Artificial Intelligence

Evolutinary Algorithms

Task Report 4

Harun GÜLEÇ

Bialystok,2017

Task 1

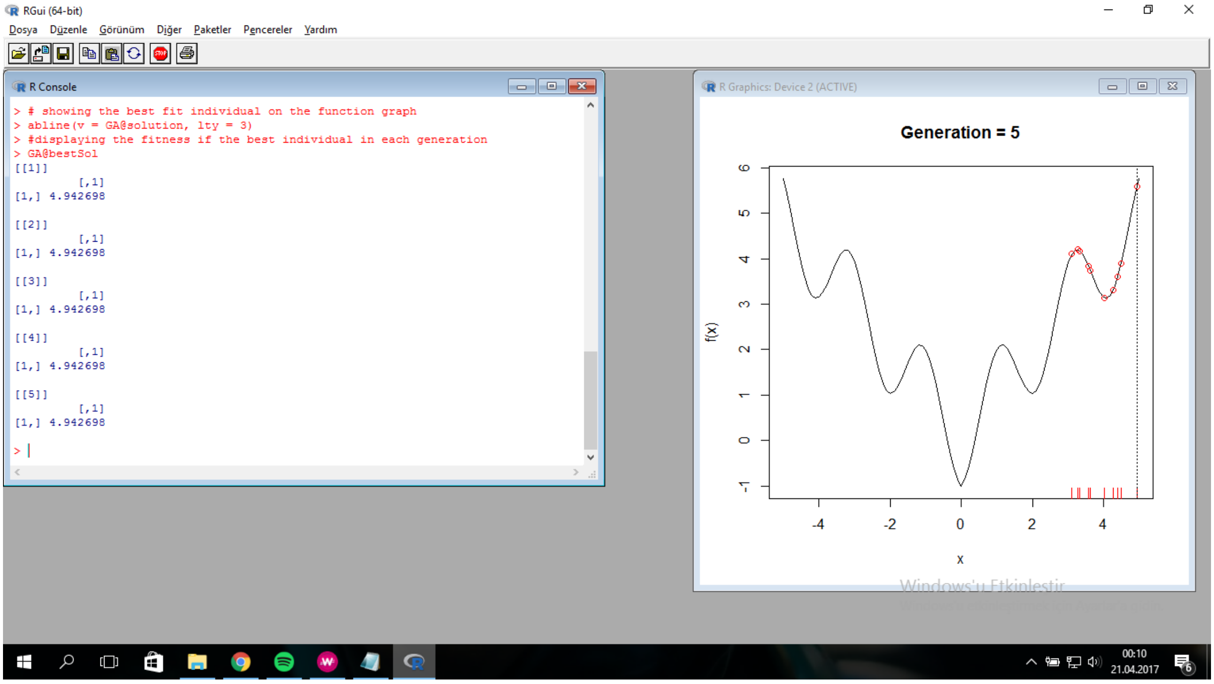
At this task our main object is understanding genetic algorithm and our exercise is using the R system and the GA package to find the maximum of the function f (x)=|x|−a(3 x) in the interval [−5,5] where a the cosine function (cos).

First of all we defined the function and declared paramaters. Our final R code is shown below.

```
library(GA)

# the library instruction loads a package and should be used only
once during one program

# running

#setting the parameters

min_x = -5

max_x = 5

pop_size = 10

pc = 0.8

pm = 0.1

maxiter = 10

sleep = 2

#sleep=a – a is delay in seconds before the next generation is to
be displayed, if a=-1 then the

#next population is shown by hitting any key (e.g. enter)

#the function to optimized

f <- function(x) abs(x)-cos(3*x)

#fitness function, the same as the optimization one if it is to be
maximized, the negation of the

#optimization one if it is to be minimized

fitness = function(x) abs(x)-cos(3*x)

#drawing the function graph

5curve(f, min_x, max_x)
```

