R Handbook

Quick Reference for Research Methods Labs

Gordon Wright

2026-01-01

About This Handbook

This handbook is a **reference companion** for your Research Methods labs. Use it to look up R code patterns when you're working on assignments.

> How to Use
>
> - **Search** (Ctrl/Cmd + F) for what you need
> - **Copy** code examples and adapt them
> - **Bookmark** this page for quick access

R Basics

Assignment and Objects

```
# Assign a value to a variable
x <- 5
my_score <- 42

# View a variable
x
print(my_score)

# Remove a variable
rm(x)

# List all objects in environment
ls()
```

Data Types

```r
# Numeric
age <- 25
height <- 1.75

# Character (text)
name <- "Alice"
group <- "control"

# Logical (TRUE/FALSE)
passed <- TRUE
is_control <- FALSE

# Check type
class(age)        # "numeric"
class(name)       # "character"
class(passed)     # "logical"
```

Vectors

```r
# Create a vector with c()
scores <- c(4, 5, 3, 2, 5)
names <- c("Alice", "Bob", "Carol")
conditions <- c(TRUE, FALSE, TRUE)

# Access elements
scores[1]         # First element: 4
scores[2:4]       # Elements 2 through 4
scores[c(1, 3)]   # Elements 1 and 3

# Useful functions
length(scores)    # Number of elements
sum(scores)       # Sum of all elements
mean(scores)      # Average
min(scores)       # Minimum
max(scores)       # Maximum
```

Basic Operations

```r
# Arithmetic
5 + 3     # Addition
10 - 4    # Subtraction
6 * 7     # Multiplication
20 / 4    # Division
2 ^ 3     # Exponent (2 cubed = 8)
17 %% 5   # Modulo (remainder: 2)

# Comparisons (return TRUE/FALSE)
```

```r
5 > 3     # Greater than
5 < 3     # Less than
5 == 5    # Equal to
5 != 3    # Not equal to
5 >= 5    # Greater than or equal
5 <= 3    # Less than or equal

# Logical operators
TRUE & FALSE   # AND: FALSE
TRUE | FALSE   # OR: TRUE
!TRUE          # NOT: FALSE
```

Working with Data

Loading Data

```r
# From CSV (readr package - recommended)
library(readr)
data <- read_csv("filename.csv")
data <- read_csv("path/to/filename.csv")

# From CSV (base R)
data <- read.csv("filename.csv")

# From URL
data <- read_csv("https://example.com/data.csv")

# Suppress column type messages
data <- read_csv("filename.csv", show_col_types = FALSE)
```

Inspecting Data

```r
# First/last rows
head(data)        # First 6 rows
head(data, 10)    # First 10 rows
tail(data)        # Last 6 rows

# Structure and dimensions
str(data)         # Structure with types
dim(data)         # Rows and columns
nrow(data)        # Number of rows
ncol(data)        # Number of columns

# Column names
names(data)       # All column names
colnames(data)    # Same as names()
```

```r
# Summary statistics
summary(data)     # Summary of all columns
```

Accessing Columns

```r
# Using $ (most common)
data$age
data$group

# Using [[ ]]
data[["age"]]

# Using [ ] with column name
data[, "age"]

# Using [ ] with column number
data[, 1]         # First column
data[, 2:4]       # Columns 2 through 4
```

Creating New Columns

```r
# Simple calculation
data$new_column <- data$column1 + data$column2

# With conditions
data$category <- ifelse(data$score > 50, "high", "low")

# Multiple conditions
data$grade <- case_when(
  data$score >= 70 ~ "A",
  data$score >= 60 ~ "B",
  data$score >= 50 ~ "C",
  TRUE ~ "F"
)
```

Data Wrangling (dplyr)

The Pipe Operator

```r
library(dplyr)

# Pipe: |> (or %>%)
# Takes output from left, passes as first argument to right

# Without pipe
summarise(group_by(data, condition), mean = mean(score))
```

```r
# With pipe (easier to read!)
data |>
  group_by(condition) |>
  summarise(mean = mean(score))
```

Selecting Columns

```r
# Select specific columns
data |> select(id, score, group)

# Select range of columns
data |> select(p1:p5)

# Remove columns
data |> select(-id, -notes)

# Select by pattern
data |> select(starts_with("p"))
data |> select(ends_with("score"))
data |> select(contains("time"))
```

Filtering Rows

```r
# Single condition
data |> filter(group == "control")
data |> filter(age > 25)

# Multiple conditions (AND)
data |> filter(group == "control", time == "pre")
data |> filter(group == "control" & time == "pre")

# Multiple conditions (OR)
data |> filter(group == "control" | group == "treatment")
data |> filter(group %in% c("control", "treatment"))

# Exclude missing values
data |> filter(!is.na(score))
```

