Intro to R Programming: Assignment 1

You can obtain a total of 30 marks for this problem sheet. Please upload your answers **before 30th October 2018, 12noon**. To upload your answers, log on to Moodle and go to *Introduction to R Programming*. Under *Assignments*, there is a link *Upload answers for assignment 1*. Click on this link to upload the R file containing your R code and your comments. You do not need to upload R output. Please do not upload Word documents or zip files. The name of the file **must** be your student ID number.

Under *Assignments* on Moodle there is also a folder *Data files for assignment 1* which includes all data files needed for this assignment.

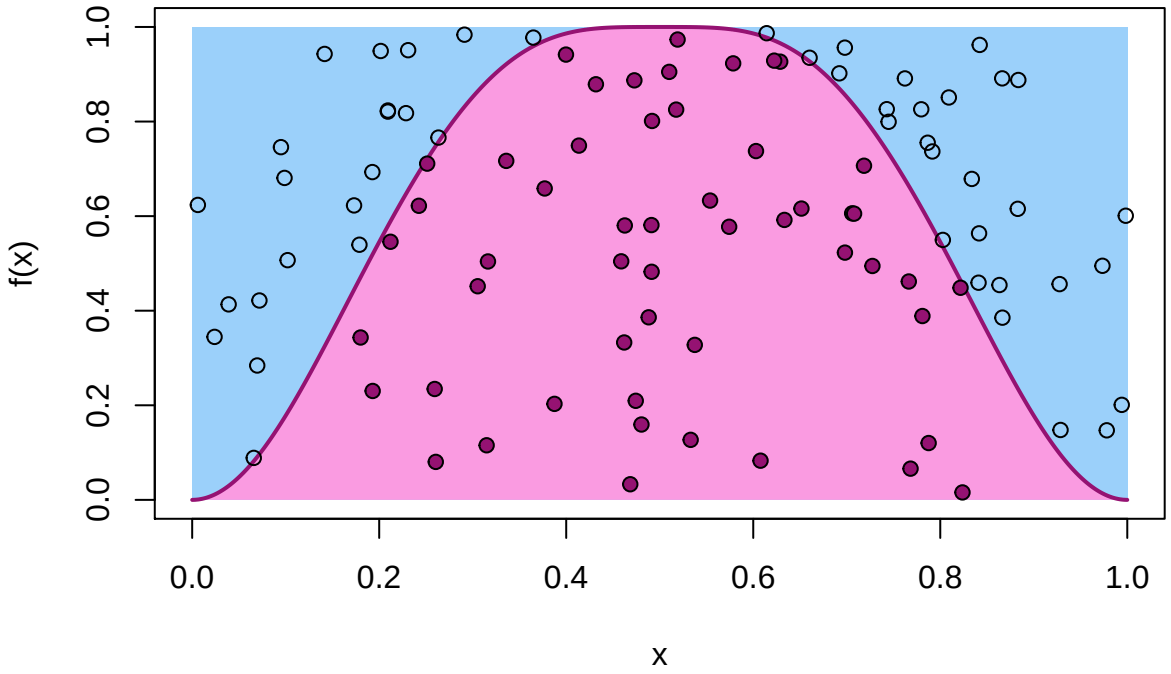
Task 1

Suppose we want to calculate the integral $\int_0^1 f(x)\,dx$ for the function

$$f:\ [0,1] \to [0,1],\ x \mapsto f(x) = 1 - \sqrt{\exp(\sin(\pi x)^4)}\cos(\pi x)^6.$$

Integrating this function is non-trivial, so we will compute this integral using a simulation ("Monte Carlo integration").

Suppose we consider uniform "rain" falling on the unit square $[0,1] \times [0,1]$. The probability of a "rain drop" falling below $f(x)$ (pink area in the figure below) is just the area under the function, which is nothing other than the integral of interest, $\int_0^1 f(x)\,dx$.



