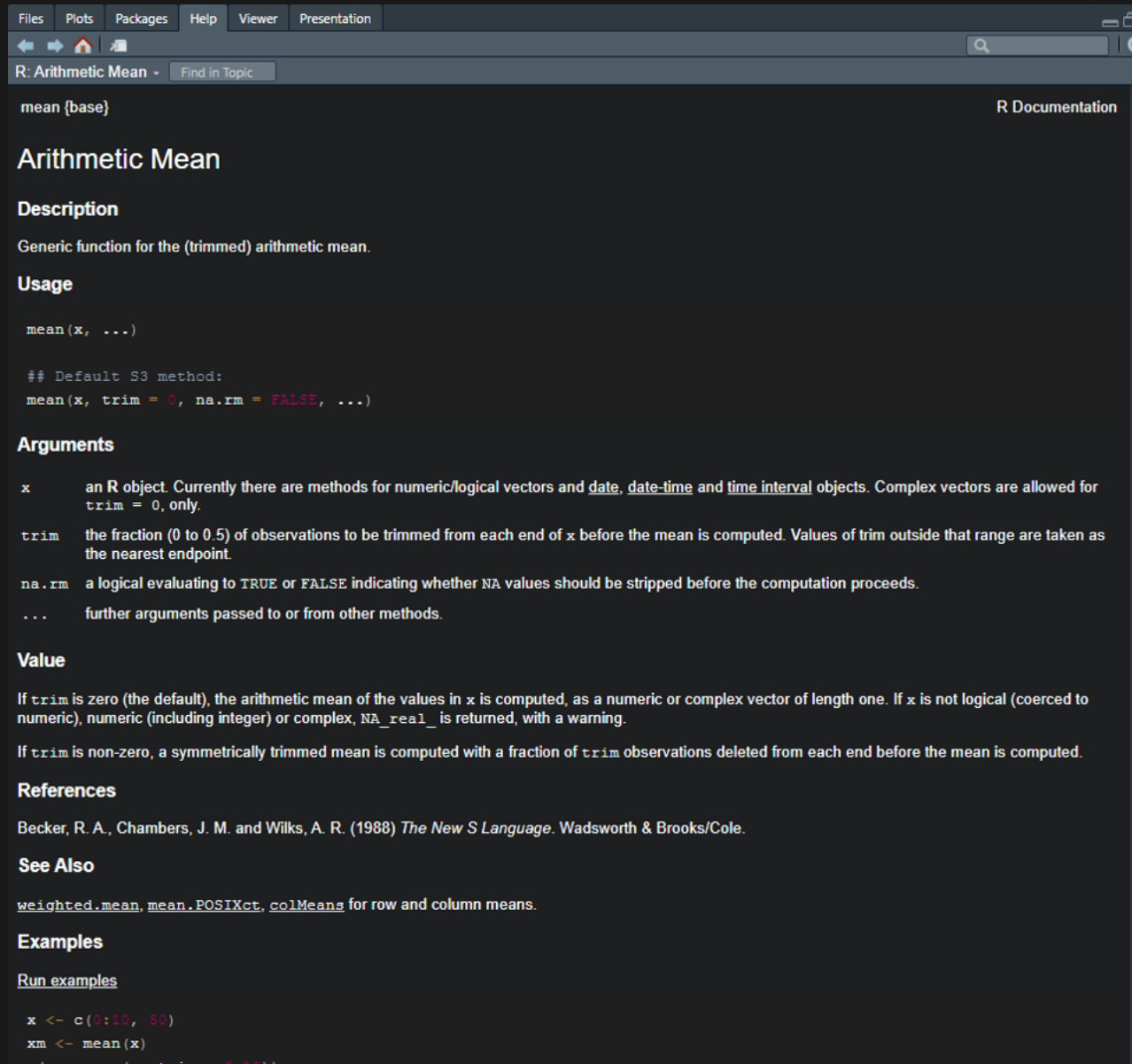
- The ## and #### make it a chunk!
- Organised and good for navigation

Help

- In R Studio help

```
1 # Display help page for the function 'mean'  
2 help(mean)  
3  
4 # Alternative way of getting help  
5 ?mean
```

Anatomy of help pages



The screenshot shows the R Documentation page for the `mean` function in the `base` package. The page is displayed in a web browser window with tabs for Files, Plots, Packages, Help, Viewer, and Presentation. The address bar shows the URL `R: Arithmetic Mean` and a search bar. The page content includes the following sections:

- mean (base)** (top right)
- Arithmetic Mean** (Section Header)
- Description**

Generic function for the (trimmed) arithmetic mean.
- Usage**

```
mean(x, ...)
```

Default S3 method:
`mean(x, trim = 0, na.rm = FALSE, ...)`
- Arguments**

<code>x</code>	an R object. Currently there are methods for numeric/logical vectors and date , date-time and time interval objects. Complex vectors are allowed for <code>trim = 0</code> , only.
<code>trim</code>	the fraction (0 to 0.5) of observations to be trimmed from each end of <code>x</code> before the mean is computed. Values of <code>trim</code> outside that range are taken as the nearest endpoint.
<code>na.rm</code>	a logical evaluating to <code>TRUE</code> or <code>FALSE</code> indicating whether NA values should be stripped before the computation proceeds.
<code>...</code>	further arguments passed to or from other methods.
- Value**

If `trim` is zero (the default), the arithmetic mean of the values in `x` is computed, as a numeric or complex vector of length one. If `x` is not logical (coerced to numeric), numeric (including integer) or complex, `NA_real_` is returned, with a warning.

If `trim` is non-zero, a symmetrically trimmed mean is computed with a fraction of `trim` observations deleted from each end before the mean is computed.
- References**

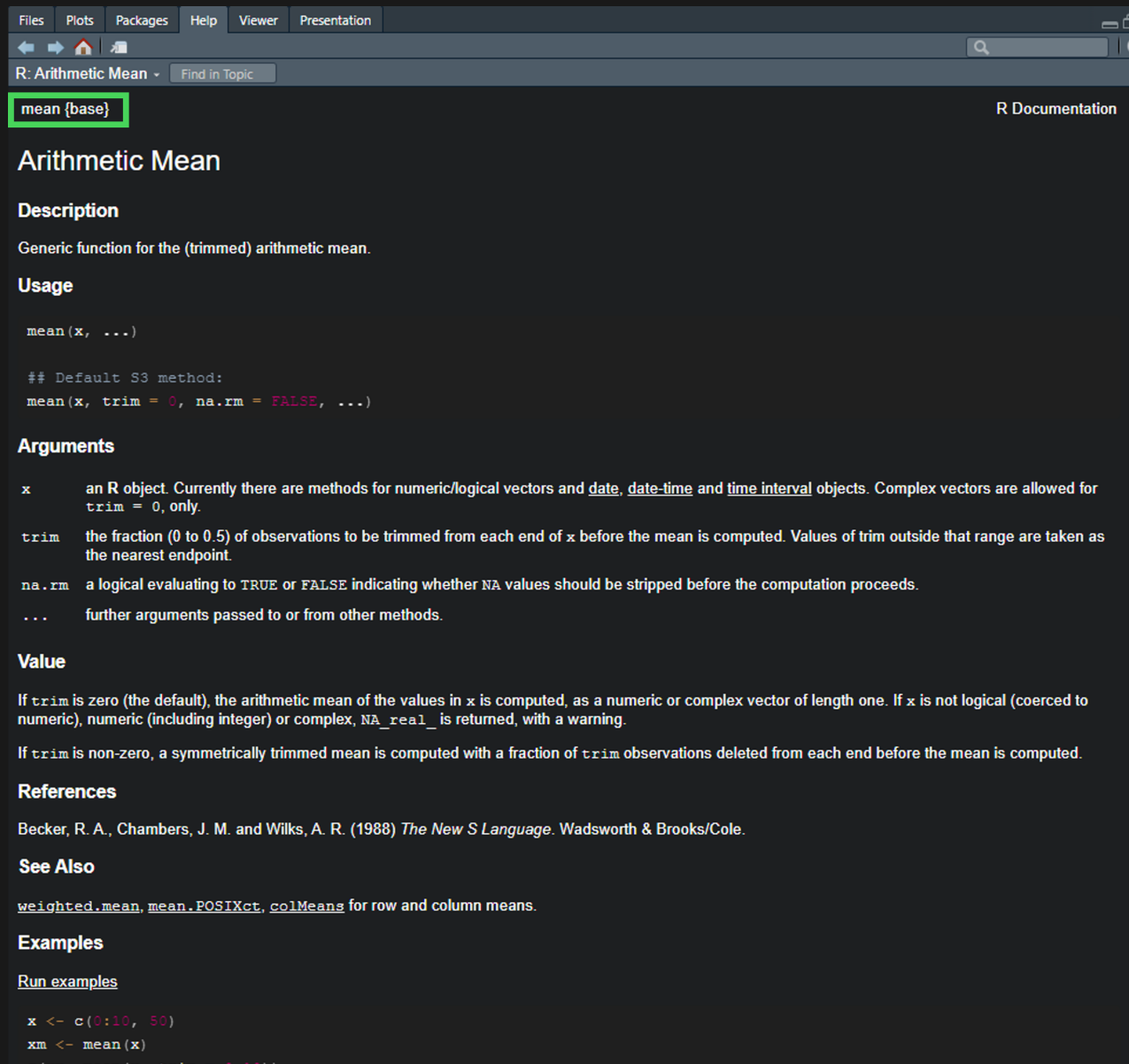
Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole.
- See Also**

