Assignment Zero: Practicing the basics of R

Introduction To Chaos Applied To Systems, Processes And Products (ETSIDI, UPM)

Alfonso Allen-Perkins, Juan Carlos Bueno and Eduardo Faleiro

2025-02-18

Contents

Exercise 1: Setting up R and RStudio

1. Install **R** and **RStudio** (if not already installed).
2. Load the necessary libraries:

```
library(ggplot2)
```

3. Create a simple **vector**, **matrix**, and **data frame** in R and display their structure.

```
my_vector <- c(1, 2, 3, 4, 5)
my_matrix <- matrix(1:9, nrow=3)
my_data_frame <- data.frame(Name = c("Alice", "Bob"), Age = c(25, 30))
```

4. Print each object and check their class using `class()`.

Exercise 2: Creating functions in R

1. Define a function to compute a parabola:

```
parabola <- function(x) {
  return(x^2)
}
```

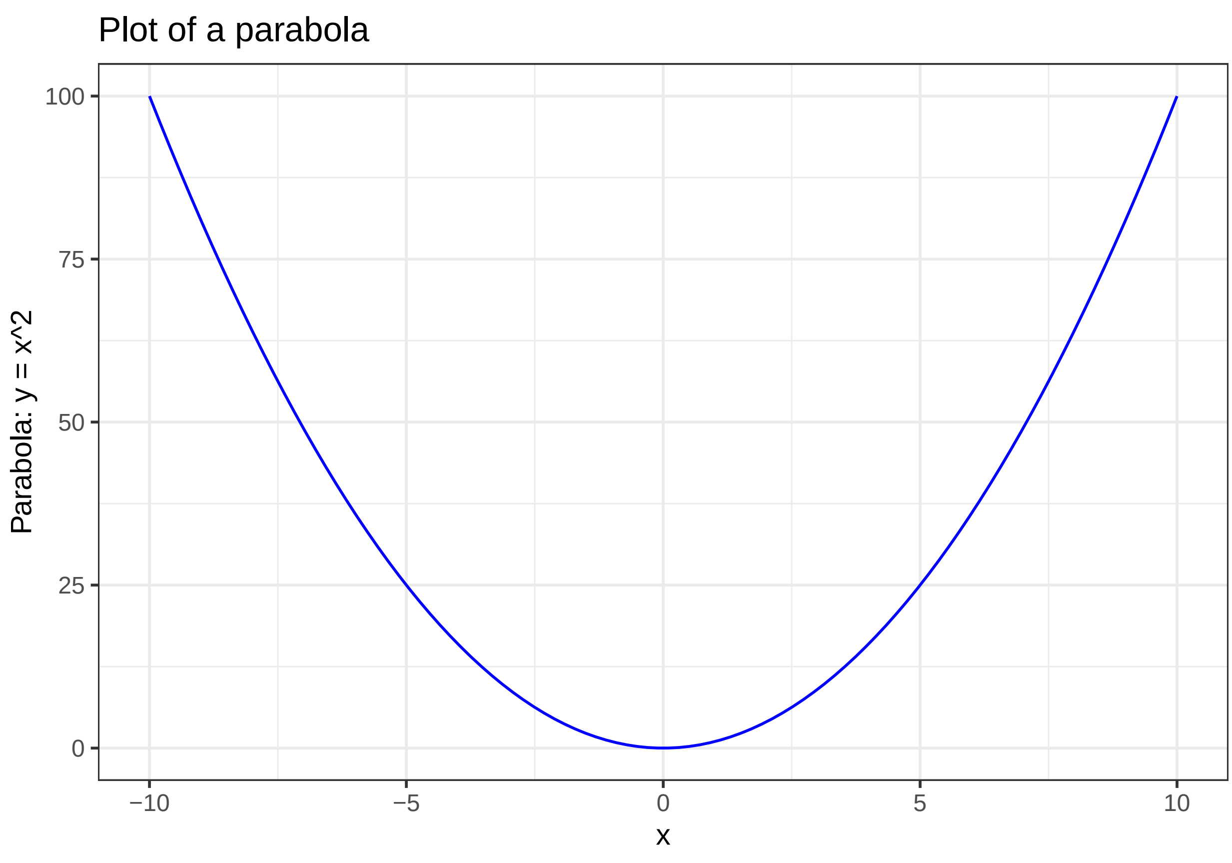
2. Define a function for a straight line:

```r
straight_line <- function(x, m=1, b=0) {
  return(m * x + b)
}
```

3. Apply these functions to a sequence of values and store results.
4. Use **ggplot2** to plot both functions over the range [-10, 10].

