Non Linear Programming Assignment

Cesareo Giacomo 805716

Contents

Problem 1

Using the bisection method calculate at least one zero for

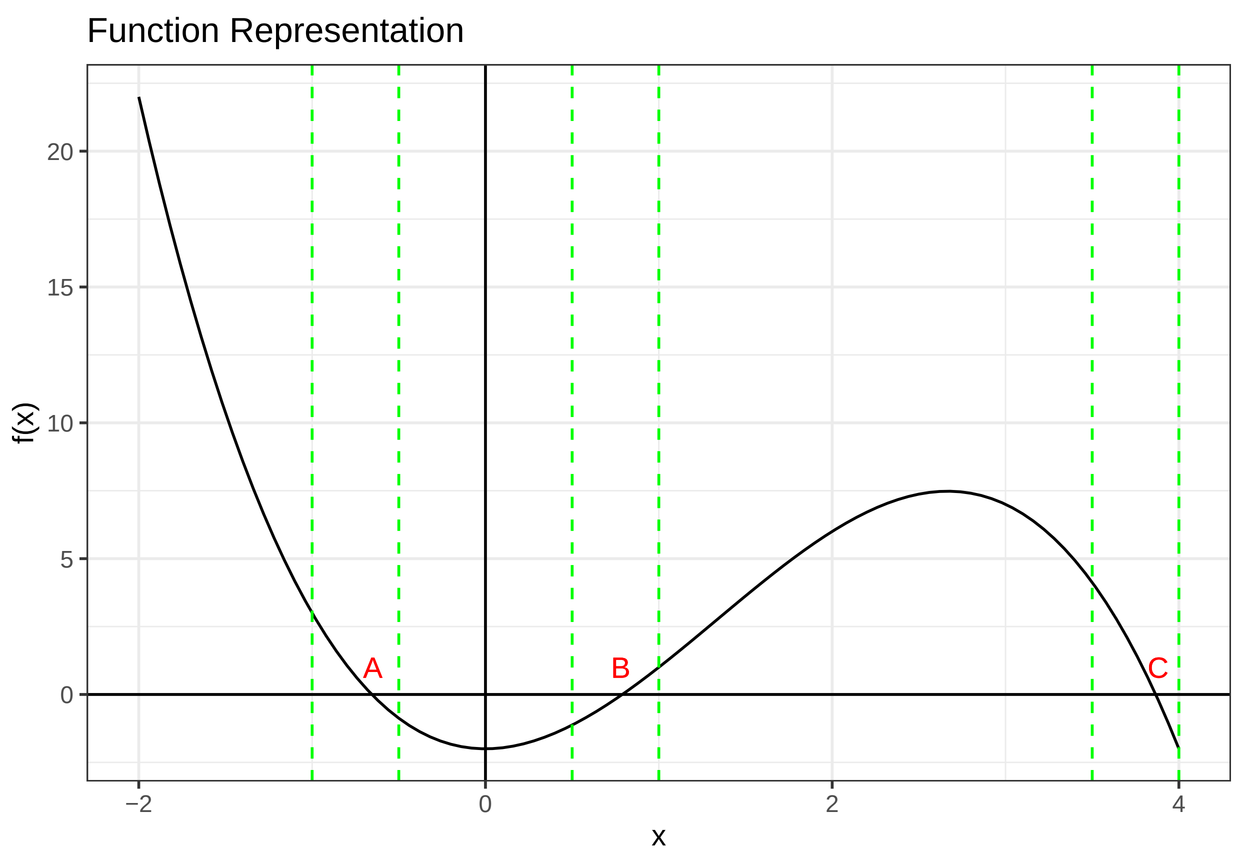
$$f(x) = -x^3 + 4x^2 - 2$$

starting for a suitable initial guess. You may want to reuse the code provided in the exercise session.

