R Take-Home Assignments

PhD Course — R Programming

2025-10-01

Introduction

These take-home assignments are designed as **practice material and a long-term support resource**. Many researchers experience that the skills they learn in an R course fade quickly without repeated use. Each exercise is therefore written both as a *learning task now* and as a *self-guided refresher later*. The assignments provide clear, story-driven introductions, guided hints, and structured solutions so you can return to them whenever you need to analyse your own data.

You will also receive **separate answer documents**. We **strongly recommend** you first complete the assignments yourself and **only** consult the answers **afterwards**—or if you are really stuck. Treat the answer key as a check and learning aid, not a shortcut.

Working with the provided `.Rmd` files

All assignments are provided as **RMarkdown (`.Rmd`) files**. These are special text documents that combine **narrative text, code, and outputs** in one place.

- Each `.Rmd` has sections with explanations, tasks, and code blocks.
- The student versions contain **hints in the code blocks** (commented out).
- You should **fill in the missing code** yourself where the comments indicate a task.
- When you "knit" the file, RStudio runs the code and produces a **PDF or HTML document** with both your text and the results.

Why use RMarkdown?

- Keeps **code, results, and explanations together**.
- Produces a professional document you can keep for later reference.
- Ensures your work is **reproducible**: anyone (including you in the future) can rerun the file and get the same results.

How to knit in RStudio

1. Open the `.Rmd` file in RStudio.

2. At the top, select the **Knit** button.

3. Choose your output format: **Knit to PDF** (requires LaTeX) or **Knit to HTML** (works by default).

4. RStudio runs all the code chunks and creates a finished document in the same folder.

Prefer not the use RMarkdown?

Although it is a nice exercise to start working with RMarkdown straight away to make the process of making reports in the future easier, it is by no means required. Do you prefer not to work with RMarkdown? Just use the supplied pdf files and a simple non RMarkdown script to make the exercises.

Mini tutorial: how to fill in an exercise

Inside the `.Rmd` you will see code chunks like this:

```
#### Task: filter the dataset to year 2019
#### penguins %>% filter(year == 2019)
```

- Lines starting with `####` are **hints or instructions**.

- To answer the exercise, replace the hints with your own code. For example:

```
# penguins_2019 <- penguins %>% filter(year == 2019)
```

- You can run code chunks one by one with **Ctrl+Shift+Enter** (Windows) or **Cmd+Shift+Enter** (Mac).

- When you are satisfied, knit the whole file to produce your report.

What you will learn

Over six assignments you will:

- Get comfortable with **R / RStudio**, reading and writing CSVs, and working with real datasets.

- Practise **data wrangling** and **visualisation** with `dplyr` and `ggplot2`.

- Use **conditionals, loops, and functions** to express reproducible workflows.

- Explore open datasets (world development, COVID-19, penguins) and make **clean, interpretable plots**.

- Take first steps in **statistical testing in R** (t-tests, ANOVA), report **effect sizes**, and do basic **model checks**.

- Build a small **regression model** linking socioeconomic indicators to health outcomes.

A note on Assignment 5

Assignment 5 is **not part of the official R Carpentry lessons**. It is included to provide a **gentle introduction to statistics in R**. The aim is practical literacy: how to run standard tests, what their assumptions mean, and how to interpret results for your own research. You can see this as a bridge from what you have learned in the course towards what you will use R for in your own work.

Getting help inside R (quick tips)

When you need more detail about a function, use R's built-in help:

- **Help for a specific function:** `?drop_na` or `?dplyr::filter`

- **Search help topics (fuzzy):** `??"linear model"`

- **See examples for a function:** `example(lm)`

- **Package vignettes:** `vignette(package = "dplyr")` or `vignette("dplyr")`

- **RStudio shortcut:** place the cursor on a function name and press **F1** to open its help page.

Final remark

Use these assignments as a **mini-syllabus** you can revisit. Work through the tasks first; then read the answers to confirm, compare, and deepen your understanding.

R Take-Home Assignment 1: Gapminder (Wrangling & Visualization)

Your Name

2025-09-30

Overview

In this first assignment, you will work with the Gapminder dataset, a classic resource for studying global development. Originally compiled by the Gapminder Foundation, this dataset brings together health, population, and economic data for 142 countries between 1952 and 2007, measured at five-year intervals.

