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
Tiulkov Nikita
J4133c

Accepted by
Dr Petr Chunaev

St.Petersburg
2021

1. Goal

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

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

II. 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. 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:

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

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.).

3. Brief theoretical part

3.1. Exhaustive search

In computer science, brute-force search or exhaustive search, also known as generate and test, is a very general problem-solving technique and algorithmic paradigm that consists of systematically enumerating all possible candidates for the solution and checking whether each candidate satisfies the problem's statement. A brute-force algorithm to find the divisors of a natural number n would enumerate all integers from 1 to n, and check whether each of them divides n without remainder.

3.2. Dichotomy method

Dichotomy method is a root-finding method that applies to any continuous functions for which one knows two values with opposite signs. The method consists of repeatedly bisecting the interval defined by these values and then selecting the subinterval in which the function changes sign, and therefore must contain a root.

3.3. Golden-section search

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.

3.4. Gauss method

Given m functions $r = (r_1, \dots, r_m)$ of n variables $\beta = (\beta_1, \dots, \beta)$, with $n \leq m$, algorithm iteratively finds the value of the variables that minimizes the sum of squares.

$$S(\beta) = \sum_{i=1}^{m} r_i^2(\beta)$$

Starting with an initial guess $\beta^{(0)}$ for the minimum, the method proceeds by the iterations

$$\beta^{(s+1)} = \beta^{(s)} - (J_r^T J_r)^{-1} J_r^T r\beta^{(s)}$$

where, if r and β are column vectors, the entries of the Jacobian matrix are

$$(J_r)_{ij} = \frac{\partial r_i(\beta^{(s)})}{\partial \beta_j}$$

In data fitting, where the goal is to find the parameters β such that a given model function $y = f(x, \beta)$ best fits some data points (x_i, y_i), the functions r_i are the residuals:

$$r_i(\beta) = y_i - f(x_i, \beta)$$

.

3.5. The Nelder–Mead method

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 method uses the concept of a simplex, which is a special polytope of n + 1 vertices in n dimensions. Examples of simplices include a line segment on a line, a triangle on a plane, a tetrahedron in three-dimensional space and so forth. The method approximates a local optimum of a problem with n variables when the objective function varies smoothly and is unimodal. Typical implementations minimize functions, and we maximize $f(x)$ by minimizing $-f(x)$. Nelder–Mead in n dimensions maintains a set of $n + 1$ test points arranged as a simplex. It then extrapolates the behavior of the objective function measured at each test point in order to find a new test point and to replace one of the old test points with the new one, and so the technique progresses. The simplest approach is to replace the worst

point with a point reflected through the centroid of the remaining n points. If this point is better than the best current point, then we can try stretching exponentially out along this line. On the other hand, if this new point isn't much better than the previous value, then we are stepping across a valley, so we shrink the simplex towards a better point.

4. Results

4.1. I, Exhaustive search

At Figures 1, 2, 3 we can see minimum of function (red point) and steps for finding minimum (black points). Since we are using exhaustive search and check all point of segment, there are a lot of iteration steps. For cube - 1000, for module - 1000, and for $x * sinus$ - 990 (less than on previous steps in case of watched segment is smaller). F-calculations in all methods equals to iteration steps cause on each step we calculate function once.