Solution

The goal is to find the points wich nullify the given function. The first step is to initialize the given function and then plot it as to have a general overview.

```r
library(ggplot2)
library(pander)

func <- function(x){-x^3 + 4*x^2 - 2}

ggplot(data.frame(x = c(-2, 4)), aes(x=x)) + stat_function(fun=func) + ylab('f(x)') +
  geom_vline(xintercept = 0, linetype = 1, color = 'black', size = 0.5) +
  geom_hline(yintercept = 0, linetype = 1, color = "black", size = 0.5) +
  geom_vline(xintercept = -1, linetype = 2, color = 'green', size = 0.5) +
  geom_vline(xintercept = -0.5, linetype = 2, color = 'green', size = 0.5) +
  geom_vline(xintercept = 0.5, linetype = 2, color = 'green', size = 0.5) +
  geom_vline(xintercept = 1, linetype = 2, color = 'green', size = 0.5) +
  geom_vline(xintercept = 3.5, linetype = 2, color = 'green', size = 0.5) +
  geom_vline(xintercept = 4, linetype = 2, color = 'green', size = 0.5) +
  geom_text(x=-0.65, y=1, label="A",col='red') +
  geom_text(x=0.78, y=1, label="B",col='red') +
  geom_text(x=3.88, y=1, label="C",col='red') +
  ggtitle('Function Representation') + theme_bw()
```



From the plot above it is clear that there are three points that nullify the function. Let's try to get a solution by reusing the code provided in the exercise session: the bisection function will check if in a given interval $[a, b]$ the function change is sign meaning that a zero has been found.

```r
bisection <- function(f, a, b, n = 1000, tol = 1e-7) {

  if (sign(f(a) == sign(f(b)))) {
    stop('signs of f(a) and f(b) must differ')
  }

  for (i in 1:n) {
    c <- (a + b) / 2

    if ((f(c) == 0) || ((b - a) / 2) < tol) {
      return(c)
    }

    ifelse(sign(f(c)) == sign(f(a)),
           a <- c,
           b <- c)
  }
  print('Too many iterations')
}
```

By using this *bisection* function annother one more complex can be build. In this particolar case is needed a function that can take as input 3 interval: $[-1, -0.5]$, $[0.5, 1]$, $[3.5, 4]$ since the zeros are located into those intervals.

```
Bisection <- function(f, vec){
  half <- length(vec)/2
  result <- c()
  for (k in 1:half){
  result[k] <- bisection(f, vec[(k*2)-1], vec[k*2])
  }
  return(pander(data.frame(Points=c('A','B','C'), Zeros=result)))
}

intervals <- c(-1,-0.5,0.5,1,3.5,4)
Bisection(func, intervals)
```

Points	Zeros
A	-0.6554
B	0.7892
C	3.866

In the table above are presented the x_i values for each point that nullify the given function.
There is also the package NLRoot that can easily compute the zeros of a function in just one line of code, let's compare the results just obtained. Now is considered the whole interval ranging between $[-0.8, 4]$.

```
library(NLRoot)

BFfzero(func, -0.8, 4.1)
```

```
## [1] 1
## [1] -0.6554397
## [1] -1.763432e-05
## [1] 2
## [1] 0.7892439
## [1] -1.192802e-06
## [1] 3
## [1] 3.866197
## [1] 2.42719e-05
## [1] "finding root is successful"
```

As expected the results are the same as those computed by the *Bisection* function.

Problem 2

Consider the following minimization problem:

$$MIN : f(x_1, x_2) = 2(x_1)^2 + x_1x_2 + 2(x_2 - 3)^2$$

1) Apply an iteration of the gradient method by performing the line search in an exact way, starting from the point $A(-1, 4)^T$. Report all the steps of the method, not just the result.
2) Apply an iteration of Newton's method from point A. Verify that the point found is the minimum of function f. Report all the steps of the method, not just the result.
3) How many iterations of Newton's method are required to optimize a quadratic function?

Solution

For convenience the function can be rewritten as:

$$MIN : f(x_1, x_2) = 2(x_1)^2 + x_1x_2 + 2(x_2)^2 - 12x_2 + 18$$

The first thing to do is to compute the gradient $(\nabla f(x_k))$ as considering the partial derivatives with respect of each component.

$$\nabla f(x_k) = [4x_1 + x_2;\ x_1 + 4x_2 - 12]$$

The first component is the function's derivative with respect of x_1 while the second component is the partial derivative with respect of x_2

The algorithm start by computing the gradient in the starting point x_0, in this case this is the given point $A(-1, 4)^T$. The gradient in x_0 is obtained by subsituting his own coordinates coordinates into the gradient function obtaining:

$$\nabla f(x_0) = [0;\ 3]$$

At this point the algorithm's direction have to be chosen: in order to minimize the function the direction must be opposite to the grandient vector.
In this case $d_0 = -\nabla f(x_0)$. Whit those information x_1 can easely be computed as follow:

$$x_1 = [-1,\ 4] - a_0[0,\ 3] = [-1;\ 4 - 3a_0]$$

The next step is to determine the step size a_0 We substitute x_1 in the initial function and then compute the root for which the first derivative is equal to zero.

