# Computer Lab 6

Computational Statistics

Group 4

2021-12-08

## Question 1: Genetic algorithm

### 1.1

```
f_x <- function(x) {
  return((x^2 / exp(x)) - 2 * exp(-(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)
}
```

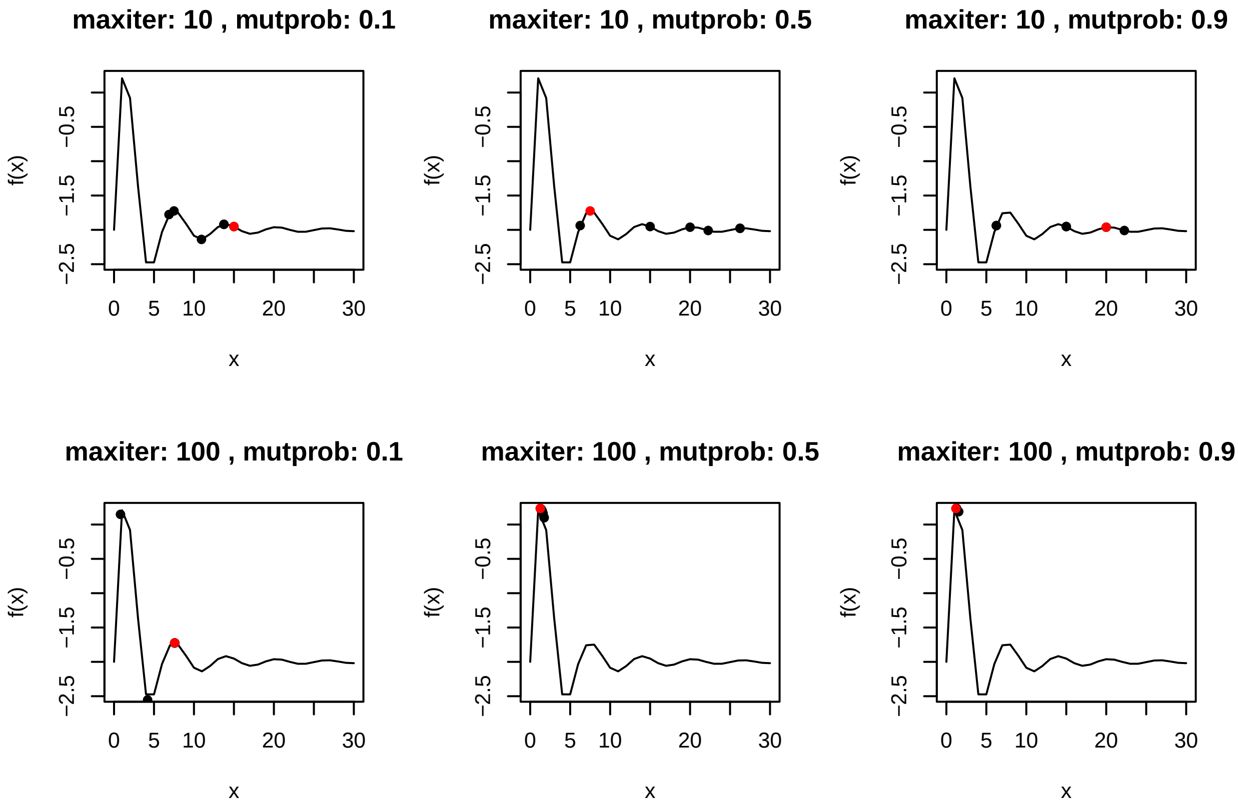
### 1.4

```
my_func <- function(maxiter, mutprob) {
  # 1.4 a)
  plot(seq(0, 30), f_x(seq(0, 30)),
       ylab = "f(x)", xlab = "x", type = "l",
       main = paste("maxiter:", maxiter, ", mutprob:", mutprob))
  # 1.4 b)
  pop <- seq(0, 30, by = 5)
  # 1.4 c)
  Values <- f_x(pop)
```

```r
  max_val <- Values[which.max(Values)]
  # 1.4 d)
  for(i in 1:maxiter) {
    # 1.4 d) i.
    parents <- sample(1:length(pop), size = 2)
    # 1.4 d) ii.
    victim <- order(Values)[1]
    # 1.4 d) iii.
    kid <- crossover(x = pop[parents[1]],
                     y = pop[parents[2]])
    if(rbinom(n = 1, size = 1, prob = mutprob) == 1) {
      kid <- mutate(kid)
    }
    # 1.4 d) iv.
    pop[victim] <- kid
    Values <- f_x(pop)
    # 1.4 d) v.
    if(Values[which.max(Values)] > max_val) {
      max_val <- Values[which.max(Values)]
    }
  }
  # 1.4 e)
  points(pop[-length(pop)], Values[-length(Values)], col = "black", pch = 16)
  points(pop[length(pop)], Values[length(Values)], col = "red", pch = 16)
}
```

### 1.5

```r
par(mfrow=c(2,3))
my_func(10,0.1)
my_func(10,0.5)
my_func(10,0.9)
my_func(100,0.1)
my_func(100,0.5)
my_func(100,0.9)
```




The algorithm manages to find the maximum of the function *f(x)* when the maximum number of iterations (*maxiter*) is set to 100, and the probability of a kid being mutated (*mutprob*) is set to 0.5 or higher.