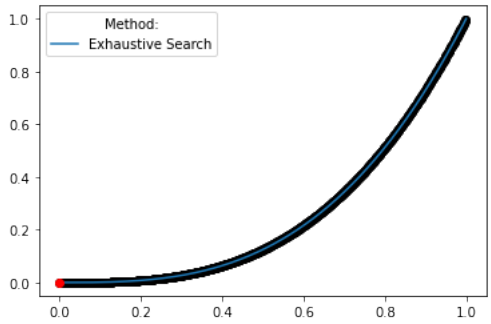

Figure 1: Exhaustive search for cube function

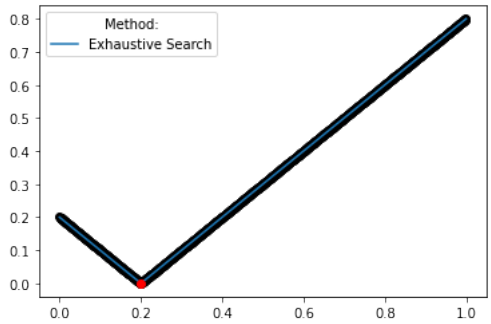

Figure 2: Exhaustive search for module function

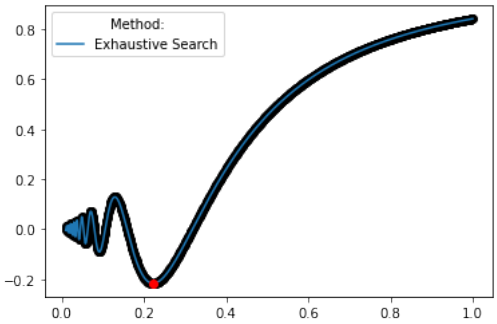

Figure 3: Exhaustive search for x*sinus function

4.2. I, Dichotomy

At Figures 4, 5, 6 we can see minimum of function (red point) and intermediate steps for finding minimum (black points) for dichotomy method. Compared to the previous method this method shows better results. For each function number of iterations equals 11, f-calculations equals 22, since we calculate function twice in one iteration.

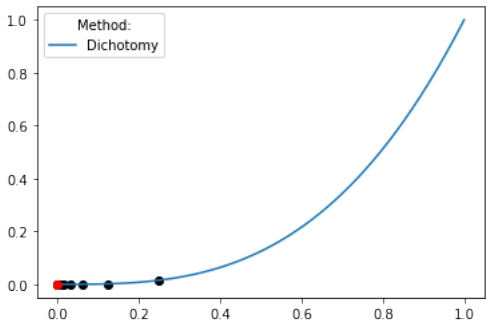

Figure 4: Dichotomy for cube function

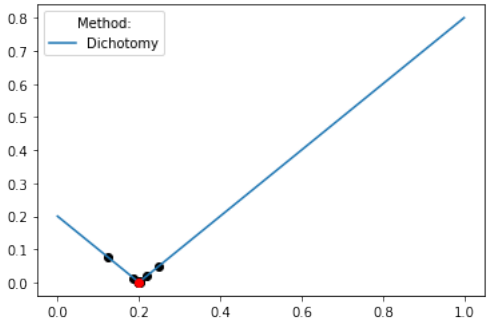

Figure 5: Dichotomy for module function

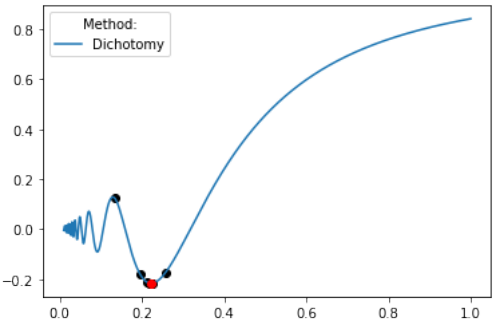

Figure 6: Dichotomy for x*sinus function

4.3. I, Golden section

At Figures 7, 8, 9 we can see minimum of function (red point) and intermediate steps for finding minimum (black points) for golden section. Compared to the previous method this method shows worse results, but also better than exhaustive search. For each function number of iterations equals 15, f-calculations equals 17, since we calculate function twice at the beginning and once in one iteration. So f-calculations less than in dichotomy method.

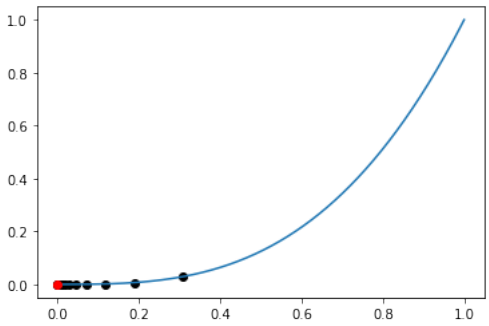

Figure 7: Golden section for cube function

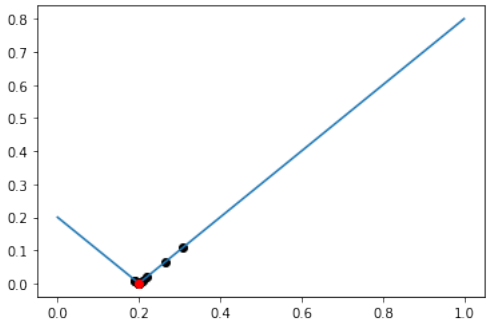

Figure 8: Golden section for module function

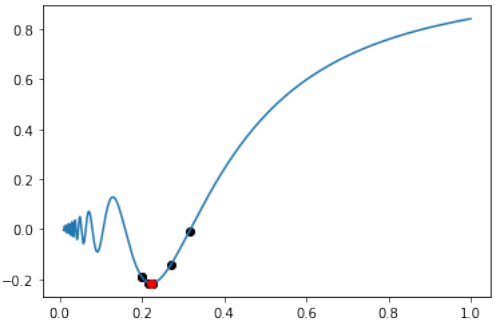

Figure 9: Golden section for x*sinus function

4.4. II, Exhaustive search

At Figures 10, 11 we can see the plots of linear and rational approximants and generating line. At both cases it took 4004001 steps to find a minimum value, because we need to check every value of a grid.