$$f(-1;\ 4 - 3a_0) = 2 - 4 + 3a_0 + 2(4 - 3a_0)^2 - 12(4 - 3a_0) + 18$$

$$f(-1;\ 4 - 3a_0) = 18a_0^2 - 9a_0 = 0$$

At this point in order to optimize this function the derivative is computed.

$$f'(-1;\ 4 - 3a_0) = 36a_0 + 9$$

$$a_0 = \frac{1}{4}$$

The point x_1, representing the new point where the algorithm move, can easily be computed by substitutng a_0.

$$x_1 = [-1;\ \frac{13}{4}]$$

Now the last step is to check if this new point make the gradient equal to zero. The gradient in the next point can be obtained by substituting the values just found in the gradient vector function. It being equal to:

$$\nabla f(x_1) = [-\frac{3}{4};\ 0]$$

In the last step to evaluate if the stop criteria is satisfied It is necessary to set the stoping criteria ε, in this case $\varepsilon = 0.1$.

$$||\nabla f(x_{k+1})|| < \varepsilon$$

$$||\nabla f(x_{k+1})|| = \sqrt{(-\frac{3}{4})^2 + 0^2} = 0.75 < 0.1$$

Since the stopping criteria is not respected annother iteration, starting from x_1, is needed.

In this second iteration the algorithm now starts from x_1 and procedes in the same way as seen above. This time some calculations are omitted.

$$x_2 = [-1,\ \frac{13}{4}] - a_1[\frac{3}{4},\ 0] = [-1 + \frac{3}{4}a_1;\ \frac{13}{4}]$$

Let's find the value of a_1:

$$f(-1 + \frac{3}{4}a_1;\ \frac{13}{4}) = \frac{9}{8}a_1^2 - \frac{9}{16}a_1 + \frac{9}{8}$$

$$f'(-1 + \frac{3}{4}a_1;\ \frac{13}{4}) = \frac{9}{4}a_1 - \frac{9}{16}$$
$$a_1 = \frac{1}{4}$$

And now by substituting a_1 the value of x_2 can easily be computed:

$$x_2 = [-1 + \frac{3}{4}a_1;\ \frac{13}{4}] = [-\frac{13}{16};\ \frac{13}{4}]$$

The gradient in x_2 can be obtained by substituting the values just found into the gradient vector function. It being equal to:

$$\nabla f(x_2) = [0;\ \frac{3}{16}]$$

In this second iteration too the stop criteria is not respected, so annother iteration of the algorithm is needed.

$$||\nabla f(x_{k+1})|| = \sqrt{(\frac{3}{16})^2 + 0^2} = 0.1875 < 0.1$$

Let's procede straight forward with the third iteration: Again the algorithm start from the previous point, x_2 in this case.

$$x_3 = [-\frac{13}{16},\ \frac{13}{4}] - a_2[0,\ \frac{13}{16}] = [-\frac{13}{16};\ \frac{13}{4} - \frac{13}{16}a_2]$$

Solve again the equation with respect of a_2:

$$f(-\frac{13}{16};\ \frac{13}{4} - \frac{13}{16}a_2) = \frac{169}{128}a_2^2 - \frac{39}{156}a_2 + \frac{153}{128}$$

$$f(-\frac{13}{16};\ \frac{13}{4} - \frac{13}{16}a_2) = \frac{169}{64}a_2^2 - \frac{39}{256}$$

$$a_2 = \frac{3}{52}$$

At last x_3 and the gradient at that point are computed as follow:

$$x_3 = [-\frac{13}{16};\ \frac{13}{4} - \frac{13}{16}a_2] = [-\frac{13}{16};\ \frac{205}{64}] = [-0.8125,\ 3.20]$$

$$\nabla f(x_3) = [-\frac{3}{64};\ 0]$$

$$||\nabla f(x_{k+1})|| = \sqrt{(-\frac{3}{64})^2 + 0^2} = 0.046875 < 0.1$$

At this stage a solution has been found and the function has been minimized in $[-0.8125,\ 3.20]$.