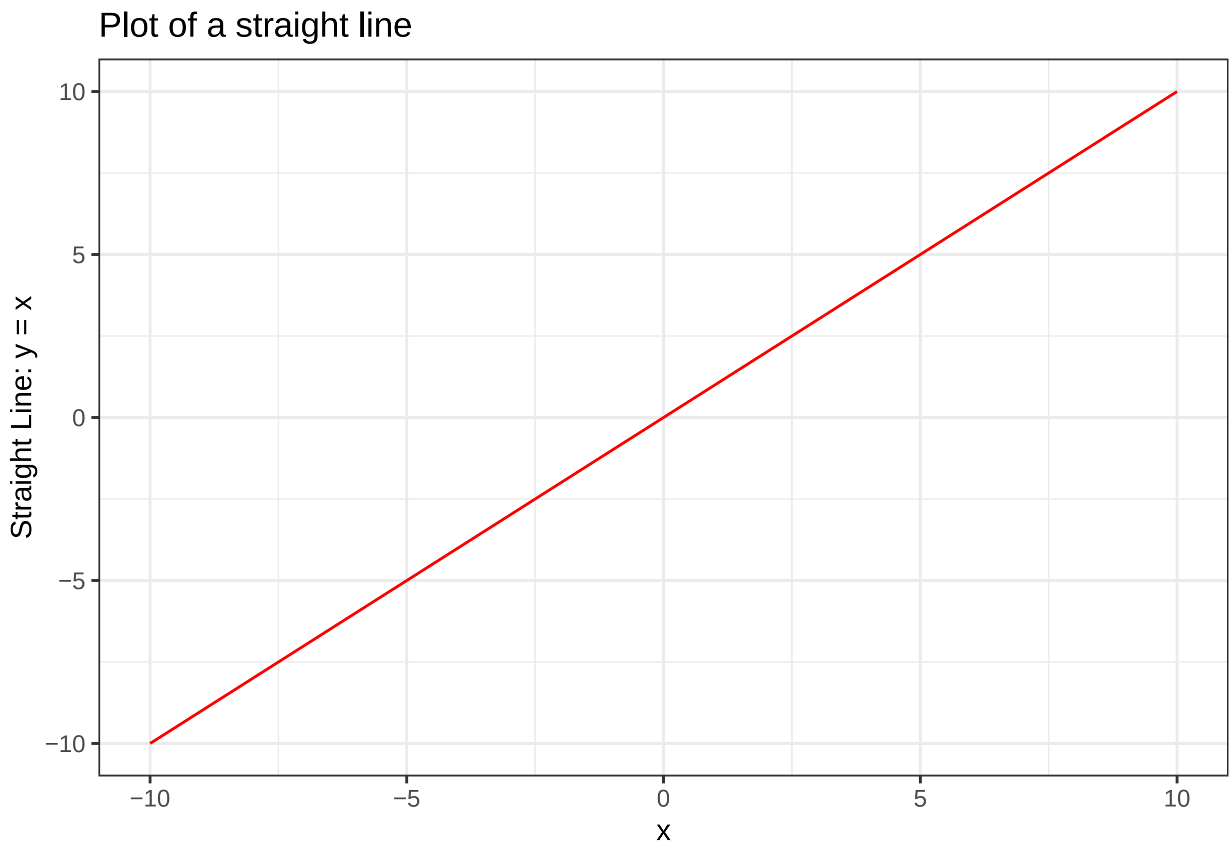
```r
 x_values <- seq(-10, 10, length.out = 100)
datafr_parabola <- data.frame(x = x_values, y = parabola(x_values))
datafr_line <- data.frame(x = x_values, y = straight_line(x_values, m=1, b=0))

 # Parabola plot
 ggplot(data = datafr_parabola, aes(x = x, y = y)) +
   geom_line(color = "blue") +
   xlab("x") +
   ylab("Parabola: y = x^2") +
   ggtitle("Plot of a parabola") +
   theme_bw()
```




```r
 # Straight line plot
 ggplot(data = datafr_line, aes(x = x, y = y)) +
   geom_line(color = "red") +
   xlab("x") +
   ylab("Straight Line: y = x") +
```

```
  ggtitle("Plot of a straight line") +
  theme_bw()
```



Exercise 3: Using a For Loop

1. Implement a **for loop** to compute the orbit of the **logistic map**:

$$x_{n+1} = rx_n(1 - x_n)$$

   using an initial condition x_0 and a given value of r.

