Exercise Sheet - Function Debugging

1. Debugging

a) Have a look at the following functions. You can find the code on

```
ftp://stat.ethz.ch/U/sfs/RKurs/Datasets/Fun.R
```

Copy the file onto your desktop and open it with R.

```
fun1 <- function(x){
  r <- x - 1 / fun2(x)
  r
}

fun2 <- function(y){
  r <- y * log(fun3(y))
}

fun3 <- function(z){
  z <- z^2
  r <- 1 - sqrt(z^2) / z
  if (r < 10)
     r^2
  else r^3
  r
}
```

b) Calculate by hand the output of `fun1()` if you would enter $x = 2$. Check the result using R with the command `fun1(2)`.

c) Where is the error when passing $x = 0$ to `fun1()`? Use `traceback()` and `debug()` to find the error in the functions.

2. Fibonacci

Have a look at the following function `Fibonacci.R`. You can find the function code on

```
ftp://stat.ethz.ch/U/sfs/RKurs/Datasets/Fibonacci.R
```

The function is supposed to calculate the first `x` Fibonacci numbers, and to plot them in a"spiral-plot". Unfortunately it still contains an error and doesn't work. Use `traceback()` and `debug()` to find the two errors in the functions.

Hint: Make sure the function works when passing one of the following two arguments to the function `Fibonacci.R`:

```
x = 10; x = log(-1)
```

Fibonacci.R:

```
library(shape)

plot.Fib <- function(x){
  # empty plot
  plot(-max(x):max(x), -max(x):max(x), type = "n", xlab = "", ylab = "",
       main = "Fibonacchi spiral")

  # x-coordinate for the center of the circle
  x.coor <- c(0, 0, cumsum(x * rep(c(1, 0, -1, 0), length = length(x))))

  # y-coordinate for the center of the circle
  y.coor <- c(0, 0, cumsum(x * rep(c(0, 1, 0, -1), length = length(x))))

  a1 <- 0:length(x) * pi / 2 #starting angle
  a2 <- 1:(length(x) + 1) * p / 2
  for (i in 1:length(x)) {
            plotcircle(r = x[i], mid = c(x.coor[i], y.coor[i]),
                       from = a1[i], to = a2[i])
            #plot the quartercircles
       }
  }


Fibonacci <- function(n) { #function to generate Fibonacci numbers
    if (n == NaN) {
    break
    }
    if (n == 1) {
        x <- 0
    } else {
        x <- c(0, 1)
        while (length(x) < n) {
            position <- length(x)
            new <- x[position] + x[position - 1]
            x <- c(x, new)
        }
    }
    plot.Fib(x)
        return(x)
}
```

3. Functions for Graphical Output, Debugging

Our goal is to create a function that displays each variable in a data frame as a histogram, including sensible titles and axis labels for each histogram. We want to apply the function for example to the data frame `d.pcb` which can be loaded via:

```
url <- "http://stat.ethz.ch/~stahel/courses/R/pcb.txt"
d.pcb <- read.table(url, header = TRUE)
```

The `pcb` data set reports 122 measurements of Polychlorinated biphenyls (PCB) concentrations in the sediments of the North Sea outward of the Dutch coast. The data set contains the following variables:

- the year of the measurements (variable `year`)

- the UTM coordinates (variables `x` and `y`, in meters),
- the shortest distance of the location where the measurement was taken to the coast (variable `coast`, in meters),
- the water depth at the measurement location (variable `depth`, in meters), and
- the PCB content (variable `pcb`, normalized concentration).

The output of the function should look similar to Figure 1 shown below.

a) Your function should take the name of a data frame as the input variable. First develop your plotting function only for continuous variables. In other words, consider only `d.pcb[, 2:6]` as an input argument. Use the function `hist()` to plot histograms in R. See `?hist` for information on all arguments.

Hints:

- Find out how many variables there are to plot.
- Automatically arrange the histograms on one page using the `par(mfrow = ...)` command.
- Don't forget to label the axes and to add titles.

b) BONUS: As an extension to question (a), try to include an **if**-condition in your function that checks for factor variables. The function should produce a barplot for each of these factor variables and add these plots to the previous output of the function (i.e. the histograms).