Creating/Modifying Columns

```r
# Create new column
data |> mutate(total = p1 + p2 + p3)

# Modify existing column
data |> mutate(score = score / 100)
```

```r
# Multiple operations
data |> mutate(
  total = p1 + p2 + p3,
  mean_score = total / 3,
  category = ifelse(total > 10, "high", "low")
)
```

Grouping and Summarising

```r
# Group by one variable
data |>
  group_by(condition) |>
  summarise(
    mean = mean(score, na.rm = TRUE),
    sd = sd(score, na.rm = TRUE),
    n = n()
  )

# Group by multiple variables
data |>
  group_by(condition, time) |>
  summarise(
    mean = mean(score, na.rm = TRUE),
    sd = sd(score, na.rm = TRUE),
    n = n(),
    .groups = "drop"
  )

# Count observations
data |> count(condition)
data |> count(condition, time)
```

Arranging (Sorting)

```r
# Sort ascending
data |> arrange(score)

# Sort descending
data |> arrange(desc(score))

# Sort by multiple columns
data |> arrange(group, desc(score))
```

Descriptive Statistics

Central Tendency

```r
# Mean
mean(data$score)
mean(data$score, na.rm = TRUE)  # Ignore missing values

# Median
median(data$score)
median(data$score, na.rm = TRUE)

# Mode (no built-in function)
names(sort(table(data$category), decreasing = TRUE))[1]
```

Variability

```r
# Standard deviation
sd(data$score, na.rm = TRUE)

# Variance
var(data$score, na.rm = TRUE)

# Range
range(data$score, na.rm = TRUE)
min(data$score, na.rm = TRUE)
max(data$score, na.rm = TRUE)

# Interquartile range
IQR(data$score, na.rm = TRUE)
```

Other Useful Statistics

```r
# Sum
sum(data$score, na.rm = TRUE)

# Length/count
length(data$score)
n_distinct(data$id)  # Unique values (dplyr)

# Quantiles
quantile(data$score, probs = c(0.25, 0.5, 0.75), na.rm = TRUE)

# Comprehensive summary
summary(data$score)
```

Questionnaire Scoring

Reverse Scoring

```r
# For 1-5 scale: reversed = 6 - original
data$rev_item <- 6 - data$item

# For 1-7 scale: reversed = 8 - original
data$rev_item <- 8 - data$item

# For 0-4 scale: reversed = 4 - original
data$rev_item <- 4 - data$item

# General formula: reversed = (max + min) - original
# For 1-5: max + min = 6, so 6 - original
```

Creating Scale Scores

```r
# Sum score
data$total <- data$item1 + data$item2 + data$item3 +
              data$rev_item4 + data$rev_item5

# Using rowSums (more efficient)
data$total <- rowSums(data[, c("item1", "item2", "item3",
                               "rev_item4", "rev_item5")],
                      na.rm = TRUE)

# Mean score
data$mean_score <- rowMeans(data[, c("item1", "item2", "item3",
                                     "rev_item4", "rev_item5")],
                            na.rm = TRUE)
```

Visualisation (ggplot2)

Basic Structure

```r
library(ggplot2)

# Basic template
ggplot(data, aes(x = variable1, y = variable2)) +
  geom_XXXXX() +
  theme_minimal() +
  labs(title = "Title", x = "X Label", y = "Y Label")
```

Histogram

```r
# Basic histogram
ggplot(data, aes(x = score)) +
  geom_histogram(bins = 30)
```

```r
# Styled histogram
ggplot(data, aes(x = score)) +
  geom_histogram(bins = 20, fill = "#4A90A4", colour = "white") +
  theme_minimal() +
  labs(title = "Distribution of Scores", x = "Score", y = "Count")

# Histogram by group
ggplot(data, aes(x = score, fill = group)) +
  geom_histogram(bins = 20, alpha = 0.6, position = "identity") +
  theme_minimal()
```

Boxplot

```r
# Basic boxplot
ggplot(data, aes(x = group, y = score)) +
  geom_boxplot()

# With fill colour
ggplot(data, aes(x = group, y = score, fill = group)) +
  geom_boxplot(alpha = 0.7)

# Grouped by two variables
ggplot(data, aes(x = group, y = score, fill = time)) +
  geom_boxplot(alpha = 0.7) +
  theme_minimal() +
  labs(title = "Scores by Group and Time",
       x = "Group", y = "Score", fill = "Time")
```

Bar Chart

```r
# Counts
ggplot(data, aes(x = group)) +
  geom_bar()

# From summary data (means)
ggplot(summary_data, aes(x = group, y = mean)) +
  geom_col(fill = "#4A90A4") +
  theme_minimal()

# With error bars
ggplot(summary_data, aes(x = group, y = mean)) +
  geom_col(fill = "#4A90A4", alpha = 0.7) +
  geom_errorbar(aes(ymin = mean - se, ymax = mean + se),
                width = 0.2) +
  theme_minimal()
```