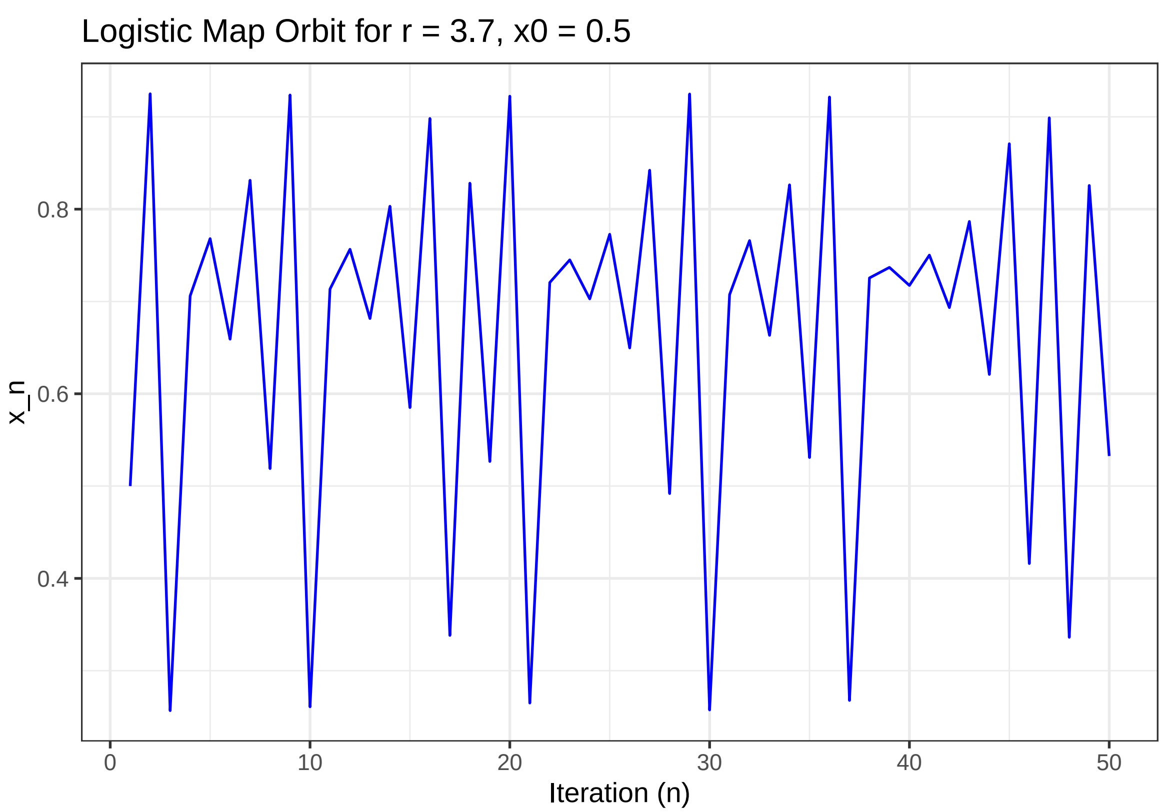
2. Store the first **50 iterations** and plot x_n **vs.** n using `ggplot2`.

```
logistic_map <- function(r, x0, n) {
  x_values <- numeric(n)
  x_values[1] <- x0
  for (i in 2:n) {
    x_values[i] <- r * x_values[i-1] * (1 - x_values[i-1])
  }
  return(data.frame(n = 1:n, x = x_values))
}
```

```r
# Example usage
r_value <- 3.7
x0_value <- 0.5
n_iter <- 50

orbit_data <- logistic_map(r_value, x0_value, n_iter)

ggplot(orbit_data, aes(x = n, y = x)) +
  geom_line(color = "blue") +
  xlab("Iteration (n)") +
  ylab("x_n") +
  ggtitle("Logistic Map Orbit for r = 3.7, x0 = 0.5") +
  theme_bw()
```



3. Try different values of r (e.g., 2.5, 3.0, 3.5, 4.0) and observe the behavior.

Exercise 4: Finding and visualizing roots of functions

Task 1: Finding roots of a polynomial

1. Find all the roots of the polynomial $x^2 - 1$:

```r
fixed_points <- polyroot(c(-1, 0, 1))
print(fixed_points)
```

```
## [1]  1+0i -1+0i
```