[weighted.mean](#), [mean.POSIXct](#), [colMeans](#) for row and column means.
- Examples**

[Run examples](#)

```
x <- c(0:10, 50)
xm <- mean(x)
#> [1] 5.5
```

Anatomy of help pages



The screenshot shows the R Documentation website for the `mean` function in the `base` package. The page is viewed in a web browser with the address bar showing "R: Arithmetic Mean". The title "mean {base}" is highlighted with a green box. The page content includes sections for Description, Usage, Arguments, Value, References, See Also, and Examples.

mean {base} R Documentation

Arithmetic Mean

Description

Generic function for the (trimmed) arithmetic mean.

Usage

```
mean(x, ...)
```

Default S3 method:
`mean(x, trim = 0, na.rm = FALSE, ...)`

Arguments

x an R object. Currently there are methods for numeric/logical vectors and [date](#), [date-time](#) and [time interval](#) objects. Complex vectors are allowed for `trim = 0`, only.

trim the fraction (0 to 0.5) of observations to be trimmed from each end of `x` before the mean is computed. Values of `trim` outside that range are taken as the nearest endpoint.

na.rm a logical evaluating to `TRUE` or `FALSE` indicating whether NA values should be stripped before the computation proceeds.

... further arguments passed to or from other methods.

Value

If `trim` is zero (the default), the arithmetic mean of the values in `x` is computed, as a numeric or complex vector of length one. If `x` is not logical (coerced to numeric), numeric (including integer) or complex, `NA_real_` is returned, with a warning.

If `trim` is non-zero, a symmetrically trimmed mean is computed with a fraction of `trim` observations deleted from each end before the mean is computed.

References

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole.

See Also

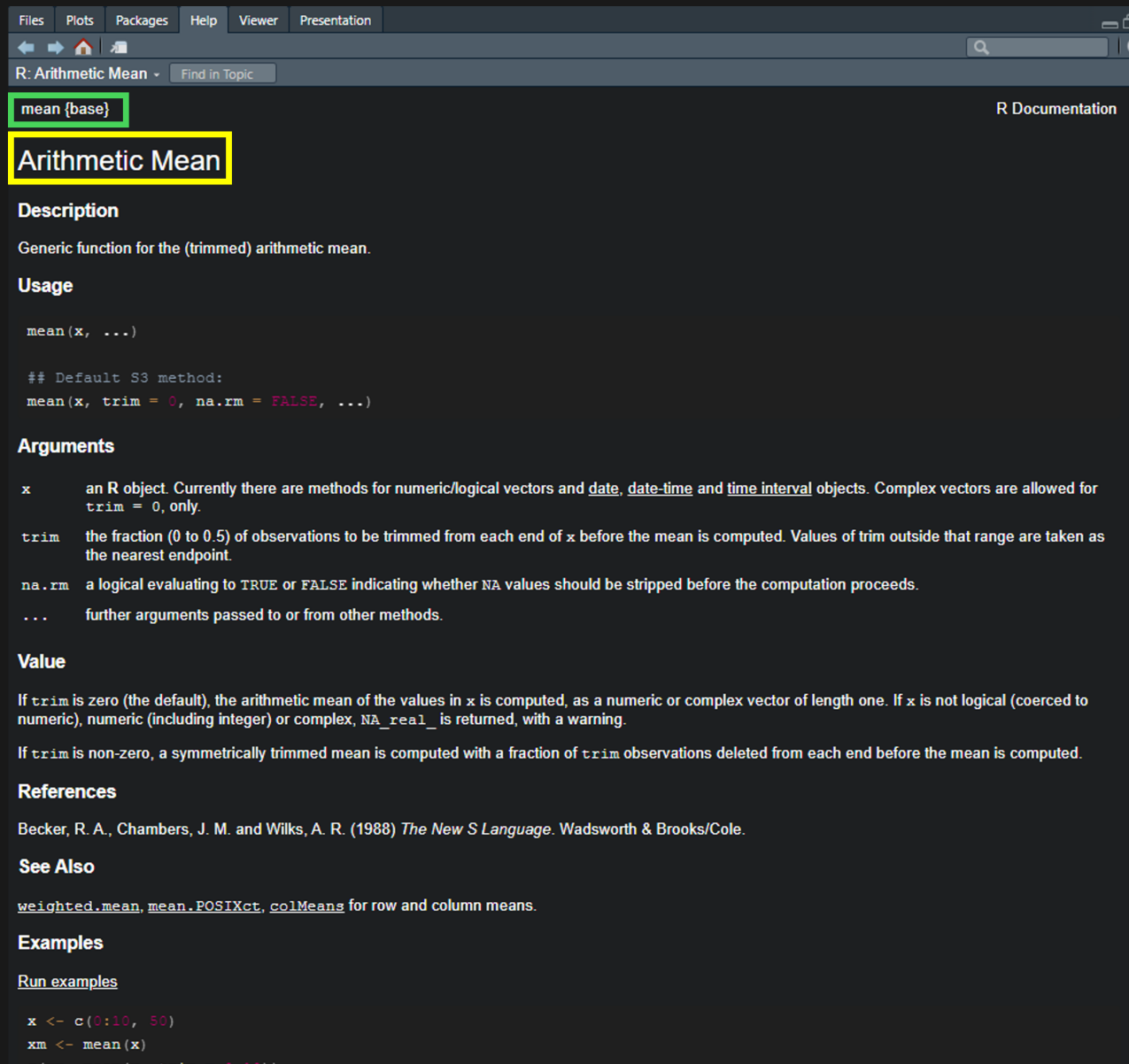
[weighted.mean](#), [mean.POSIXct](#), [colMeans](#) for row and column means.

