FEDERAL STATE AUTONOMOUS EDUCATIONAL INSTITUTION
OF HIGHER EDUCATION
ITMO UNIVERSITY

# Report
on the practical task No. 2
“Algorithms for unconstrained nonlinear optimization. Direct methods”

Performed by
*Anastasia Miroshnikova*
*j4133c*
Accepted by
Dr Petr Chunaev

St. Petersburg
2020

# Goal

The use of direct methods (one-dimensional methods of exhaustive search, dichotomy, golden section search; multidimensional methods of exhaustive search, Gauss, Nelder-Mead) in the tasks of unconstrained nonlinear.

# Formulation of the problem

**I.** Use the one-dimensional methods of exhaustive search, dichotomy and golden section search to find an approximate (with precision $\varepsilon=0.001$) solution $x: f(x) \to min$ for the following functions and domains:

1. $f(x)=x^3$, $x\in[0,1]$;
2. $f(x)=|x-0.2|$, $x\in[0,1]$;
3. $f(x)=x\sin\frac{1}{x}$, $x\in[0.01,1]$.

Calculate the number of $f$-calculations and the number of iterations performed in each method and analyze the results. Explain differences (if any) in the results obtained.

**II.** Generate random numbers $\alpha\in(0,1)$ and $\beta\in(0,1)$. Furthermore, generate the noisy data $\{x_k, y_k\}$, where $k=0,\ldots,100$, according to the following rule:

$$y_k=\alpha x_k+\beta+\delta_k, x_k=\frac{k}{100},$$

where $\delta_k$ $N(0,1)$ are values of a random variable with standard normal distribution. Approximate the data by the following linear and rational functions:

1. $F(x,a,b)=ax+b$ (linear approximant),
2. $F(x,a,b)=\frac{a}{1+bx}$ (rational approximant),

by means of least squares through the numerical minimization (with precision $\varepsilon=0.001$) of the following function:

$$D(a,b)=\sum_{k=0}^{100}(F(x_k,a,b)-y_k)^2.$$

To solve the minimization problem, use the methods of exhaustive search, Gauss and Nelder-Mead. If necessary, set the initial approximations and other parameters of the methods. Visualize the data and the approximants obtained in a plot separately for each type of approximant. Analyze the results obtained (in terms of number of iterations, precision, number of function evaluations, etc.)

## Brief theoretical part

**Exhaustive search** is a brute-force method of finding optimal solution of the problem. When finding optimum of one-dimensional function, it calculates the value of the function at each point in the given range with defined precision. For example, if we want to find minimum of the function in the range from 0 to 1 with precision 0.001, we have to make 1000 calls of the function.

For the multiple-dimensional problems the number of function calls needed to find optimum using exhaustive search increases as exponential function of number of dimensions. If we wanted to find minimum of the 2D function in the range from 0 to 1 with precision 0.001, we should make $10^6$ calls. In practice, this method is used to find initial approximations in broad search space.

**Dichotomic search** in a broad sense is a search algorithm that operates by selecting between two distinct alternatives (dichotomies) at each step. It is a specific type of divide and conquer algorithm. Therefore, it has to make substantially less steps than exhaustive search to find an optimum of the function.

The **golden-section search** is a technique for finding an extremum (minimum or maximum) of a function inside a specified interval. For a strictly unimodal function with an extremum inside the interval, it will find that extremum, while for an interval containing multiple extrema (possibly including the interval boundaries), it will converge to one of them. If the only extremum on the interval is on a boundary of the interval, it will converge to that boundary point. The method operates by successively narrowing the range of values on the specified interval, which makes it relatively slow, but very robust. The technique derives its name from the fact that the algorithm maintains the function values for four points whose three interval widths are in the ratio $2-\varphi:2\varphi-3:2-\varphi$ where $\varphi$ is the **golden ratio**. These ratios are maintained for each iteration and are maximally efficient. Excepting boundary points, when searching for a minimum, the central point is always less than or equal to the outer points, assuring that a minimum is contained between the outer points.