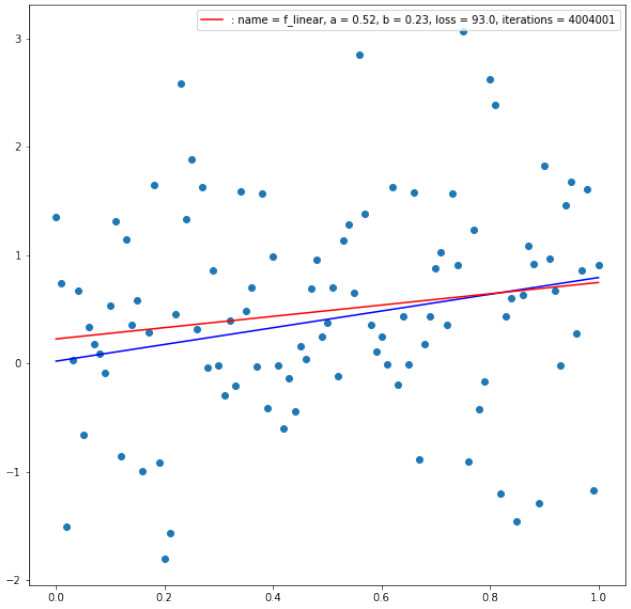

Figure 10: Exhaustive search for linear approximant

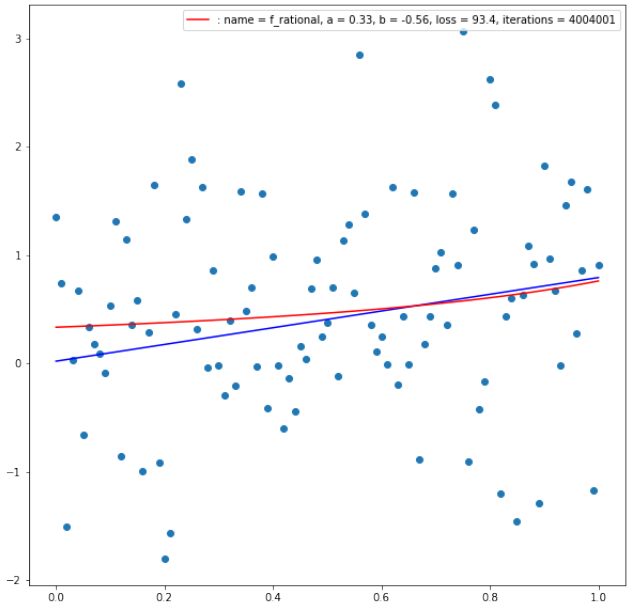

Figure 11: Exhaustive search for rational approximant

4.5. II, Gauss

At Figures 12, 13 we can see the plots of linear and rational approximants and generating line. At linear case it took 613 steps and at rational 469 to find a minimum value. It is better result than exhaustive search as expected.

