**Base R Cheat Sheet**

### Getting help
- `?mean`: Get help for a particular function.
- `help.search('weighted mean')`: Search the help files for a word or phrase.
- `help(package = 'dplyr')`: Find help for a package.

### Accessing help files
- `str(iris)`: Get a summary of an object's structure.
- `class(iris)`: Find the class an object belongs to.

### Using packages
- `install.packages('dplyr')`: Download and install a package from CRAN.
- `library(dplyr)`: Load the package into the session, making all its functions available to use.
- `dplyr::select`: Use a particular function from a package.
- `data(iris)`: Load a built-in dataset into the environment.

### More about an object
- `Variable assignment`
  - `a <- 'apple'`  
  - `a[1]` 'apple'

### Working directory
- `getwd()`: Find the current working directory (where inputs are found and outputs are sent).
- `setwd('C:/file/path')`: Change the current working directory.

### Conditions
- `a == b`: Equal
- `a != b`: Not equal
- `a < b, a <= b`: Less than or equal to
- `a > b, a >= b`: Greater than or equal to
- `is.na(a)`: Is missing
- `is.null(a)`: Is null

### Basic syntax
- `# This is ignored by R`
  - `print()`: Print in console
    - `print in console` allows to specify formatting

### Vectors
**Creating vectors**
- `c(2, 4, 6)`: Join elements into a vector
- `2:6`: An integer sequence
- `seq(2, 3, by=0.5)`: A complex sequence
- `rep(1:2, times=3)`: Repeat a vector
- `rep(1:2, each=3)`: Repeat elements of a vector

### Vector functions
- `sort(x)`: Return `x` sorted.
- `rev(x)`: Return `x` reversed.
- `table(x)`: See counts of values.
- `unique(x)`: See unique values.

### The environment
- `ls()`: List all variables in the environment.
- `rm(x)`: Remove `x` from the environment.
- `rm(list = ls())`: Remove all variables from the environment.

You can use the environment panel in RStudio to browse variables in your environment.

### Accessing help files
- `Getting help`
- `Accessing help files`
- `More about an object`
- `Using packages`
- `More about an object`

### Lists
- `l <- list(x = 1:5, y = c('a', 'b'))`: A list is a collection of elements which can be of different types.
- `l$x`: New list with only the first element.
- `l[y]`: New list with only element named `y`.

### Data frames
- `df <- data.frame(x = 1:3, y = c('a', 'b', 'c'))`: A special case of a list where all elements are the same length.
- `df$X`: New list with only the first element.
- `nrow(df)`: Number of rows.
- `ncol(df)`: Number of columns.
- `dim(df)`: Number of rows and columns.
- `cbind - Bind columns.
- `rbind - Bind rows.`

### Matrices
- `m <- matrix(x, nrow = 3, ncol = 3)`: Create a matrix from `x`.
- `t(m)`: Transpose
- `m %*% n`: Matrix Multiplication
- `solve(m, n)`: Find `x` in `m` \( \times n \)
Descriptive functions

- `colnames()`: retrieve/set column labels
- `rownames()`: retrieve/set row labels
- `length()`: number of elements/columns
- `rev()`: reverse order of elements
- `sample()`: randomize order of elements
- `sort()`: sort elements ascending order
- `merge()`: join data frames
- `na.omit()`: remove rows with NA values

Factors

- `factor(x)`: turn a vector into a factor. Can set the levels of the factor and the order.
- `cut(x, breaks = 4)`: turn a numeric vector into a factor by 'cutting' into sections.

