R plotting exercises

Exercise 1

- How can you find plotting parameters that you can change?
- What is the default value for the setting for the plotting parameter `bty`?
- How can you query the current setting for a plotting parameter?
- Can you find the argument to supply to `plot()` if you want to suppress only plotting of the x or y axis?

```
# Suppressing only the x or y axes:
xaxt = "n"
yaxt = "n"
```

Bonus tasks: What is the argument you would use to change the line type of the whiskers when using `boxplot()` and where can you find it in the help pages? (It's pretty hard to find!)

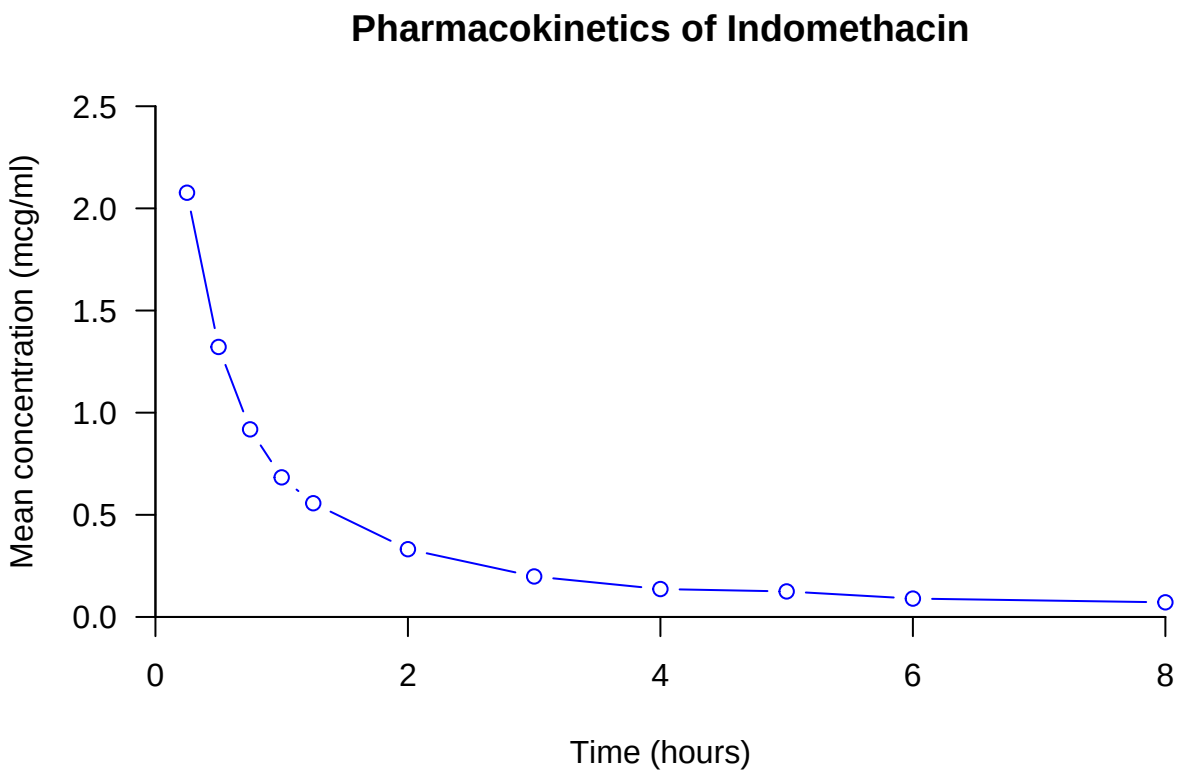
```
whisklty
# You needed to look under see also and navigate to the "bxp" help page.
```

Exercise 2

Use the R dataset `Indometh` to reproduce the following plot showing the mean pharmacokinetics of Indomethacin in 6 subjects.

```
# Get means for each time point
timepoints <- unique(Indometh$time)
mean_conc  <- rep(NA, length(timepoints))

for(n in 1:length(timepoints)){
  tpoint       <- timepoints[n]
  mean_conc[n] <- mean(Indometh$conc[Indometh$time == tpoint])
}

# Plot the average data.
plot(x = timepoints,
     y = mean_conc,
     # y = log(mean_conc), <- One option for a logged y axis
     # (but also change axis label!)
     xlab = "Time (hours)",
     ylab = "Mean concentration (mcg/ml)",
     xlim = c(0,8),
     ylim = c(0,2.5),
     xaxs = "i",
     yaxs = "i",
     xpd = TRUE,
     # ylab = expression("Mean concentration ("*mu*"g/L)"),
     # log = "y", <- With logged y axis but unlogged y axis labels.
     col = "blue",
     main = "Pharmacokinetics of Indomethacin",
     las  = 1,
```

```
     bty = "l",
     type = "b")
```