Scatterplot

```
# Basic scatterplot
ggplot(data, aes(x = age, y = score)) +
  geom_point()

# With regression line
ggplot(data, aes(x = age, y = score)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "lm", se = TRUE) +
  theme_minimal()

# Coloured by group
ggplot(data, aes(x = age, y = score, colour = group)) +
  geom_point(alpha = 0.6) +
  geom_smooth(method = "lm", se = FALSE) +
  theme_minimal()
```

Customisation

```
# Themes
+ theme_minimal()
+ theme_classic()
+ theme_bw()

# Labels
+ labs(
    title = "Main Title",
    subtitle = "Subtitle",
    caption = "Data source",
    x = "X-axis Label",
    y = "Y-axis Label",
    fill = "Legend Title",
    colour = "Legend Title"
  )

# Colours
+ scale_fill_manual(values = c("control" = "#E8E8E8",
                               "treatment" = "#4A90A4"))
+ scale_colour_manual(values = c("pre" = "#B8D4E3",
                                 "post" = "#4A90A4"))

# Axis limits
+ xlim(0, 100)
+ ylim(0, 50)
+ coord_cartesian(xlim = c(0, 100), ylim = c(0, 50))
```

Common Errors and Fixes

Object not found

```
# Error: object 'x' not found

# CAUSE: Variable does not exist
# FIX: Create it first, or check spelling

x <- 5  # Create the object
x       # Now it works
```

Could not find function

```
# Error: could not find function "read_csv"

# CAUSE: Package not loaded
# FIX: Load the package first

library(readr)   # Load the package
read_csv("data.csv")  # Now it works
```

NA result

```
# Getting NA when calculating statistics

# CAUSE: Missing values in data
# FIX: Add na.rm = TRUE

mean(data$score)                # Returns NA
mean(data$score, na.rm = TRUE)   # Returns the mean
```

Unexpected symbol

```
# Error: unexpected symbol in...

# CAUSE: Usually a typo or missing comma/operator
# FIX: Check for:
#   - Missing commas between arguments
#   - Unclosed brackets or quotes
#   - Typos in function names
```

Object of type closure

```
# Error: object of type 'closure' is not subsettable

# CAUSE: Using a function name as if it were data
```

```r
# Common with: data$column (when 'data' is the function)

# FIX: Use a different variable name
my_data <- read_csv("file.csv")
my_data$column  # Works
```

Useful Patterns

Complete Workflow Template

```r
# 1. Load packages
library(readr)
library(dplyr)
library(ggplot2)

# 2. Load data
data <- read_csv("my_data.csv", show_col_types = FALSE)

# 3. Inspect
head(data)
str(data)

# 4. Clean/transform
data <- data |>
  mutate(
    rev_item4 = 6 - item4,
    rev_item5 = 6 - item5,
    total = item1 + item2 + item3 + rev_item4 + rev_item5
  )

# 5. Summarise
summary_stats <- data |>
  group_by(condition, time) |>
  summarise(
    mean = mean(total, na.rm = TRUE),
    sd = sd(total, na.rm = TRUE),
    n = n(),
    .groups = "drop"
  )

# 6. Visualise
ggplot(data, aes(x = condition, y = total, fill = time)) +
  geom_boxplot(alpha = 0.7) +
  theme_minimal() +
  labs(title = "My Title", x = "Condition", y = "Score")
```

Reporting Statistics in APA

```r
# Get values
m <- mean(data$score, na.rm = TRUE)
s <- sd(data$score, na.rm = TRUE)
n <- sum(!is.na(data$score))

# Format for reporting
sprintf("M = %.2f, SD = %.2f", m, s)
# Output: "M = 3.45, SD = 0.89"

cat("Participants scored moderately (M =", round(m, 2),
    ", SD =", round(s, 2),
    ", n =", n, ")")
```

Keyboard Shortcuts

Action	Windows/Linux	Mac
Run current line	Ctrl + Enter	Cmd + Enter
Run all code	Ctrl + Shift + Enter	Cmd + Shift + Enter
Insert assignment	Alt + -	Option + -
Insert pipe	Ctrl + Shift + M	Cmd + Shift + M
Comment/uncomment	Ctrl + Shift + C	Cmd + Shift + C
Save	Ctrl + S	Cmd + S
Render document	Ctrl + Shift + K	Cmd + Shift + K
Find	Ctrl + F	Cmd + F

Resources

Official Documentation

- R for Data Science - Free online book
- ggplot2 documentation
- dplyr documentation
- Quarto guide

Getting Help

```r
# Help for a function
?mean
help(mean)

# Search help
```

```
??regression

# Examples
example(mean)
```

Quick Reference Card

Print this page or save it as PDF for offline reference!