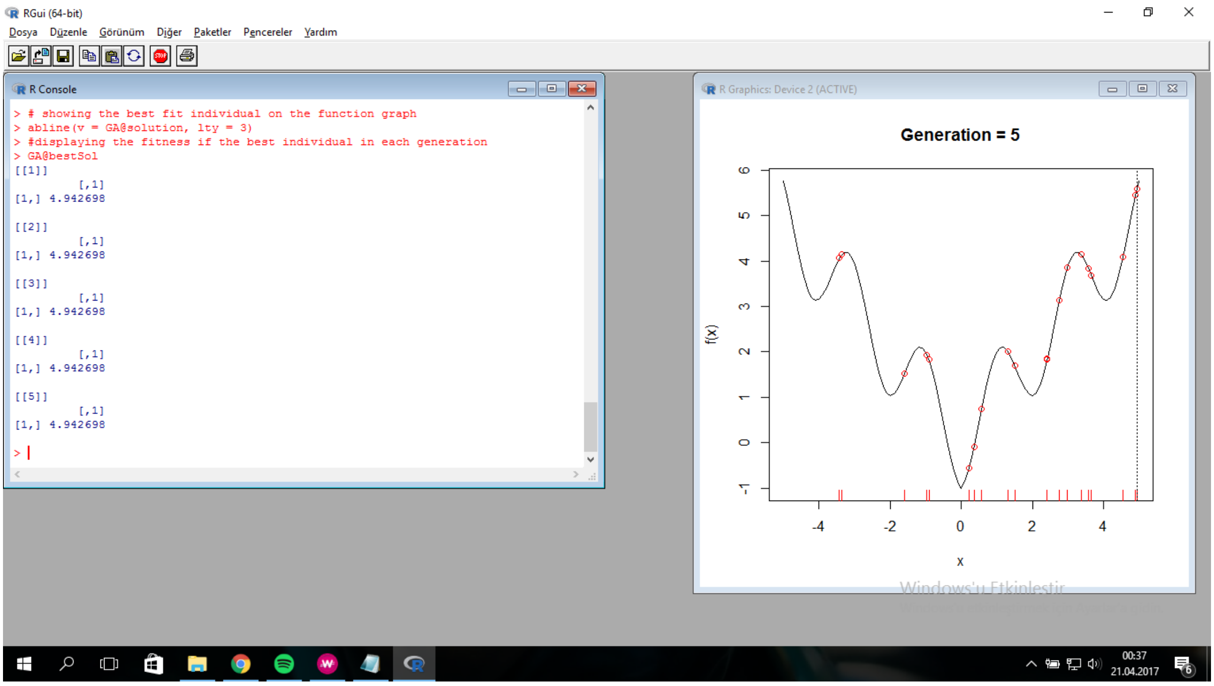
```
# monitoring function showing succeeding steps of the algorithm

monitor <- function(obj)

{

curve(f, min_x, max_x, main = paste("Generation =", obj@iter))

points(obj@population, f(obj@population), pch = 1, col = 2)

rug(obj@population, col = 2)

if (sleep<0)

readline()

else

Sys.sleep(sleep)

}

# the algorithm running

GA <- ga(type = "real-valued",fitness = fitness,min = min_x,max =
max_x,popSize = pop_size,

pcrossover = pc,pmutation = pm,maxiter = maxiter,monitor =
monitor,keepBest = TRUE ,seed =

123)

#displaying the summary

summary(GA)

# showing the best fit individual on the function graph

abline(v = GA@solution, lty = 3)

#displaying the fitness if the best individual in each generation

GA@bestSol
```

Solutions disturibitions are like that;