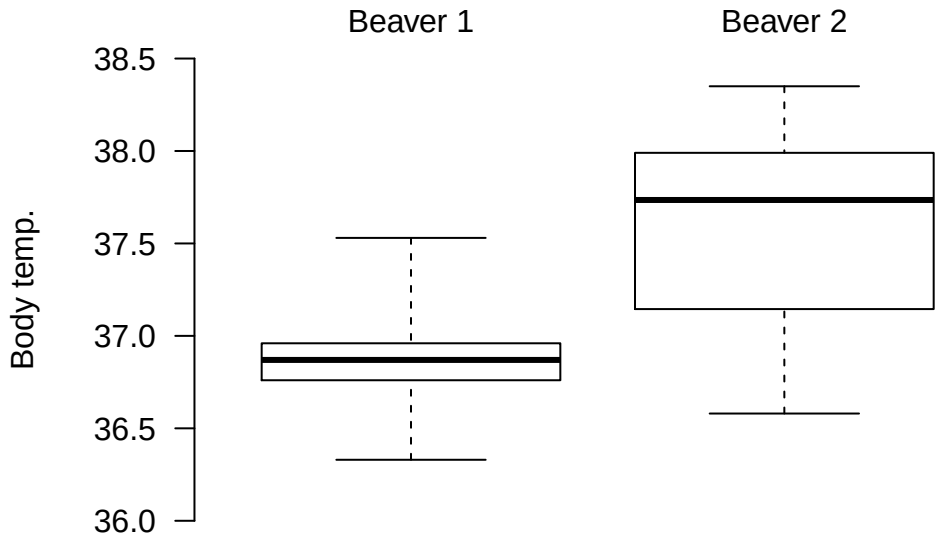
Bonus tasks: How would you plot the y-axis data on the log scale? Can you think of more than one approach? How would you make the y-axis label show "Concentration (μg/L)" using scientific notation?

Exercise 3

Using data from the datasets `beaver1` and `beaver2`, reproduce the following plot using boxplots to compare the body temperature of the two beavers during the experiment.

```
# Set margin sizes
par(mar=c(2,7,5,2))

# Produce the boxplot without axes
boxplot(x = list(beaver1$temp, beaver2$temp),
        names = c("Beaver 1", "Beaver 2"),
        ylim = c(36,38.5),
        range = 0,
        axes = FALSE)

# Add in the y axis and label manually
axis(2, las = 1)
mtext(text = "Body temp.",
      side = 2,
      line = 4)
```

```r
# Label the two boxplots
mtext(text = "Beaver 1",
      side = 3,
      at = 1)

mtext(text = "Beaver 2",
      side = 3,
      at = 2)
```



Exercise 4

(a) What is the problem with the following code attempting to plot data on the same x axis but using different y axes?

```r
plot(x = x_data,
     y = y_data1,
     xlim = c(0,100),
     ylim = c(10,80),
     xaxs = "i",
     yaxs = "i")

plot.window(xlim = c(0,100),
            ylim = c(200, 800))

points(x = x_data,
       y = y_data2)
```

```r
# When the first plot is done xaxs = "i" was called so that no padding is
# added to the x limits, however when plot.window is called xaxs was
# omitted and will default to "r" and add padding to the x limits. This is
# not a problem for y since you are replotting this axis scale anyway, but
# the new x axis used will not be the same as the first plot anymore. To
# correct it you need to either remove xaxs = "i" from the plot function or
# add it into the plot.window function.
```