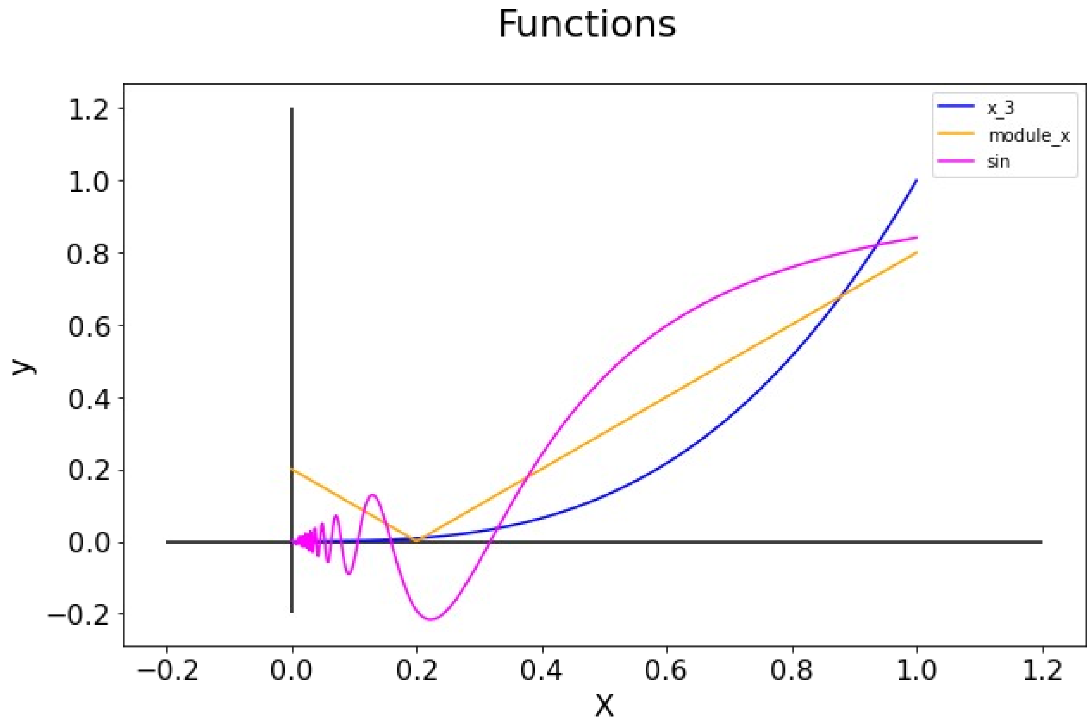
The idea of **Gauss method** of multidimensional function optimization is that in each iteration, the minimisation is carried out only with respect to one vector component of the multidimensional variable x. The method is simple but hardly efficient. Problems appear when the level lines of a function to be optimized are strongly elongated along the “diagonal” line $x_1 = x_2$. If the initial approximation is on $x_1 = x_2$ , then the process gets stuck.

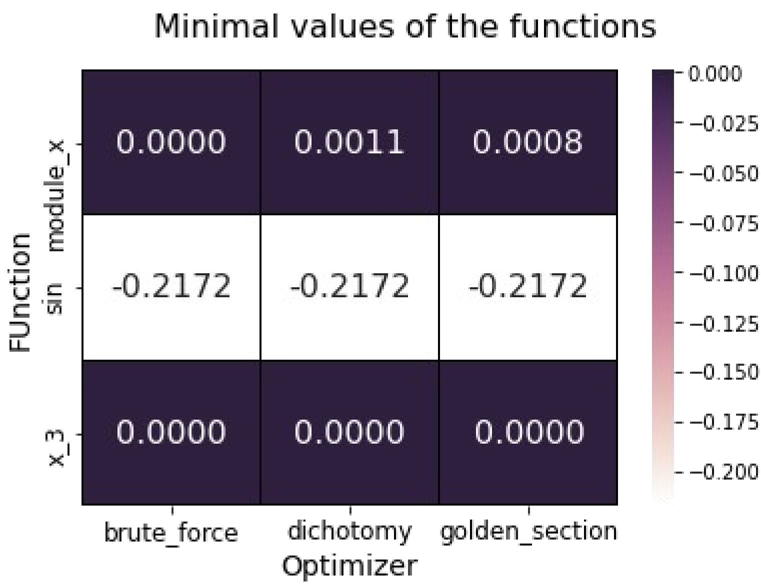
The **Nelder–Mead method** is a commonly applied numerical method used to find the minimum or maximum of an objective function in a multidimensional space. The Nelder–Mead technique is a heuristic search method that can converge to non-stationary points on problems that can be solved by alternative methods.
The idea of the method is to find function values at the vertices of simplex in the search space, define a vertex with maximum function value, and then reflect this point with respect to the gravity center of the other points.