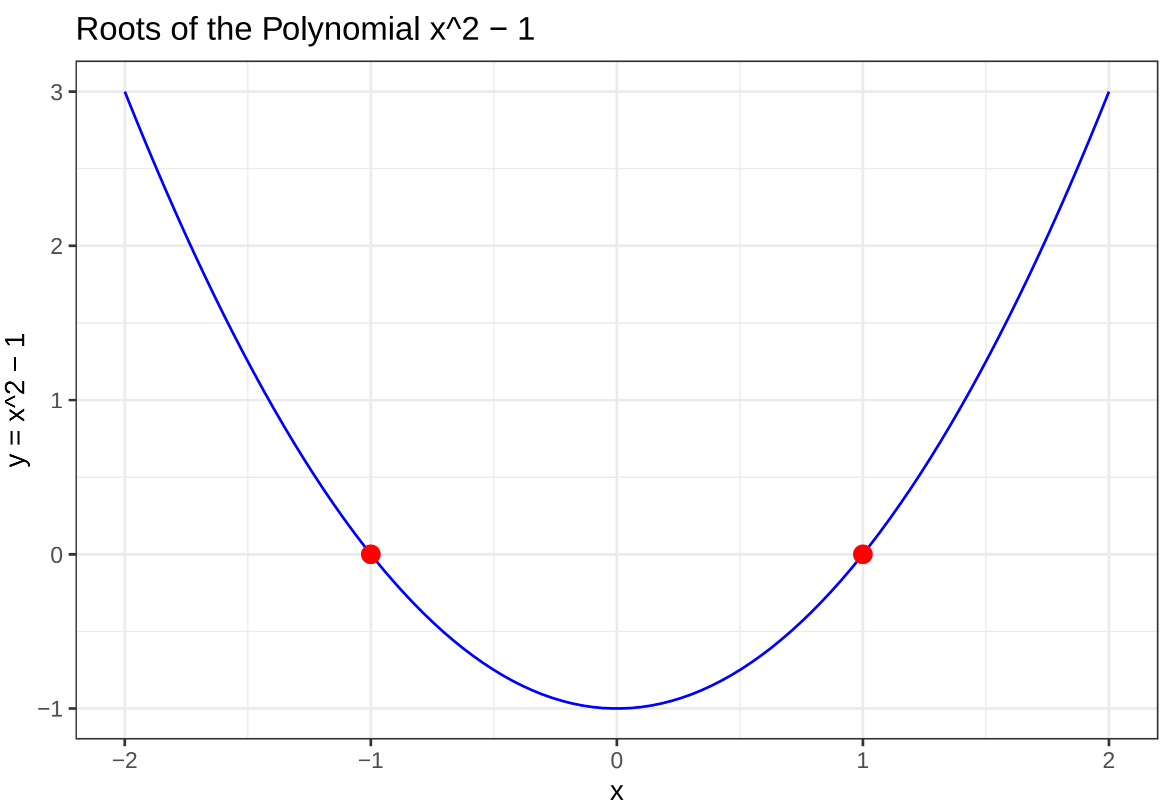
Note: To learn how to use the polyroot function, type `?polyroot` in the console.

2. Plot the polynomial function and highlight the roots:

```r
x_values <- seq(-2, 2, length.out = 100)
y_values <- x_values^2 - 1

polynomial_curve_df <- data.frame(x = x_values, y = y_values)

ggplot(data = polynomial_curve_df, aes(x = x, y = y)) +
  geom_line(color = "blue") +
  geom_point(data = data.frame(x = Re(fixed_points), y = 0), aes(x = x, y = y), color = "red", size
  xlab("x") +
  ylab("y = x^2 - 1") +
  ggtitle("Roots of the Polynomial x^2 - 1") +
  theme_bw()
```



Task 2: Finding roots of a non-polynomial function

1. Define the function $f(x) = e^x - 10\cos(x)$ and find a root in a given interval (for example, $[0, 2]$):

```r
dx_dt <- function(x) {
  exp(x) - 10 * cos(x)
}

fixed_point_0 <- uniroot(dx_dt, interval = c(0, 2))$root
print(fixed_point_0)
```

```
## [1] 1.223853
```

Note: To learn how to use the polyroot function, type `?uniroot` in the console.

