FEDERAL STATE AUTONOMOUS EDUCATIONAL INSTITUTION OF HIGHER EDUCATION

ITMO UNIVERSITY

Report on the practical task No. 3 “Algorithms for unconstrained nonlinear optimization. First- and second- order methods”

Performed by

Alexander Yamoldin

J4134c

Accepted by

Dr Petr Chunaev

St. Petersburg

2021

Goal

The use of first- and second-order methods (Gradient Descent, Non-linear Conjugate Gradient Descent, Newton's method and Levenberg-Marquardt algorithm) in the tasks of unconstrained nonlinear optimization.

Formulation of the problem

Generate random numbers □ ∈ (0,1) and □ ∈ (0,1). Furthermore, generate the noisy data $\{x_k, y_k\}$*, where □ = 0, ... ,100, according to the following rule:*

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

where $\delta_k \sim N(0,1)$ *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 approximation),*

2. $F(x, a, b) = \dfrac{a}{1 + bx}$ *(rational approximant),*

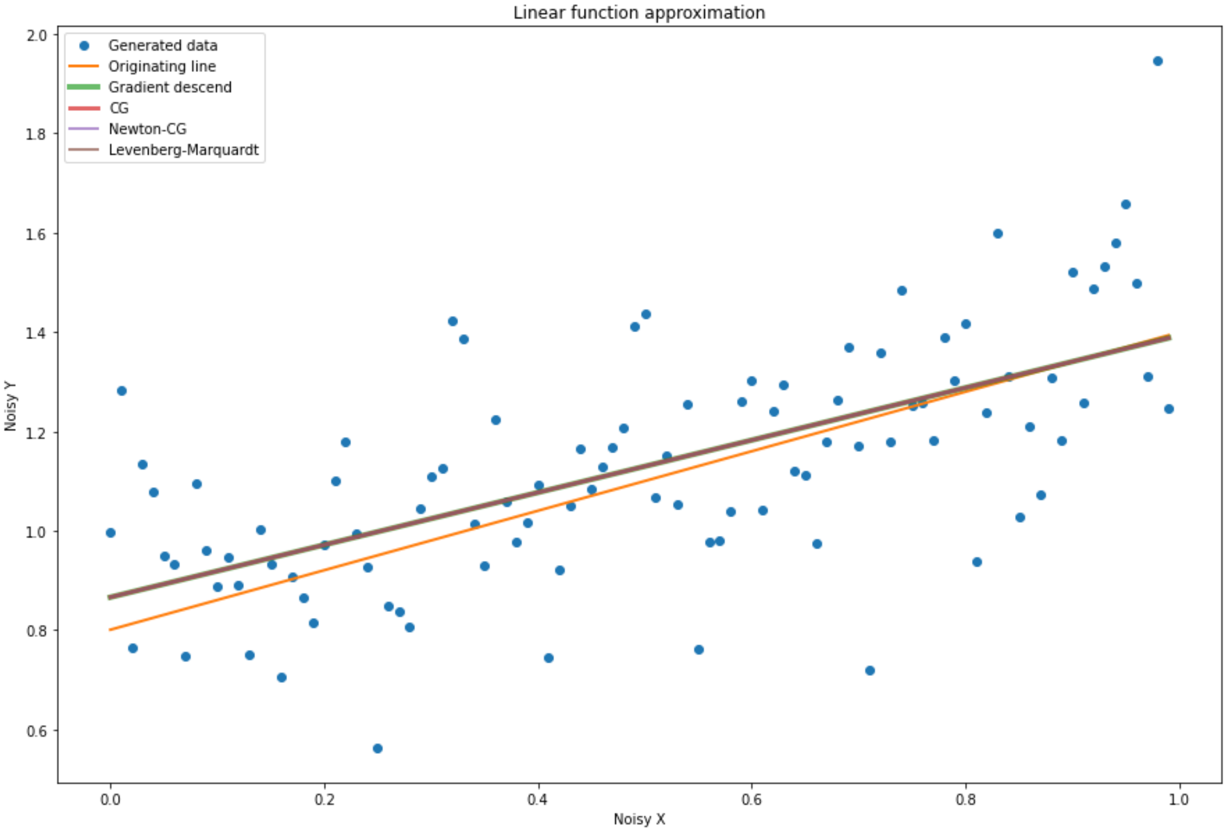
by means of least squares through the numerical minimization (with precision □ = 0.001) of the following function:

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

To solve the minimization problem, use the methods of Gradient Descent, Conjugate Gradient Descent, Newton's method and Levenberg-Marquardt algorithm. 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 so that one can compare the results for the numerical methods used. Analyze the results obtained (in terms of number of iterations, precision, number of function evaluations, etc.) and compare them with those from Task 2 for the same dataset.