Examples

[Run examples](#)

```
x <- c(0:10, 50)
xm <- mean(x)
xrm <- mean(x, trim = 0.10)
```

Anatomy of help pages



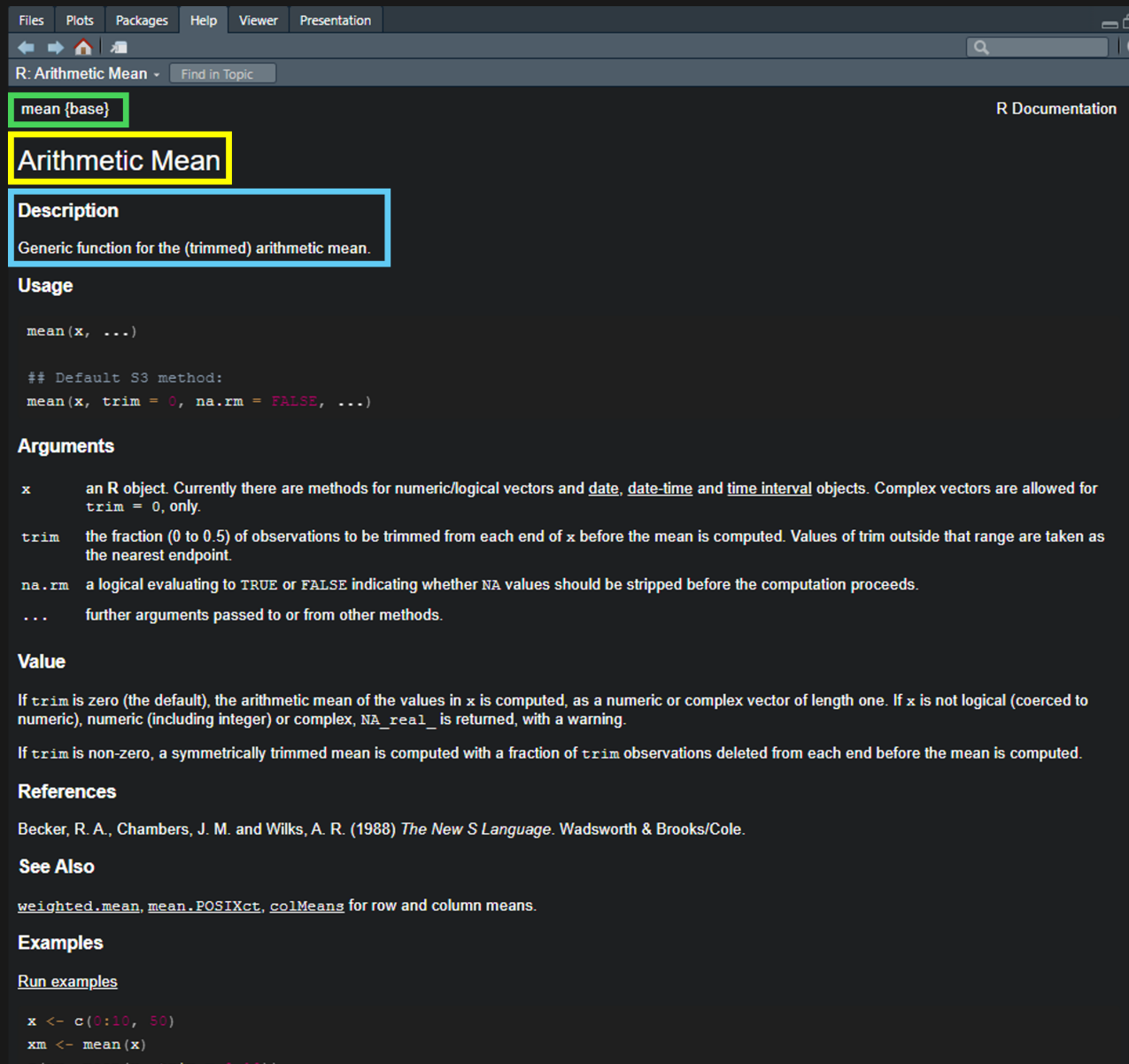
The screenshot shows the R Documentation website for the 'mean' function. The browser window has tabs for 'Files', 'Plots', 'Packages', 'Help', 'Viewer', and 'Presentation'. The address bar shows 'R: Arithmetic Mean' and a search bar. The page title is 'mean (base)' and the breadcrumb is 'R Documentation'. The main heading is 'Arithmetic Mean'. The 'Description' section states: 'Generic function for the (trimmed) arithmetic mean.' The 'Usage' section shows the function signature: `mean(x, ...)` and the default S3 method: `mean(x, trim = 0, na.rm = FALSE, ...)`. The 'Arguments' section lists:

- `x`: an R object. Currently there are methods for numeric/logical vectors and `date`, `date-time` and `time interval` objects. Complex vectors are allowed for `trim = 0`, only.
- `trim`: the fraction (0 to 0.5) of observations to be trimmed from each end of `x` before the mean is computed. Values of `trim` outside that range are taken as the nearest endpoint.
- `na.rm`: a logical evaluating to `TRUE` or `FALSE` indicating whether NA values should be stripped before the computation proceeds.
- `...`: further arguments passed to or from other methods.

 The 'Value' section states: 'If `trim` is zero (the default), the arithmetic mean of the values in `x` is computed, as a numeric or complex vector of length one. If `x` is not logical (coerced to numeric), numeric (including integer) or complex, `NA_real_` is returned, with a warning. If `trim` is non-zero, a symmetrically trimmed mean is computed with a fraction of `trim` observations deleted from each end before the mean is computed.' The 'References' section lists: 'Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole.' The 'See Also' section lists: '`weighted.mean`, `mean.POSIXct`, `colMeans` for row and column means.' The 'Examples' section has a link 'Run examples' and shows code:

```
x <- c(0:10, 50)
xm <- mean(x)
x/length(x) # trim = 0.1011
```

Anatomy of help pages



The screenshot shows the R documentation page for the `mean` function. Several elements are highlighted with colored boxes: a green box around the `mean {base}` link, a yellow box around the `Arithmetic Mean` title, and a blue box around the `Description` section. The page content includes the function's description, usage, arguments, value, references, and examples.

Files Plots Packages Help Viewer Presentation

R: Arithmetic Mean Find in Topic

[mean {base}](#) R Documentation

Arithmetic Mean

Description

Generic function for the (trimmed) arithmetic mean.

Usage

```
mean(x, ...)
```

Default S3 method:
`mean(x, trim = 0, na.rm = FALSE, ...)`

Arguments

x an R object. Currently there are methods for numeric/logical vectors and [date](#), [date-time](#) and [time interval](#) objects. Complex vectors are allowed for `trim = 0`, only.

trim the fraction (0 to 0.5) of observations to be trimmed from each end of `x` before the mean is computed. Values of `trim` outside that range are taken as the nearest endpoint.

na.rm a logical evaluating to `TRUE` or `FALSE` indicating whether NA values should be stripped before the computation proceeds.

... further arguments passed to or from other methods.

Value

If `trim` is zero (the default), the arithmetic mean of the values in `x` is computed, as a numeric or complex vector of length one. If `x` is not logical (coerced to numeric), numeric (including integer) or complex, `NA_real_` is returned, with a warning.