We can thus estimate $\int_0^1 f(x)\,dx$ as follows. We first draw a large number (say $n = 10,000$) points (x_i, y_i) uniformly from the unit square $[0,1] \times [0,1]$. The integral $\int_0^1 f(x)\,dx$ is then estimated by the proportion of simulated points for which $y_i \le f(x_i)$.

Implement this method to estimate the integral $\int_0^1 f(x)\,dx$ following the steps below:

1. **[1 mark]** Create a 10000×2 data frame `U` with random numbers from the uniform distribution (*Hint: The function* `runif(n)` *generates* n *random numbers, uniformly distributed between 0 and 1.*).
2. **[1 mark]** Give to the columns of `U` the names `x` and `y`.

3. **[1 mark]** Add a column to `U` with the value of the function $f(x)$ for the values in column `x`. Name the column `fx`.

4. **[1 mark]** Calculate the proportion of rows of `U` for which the entry in column `y` is smaller than the entry in column `fx`.

Task 2

In this question we will work with the data set `starwars`.

Answer the questions below without manually extracting information from the data.

1. **[1 mark]** Give the code to read the data correctly into R.

2. **[1 mark]** Add a column `BMI` to the dataset: it should contain the body mass index ($10000 \times \text{mass}/\text{height}^2$ when mass is measured in kg, and height measured in cm).

3. **[1 mark]** Find the character with the highest BMI.

4. **[1 mark]** For every homeworld, find how many characters come from it.

5. **[1 mark]** Find the names of all characters which are from the same homeworld as Chewbacca.

6. **[1 mark]** Identify all characters taller than 2 metres.

Task 3

The first digit of numbers in many real-word data sets (like the population sizes of towns, lengths of rivers, numbers of votes in an election) does not have a uniform distribution. The digit `1` for example occurs much more often than the digit `9`. In 1938, the physicist Frank Benford suggested that the first digit of most empirical data has a distribution whose probability mass function (p.m.f.) is given by

$$p(x) = \begin{cases} \log_{10}\left(1 + \frac{1}{x}\right) & \text{for } x \in \{1, 2, \ldots, 9\} \\ 0 & \text{otherwise,} \end{cases}$$

where $\log_{10}(\cdot)$ denotes the logarithm with base 10.

Hint: $\log_{10}(x)$ *can be found in R using* `log(x, base=10)`.

1. **[1 mark]** Define a vector `x` containing the numbers $x = (1, \ldots 9)$ and a vector `p` containing the values of $p(x)$ for these x.

2. **[1 mark]** Use R to verify that the entries of `p` sum to one.

3. **[1 mark]** Use R to identify the entry of `x` for which $p(x)$ is maximum.

4. **[1 mark]** Use R to compute the expected value $E(X) = \sum_{x=1}^{9} xp(x)$. Do not use the built-in function `mean`.

5. **[1 mark]** Use R to compute the variance $\text{Var}(X) = \sum_{x=1}^{9}(x-\mu)^2 p(x)$, where $\mu = E(X)$. Do not use the built-in function `var`.