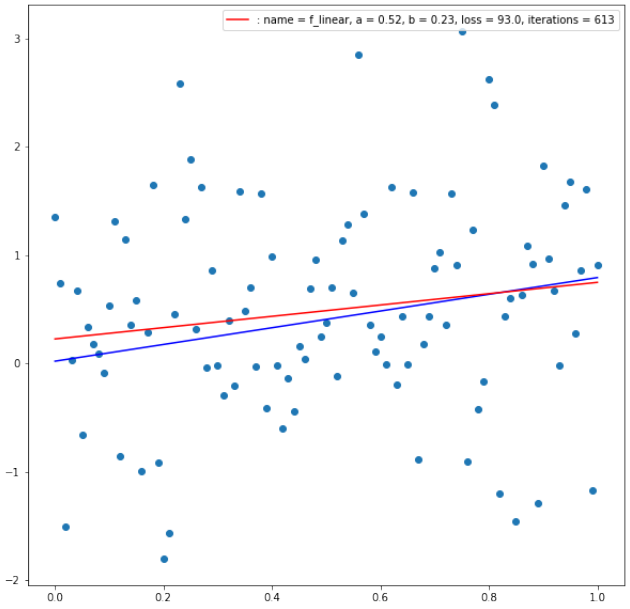

Figure 12: Gauss for linear approximant

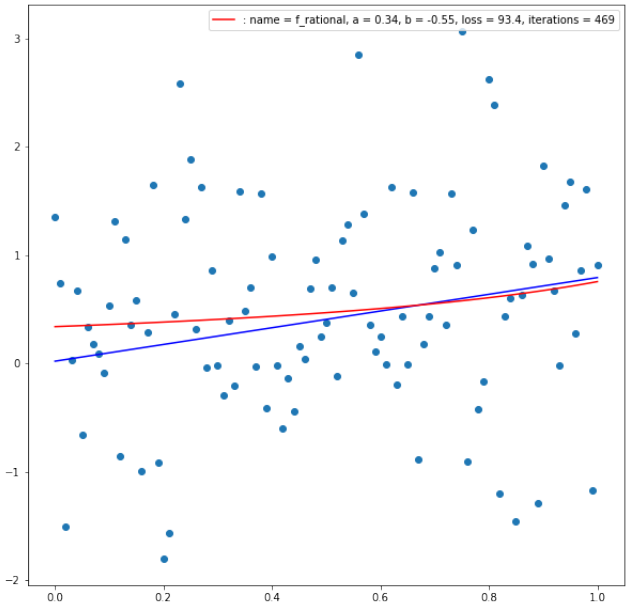

Figure 13: Gauss for rational approximant

4.6. II, Nelder-Mead

At Figures 14, 15 we can see the plots of linear and rational approximants and generating line. At linear case it took 98 steps and at rational 96 to find a minimum value. It is better result than all previous methods showed.

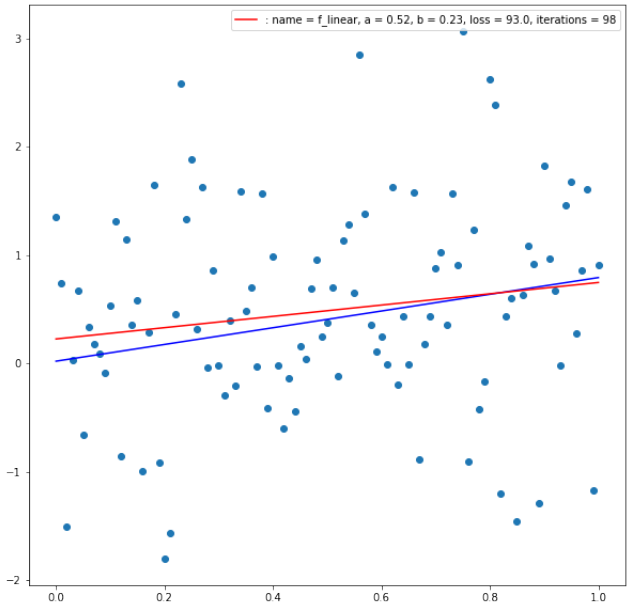

Figure 14: Nelder-Mead for linear approximant

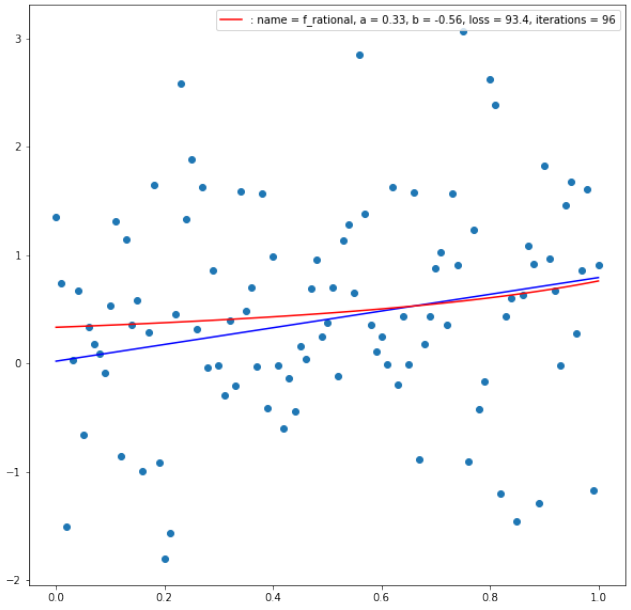

Figure 15: Nelder-Mead for rational approximant

5. Conclusions

To sum it up, we have measured the amount of iterations of well-known algorithm implementations with specified values, using data plots, representing function and points of steps. Brute-force method showed the most inefficient results in all dimensions, dichotomy and golden section showed good results as one-dimension algorithms. Gauss and Nelder-Mead methods also showed good results as two-dimension algorithms, but the last one was the most effective (in iterations count).

Appendix

The code of algorithms you can find on GitHub: https://github.com/FranticLOL/ITMO_Algorithms