STATS 782, 2017 **Assignment 1** **Due: 1pm Tue 15 Aug**

Note:

- Hand in your assignment in class (before the Tuesday lecture starts!).
- Your answers should include all the R code, and outputs of running the code, including error/warning messages (if any).
- Wherever necessary, include some intermediate results (e.g., using the `print` function) to show how your code works step by step.
- Comment your code wherever appropriate.

1. **[10 marks]** Use `:`, `seq()`, `rep()` and possibly other commonly-used operators/functions, but definitely not `c()` or any explicit loop, to create the following sequences:

   (a) `1 4 7 10 13 16 19 22 25 28`

   (b) `"xa" "xb" "xc" "ya" "yb" "zc"`

   (c) `TRUE FALSE FALSE TRUE FALSE FALSE`

   (d) `1 101 10101 1010101 101010101`

   (e) `1 2 3 4 2 3 4 5 3 4 5 6 4 5 6 7`

2. **[10 marks]** The file `eurocities.csv` on Canvas contains the road distances (in km) between each pair of 21 cities in Europe. Download the data file onto your computer and find a way to read in the data appropriately into R (check `?read.csv`). Use R expressions/functions to find:

   (a) The mean road distance between a pair of cities.

   (b) The city pair that are farthest apart.

   (c) The number of cities from which Paris is within 300km by driving.

   (d) The three cities that are closest to Copenhagen.

   (e) The city that has the shortest total road distance to Rome and Stockholm.

3. **[10 marks]** The Hermite polynomials are given by

$$H_n(x) = n! \sum_{m=0}^{\lfloor n/2 \rfloor} \frac{(-1)^m}{m!(n-2m)!} \frac{x^{n-2m}}{2^m},$$

   where $\lfloor \cdot \rfloor$ denotes the floor function. Calculate $H_n(x)$, for $n = 5$ and $x = -2, -1.8, -1.6, \dots, 2$, in the following three different ways of coding:

   (a) Using a double loop, for x and m, respectively. [3 marks]

   (b) Using a single loop, for x only. [3 marks]

   (c) Using no loop. (Hint: Use `outer` and `apply`.) [4 marks]

   [You may or may not package your code in R functions.]

4. **[10 marks]** The Hermite polynomials have the following recurrence relation:

$$H_{n+1}(x) = xH_n(x) - H_n'(x),$$

where $H_0(x) = 1$. Write an R function that uses this recurrence relation to find the coefficients of $x^0, x^1, \ldots, x^n$ for $n = 0, 1, \ldots, N$, and returns them in the form of a matrix. For example, when $N = 3$, running your function should give a similar result to the following:

```
> hermite(3)
   x^0 x^1 x^2 x^3
H0   1   0   0   0
H1   0   1   0   0
H2  -1   0   1   0
H3   0  -3   0   1
```

Note that you need not use the recurrence relation to update the coefficients that are 0. Use only an explicit loop for $n = 0, 1, \ldots, N$. Demonstrate your code for $N = 0, 1, 3, 10$, respectively.

5. **[10 marks]** A root of a function $f(x)$ is a solution of the equation $f(x) = 0$. For example, function $f(x) = \exp(x) - 2$ has a single root $x = \log 2$, and function $\sin(x)$ has an infinite number of roots at $x = k\pi$, k being an integer. When the equation $f(x) = 0$ can not be solved analytically, one has to use an iterative method to find the roots of a function. A simple root-finding method, known as the *bisection method*, works as follows. Initially, an interval $[a, b]$ is provided, which brackets a root in the sense that $f(a)f(b) < 0$. Then f is evaluated at the midpoint $c = (a + b)/2$. If $f(c)$ has the same sign as $f(a)$, a is replaced with c, i.e., the bracketing interval $[a, b]$ is replaced with $[c, b]$; otherwise, $[a, b]$ is replaced with $[a, c]$. This process of halving the bracketing interval continues until the interval is sufficiently narrow, indicating that a root has been accurately found and the method can stop.

   (a) [5 marks] Implement the bisection method in an R function and use it to find the root of

   $$f(x) = \sin(x) + \exp(-x/10)$$

   between 0 and 5. Make sure that your R function works for a general function f and a bracketing interval $[a, b]$.

   (b) [5 marks] Modify your function implemented for part (a) so that it now finds all the roots of f on a given interval $[a, b]$. To obtain the initial bracketing intervals for the roots, first evaluate f on a fine grid on $[a, b]$, say, 1000 evenly spaced x-values, and then locate the pairs of consecutive points that have opposite signs in f. Use vectorised computation as efficiently as you can. Demonstrate that your modified function works for finding all the roots of the function given in part (a) on $[0, 50]$.