Each record in the dataset contains:

- Life expectancy (`lifeExp`): the average number of years a newborn is expected to live, given current mortality rates.
- Population size (`pop`): the total number of inhabitants in the country that year.
- GDP per capita (`gdpPercap`): the average economic output per person, adjusted for inflation and expressed in US dollars.
- `Country` and `continent` information: identifiers for grouping and comparison.

The dataset is widely used because it allows us to explore broad questions: How did life expectancy change across different continents over the second half of the 20th century? Do wealthier countries consistently have higher life expectancy? And how much variation exists within regions?

In this assignment, you will practice the core steps of data analysis in R: importing data, exploring its structure, creating subsets, calculating summaries, and producing visualisations. These are fundamental skills you will use again and again in your own research.

Source

`https://github.com/resbaz/r-novice-gapminder-files/raw/master/data/gapminder-FiveYearData.csv`

Skills

`loading CSVs, subsetting, summarising, plotting`

Preparation

You will need the following R packages:

- `dplyr`, `ggplot2`, `readr`

```r
# Task: load the required packages (hint: use library())
# library(dplyr)
# library(ggplot2)
# library(readr)
```

1. Load the data

Every analysis begins with bringing the data into R. Here, you will import the Gapminder dataset directly from its online source.

Task: Load the Gapminder dataset into R and store it as `gapminder`.

```
# Task: load dataset here (hint: read_csv() from readr)
```

2. Explore the dataset

Before we can analyse, we need to understand what the dataset looks like. How many countries are covered, what years are available, and what variables can we work with?

Task: Report the number of countries, the range of years, and the variable names.

```
# Task: explore the dataset (hint: n_distinct(), range(), names())
```

Write your answers here:

```
Countries: ...
Years: ...
Variables: ...
```

3. Subset the data

Analyses are often limited to a particular region of interest. Suppose you are asked to focus only on Europe.

Task: Create a subset containing only the rows where `continent` is `"Europe"`. Save this as `europe`.

```
# Task: subset the data (hint: filter(continent == "Europe"))
```

4. Summarise life expectancy

To see how regions compare, we want to calculate average life expectancy for each continent over time. This will let us track differences across the decades.

Task: Calculate the mean life expectancy per continent per year. Save the result as `lifeexp_summary`.

```
# Task: summarise data (hint: group_by() + summarise())
```

5. Visualise trends

Summaries are useful, but a plot makes patterns easier to see. We will now compare how life expectancy has changed in different regions.

Task: Produce a line plot of life expectancy over time for **two continents of your choice**. Label the axes and add a title.

```
# Task: plot trends (hint: ggplot() + geom_line())
```

6. Reflection

Finally, consider what your analysis shows. Think about whether continents improved at the same pace, and whether any regions lag behind.

Task: Write 3–4 sentences describing the patterns you see across continents in life expectancy trends.

Write your reflection here…

Appendix

```r
sessionInfo()
```

```
## R version 4.5.1 (2025-06-13)
## Platform: aarch64-apple-darwin24.4.0
## Running under: macOS Sequoia 15.6.1
## 
## Matrix products: default
## BLAS:   /opt/homebrew/Cellar/openblas/0.3.30/lib/libopenblasp-r0.3.30.dylib 
## LAPACK: /opt/homebrew/Cellar/r/4.5.1/lib/R/lib/libRlapack.dylib;  LAPACK version 3.12.1
## 
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
## 
## time zone: Europe/Amsterdam
## tzcode source: internal
## 
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base     
## 
## loaded via a namespace (and not attached):
##  [1] compiler_4.5.1    fastmap_1.2.0     cli_3.6.5         tools_4.5.1      
##  [5] htmltools_0.5.8.1 yaml_2.3.10       rmarkdown_2.30    knitr_1.50       
##  [9] xfun_0.52         digest_0.6.37     rlang_1.1.6       evaluate_1.0.4
```

R Take-Home Assignment 2: Palmer Penguins (Functions & Visualization)

Your Name

2025-10-01

Overview

In this assignment you will explore the Palmer Penguins dataset, which contains measurements for three penguin species living in the Palmer Archipelago in Antarctica. Researchers collected data on their bill size, flipper length, body mass, sex, and island of origin. The dataset is a favourite in data science because it provides a simple but realistic setting to practise exploring biological variation. You will use this dataset to practise creating functions in R and visualising data in meaningful ways.