If `trim` is non-zero, a symmetrically trimmed mean is computed with a fraction of `trim` observations deleted from each end before the mean is computed.

References

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole.

See Also

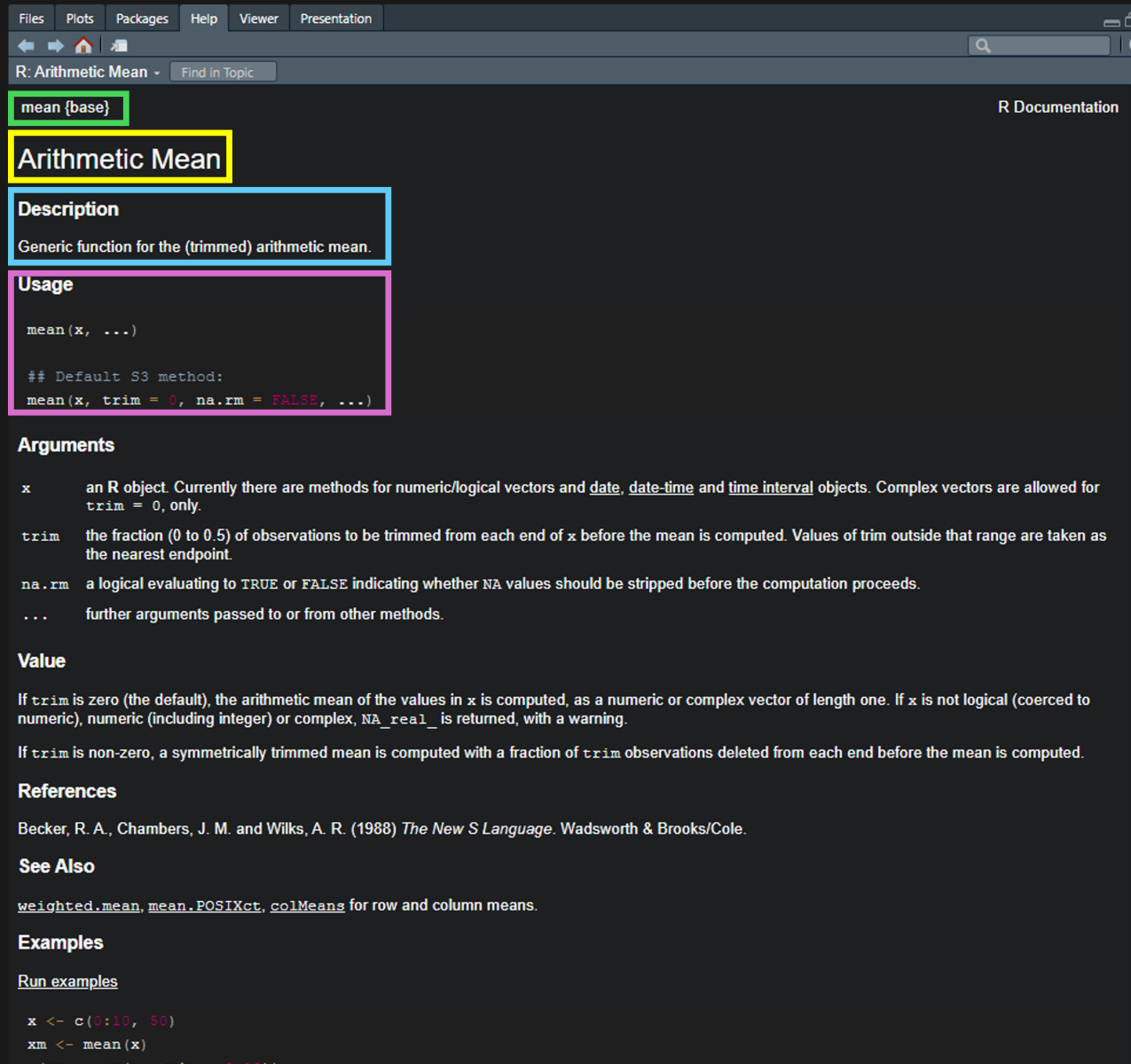
[weighted.mean](#), [mean.POSIXct](#), [colMeans](#) for row and column means.

