# 732A90: Computational Statistics

Computer lab6 - Group11

*Sofie Jörgensen, Oriol Garrobé Guilera, David Hrabovszki*

*29 February 2020*

## Question 1: Genetic algorithm

In this exercise we are going to perform one-dimensional maximization by using a genetic algorithm.

### 1.

Firstly, we define the function `f()` as

$$f(x) := \frac{x^2}{e^x} - 2\,\exp(-(9\sin x)/(x^2 + x + 1)).$$

### 2.

Secondly, we define the function `crossover()`, that takes two scalars $x$ and $y$ as inputs, and returns a child as $\frac{x+y}{2}$.

### 3.

Thirdly, we define the function `mutate()`, that performs the integer division $x^2 \bmod 30$, for a scalar input $x$.

### 4.

Further, we will create a function called `genetic()`, with the parameters `maxiter` and `mutprob`. The settings of this `genetic()` function, as well as its output results, are presented in (a)-(e). The code can be found in the Appendix.

(a). The function $f()$ is plotted in the range from 0 to 30 in Figure X, and we can observe that there is a maximum value at ...

(b). An initial population for the genetic algorithm is defined as $X = (0, 5, 10, 15, ..., 30)$.

(c). A vector called `Values` are computed, containing the function values for each population point.

(d). The `genetic()` function performs `maxiter` iterations. For each iteration...

(e).

### 5.

By using the defined functions from previous tasks (1.1-1.4), we are going to observe the initial population and final population. This is done by running the code with different combinations of `maxiter`$= 10, 100$ and `mutprob`$= 0.1, 0.5, 0.9$.

## Question 2: EM algorithm

**1.**

**2.**

**3.**

**4.**

---

## Appendix

```r
knitr::opts_chunk$set(echo = FALSE)
# R version
RNGversion('3.5.1')
#1.1
f <- function(x){
  return(x^2/exp(x) - 2*exp(-1*(9*sin(x)) / (x^2 + x + 1)))
}

#1.2
crossover <- function(x,y){
  return((x+y) / 2)
}

#1.3
mutate <- function(x){
  return(x^2 %% 30)
}
#4
genetic <- function(maxiter, mutprob){
  #a
  plot(x = seq(0,30), y = f(seq(0,30)), type = "l", xlab = "x", ylab = "f(x)")
  abline(v=1.05, col = "red")

  #b
  X = seq(0,30,5)

  #c
  Values = f(X)

  #d
  #set seed
  set.seed(1234567890)
  for (i in 1:maxiter) {
    #i
    parents = match(sample(X, 2),X)

    #ii
    victim = order(Values)[1]

    #iii
    kid = round(crossover(parents[1],parents[2]))
    p = runif(1)
    if (p < mutprob) {
      kid = mutate(kid)
    }

    #iv
    X[victim] = kid
    Values = f(X)

    #v
    max = max(Values)
  }

  #e
  print(X)
  print(Values)
  plot(x = seq(0,30), y = f(seq(0,30)), type = "l", xlab = "x", ylab = "f(x)")
  points(x = X, y = Values, col = "red", pch = 19)
}

# R version
RNGversion('3.5.1')
# Packages
```