Source

`palmerpenguins` R package
### Skills

`functions, grouped summaries, scatterplots, faceting`

Preparation

You will need: `dplyr`, `ggplot2`, `palmerpenguins`.

```
## Task: load packages (hint: library())
## library(dplyr)
## library(ggplot2)
## library(palmerpenguins)
```

1. Load and inspect the data

Before beginning any analysis, the first step is always to bring the data into R and take a first look. By knowing how many rows we have, which species are included, and what variables are measured, we can plan the kinds of questions we might answer.

Task: Load the `penguins` dataset from the `palmerpenguins` package. Then:

- Report the number of rows in the dataset.

- List the unique species included.

- Identify which variables are numeric.

```
## Task: load data (hint: data(penguins); or use penguins object)
## Task: inspect (hint: nrow(), dplyr::glimpse(), table(penguins$species))
```

2. Plot flipper length vs body mass

Biologists often wonder whether certain traits scale together. For penguins, larger flippers might be expected to support a heavier body. If this is true, we should see a clear relationship between flipper length and body mass. By colouring the points by species, we can also see whether the relationship holds consistently across Adelie, Chinstrap, and Gentoo penguins.

Task: Make a scatterplot of `flipper_length_mm` (x-axis) against `body_mass_g` (y-axis). Colour the points by `species`. Add axis labels and a title.

```
## Task: scatterplot (hint: ggplot() + geom_point())
```

3. Facet by island

Species sometimes live on different islands, and environmental conditions (like food availability) can influence body size. To check whether penguins differ not just by species but also by location, we can break the plot into separate panels for each island.

Task: Recreate the scatterplot and facet by `island`.

```
## Task: facet (hint: facet_wrap())
```

4. Write a summary function

Analyses are more powerful when we can repeat them easily. Suppose we want to compare average bill length, flipper length, and body mass for each species. Instead of writing separate code each time, we can create a function that computes these summaries for any species we provide.

Task: Write `species_summary(sp)` that returns mean bill length, flipper length, and body mass for species `sp`.

```
## Task: write function (hint: function() & summarise())
```

5. Apply your function

Now that you have written your function, you can quickly generate summaries for all species and combine them into one table. This gives a compact overview of how species differ.

Task: Apply your function to all species and bind results into one table.

```
## Task: apply (hint: unique(), lapply() & bind_rows())
```

6. Reflection

Finally, think about what these analyses reveal. Did larger flippers usually go hand in hand with heavier bodies? Were differences between species clear? Did location appear to matter?

Task: Write 3–4 sentences describing species differences and any island effects you observed.

Write your reflection here…

R Take-Home Assignment 3: OWID COVID-19 (Loops & Conditionals)

Your Name

2025-10-01

Overview

In this assignment you will work with the **Our World in Data (OWID) COVID-19 dataset**, which tracks the spread and impact of the pandemic globally. We will focus on daily new cases for three countries: the Netherlands, Germany, and Italy, to explore and compare how case numbers change over time.

Because daily counts can be noisy (for example, due to weekend reporting effects), you will use rolling averages and logical flags to highlight important peaks. This exercise gives practice with **loops and conditionals**, two fundamental programming tools that help automate repetitive tasks and make decisions based on data.

Source

`https://catalog.ourworldindata.org/garden/covid/latest/compact/compact.csv`

Skills

`importing CSVs, subsetting, rolling averages, logical flags, time-series plots`

Preparation

You will need: `dplyr`, `ggplot2`, `readr`, `zoo`.

```
## Task: load packages (hint: library())
## library(dplyr)
## library(ggplot2)
## library(readr)
## library(zoo)
```

1. Load and subset the data

The dataset is large and contains many countries. Since we want to compare trends in only three European countries, we first need to import the full dataset and then narrow it down.