# Results

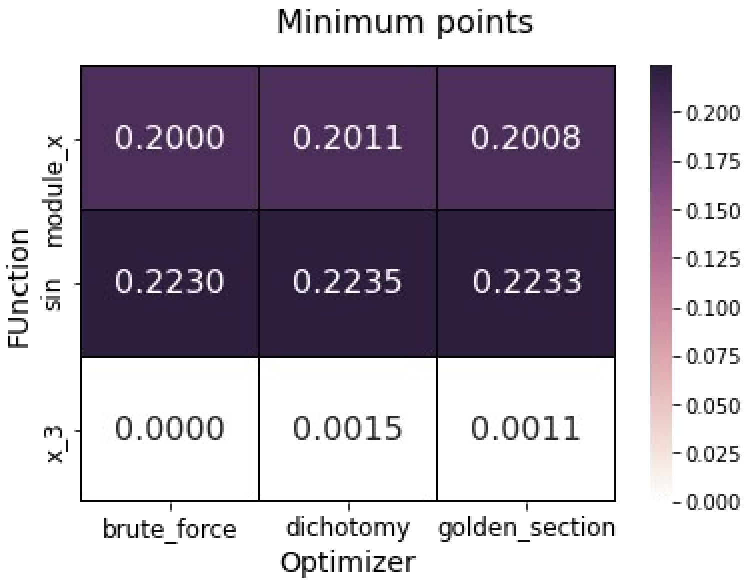
## One-dimensional optimization problem

The methods of exhaustive search, dichotomy, and golden-section search were implemented using Python 3.6.9 programming language and numpy library.

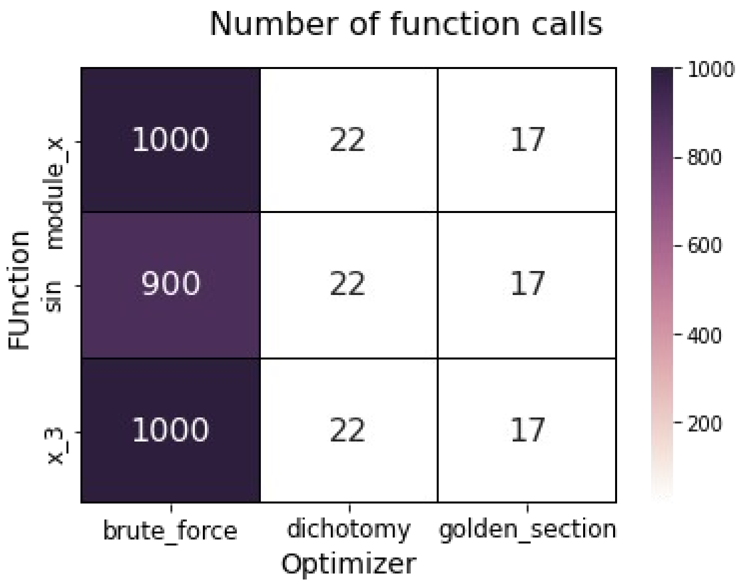



All algorithms converged to approximately the same minimal values of the functions. The calculated values are in agreement with theoretically expected values of minimums of the functions.




The minimum points for the algorithms are approximately the same and are in concordance with real minimum points.



