R Programming: Worksheet 2

1. **Some simple functions**

   (a) Write a function which takes a numeric vector `x`, and returns a named list containing the mean, median, variance, and range of the values in `x`. Try your function out on some random uniform numbers. *If you're not sure what the name of a function is, try using fuzzy search: e.g.* `??variance`.

   (b) Write a function with arguments `x` and `n`, which evaulates $\sum_{i=0}^{n} \frac{e^{-x}x^i}{i!}$ (you can use `factorial()` for this).

   (c) Write a function which goes through every entry in a list, checks whether it is a character vector (`is.character()`), and if so prints it (`print()` or `cat()`).

   (d) Write a function with an argument k which simulates a symmetric random walk (see Sheet 1, Question 3), but that stops when the walk reaches k (or $-k$), and returns the walk up to that point

2. **Lynxes and hares**

   Here we use `hares.txt`, a file with the annual number of lynx and hare pelts recovered by the Hudson's Bay Company in Canada over many years. It is often used to study differential equations.

   Read the data into R, and produce a single plot of high quality, using the data, with the following requirements

   - One line per species, which should be different colours
   - The colours are colour blind friendly (suggestion try `c("#E69F00", "#56B4E9")`, or just blue and orange)
   - the y-axis starts at 0
   - Both axes are labelled
   - The plot has a title
   - There is a legend
   - The lines have a width greater than 1

3. **Beavers**

   Load the data `beav1` from the `MASS` package.

```
> library(MASS)
> data(beav1)
```

   Have a look at the field `temp` which is short for temperature.

   (a) Write a function to calculate the moving averages of length 3 of a vector $(x_1, \ldots, x_n)^T$. That is, it should return a vector $(z_1, \ldots, z_{n-2})^T$, where $z_i = \frac{1}{3}(x_i + x_{i+1} + x_{i+2}), i = 1, ..., n-2$. Call this function `ma3()`. Use it to calculate the moving average of length 3 for the beaver `temp` column.

   (b) Make a simple line plot of the original temperature data using `plot()` with the option `type = "l"`

(c) On the same graph, add the moving average of length 3, using `ma3` and the function `points`.

(d) Write a function `ma` which takes two arguments, `x` and `k`, that calculates the moving average of `x` for the value of `k`

(e) Write a function `plotMA` which takes three arguments, `x`, `k`, and `ylab`. In this function, first plot `x`, then calculate the moving average of `x` for each entry in `k` using the `ma` function, and add them to the same plot. Make each line a different colour, add a legend, set the the y-axis label to `ylab`. Once done, apply this to `x = beav1[, "temp"]` with `k = c(5, 10)`. *Optional, use a colour blind suitable palette*

4. **Monte Carlo and Importance Sampling**

This example will be based on Monte Carlo and Importance Sampling, itself a type of Monte Carlo. Briefly, when applied to probabilities, Monte Carlo is a method for estimating a probability using repeatedly sampled random variables from the underlying distribution. Knowing how and why both topics work is beyond the scope of this course, but it is a useful example for thinking about functions and plotting.

Here we want to estimate $\theta = P(X \in [3, 5])$ for X from a Cauchy distribution with probability density function $f_X(x) = \frac{1}{\pi(1+x^2)}$. We can calculate this analytically to yield $\frac{1}{\pi}(\tan^{-1}(5) - \tan^{-1}(3))$.

(a) Make a function called `phi` that takes a single argument `x` a vector of realized draws of X, and returns a logical vector of whether or not $x \in [3, 5]$. Try it on some Cauchy distributed numbers using `rcauchy`

(b) Here we want to make a cumulative average Monte Carlo estimate of θ using draws from a Cauchy distribution. Make a function with argument `n` that first makes a vector `x` of length n with draws from a Cauchy distribution. Next, calculate and return a vector of length n whose j^{th} entry is the running average Monte Carlo estimate for the first j entries, *i.e* the j^{th} returned value is $\frac{1}{j}\sum_{i=1}^{j}\phi(x_i)$.

(c) Here we want to make a cumulative average Importance Sampling estimate of θ using draws from a Uniform distribution. Make a function with argument `n` that first makes a vector `y` of length n with draws from $U(3, 5)$ (see `runif`). Next, make a vector of weights, here defined with entries $w_i = 2f_X(y_i)$ (here $f_X(x)$ is `dcauchy`). Finally, return a vector of length n with the running average Importance Sampling Monte Carlo estimate, defined as $\frac{1}{n}\sum_{i=1}^{n}\phi(y_i)w_i$

(d) For $n = 100$, make a simple plot with both the regular Monte Carlo and the Importance Sampler Monte Carlo running averages, and compare that to the truth value defined above. Which one looks better?

(e) Take your plotting code, and wrap it in a function, that takes as input the Monte Carlo and Importance Sampler estimates, and plots them. Clean up your plotting code so the plot is of high quality, using a colour blind palette, appropriate label naming, a legend, etc. Try to use sensible function and variable names.

(f) Make a function that performs both the simulation and plotting code, taking a single argument n. Try out your function with a few values of n. Try using `par(mfrow)` to show a few values of n on the same plot.

(g) *Optional* Make the interval endpoints arguments with default values of 3 and 5, that get propagated through the functions as appropriate. *Note this means you need to*

change a lot of things, including the truth value. You should know you've changed everything when you don't get errors, and the new Monte Carlo and Importance Sampling estimates converge with the new truth value