(b) Imagine you have been given the dataset `airquality` and you would like to visualise how Solar radiation varied with temperature in the month of July. Plot two lines showing the data on the same plot, but using different y axis scales. Add a legend to help distinguish between the line showing temparature and that showing solar radiation.

```
# Get the subset of data for the month of July
july_data <- airquality[airquality$Month == 7,]

# Set the plotting margins to allow space for the extra y axis on the right
par(mar = c(5,5,4,5))

# Do an initial plot of temperature over time
plot(x = july_data$Day,
     y = july_data$Temp,
     type = "l",
     col = "red",
     main = "Temperature and Solar radiation in July",
     xlab = "Day",
     ylab = expression("Temperature ("*phantom()^"o"*"F)"),
     las = 1)

# Redefine the plotting space, changing the y limits
plot.window(xlim = range(july_data$Day),
            ylim = range(july_data$Solar.R))

# Plot the new y axis and label
axis(side = 4, las = 1)
mtext(text = "Solar radiation (lang)",
      side = 4,
      line = 3)

# Plot the solar radiation data
lines(x = july_data$Day,
      y = july_data$Solar.R,
      col = "blue")

# Plot a legend
legend("bottomleft",
       legend = c("Temperature","Solar radiation"),
       fill = c("red", "blue"),
       bty = "n")
```

Bonus tasks: Can you think of another way to visualise the relationship between temperature and solar radiation? How could you use such data to go further and answer the question of whether temparature and solar radiation are linked?

```
# There are a few way in which you could approach this, a good start would
# be a scatterplot of solar radiation against temperature to first examine
# for any striking patterns. The data is quite noisy though so it might be
# hard to see a relationship by eye. You could fit a linear model comparing
# the two variables using the lm() function and see if there is a
# significant relationship, or (perhaps a better start than a linear model)
# you could simply check for correlation between the two using the cor()
# function. Always try and visualise the data first though before applying
# statistical tests! Outliers that would otherwise have unfairly skewed
# things may become obvious and most statistical tests will fail if the
# relationship is not straightforward, for example a "u" shaped
```

```
# relationship. Also think about more than one way to visualise the data.
# For example even a scatterplot of temperature against solar radiation
# could be misleading if solar radiation had a delayed effect and was
# highly correlated with temparature readings taken a week later rather
# than the same day, but this may be obvious when both are plotted together
# against time.
```

Bonus exercise 1: Making your own plotting functions

Create your own plotting function that by default plots horizontal axis labels, axes that extend exactly to the ranges specified and an L-shaped box by default. Appropriately use the … argument to pass on other arguments from your function to plot().

```
myplot <- function(x, y,
                   xaxs = "i",
                   yaxs = "i",
                   las  = 1,
                   bty = "l",
                   ...){

  plot(x = x,
       y = y,
       xaxs = xaxs,
       yaxs = yaxs,
       las  = las,
       bty  = bty,
       ...)

}
```

Bonus exercise 2: Bringing it all together

Use data provided by the World Health Organisation to plot the incidence of different subtypes of influenza circulating over time.

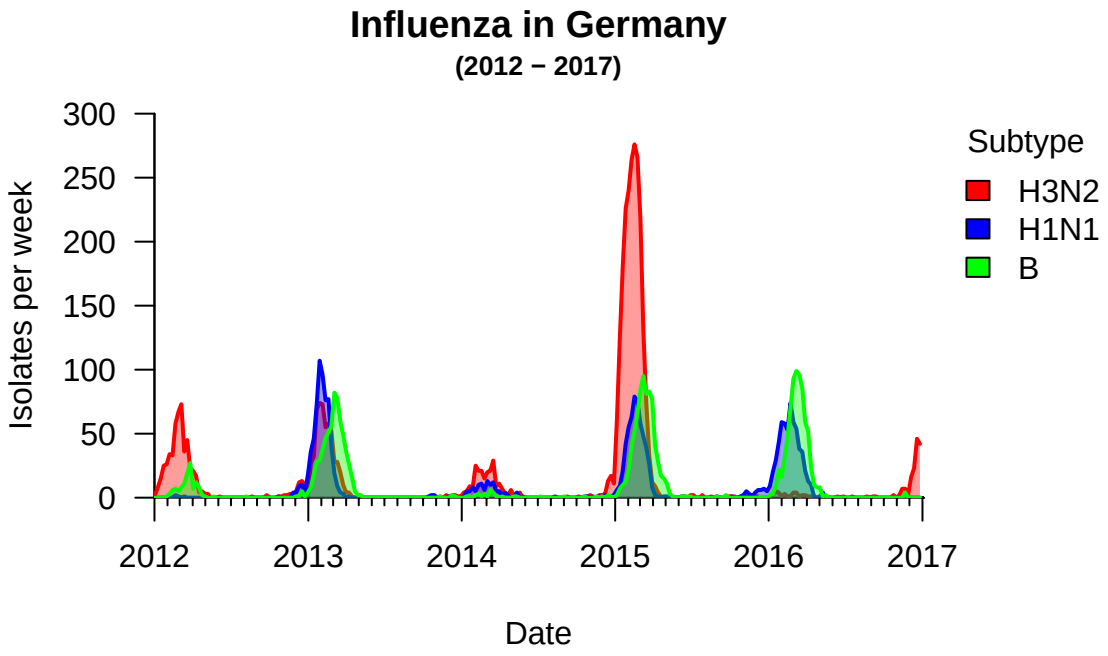
You can download the data yourself and select your own country of interest and date-range by going to http://www.who.int/influenza/resources/charts/en/ and clicking on "Download data for any time period with country selection" in the right panel. Alternatively you can use the data for Germany between 2012 and 2017 included in the file "FluNetInteractiveReport.csv".