Hint: Use the function `is.factor()` to check if a variable is a factor and the function `barplot()` to create barplots in R.

c) Now apply your function to a new data set. Check whether your function works for the data frame `d.sport` which you can load via:

```
url <- "http://stat.ethz.ch/Teaching/Datasets/NDK/sport.dat"
d.sport <- read.table(url, header = TRUE)
```

d) Debug your code if an error occured or create an error and use `browser()` to debug it.

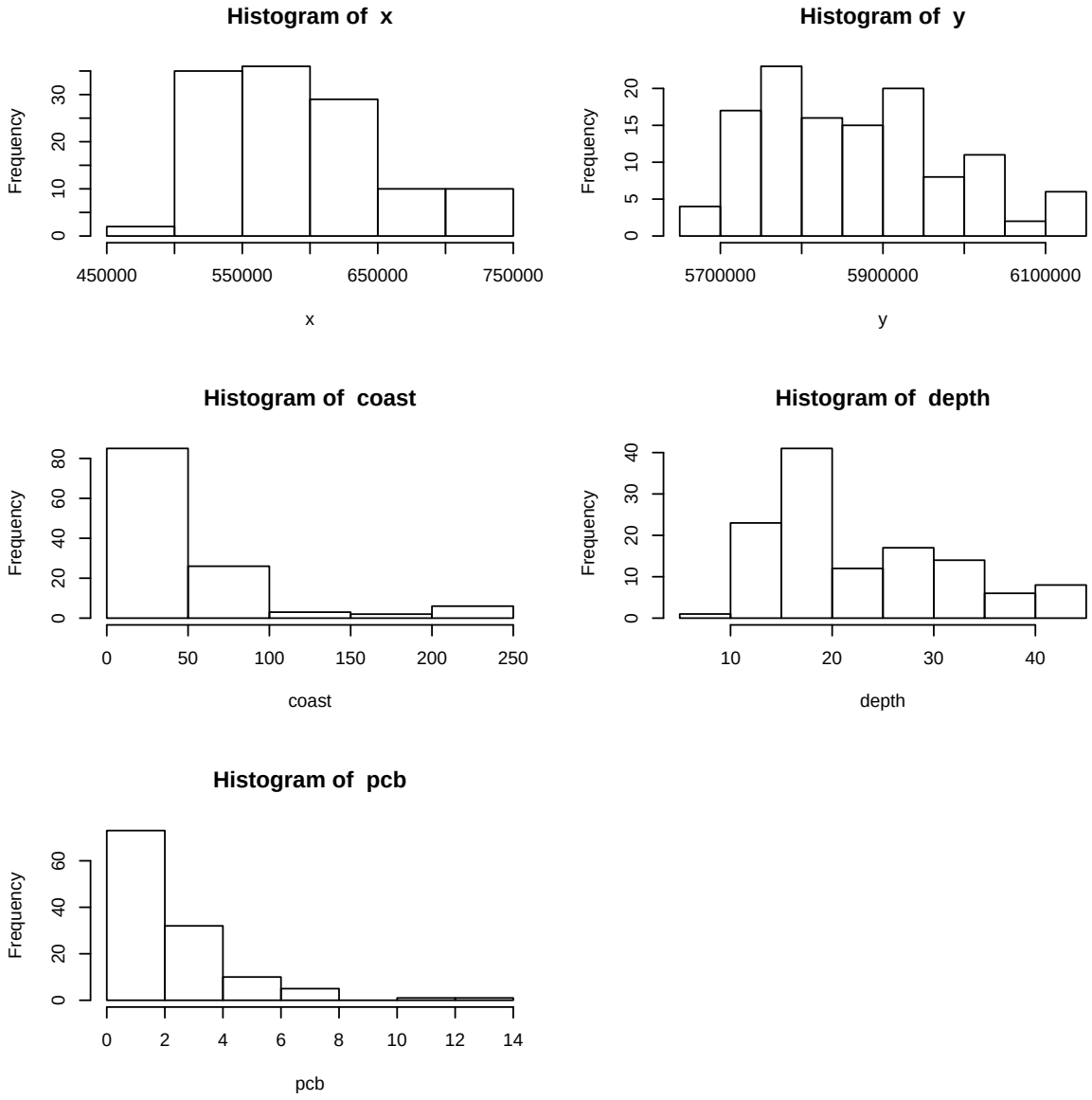

Figure 1: Histograms of the continuous variables of the data frame `d.pcb`.