Task 4

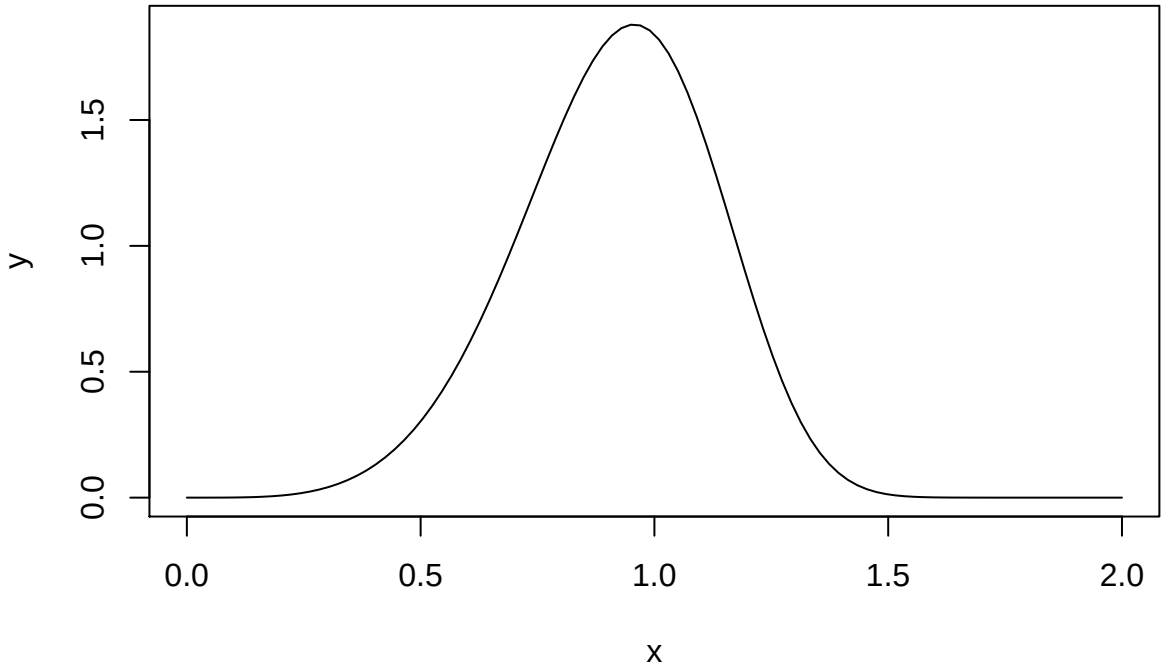
The Weibull distribution is a continuous probability distribution, named after Swedish mathematician Waloddi Weibull, who described it in detail in 1951. The probability density function of a Weibull random variable X with parameters λ and k is

$$f(x) = \begin{cases} \frac{k}{\lambda}\left(\frac{x}{\lambda}\right)^{k-1} \exp(-(x/\lambda)^k), & \text{if } x \geq 0 \\ 0, & \text{if } x < 0 \end{cases}$$

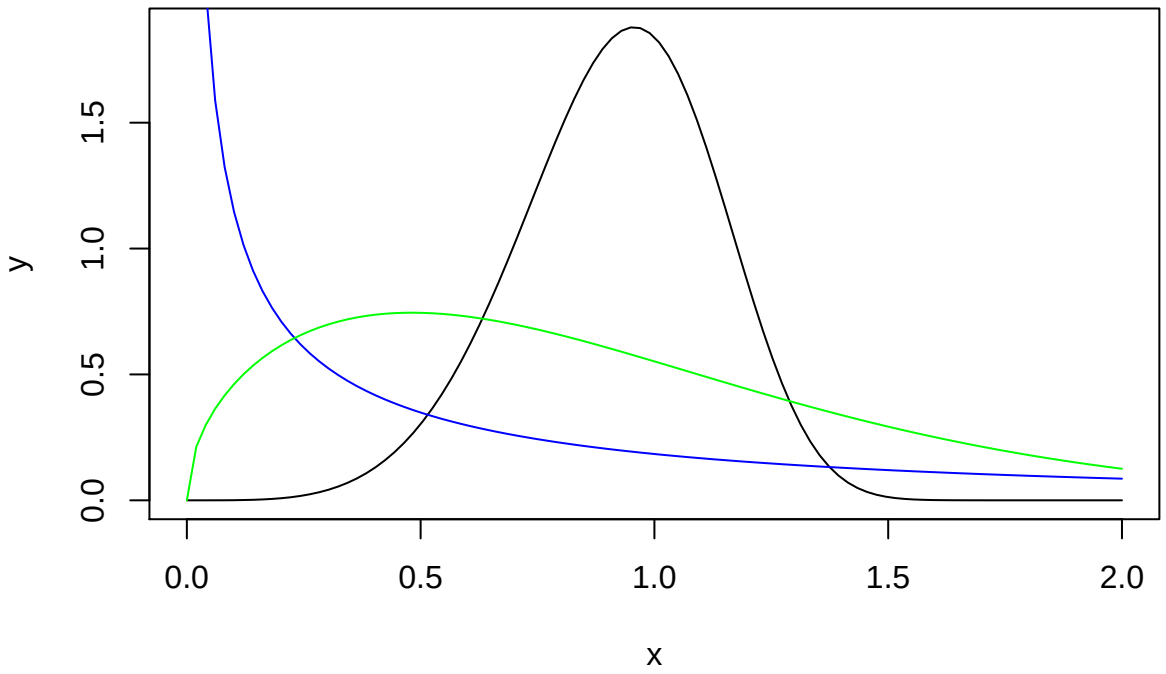
1. **[1 mark]** Create a vector `x` of length 100 with equally spaced numbers between 0 and 2.