Try and produce a plot similar to the one below:

```
# Read in the influenza circulation data
flu_data <- read.csv(file   = "data/FluNetInteractiveReport.csv",
                     header = TRUE,
                     skip   = 2,
                     stringsAsFactors = FALSE)

# We want to convert some of the date data into the "date" data type
flu_data$SDATE <- as.Date(flu_data$SDATE)

# Define the date range you wish to plot
date_range <- as.Date(c("2012-01-01", "2017-01-01"))

# Set plot margins
```

```r
par(mar = c(5,7,5,8))

# Setup the plot
plot.new()
plot.window(xlim = date_range,
            ylim = c(0, 300),
            xaxs = "i",
            yaxs = "i")

# Plot an l-shaped box
box(bty = "l")

# Label y axis
axis(side = 2,
     las  = 1)

# Now add tick marks and axis labels by month and year
for(year in 2012:2017){
  for(month in 1:12){

    # You want the label date to be the first day of each month
    label_date <- as.Date(paste(year, month, "01", sep = "-"))

    # You want special behaviour if it is the first month of the year
    if(month == 1){
      axis_label <- year
      axis_tcl   <- -0.6
    }
    else{
      axis_label <- FALSE
      axis_tcl   <- -0.2
    }

    # Now plot an axis tick at that label position
    axis(side   = 1,
         at     = label_date,
         labels = axis_label,
         tcl    = axis_tcl)

  }
}


# Add axis labels and title
title(main = "Influenza in Germany",
      line = 2)
title(main = "(2012 - 2017)",
      line = 1,
      cex.main = 0.8)
title(xlab = "Date",
      ylab = "Isolates per week")
```

```r
# Setup a function to plot your flu incidence data
plot_incidence <- function(week_dates,
                           flu_incidence,
                           plot_color){

  # You can first use the col2rgb function to create a faded colour for the
  # polygon fill
  faded_plot_color <- rgb(red   = col2rgb(plot_color)[1],
                          green = col2rgb(plot_color)[2],
                          blue  = col2rgb(plot_color)[3],
                          alpha = 100,
                          maxColorValue = 255)

  # You can use a polygon to fill in the shape underneath the lines
  polygon(x = c(week_dates, rev(week_dates)),
          y = c(flu_incidence, rep(0, length(flu_incidence))),
          col    = faded_plot_color,
          border = NA)

  # You can then simply plot the line separately
  lines(x = week_dates,
        y = flu_incidence,
        col = plot_color,
        lwd = 2)

}

# Plot incidence for each type of flu
plot_incidence(week_dates    = flu_data$SDATE,
               flu_incidence = flu_data$AH3,
               plot_color    = "red")

plot_incidence(week_dates    = flu_data$SDATE,
               flu_incidence = flu_data$AH1N12009,
               plot_color    = "blue")

plot_incidence(week_dates    = flu_data$SDATE,
               flu_incidence = flu_data$INF_B,
               plot_color    = "green")


# Plot the legend
legend(x = "topright",
       inset = c(-0.25, 0),
       title = "Subtype",
       legend = c("H3N2",
                  "H1N1",
                  "B"),
       fill = c("red",
                "blue",
                "green"),
       bty = "n",
       xpd = TRUE)
```


Influenza in Germany
(2012 – 2017)
Isolates per week
300
250
200
150
100
50
0
2012
2013
2014
2015
2016
2017
Date
Subtype
H3N2
H1N1
B