Let's try now with the Newton multivariate method: it uses a quadratic model to approximate the function with a quadratic function using the Taylor series expansion. The gradient vector and the value it takes in $(-1;\ 4)^T$ has already been computed:

$$\nabla f(x_0) = [4x_1 + x_2;\ x_1 + 4x_2 - 12] = [0,\ 3]$$

The Hessian matrix and his inverse are presented below, it is important to notice that if $H(x_0)$ is definite positive then only one iteration of the algorithm is needed to find the optimal solution.

$$H(x_0) = \begin{bmatrix} 4 & 1 \\ 1 & 4 \end{bmatrix} \qquad H^{-1}(x_0) = \begin{bmatrix} \frac{4}{15} & -\frac{1}{15} \\ -\frac{1}{15} & \frac{4}{15} \end{bmatrix}$$

The point x_1 can be computed in the following way:

$$x_1 = x_0 - H(x_0)^{-1}\nabla f(x_0)$$

$$x_1 = \begin{bmatrix} -1 \\ 4 \end{bmatrix} - \begin{bmatrix} \frac{4}{15} & -\frac{1}{15} \\ -\frac{1}{15} & \frac{4}{15} \end{bmatrix} \begin{bmatrix} 0 \\ 3 \end{bmatrix} = \begin{bmatrix} -1 \\ 4 \end{bmatrix} - \begin{bmatrix} -\frac{1}{5} \\ \frac{4}{5} \end{bmatrix} = \begin{bmatrix} -\frac{4}{5} \\ \frac{16}{5} \end{bmatrix} = \begin{bmatrix} -0.8 \\ 3.2 \end{bmatrix}$$

The solution has been found and it is very close to the one found in the previous page.

$$\nabla f(x_1) = [4x_1 + x_2;\ x_1 + 4x_2 - 12] = [0,\ 0]$$

The algorithm has converged in x_1 and the solutions are identical to the ones found using the gradient method. To conclude, it's worth mentioning that gradient method required 3 iterations while Newton's needed only 1 step, as expected.

Problem 3

Use the Simulated annealing algorithm to find the global minimum of the following function.

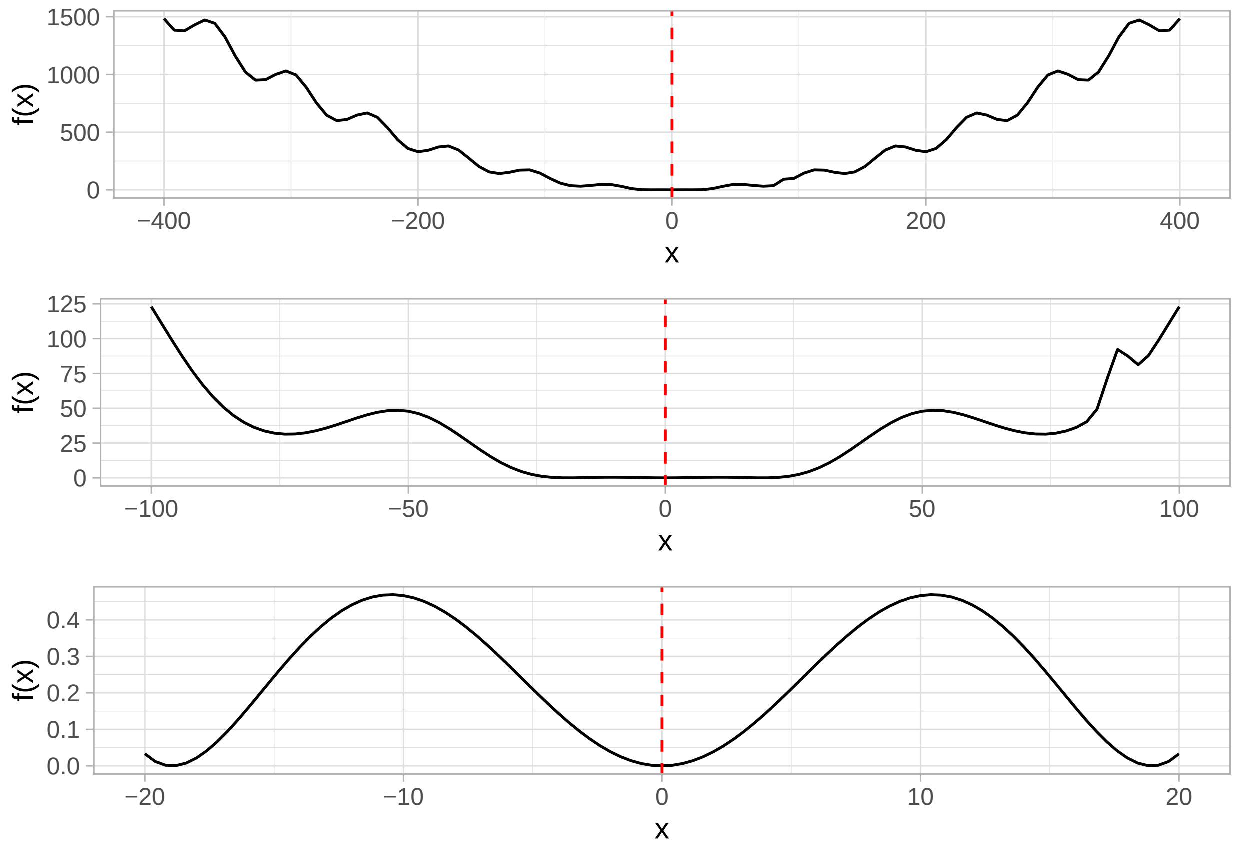
$$f(x) = 34 * e^{-\frac{1}{2}(\frac{x-88}{2})^2} + [\frac{x}{10} - 2 * \sin(\frac{x}{10})]^2$$