2. **[1 mark]** Create a vector `y` with the values of the probability density function of a Weibull distribution with parameters $\lambda = 1$ and $k = 5$.

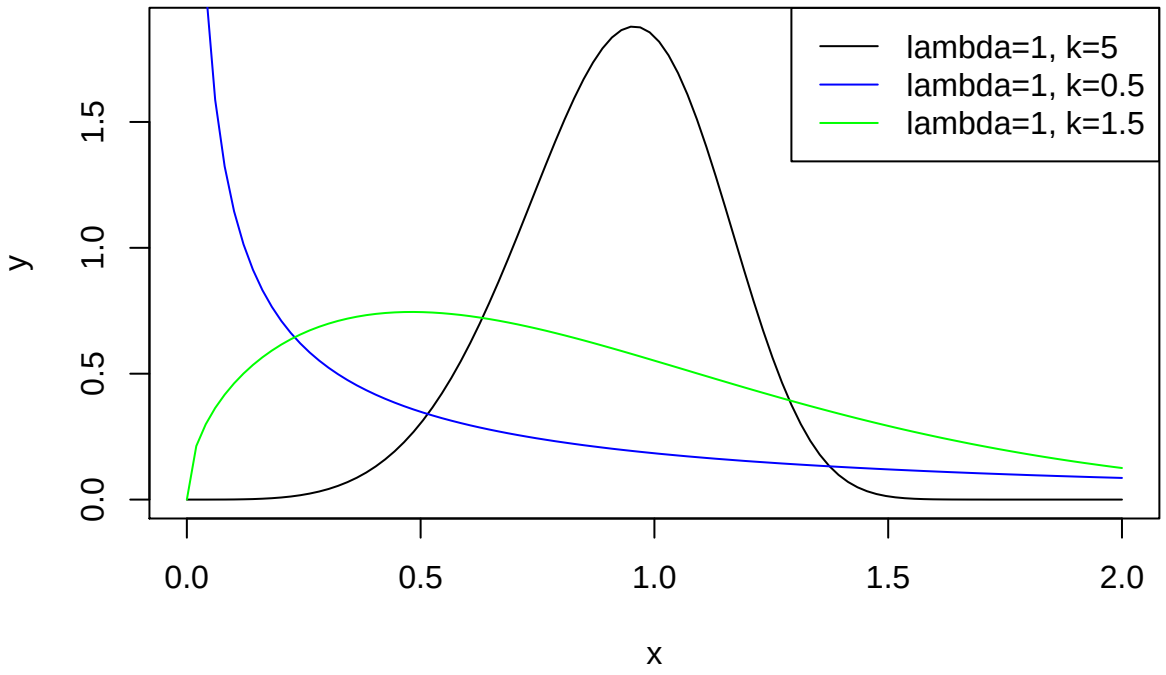
3. **[1 mark]** Plot the density of the Weibull distribution from part 2. *Your plot should look similar the one below*



4. **[1 mark]** Add the density of a Weibull distribution with parameters $\lambda = 1$ and $k = 0.5$ (in blue) and the density of a Weibull distribution with parameters $\lambda = 1$ and $k = 1.5$ (in green) to the plot from part 3. *Your plot should look similar the one below.*



5. **[1 mark]** Add a legend to the plot from part 4. *Your plot should look similar the one below.*



Task 5

The tweets posted by `@realDonaldTrump` are either written by himself or his team.