Task: Load the OWID dataset from the provided URL. Subset it to the Netherlands, Germany, and Italy. Convert the `date` column to type `Date`.

```
## Task: read CSV (hint: read_csv(url))
## Task: subset to 3 countries (hint: filter(country %in% c("Netherlands","Germany","Italy")))
## Task: convert date (hint: mutate(date=as.Date(date)))
```

2. Compute a 7-day moving average

Raw daily case counts jump up and down due to reporting schedules. A moving average smooths these fluctuations, making it easier to see real trends.

Task: For each of the three countries, calculate a 7-day rolling average of `new_cases`. Store it in a new column `ma7`.

```
## Task: compute moving average (hint: group_by(country); arrange(date); mutate(ma7=zoo::rollmean(new_c
```

3. Flag large values

Public health authorities often monitor when case numbers cross certain thresholds. Here, we will flag days with especially high case numbers.

Task: Add a logical column `flag_high` that is `TRUE` if `ma7 > 10000` and `FALSE` otherwise.

```
## Task: add flag column (hint: mutate(flag_high = ma7 > 10000))
```

4. Plot with flagged points

A visualisation makes it easy to compare the three countries. Highlighting the flagged points will help you see when major peaks occurred.

Task: Create a line plot of the 7-day averages (`ma7`) over time for each country. Use different colours for the countries, and mark flagged points with an additional symbol (e.g. points in red).

```
## Task: plot (hint: ggplot() + geom_line(); add geom_point(data=subset(..., flag_high)))
```

5. Reflection

Finally, think about the patterns. Which country had the highest peaks? Did they occur at the same time? Did the threshold capture the waves you would expect?

Task: Write 3–4 sentences comparing the trends across the three countries and reflecting on how loops and conditionals helped structure the analysis.

Write your reflection here…

R Take-Home Assignment 4: World Development Data (Multiple Indicators)

Your Name

2025-10-01

Overview

In this assignment you will explore a dataset combining three important development indicators for almost all countries in the world between 1990 and 2020:

- **GDP per capita** – a measure of economic output per person
- **Life expectancy** – an indicator of population health
- **CO_2 emissions per capita** – a measure of environmental impact

The dataset `world_data.csv` has already been prepared for you. It allows you to study how wealth, health, and environmental outcomes are related across countries and over time.

Source

File on disk `world_data.csv` (should be stored in the same folder as this `.Rmd`)

Skills

`reading local CSVs, subsetting, cleaning, plotting, interpreting relationships`

Preparation

You will need: `dplyr`, `ggplot2`, `readr`.

```r
## Task: load packages (hint: library())
## library(dplyr)
## library(ggplot2)
## library(readr)
```

1. Load the dataset

Before we can do any analysis, the dataset needs to be loaded into R. Since this file is stored locally, you'll also practise one of the most common first steps in data analysis: reading data from disk.

Task: Load `world_data.csv` into R and store it as `world_data`. Print the number of rows and columns, and look at the first few lines of the data.

```r
## Task: read CSV (hint: read_csv("world_data.csv"))
## Task: check dimensions (hint: dim())
## Task: preview (hint: head())
```

2. Subset the dataset

Instead of analysing every year, it's common to take a snapshot of one year for cross-country comparisons. Here we'll use 2015, as it provides recent data but avoids issues with missing values in the latest years.

Task: Subset the dataset to the year 2015. Show the resulting table (a few rows is enough).

```
## Task: filter year == 2015 (hint: filter(year == 2015))
```

3. Explore summary statistics

Before making plots, it's useful to see some descriptive statistics. Averages and ranges can reveal whether values look reasonable, and whether there is a lot of variation between countries.

Task: For all countries in 2015, calculate the mean and range for GDP per capita, life expectancy, and CO_2 emissions per capita.

```
## Task: summarise mean and range for all 3 variables (hint: summarise(mean(...), range(...)))
```

4. Plot GDP vs life expectancy

A classic question in development studies is whether economic prosperity translates into better health outcomes. Plotting GDP per capita against life expectancy lets us explore this visually. Adding CO_2 emissions as colour allows us to consider whether higher prosperity comes with environmental costs.

Task: Create a scatterplot of GDP per capita (x-axis, log scale) vs life expectancy (y-axis). Colour the points by CO_2 emissions per capita. Add informative axis labels and a title.

```
## Task: plot scatter (hint: ggplot(..., aes(x=gdp_per_capita, y=life_expectancy, color=co2_per_capita)
```

5. Reflection

The final step in any analysis is to connect numbers and plots back to real-world meaning. Does wealth always mean health? Are there exceptions? Do you notice patterns in CO_2 emissions?