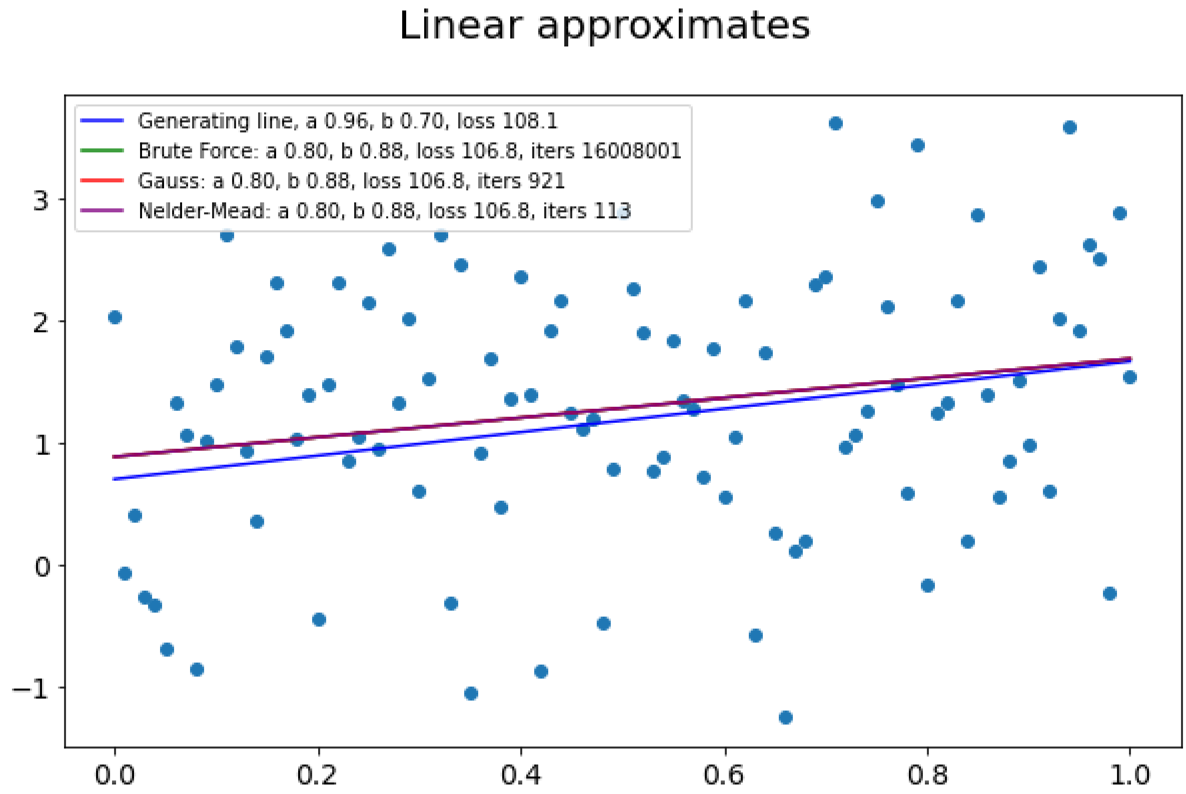

Brute force proved to be the most ineffective in terms of "error – number of function calls" trade-off. Although it had found the most accurate minimum points, it required 50 times more function calls that two other direct methods.

The most effective method was golden-section search – it required the least number of function calls and provided acceptable accuracy.

## Two-dimensional optimization problem

The multidimensional brute-force search and Gauss method were implemented using Python 3.6.9. The Gauss method used golden-section search as one-dimensional algorithm under the hood. For Nelder-Mead algorithm scipy library implementation was used. This algorithm requires no bounds, so the value of the loss function returned was artificially set to a high number if the search algorithm tried to escape predefined search boundaries.