Donald Trump is known to have used a Samsung Galaxy smartphone to write his tweets, whereas his team uses iOS devices. The device used to send a tweet can be retrieved using the Twitter API, so it used to be easy to determine whether a tweet was written by Donald Trump himself or his team. However, in March 2017, Donald Trump switched to an iPhone, so these days there is no way of telling whether Donald Trump has written a tweet himself.

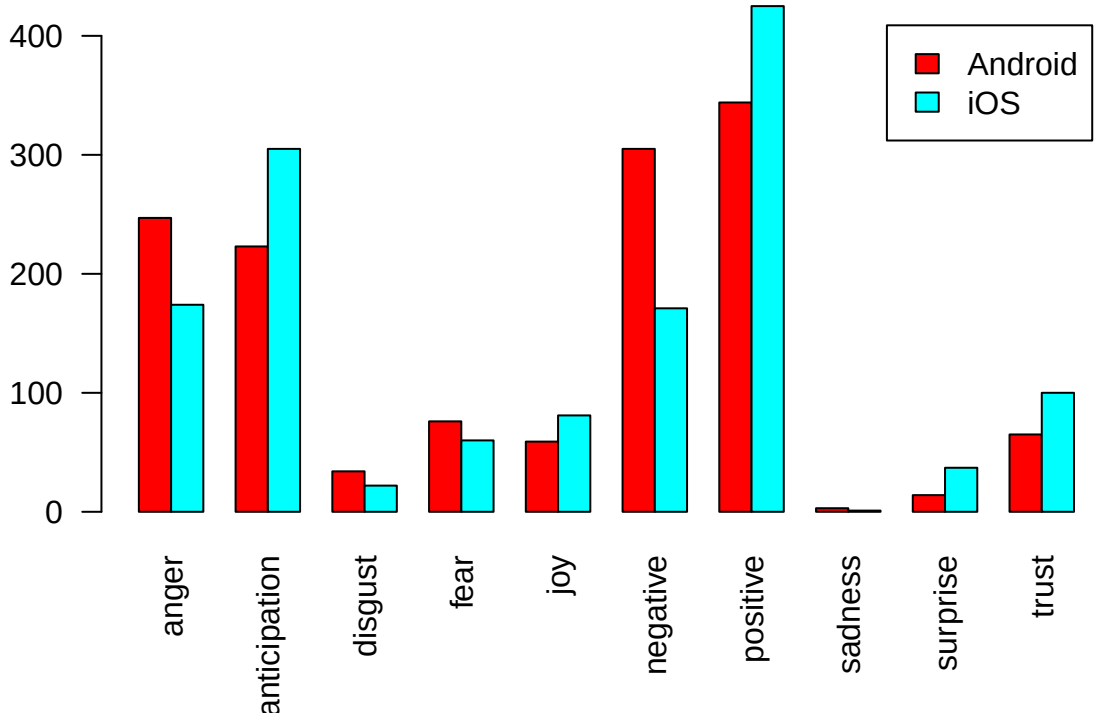
In this question you will visualise properties of Donald Trump's tweets from January 2016 to February 2017 (when he still used an Android phone), which can help predict whether a tweet was written by him (see for example `@ReallyIsTrump`).

