

Coding assignment

11.12.2009 SGN-6166 Introduction to R programming

Guidelines

The tasks are given in two files (Assignments.r and GeneticAlgorithm.r). Add your name to both of them. You should write your codes to those files, too.

In the files, the tasks are indicated by the notation:

```
##### TASK BEGINS #####
```

```
# Task x (y points)
```

```
#
```

```
# Task description...
```

```
#
```

```
##### TASK ENDS #####
```

Tasks 4 and 5 are located in the file GeneticAlgorithm.r while the other tasks are located in file Assignments.r. This printed document gives additional information for Tasks 2-6. Note that all the tasks are independent on each other. In particular, Tasks 4-6 relate to each other but still you may complete e.g. Task 6 without completing Tasks 4 or 5. In that case, the result graphs of Task 6 just don't look the same.

After you have completed the tasks (11:45 at latest):

- 1) Save the files.
- 2) Close R and your R editor.
- 3) Then email both of your result files (Assignments.r and GeneticAlgorithm.r) to tommi.aho@tut.fi.

Start by downloading the files:

<http://www.cs.tut.fi/~aho2/Assignments.r>

<http://www.cs.tut.fi/~aho2/GeneticAlgorithm.r>

<http://www.cs.tut.fi/~aho2/lecture 4 houses.data>

Task 1

Task given in the file Assignments.r.

Task 2

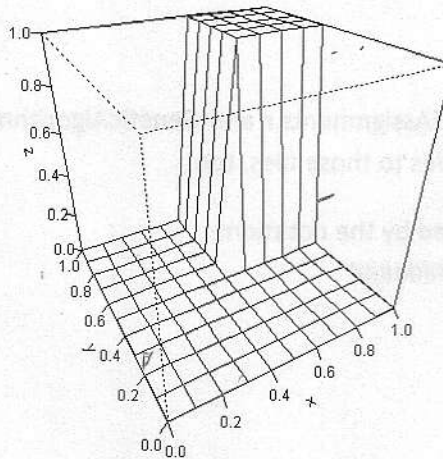
Task given in the file Assignments.r. Here is a definition for the Fibonacci numbers:

Fibonacci numbers are the sequence of numbers: 1, 1, 2, 3, 5, 8, 13, 21, ...

The sequence can be produced as $F_n = F_{n-1} + F_{n-2}$ with $F_1 = F_2 = 1$. In other words, the first two numbers are ones and each remaining number is the sum of the previous two numbers.

Task 3

Task is given in the file Assignments.r. The figure should look like this:

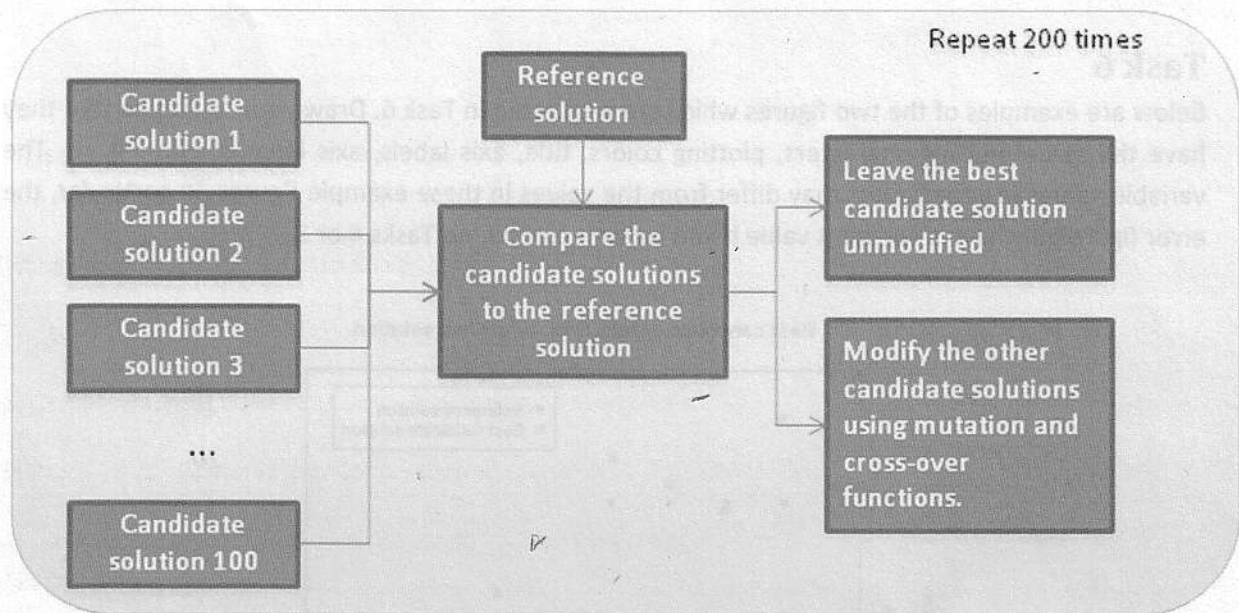


Tasks 4-5

In this task you implement two functions that complete a simple version of an optimization algorithm called genetic algorithm (see e.g. http://en.wikipedia.org/wiki/Genetic_algorithm ; the algorithm is inspired by genetics but here you don't need to know anything about genetics). The idea is to iteratively modify so-called candidate solutions to be similar to a particular reference solution. Our algorithm has 100 candidate solutions (each of which is a vector of 10 random numbers) and a reference solution (which is a 10-element vector of zeros and ones). The algorithm has the following steps:

- 1) The candidate solutions are created.
- 2) The candidate solutions are compared element-wise to the reference solution.
- 3) The candidate solution closest to the reference solution remains unmodified.
- 4) The other candidate solutions are modified using functions called `mutation()` and `cross.over()`.
- 5) The steps 2-4 are repeated 200 times.
- 6) Results are plotted.

The algorithm is already executable and you can run it (see the files `Assignments.r` and `GeneticAlgorithm.r`). The only things not working yet are the functions `mutation()` and `cross.over()` which do not modify the candidate solutions at all. Your task is to implement the contents of those two functions (Tasks 4-5). (In Task 6 your task is to plot the optimization results as requested). More information is given below and in the files `Assignments.r` and `GeneticAlgorithm.r`.



Principles for the functions `mutate()` and `cross.over()`

These examples illustrate the principles how you should modify the candidate solutions within the two functions. The shown numeric values are just examples.

Principle of the modification function `mutate()`

- 1) Candidate solution (a vector of length 10) before mutation: [2, 4, 1, -3, -2, 1, 0, 0, -1, -2]
- 2) Choose one element randomly. E.g. the second element.
- 3) Assign a random number to the second element.
- 4) Candidate solution after mutation (the bolded element has been modified): [2, **-1**, 1, -3, -2, 1, 0, 0, -1, -2]

Principle of the modification function `cross.over()`

- 1) Two candidate solutions before cross-over: [2, 4, 1, -3, -2, 1, 0, 0, -1, -2]
[3, -1, -6, 1, 0, -2, 1, 0, -3, 4]
- 2) Randomly select two numbers which determine the starting element and ending element of cross-over. E.g. 2 and 4.
- 3) Change the respective elements (all elements from the starting element to the ending element) between the candidate solutions.
- 4) The same two candidate solutions after cross-over (note how the bolded elements have been changed between the candidates): [2, **-1**, **-6**, **1**, -2, 1, 0, 0, -1, -2]
[3, **4**, **1**, **-3**, 0, -2, 1, 0, -3, 4]

Task 6

Below are examples of the two figures which are requested in Task 6. Draw your figures so that they have the same plotting characters, plotting colors, title, axis labels, axis lengths, and legend. The variable values in your figures may differ from the values in these example figures. In particular, the error figure may have a constant value if you haven't completed Tasks 4 or 5.