Maths functions

- `log(x)`: Natural logarithm
- `exp(x)`: Exponential function
- `max(x)`: Largest element
- `min(x)`: Smallest element
- `round(x, n)`: Round to n decimal places
- `cor(x, y)`: Correlation
- `log2(x)`: log with base 2
- `abs(x)`: absolute value
- `sqrt(x)`: square root
- `rowSums(x)`: sum of each row
- `colSums(x)`: sum of each column
- `rowMeans(x)`: mean of each row
- `colMeans(x)`: mean of each column
- `IQR(x)`: interquartile range
- `summary(x)`: statistical overview
- `pastecs::stat.desc()`: extensive statistical overview
- `DescTools::Desc()`: extensive statistical overview + plots

Index functions

- `which(condition)`: indices of elements where condition is TRUE
- `which.min()`: index of smallest element
- `which.max()`: index of largest element
- `order()`: indices of elements in ascending order

Repeating a function

- `apply(D, margin, function)`: apply function
  - `margin=1`: apply on rows
  - `margin=2`: apply on columns
  - `margin=1:2`: apply on elements

Formatting data

- `reshape2::melt(D)`: from wide to long format
- `reshape2::dcast(D, formula)`: from long to wide format

Writing your own function

```r
function_name <- function(formula) {
  Do something
  return(new_variable)
}
```

Example

```r
square <- function(x) {
  squared <- x * x
  return(squared)
}
```
ggplot function for 2 factors

ggplot(D, aes(x=factor1, y=data, fill=factor2)) +
fill for geoms that have an area that can be filled

ggplot(D, aes(x=factor1, y=data, color=factor2)) +
color for geoms that do not have an area that can be filled

ggplot(data, aes(Sepal.Length, density, fill=Species)) +

stats – An alternative way to build a layer

Some plots visualize a transformation of the original data set.
Use a stat to choose a common transformation to visualize,
stat_summary() to plot statistics

fun.y to plot 1 statistic:: mean, median, or identity
fun.data to plot multiple statistics:
mean_cl_normal for 95% CI
mean_se for error bars representing sem
mean_sdl for error bars representing 2 SD
add fun.args=list(mult=1) for 1 SD
median_hilow and fun.args=list(conf.int=0.5) for IQR

geom defines type of plot. bar, errorbar or point

Basics

ggplot2 is based on the grammar of graphics, the idea that you can build every graph from the same few components: a data set, a set of geoms—visual marks that represent data points, and a coordinate system.

To display data values, map variables in the data set to aesthetic properties of the geom like size, color, and x and y locations.

ggplot(data = mpg, aes(x=cyl, y=hwy))

Begin a plot that you finish by adding layers to, no defaults, but more provided than gglplot

ggplot(mpg, aes(hwy, cty)) +
geom_point(aes(color = cyl)) +
geom_smooth(method = "loess") +
coord_cartesian() +
scale_color_gradient() + theme_bw()

Add a new layer to a plot with a geom_*() or stat_*() function. Each provides a geom, a set of aesthetic mappings, and a default stat and position adjustment.

last_plot()
Returns the last plot
ggsave("plot.png", width = 5, height = 5)
Saves last plot as 5” x 5” file named “plot.png” in working directory. Matches file type to file extension.

Stats – An alternative way to build a layer

Some plots visualize a transformation of the original data set.
Use a stat to choose a common transformation to visualize,
stat_summary() to plot statistics

fun.y to plot 1 statistic:: mean, median, or identity
fun.data to plot multiple statistics:
mean_cl_normal for 95% CI
mean_se for error bars representing sem
mean_sdl for error bars representing 2 SD
add fun.args=list(mult=1) for 1 SD
median_hilow and fun.args=list(conf.int=0.5) for IQR

geom defines type of plot. bar, errorbar or point

Geoms – Use a geom to represent data points, use the geom’s aesthetic properties to represent variables. Each function returns a layer.

One Variable

Continuous

a <- ggpplot(mpg, aes(hwy))

a + geom_area(stat = "bin")

x, y, alpha, color, fill, linetype, size
b + geom_area(aes = density), stat = "bin"

a + geom_density(kernel = "gaussian")

x, y, alpha, color, fill, linetype, size, weight
b + geom_density(aes = y = ..density..)

a + geom_dotplot()

x, y, alpha, color, fill

a + geom_freqpoly()

x, y, alpha, color, linetype, size
b + geom_freqpoly(aes = ..density..)

a + geom_histogram(binwidth = 5)

x, y, alpha, color, fill, linetype, size, weight
b + geom_histogram(aes = y = ..density..)