The dataset `trump.txt` contains the following columns:

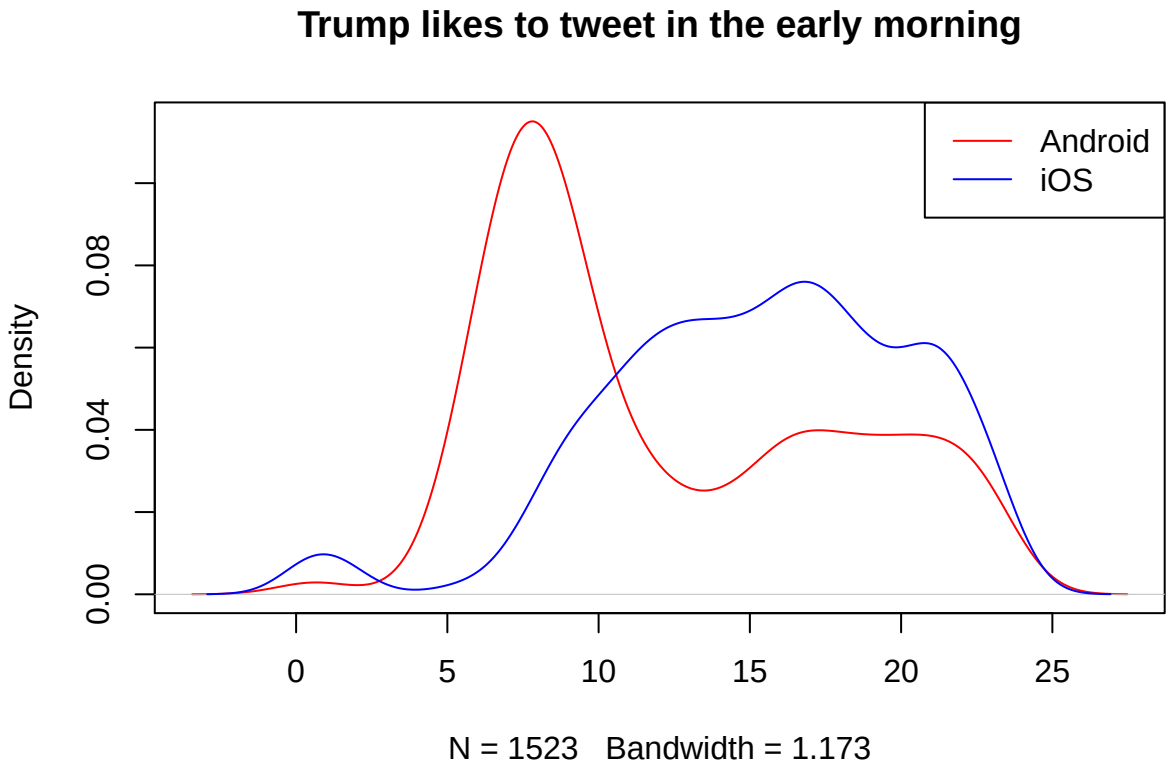
`source`	Device used: `"Android"` (Trump) or `"iOS"` (team)
`text`	Text of the tweet
`nwords`	Number of words in the tweet
`contains_url`	Whether the tweet contains a URL
`sentiment`	Sentiment obtained from a simple sentiment analysis
`dow`	Day of the week (`"Monday"` to `"Sunday"`)
`day`	Day (in days since 1 January 2016)
`hour`	Decimal hour in the day

1. **[1 mark]** Give the code to load the data into R correctly.
2. **[2 marks]** Add the column `nwords_new` to the dataset from part 1, which categorizes the length of tweets using the following rule: `short` if `nwords`<15; `medium` if $15 \leq$`nwords`≤ 25; `long` if `nwords`>25.
3. **[1 mark]** Create a bar plot illustrating the number of tweets from each source (iOS and Android). *Your plot should look similar to the plot below.*

4. [**3 marks**] Adapt your command from part 3 so that it shows the number of tweets from each source by sentiment. Different sources should be plotted using a different color. *Your plot should look similar to the plot below.*



5. [**2 marks**] For both sources, plot the density of at what time the tweets are sent using a different color. Add the title `Trump likes to tweet in the early morning`. *Your plot should look similar to the plot below.*



6. [**1 mark**] Create a boxplot of the number of words for each source. *Your plot should look similar to the plot below.*