Results

Linear function approximation

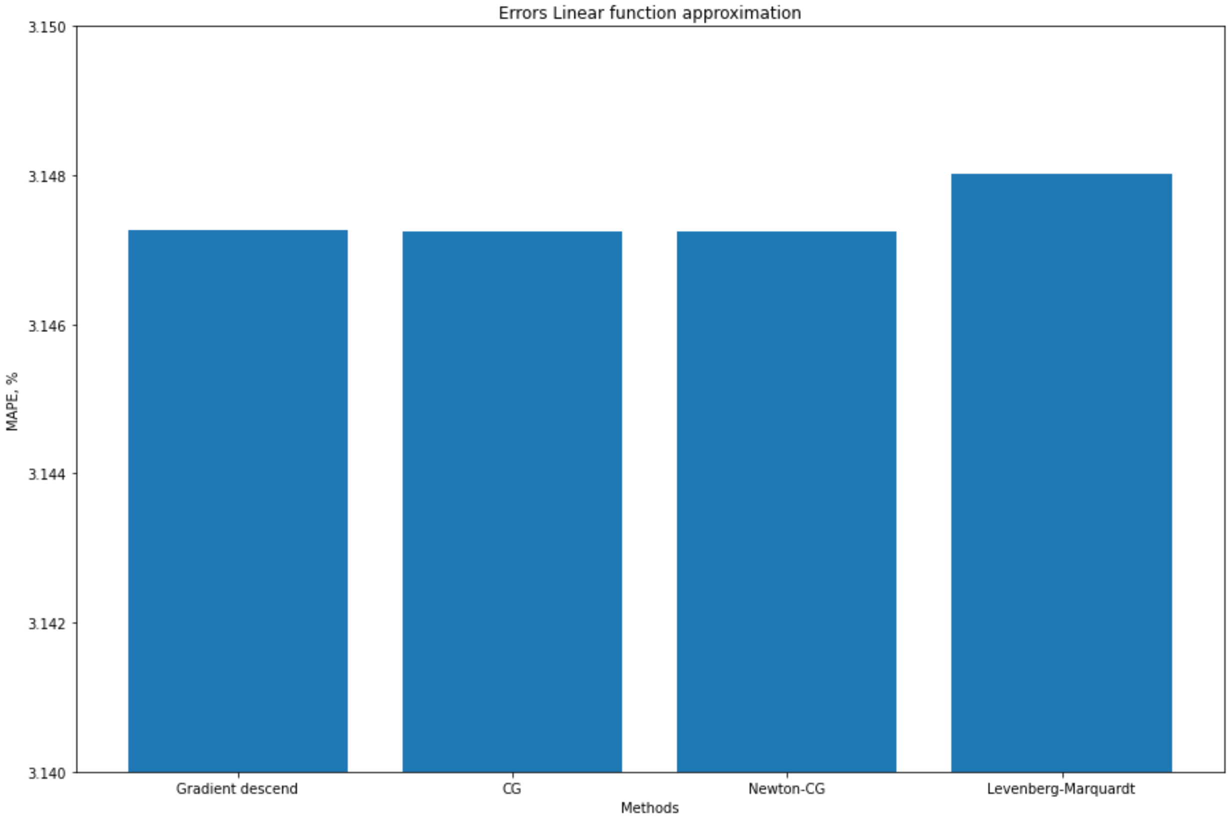


Results table:

Method	*a*	*b*	*Number of iterations*	*Number of function calls*	*Number of gradient calls*	*Number of Hessian calls*
Original linear function	*0.59*	*0.80*	-	-	-	-
Gradient descent	*0.53*	*0.86*	*763*	*4578*	*763*	*0*
Non-linear conjugate gradient descent	*0.53*	*0.86*	*3*	*30*	*10*	*0*
Newton's method	*0.53*	*0.86*	*4*	*5*	*11*	*0*
Levenberg-Marquardt algorithm	*0.53*	*0.86*	*190*	*190*	*190*	*0*