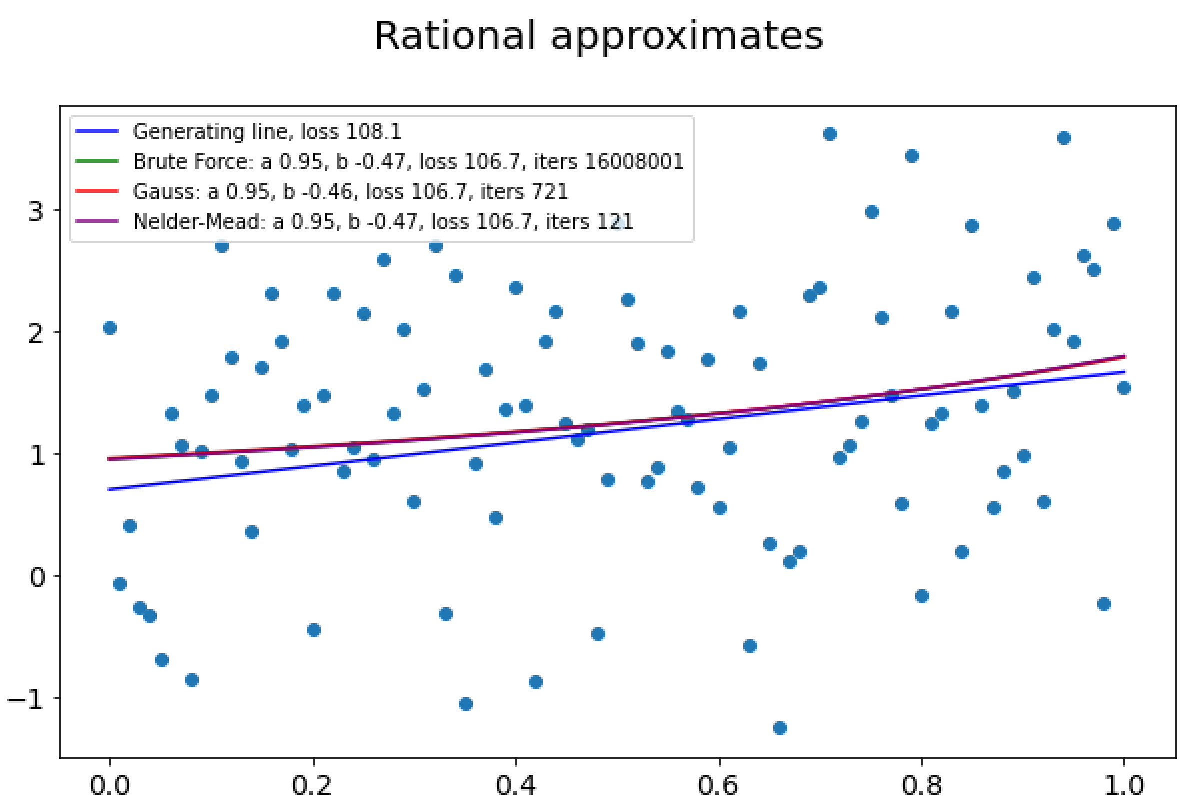
Both parameters “a” and “b” for both types of approximates had to lie within interval [-2, 2].




**Linear approximation**

All algorithms of search for optimal values of parameters “a” and “b” converged to the same values. It should be noted that the value of the loss function with ground-gruth parameters was higher than the value of the loss function with found parameters.

Brute-force algorithm for this task required 16000 times more function calls than Gauss method and 160000 times more calls than Nelder-Mead algorithm. Again it proved to be highly ineffective. Gauss method required 9 times more calls than Nelder-Mead algorithm, which is still huge amount.




**Rational approximation**

In the first experiments where coefficients “a” and “b” had to lie within interval [0, 1], the value of “b” always was zero and rational approximation was just constant. But in wider range all search algorithms found new minimum with “b” < 0. All algorithms again converged to the same values of parameters. Interesting, that the value of the loss for rational approximates is lower than the value of the loss for linear approximates, although the data were produced by noisy linear function.

As for the linear approximation, Nelder-Mead algorithm had found optimal solution using the least number of function calls.

## Conclusions

Direct methods of optimization were used to solve one-dimensional and two-dimensional optimization problems. Brute-force method proved to be the most ineffective for both types. Dichotomy method happened to be less effective than golden-section search for one-dimensional optimization. Nelder-Mead algorithm was more effective than Gauss method for two-dimensional optimization.

## Appendix

Source code can be found at https://github.com/Miffka/algo_itmo/tree/master/task2.