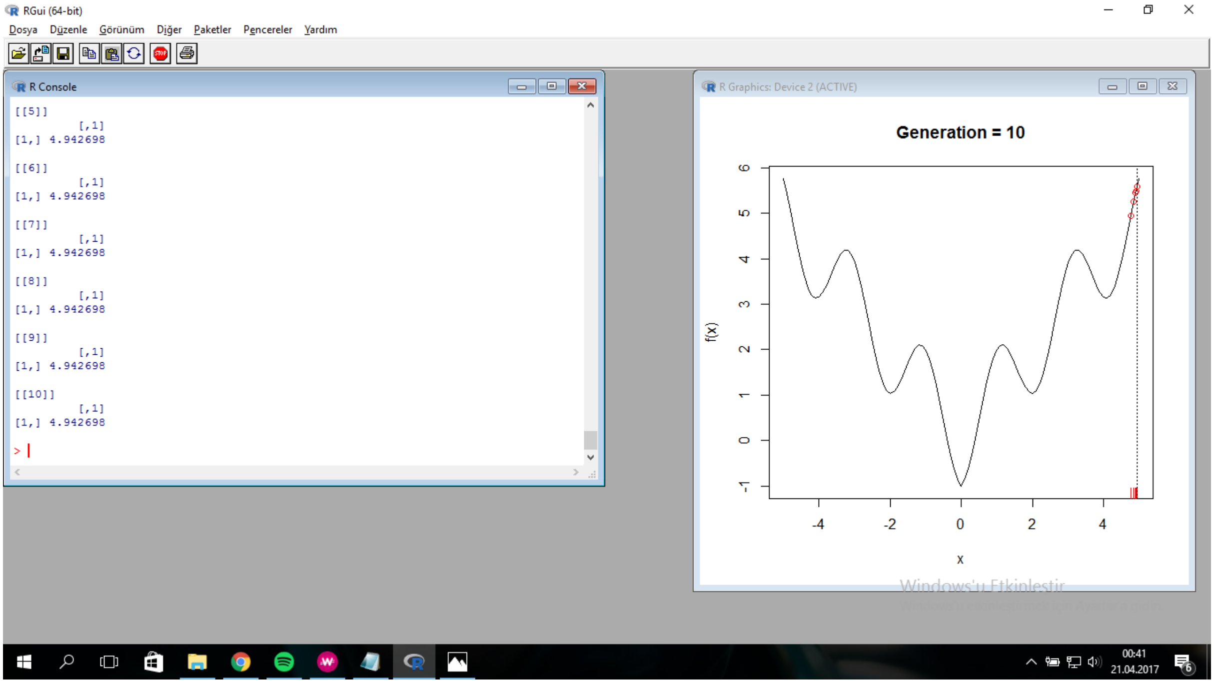
In this case generally all individuals close to maximum point.

Secondly, I changed population size and repeated computation again.



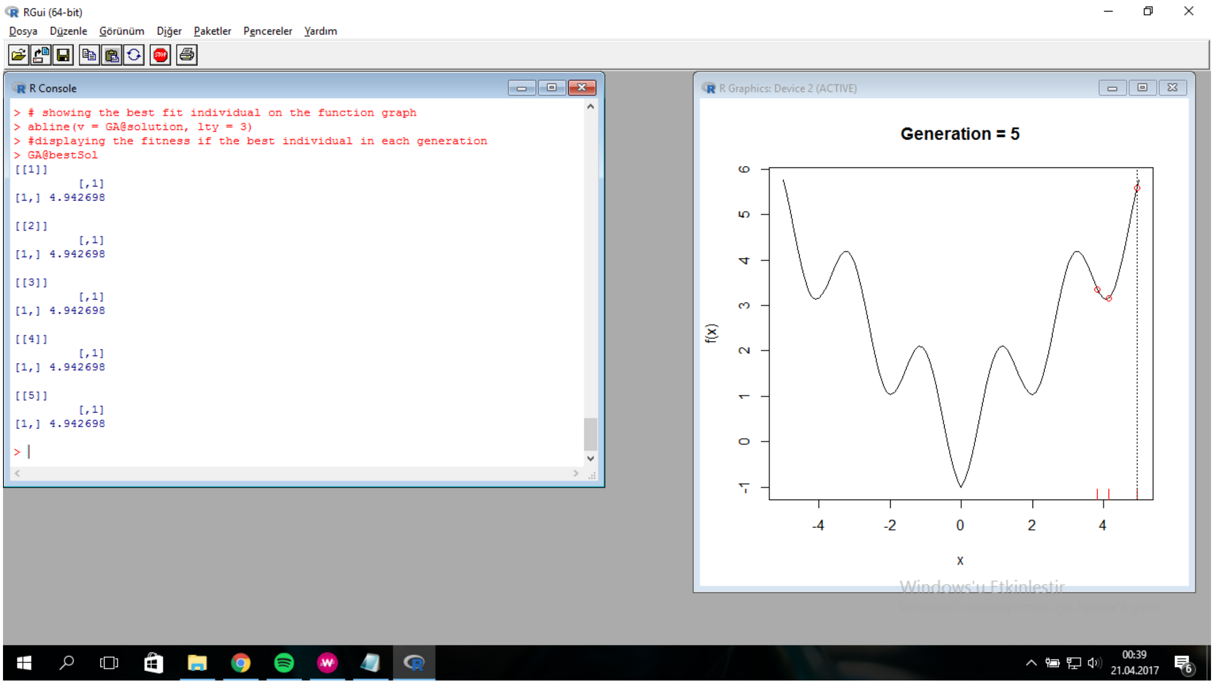
This time, individuals are distiributed on the chart.

After, I changed number of generation and repeated computation again.