Notice that f(x) may have several local optimals, thus restarting and a careful selection of the algorithm parameters may be necessary.

Solution

First the function is defined and graphically represented:

```
library(ConsPlan)
library(gridExtra)

func <- function(x) { 34*exp(-1/2*((x-88)/2)^2) + (x/10 - 2*sin(x/10))^2 }

a <- ggplot(data.frame(x = c(-400, 400)), aes(x=x)) + stat_function(fun=func) +
  ylab('f(x)') + geom_vline(xintercept = 0, linetype = 2, color = "red", size = 0.5) +
  theme_light()

b <- ggplot(data.frame(x = c(-100, 100)), aes(x=x)) + stat_function(fun=func) +
  ylab('f(x)') + geom_vline(xintercept = 0, linetype = 2, color = "red", size = 0.5) +
  theme_light()

c <- ggplot(data.frame(x = c(-20, 20)), aes(x=x)) + stat_function(fun=func) +
  ylab('f(x)') + geom_vline(xintercept = 0, linetype = 2, color = "red", size = 0.5) +
  theme_light()

grid.arrange(a,b,c,nrow=3)
```



The function is presented on three different intervals to observe how it is distributed: at first glance the function may seem symmetrical but in the right part of the function a strong asymmetry is observed.
Considering the interval $[-20, 20]$ the function is simmetric with respect to $x = 0$ and can be clearly recognized three points of minimum around $[-18,\ 0,\ 18]$.

To solve this problem will be use the sann function from the ConsPlan package setting the initial temperature equal to 10.000 changing every 5 iteration and setting the maximum number of iteration to 100.000.

The starting point is chosen randomly between $[-20, 20]$, while the candidate solution is randomly sampled from a normal distribution with zero mean and fixed standard deviation.

```r
candidate <- function(zero) { rnorm(1, mean = zero, sd = (40/6)) } # 40 = 20-(-20) -> range.

results <- c()
value <- c()

for (i in 1:10) {
  SA <- sann(start.seq = runif(1, min = -20, max = 20),
                                fn = func,
                                gr = candidate,
                                maxit = 100000,
                                REPORT = NULL,
                                tmax = 5,
                                temp = 10000)
  results[i] <- SA$sequence
  value[i] <- SA$value
}
```

```r
options(scipen = 999)
pander(data.frame(x=results,y=value))
```

x	y
-0.000711	0.000000005055
18.95	0.00000008236
18.95	0.0000000005487
-18.96	0.000000003343
-0.0001146	0.0000000001313
-18.95	0.00000001783
0.003097	0.0000000959
-18.95	0.00000004541
-0.001112	0.00000001237
-18.95	0.0000000333

From this table can be seen that there are a total of 3 global optimum points located, with some margin of error, in $x = [-18.95,\ 0,\ 18.95]$ whose y value converged to zero.