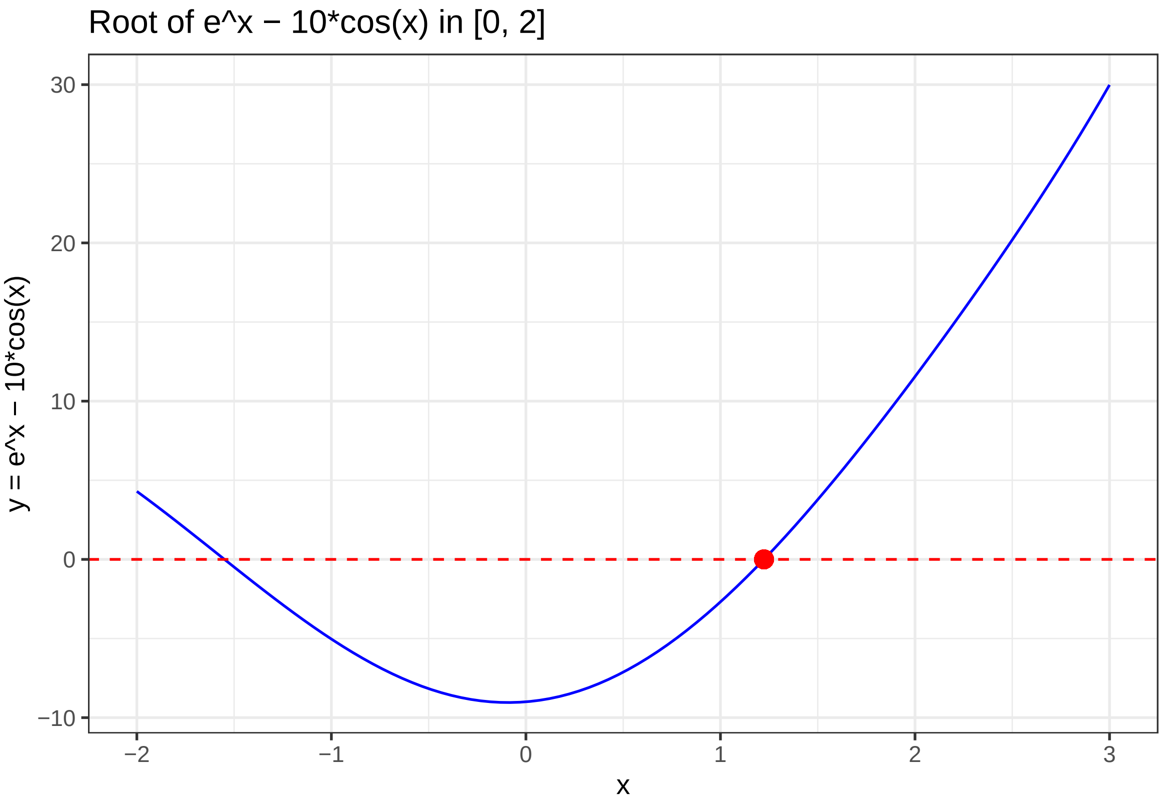
2. Plot the function and highlight the root:

```r
x_values <- seq(-2, 3, length.out = 100)
y_values <- exp(x_values) - 10 * cos(x_values)

non_polynomial_curve_df <- data.frame(x = x_values, y = y_values)

ggplot(data = non_polynomial_curve_df, aes(x = x, y = y)) +
  geom_line(color = "blue") +
  geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
  geom_point(aes(x = fixed_point_0, y = 0), color = "red", size = 3) +
  xlab("x") +
  ylab("y = e^x - 10*cos(x)") +
  ggtitle("Root of e^x - 10*cos(x) in [0, 2]") +
  theme_bw()
```

```
## Warning in geom_point(aes(x = fixed_point_0, y = 0), color = "red", size = 3): All aesthetics ha
## i Please consider using `annotate()` or provide this layer with data containing
##   a single row.
```



Exercise 5: Reproducing figures from Strogatz's book (avalaible in Moodle)

Task 1: Reproduce Figure 2.7.3

- Create an **R script** that uses `ggplot2` to reproduce **Figure 2.7.3** from Strogatz's book.
- Save the script as `figure_2_7_3.R` and generate the corresponding plot.

Task 2: Reproduce Figure 10.2.3

- Create another script, `figure_10_2_3.R`, to replicate **Figure 10.2.3** from Strogatz's book.
- Save the resulting figures as PNG or PDF files.
- Submit both **scripts and figures** via email.

Note: Ensure that your plots use sufficient data points and appropriate labels for clarity.