Examples

[Run examples](#)

```
x <- c(0:10, 50)
xm <- mean(x)
#> [1] 5.5
```

Anatomy of help pages



The screenshot shows the R Documentation page for the `mean` function. The page is titled "R: Arithmetic Mean" and includes a search bar. The following sections are highlighted with colored boxes:

- mean {base}** (green box)
- Arithmetic Mean** (yellow box)
- Description** (blue box):
Generic function for the (trimmed) arithmetic mean.
- Usage** (pink box):

```
mean(x, ...)
```


Default S3 method:

```
mean(x, trim = 0, na.rm = FALSE, ...)
```
- Arguments** (blue box):
 - x** an R object. Currently there are methods for numeric/logical vectors and [date](#), [date-time](#) and [time interval](#) objects. Complex vectors are allowed for `trim = 0`, only.
 - trim** the fraction (0 to 0.5) of observations to be trimmed from each end of `x` before the mean is computed. Values of `trim` outside that range are taken as the nearest endpoint.
 - na.rm** a logical evaluating to `TRUE` or `FALSE` indicating whether NA values should be stripped before the computation proceeds.
 - ...** further arguments passed to or from other methods.
- Value** (blue box):

If `trim` is zero (the default), the arithmetic mean of the values in `x` is computed, as a numeric or complex vector of length one. If `x` is not logical (coerced to numeric), numeric (including integer) or complex, `NA_real_` is returned, with a warning.

If `trim` is non-zero, a symmetrically trimmed mean is computed with a fraction of `trim` observations deleted from each end before the mean is computed.
- References** (blue box):

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole.
- See Also** (blue box):

[weighted.mean](#), [mean.POSIXct](#), [colMeans](#) for row and column means.
- Examples** (blue box):

[Run examples](#)

```
x <- c(0:10, 50)
xm <- mean(x)
x/10 -> xm
```

Anatomy of help pages

The screenshot shows the R help page for the `mean` function. The interface includes a top navigation bar with tabs for Files, Plots, Packages, Help, Viewer, and Presentation. Below this is a search bar and a breadcrumb trail showing the current location: R: Arithmetic Mean. The main content area is titled "mean {base}" and "Arithmetic Mean". It is divided into several sections: Description, Usage, Arguments, Value, References, See Also, and Examples. The "Arguments" section is highlighted with a red box, and the "Usage" section is highlighted with a pink box. The "Description" section is highlighted with a blue box, and the "Arithmetic Mean" title is highlighted with a yellow box.

mean {base}

Arithmetic Mean

Description

Generic function for the (trimmed) arithmetic mean.

Usage

```
mean(x, ...)
```

```
## Default S3 method:
```

```
mean(x, trim = 0, na.rm = FALSE, ...)
```

Arguments

x an R object. Currently there are methods for numeric/logical vectors and [date](#), [date-time](#) and [time interval](#) objects. Complex vectors are allowed for `trim = 0`, only.

trim the fraction (0 to 0.5) of observations to be trimmed from each end of `x` before the mean is computed. Values of `trim` outside that range are taken as the nearest endpoint.

na.rm a logical evaluating to `TRUE` or `FALSE` indicating whether NA values should be stripped before the computation proceeds.

... further arguments passed to or from other methods.

Value

If `trim` is zero (the default), the arithmetic mean of the values in `x` is computed, as a numeric or complex vector of length one. If `x` is not logical (coerced to numeric), numeric (including integer) or complex, `NA_real_` is returned, with a warning.

If `trim` is non-zero, a symmetrically trimmed mean is computed with a fraction of `trim` observations deleted from each end before the mean is computed.

References

Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) *The New S Language*. Wadsworth & Brooks/Cole.

See Also

[weighted.mean](#), [mean.POSIXct](#), [colMeans](#) for row and column means.

Examples

[Run examples](#)

```
x <- c(0:10, 50)
```

```
xm <- mean(x)
```

```
x/ym <- mean(x, trim = 0.10)
```

Pseudocode - Tell it to the duck!



- Break problem down into smaller chunks
- Helps with understanding code and problems
- Give example

Math Operators

- R as a calculator
- Addition +
- Subtraction -
- Multiplication *
- Division /

Math operators

```
1 # Add with "+"  
2 2 + 5
```

```
[1] 7
```

Math operators

```
1 # Add with "+"  
2 2 + 5
```

```
[1] 7
```

```
1 # Subtract with "-"  
2 10 - 15
```

```
[1] -5
```

Order of operation

```
1 # Try this  
2 4 + 2 * 3
```

```
[1] 10
```

Order of operation

```
1 # Try this  
2 4 + 2 * 3
```

```
[1] 10
```

```
1 # Order control - same  
2 4 + (2 * 3)
```

```
[1] 10
```

