Exercise 5: Global Search

Summer Term 2024

In this exercise, we study a very simple global optimization procedure, known as *systematic* or *exhaustive* search.

Review: Assume, you have a fitness function $f(x)$ of any kind, for *example* $f(x) = 0.5x^2$ or $f(x) = \frac{(x-2)(x-4)}{(x+5)(x^2-12)} + xe^{-x}$, and a search space $x_{\min} \leq x \leq x_{\max}$. How many test, i.e., function evaluations, have you do to in order to approach the location of the true optimum x_{opt} with a precision $\epsilon \leq |x_{\text{opt}} - x^o|$? How does the number of function evaluations required to reach a certain precision ϵ depend on the actual fitness function $f(x)$?

To Do: An in-depth analysis of the systematic search procedure.

Tasks: Please do the following tasks.

1. Now, let us consider a simple test case: (1) the number of test points in one dimension is $(x_{\max} - x_{\min})/\epsilon = 1000$, (2) you have a quite fast processor operating at 10 GHz, and (3) the implementation is extremely efficient and requires only ten instructions per fitness evaluations. Please, answer the following questions:
   (a) How many fitness evaluations can the processor perform per second?
   (b) **Guess** (just answer what you have in mind), in how many dimensions n, i.e., $f(x_1, \ldots, x_n)$, can this system complete its optimize task within two days? How many dimensions n can be solved and two years?
2. We now do the same task a bit more systematic. In order to go beyond "just guessing", you might be completing the following table:

n	1	2	3	4	5	6	7	8	9	10	20	50	100
time													

3. What does the results mean and imply? Please, discuss the following questions:
   (a) Now you know it: in how many dimensions n can this system complete its optimize task within two days and two years, respectively?
   (b) How well does this procedure scale?
   (c) What is the scaling law? $t(n) = O(\quad)$
   (d) What is the procedure's utility when considering real-world applications?

Have fun, Theo and Ralf.