Task: Write 3–4 sentences interpreting your findings. Mention at least one interesting or surprising pattern.

Write your reflection here…

R Take-Home Assignment 5: Statistics with Penguins

Your Name

2025-10-01

Overview

In this assignment you will practice simple statistical tests in R using the **palmerpenguins** dataset. We will ask biological questions about penguin body mass, learn to run standard tests in R, and interpret the results. Each section starts with a short story to explain why we are doing this step.

Source

`palmerpenguins` package

Skills

`t-tests, ANOVA, post-hoc tests, effect sizes, diagnostics`

Preparation

You will need: `dplyr`, `ggplot2`, `palmerpenguins`, `broom`, `effectsize`.

```
## Task: load packages
## library(dplyr)
## library(ggplot2)
## library(palmerpenguins)
## library(broom)
## library(effectsize)
```

1. Load the dataset

Before we can analyze penguins, we need to load the dataset. Real-world data often contain missing values, which can cause errors in analysis, so we will remove them first.

Task: Load the `penguins` dataset and remove missing values.

```
## Task: load penguins dataset (hint: data(penguins))
## Task: remove rows with missing values (hint: drop_na())
```

2. Two-sample t-test

Imagine you are a biologist asking: *Do male and female penguins of the same species differ in body mass?* The **t-test** compares the means of two groups. It answers whether the difference is large enough that it is unlikely to be due to chance.

Task: Filter the dataset to one species (e.g. Adelie) and compare male vs female body mass with a t-test.

```
## Task: filter to one species (hint: filter(species == "Adelie"))
## Task: run t.test(body_mass_g ~ sex, data = ...)
```

3. ANOVA

Now suppose we want to compare *all three species at once*. Do Gentoo, Adelie, and Chinstrap penguins differ in body mass?
An **ANOVA (Analysis of Variance)** extends the t-test to more than two groups.

Task: Run an ANOVA with species as the predictor.

```
## Task: run aov(body_mass_g ~ species, data = penguins)
## Task: check summary()
```

4. Post-hoc tests

ANOVA tells us if there is *some* difference between groups, but not *which groups differ*. For that, we run a **Tukey HSD post-hoc test**.

Task: Use `TukeyHSD()` on your ANOVA model to see which pairs of species differ.

```
## Task: run TukeyHSD() on your ANOVA model
```

5. Effect sizes

Statistical significance is not enough: we also want to know *how large the difference is*.
- For t-tests, we use **Cohen's d** (small 0.2, medium 0.5, large 0.8).
- For ANOVA, we use **^{2} (eta squared)** (small .01, medium .06, large .14).

Task: Calculate Cohen's d for your t-test and ^{2} for your ANOVA.

```
## Task: use effectsize::cohens_d() and effectsize::eta_squared()
```

6. Diagnostics (checking assumptions)

Every statistical test makes assumptions. For ANOVA, the two most important are:

1. **Equal spread of errors (homoscedasticity):**
   The variation in the data should be roughly the same across all groups.

2. **Normally distributed errors:**
   The differences between the observed data and the model's predictions (residuals) should follow a bell-shaped curve.

We can check these assumptions using two standard diagnostic plots:

- **Residuals vs Fitted plot**
  - Each dot is one observation.
  - The x-axis shows what the model predicts, the y-axis shows the "error" (residual).
  - What to look for: the dots should look like random noise.
    - If you see a curve → the relationship might not be captured well.
    - If you see a funnel shape → group variances may not be equal.
- **Normal QQ plot**
  - This compares your residuals to what would be expected if they were perfectly normal.
  - What to look for: the dots should fall roughly along the diagonal line.
    - If they bend away strongly → residuals may not be normally distributed.

Task: Create both plots for your ANOVA model and then write 2–3 sentences interpreting them.

```
## Task: residual vs fitted
## Task: QQ plot
```

7. Reflection

Statistics are not just numbers: they help answer real questions. Think like a biologist preparing a short report.

Task: Write 3–4 sentences: what did you learn about differences in penguin body mass across species and sexes? How strong are the effects?