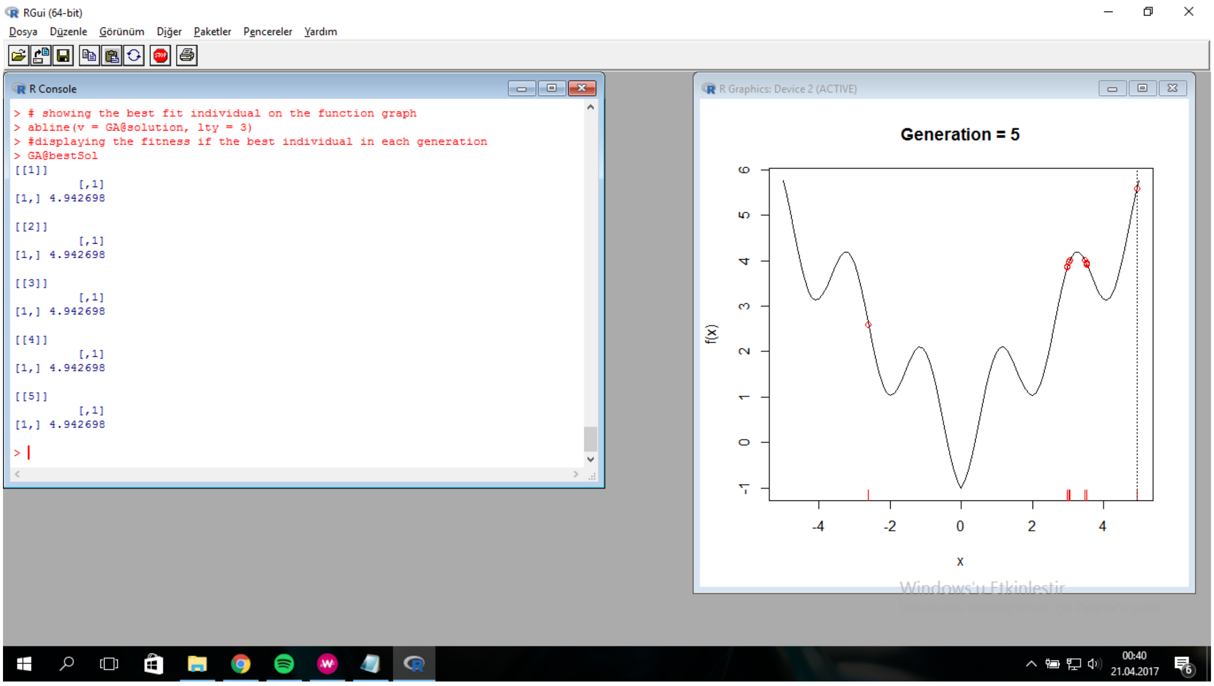
As we can see clearly, when we increase the generation, population is getting closer to maximum point.

After that I changed crossover probability and repeated computation again.



In this case, I decreased the probability of crossover and this situation caused decreasing variety of population.

Final case is changing probability of mutation and result is shown below.



When we increase the probability of mutation our population has extra ordinary individuals. For example all individuals are between 3 and 5 but one of them is about -2,5.

Task 2

For this task my proposition is about global warming function which we will find maximum temperature according to emissivity of earth.