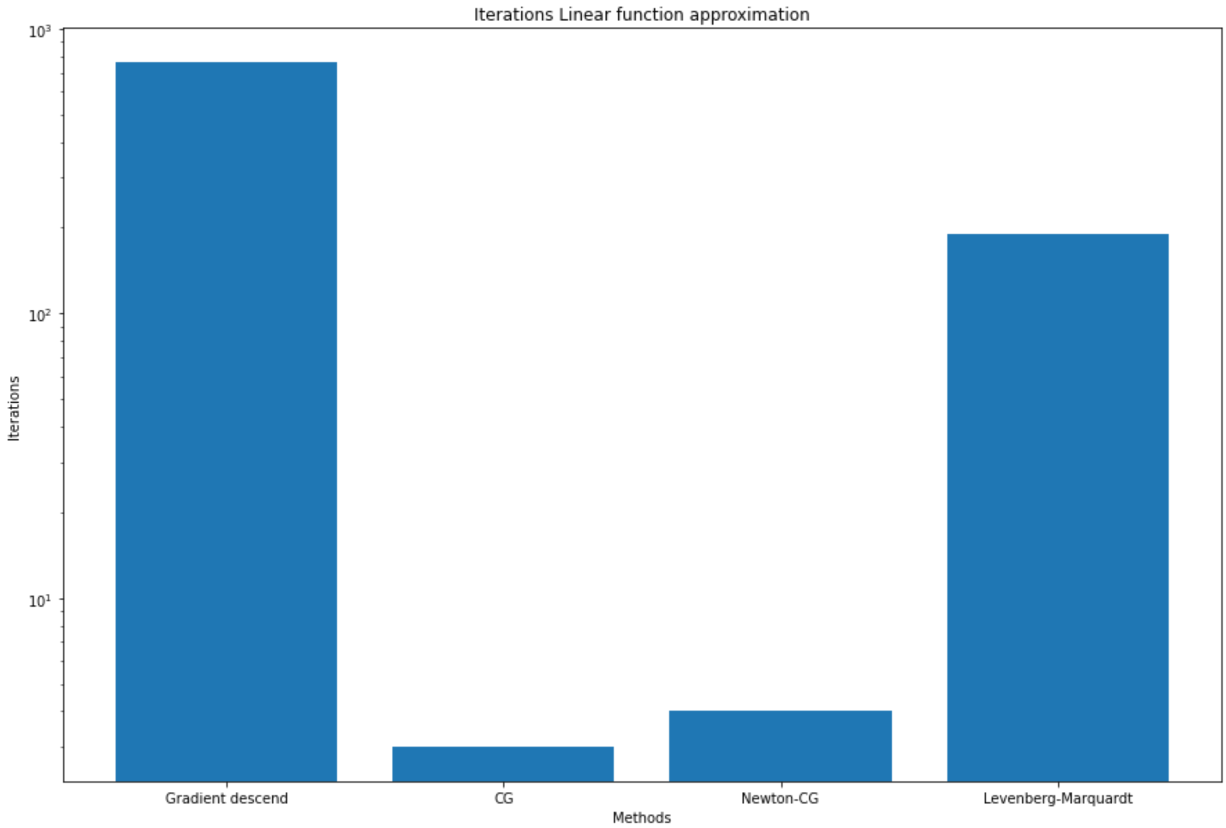
Statistical visualization:

MAPE of Linear function approximation. All algorithms have the same results.