Discrete X. Continuous Y

g <- ggpplot(mpg, aes(class, hwy))

g + geom_boxplot()

lower, middle, upper, x, ymin, ymax, alpha, color, fill, linetype, shape, size, weight

g + geom_dotplot(binaxis = "y", stackdir = "center")

x, y, alpha, color, fill

a + geom_violin(scale = "area")

x, y, alpha, color, linetype, size, weight

Two Variables

Continuous X. Continuous Y

f <- ggpplot(mpg, aes(cty, hwy))

f + geom_blank()

f + geom_jitter()

x, y, alpha, color, fill, line, shape, size

f + geom_point()

x, y, alpha, color, fill, line, shape, size

f + geom_quantile()

x, y, alpha, color, linetype, size, weight

f + geom_rug(sides = "b")

x, y, alpha, color, fill, line, shape, size

f + geom_smooth(model = lm)

x, y, alpha, color, linetype, size, weight

Continuous Bivariate Distribution

i <- ggpplot(economics, aes(date, unemploy))

i + geom_area()

x, y, alpha, color, fill, line, shape, size

i + geom_line()

x, y, alpha, color, fill, line, shape, size

Continuous Function

j <- ggpplot(economics, aes(date, unemployment))

j + geom_step(direction = "hv")

x, y, alpha, color, fill, linetype, size

j + geom_text(aes(label = cty))

x, y, label, alpha, angle, color, family, fontface, hjust, lineheight, size, values

Three Variables

scale_z <- co Sel(s, sqrt(data_long^2 + data_lat^2))
m <- ggpplot(seas, aes(lon, lat))

m + geom_raster(aes(fill = z), interpolate=FALSE)

x, y, alpha, fill

m + geom_contour(aes(z = z))

x, y, z, alpha, colour, linetype, size, weight:

m + geom_tile(aes(fill = z))

x, y, alpha, color, fill, linetype, size
### Scales

- **Arguments for axis scales:**
  - **name:** text of the title of the axis
  - **labels:** text of the tick labels
  - **limits:** range of the axis
  - **breaks:** at which values to put tick marks
  - **trans:** transformation (e.g., log, log2, log10)...

- **Scale fill colors:**
  - **scale_fill_X** for geoms that can be filled
  - **scale_color_X** for geoms that have to be colored
  - **scale_X_manual** to choose discrete colors yourself

### Coordinate system

- **Arguments for axis coordinates:**
  - **name:** text of the title of the axis
  - **labels:** text of the tick labels
  - **trans:** transformation (e.g., log, log2, log10)...

### Facets

- **Arguments for facets:**
  - **position:** method to arrange facet labels
  - **space:** spacing between facet plots

### Legends

- **Arguments for legends:**
  - **label:** text of the legend
  - **fill:** background color of legend
  - **position:** position of legend

### Themes

- **Arguments for themes:**
  - **axis_line:** line of axis
  - **axis.text:** text of axis labels
  - **legend.text:** text of legend

### Coordinate system

- **Arguments for axes:**
  - **name:** title of the axis
  - **labels:** text of the tick labels
  - **limits:** range of the axis
  - **breaks:** at which values to put tick marks
  - **trans:** transformation (e.g., log, log2, log10)...

### Scales

- **Arguments for scale fills:**
  - **name:** title of the legend
  - **values:** colors for the categories in legend
  - **labels:** new names for the categories of the legend

### Arguments for manual color scales:

- **name:** text of the title of the legend
- **values:** colors for the categories in legend
- **labels:** new names for the categories of the legend

- **Arguments for fill colors:**
  - **fill:** background color of geoms
  - **color:** color of geoms

### Zooming

- **Arguments for zooming:**
  - **enlargement:** factor to zoom
  - **clip:** whether to clip to the plot

### Arguments for shapes:

- **name:** text of the title of the legend
- **values:** colors for the categories in legend
- **labels:** new names for the categories of the legend

- **Arguments for geometric shapes:**
  - **fill:** background color of elements
  - **color:** color of elements
  - **size:** size of elements
  - **linetype:** line type of elements
  - **lineend:** line end of elements
  - **family:** font of elements
  - **hjust:** horizontal justification of elements
  - **angle:** angle of elements