Order of operation

```
1 # Try this  
2 4 + 2 * 3
```

```
[1] 10
```

```
1 # Order control - same  
2 4 + (2 * 3)
```

```
[1] 10
```

```
1 # Order control - different  
2 (4 + 2) * 3
```

```
[1] 18
```

Use of spaces

- Spaces don't matter, but style...

```
1 # Try this
2 6+10          # No spaces
[1] 16
```

Use of spaces

- Spaces don't matter, but style...

```
1 # Try this  
2 6+10          # No spaces
```

```
[1] 16
```

```
1 7      - 5      # Uneven spaces
```

```
[1] 2
```

Use of spaces

- Spaces don't matter, but style...

```
1 # Try this
2 6+10          # No spaces
```

```
[1] 16
```

```
1 7      - 5      # Uneven spaces
```

```
[1] 2
```

```
1 1.6    /    2.3 # Large spaces
```

```
[1] 0.6956522
```

Use of spaces

- Spaces don't matter, but style...

```
1 # Try this
2 6+10          # No spaces
```

```
[1] 16
```

```
1 7      - 5      # Uneven spaces
```

```
[1] 2
```

```
1 1.6    /    2.3 # Large spaces
```

```
[1] 0.6956522
```

```
1 16 * 3          # Exactly one space
```

```
[1] 48
```

Code formatting

{{< fa lightbulb >}} Formatting shortcut

Shortcut to help reformat selected/highlighted code

Windows/Linux: **Ctrl + Shift + a**

Mac: **Cmd + Shift + a**

Logical Boolean operators

TRUE and FALSE

```
1 # Try this  
2 3 > 5 # True, yes?
```

```
[1] FALSE
```

Logical Boolean operators

TRUE and FALSE

```
1 # Try this  
2 3 > 5 # True, yes?
```

```
[1] FALSE
```

```
1 # 3 is compared to each element  
2 3 < c(1, 2, 3, 4, 5, 6)
```

```
[1] FALSE FALSE FALSE TRUE TRUE TRUE
```

Logical Boolean operators

Useful booleans

```
1 # Try this
2
3 x <- c(21, 3, 5, 6, 22)
4 x
```

```
[1] 21  3  5  6 22
```

Logical Boolean operators

Useful booleans

```
1 # Try this
2
3 x <- c(21, 3, 5, 6, 22)
4 x
```

```
[1] 21 3 5 6 22
```

Selecting with Booleans

```
1 x[x > 20]
```

```
[1] 21 22
```

Logical Boolean operators

The “not” operator, ! (Sorry !sorry)

```
1 # Try this  
2 TRUE # plain true
```

```
[1] TRUE
```

Logical Boolean operators

The “not” operator, ! (Sorry !sorry)

```
1 # Try this  
2 TRUE # plain true
```

```
[1] TRUE
```

```
1 !FALSE # not false is true!
```

```
[1] TRUE
```

Logical Boolean operators

The “not” operator, ! (Sorry !sorry)

```
1 # Try this  
2 TRUE # plain true
```

```
[1] TRUE
```

```
1 !FALSE # not false is true!
```

```
[1] TRUE
```

```
1 6 < 5 #definitely false
```

```
[1] FALSE
```

Logical Boolean operators

The “not” operator, ! (Sorry !sorry)

```
1 # Try this  
2 TRUE # plain true
```

```
[1] TRUE
```

```
1 !FALSE # not false is true!
```

```
[1] TRUE
```

```
1 6 < 5 #definitely false
```

```
[1] FALSE
```

```
1 !(6 < 5) #not false...
```

```
[1] TRUE
```

Logical Boolean operators

The “not” operator, ! (Sorry !sorry)

```
1 # Try this  
2 TRUE # plain true
```

```
[1] TRUE
```

```
1 !FALSE # not false is true!
```

```
[1] TRUE
```

```
1 6 < 5 #definitely false
```

```
[1] FALSE
```

```
1 !(6 < 5) #not false...
```

```
[1] TRUE
```

```
1 !(c(23, 44, 16, 51, 12) > 50)
```

```
[1] TRUE TRUE TRUE FALSE TRUE
```

Base R vs Tidyverse

Tidyverse

- System of packages for data manipulation, exploration and visualization which share a common design philosophy
- Originally developed by Hadley Wickham [Book](#)
- Some find it difficult to understand
- Divergence of R languages

Personal preference - R Bootcamp use Base R

Practice exercises

- Understanding the first two sections of an R script
- Subset in the boxplot function
- R as a calculator
- Pseudocode
- Understanding outputs
- Boolean operators