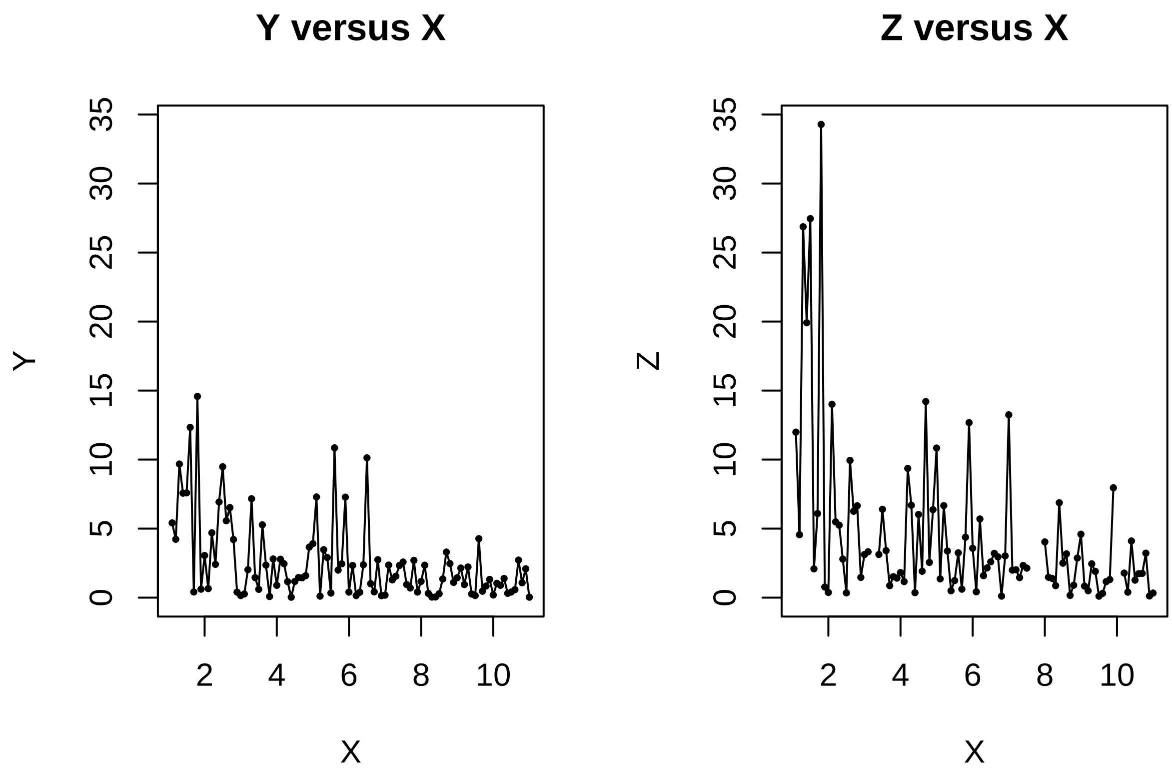
## Question 2: EM algorithm

### 2.1

```
physical1 <- read.csv("physical1.csv")

par(mfrow=c(1,2))
plot(x = physical1$X, y = physical1$Y,
     ylim = c(0,max(physical1[,2:3], na.rm = TRUE)),
     xlab = "X", ylab = "Y", main = "Y versus X",
     pch = 16, cex = 0.5)
lines(x = physical1$X, y = physical1$Y)

plot(x = physical1$X, y = physical1$Z,
     ylim = c(0,max(physical1[,2:3], na.rm = TRUE)),
     xlab = "X", ylab = "Z", main = "Z versus X",
     pch = 16, cex = 0.5)
lines(x = physical1$X, y = physical1$Z)
```




The values of $Y$ and $Z$ seems to decrease as the value of $X$ increases. Likewise the variance of $Y$ and $Z$ appear to decrease as the value of $X$ increases.

### 2.2

```
floglik <- function(data, lambda_k) {
  X <- data[,1]
  Y <- data[,2]
  Z <- data[,3]
  n <- nrow(data)
  u <- sum(is.na(data[,3]))

  return((1/(2*n)) * ( sum(X*Y) + 0.5 * sum(X*Z, na.rm = TRUE) + u*lambda_k))
}

em_algorithm <- function(data, eps, kmax) {
  X <- data[,1]
  k <- 1

  lambda_prev <- 100
  lambda_curr <- floglik(data, lambda_prev)
  print(c(k, lambda_curr))

  while((abs(lambda_prev - lambda_curr) > eps) && (k < (kmax+1))){
```

```r
    lambda_prev <- lambda_curr
    lambda_curr <- floglik(data, lambda_prev)
    k <- k + 1
    print(c(k, lambda_curr))
  }
}

em_algorithm(physical1, 0.001, 1000)
```

```
## [1]  1.00000 14.26782
## [1]  2.00000 10.83853
## [1]  3.00000 10.70136
## [1]  4.00000 10.69587
## [1]  5.00000 10.69566
```

The optimal value of $\lambda$ is 10.69566, and 5 iteration was required to compute it.