Our space is **[0.2,0.8]**

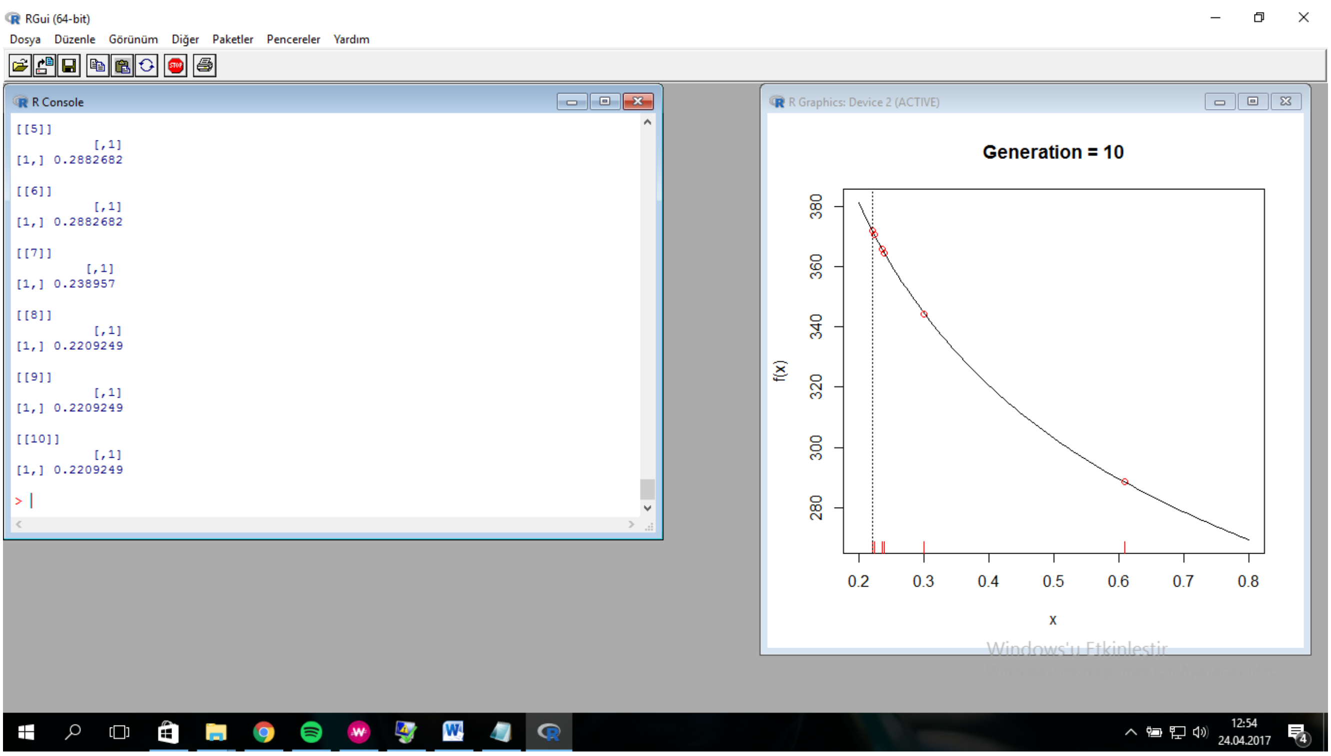
Our function is $\sqrt[4]{\frac{(1-0.3)*1367}{4x*5.67*10^{-8}}}$

Encoding for example 0.2 is **00111110 01001100 11001100 11001101**

The code for generating new generation and finding maximum temperature is shown below.

```
library(GA)
# the library instruction loads a package and should be used
only once during one program
# running
#setting the parameters
min_x = 0.2
max_x = 0.8
pop_size = 10
pc = 0.8
pm = 0.2
maxiter = 10
sleep = 2
```

```
#sleep=a - a is delay in seconds before the next generation is to be displayed,
if a=-1 then the
#next population is shown by hitting any key (e.g. enter)
#the function to optimized
f <- function(x) sqrt(sqrt((1-0.3)*1367/(4*x*5.67*10^(-8))))
#fitness function, the same as the optimization one if it is to be maximized,
the negation of the
#optimization one if it is to be minimized
fitness = function(x) sqrt(sqrt((1-0.3)*1367/(4*x*5.67*10^(-8))))
#drawing the function graph
5
curve(f, min_x, max_x)
# monitoring function showing succeeding steps of the algorithm
monitor <- function(obj)
{
curve(f, min_x, max_x, main = paste("Generation =", obj@iter))
points(obj@population, f(obj@population), pch = 1, col = 2)
rug(obj@population, col = 2)
if (sleep<0)
readline()
else
Sys.sleep(sleep)
}
# the algorithm running
GA <- ga(type = "real-valued",fitness = fitness,min = min_x,max = max_x,popSize
= pop_size,
pcrossover = pc,pmutation = pm,maxiter = maxiter,monitor = monitor,keepBest =
TRUE ,seed =
123)
#displaying the summary
summary(GA)
# showing the best fit individual on the function graph
abline(v = GA@solution, lty = 3)
#displaying the fitness if the best individual in each generation
GA@bestSol
```



Conclusion

In genetic algorithm, we have different parameters which effect the result. These are population size, maximum generation, probability of crossover, probability of mutation. When we increase the population size individuals distrubuted on the chart. When we increase the maximum generation individuals more closer to goal point. When we increase probability of crossover genetic variety will increase also. When we increase the last parameter, probability of mutation, population will has more extra ordinary individuals.