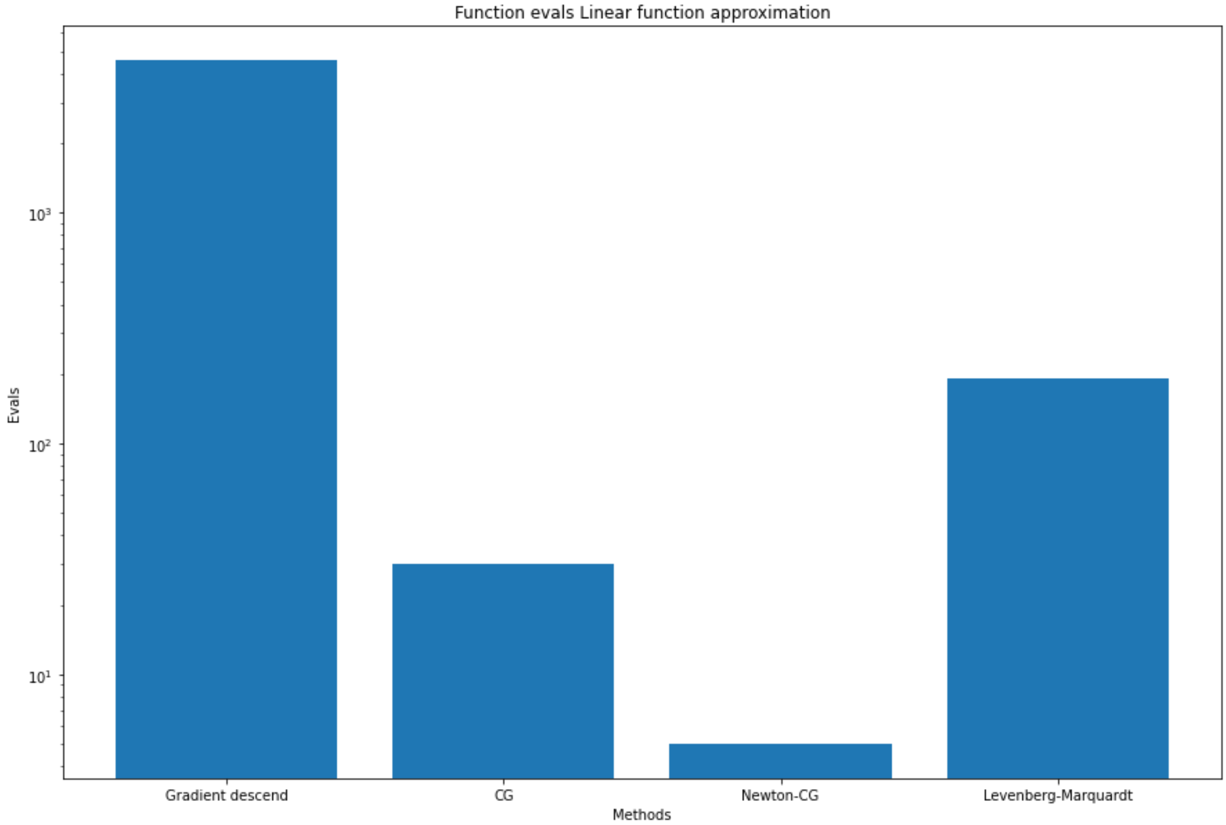
Numbers of iterations

As expected by theory, gradient descent has the most amount of iterations by solving optimization problem in comparing of other algorithms. The least is CG.

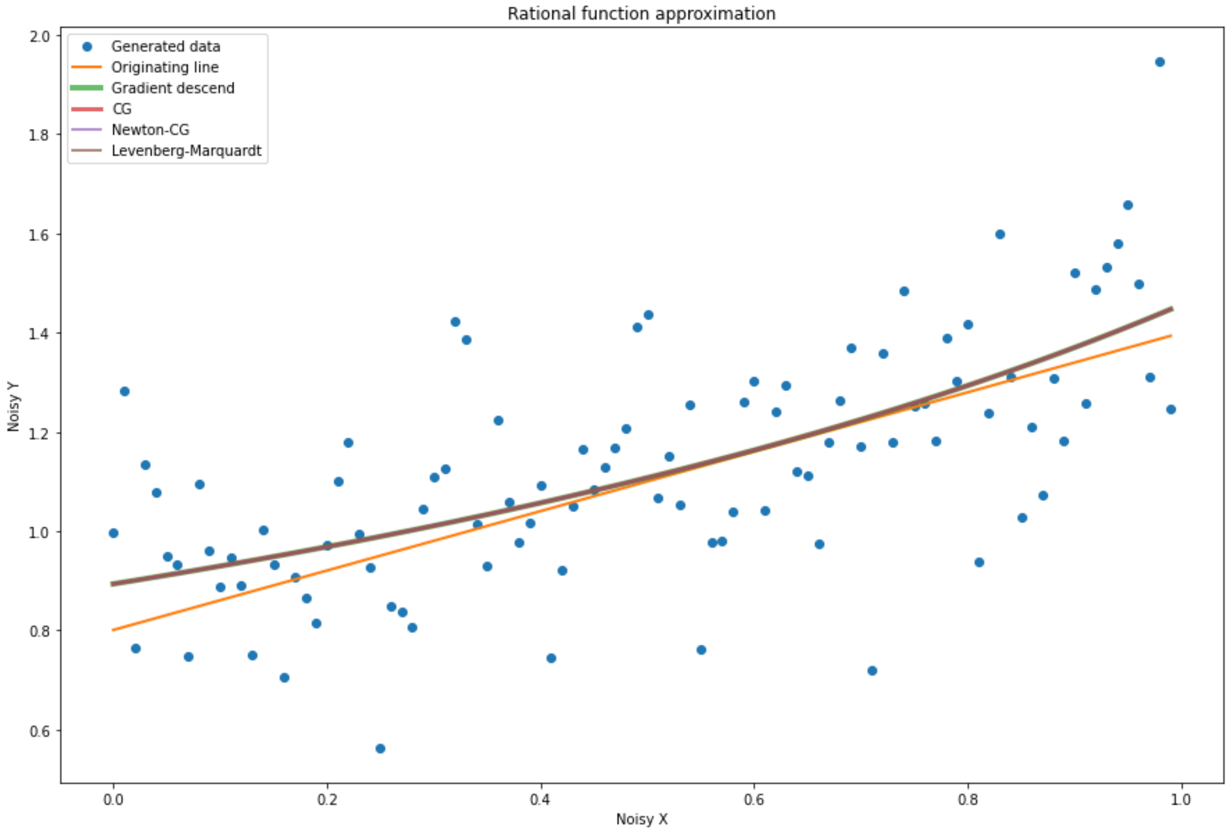


Number of function evals

As expected by theory, gradient descent has the greatest number of functions evals in comparing of other algorithms. The least number is Newton-CG.



Rational function approximation

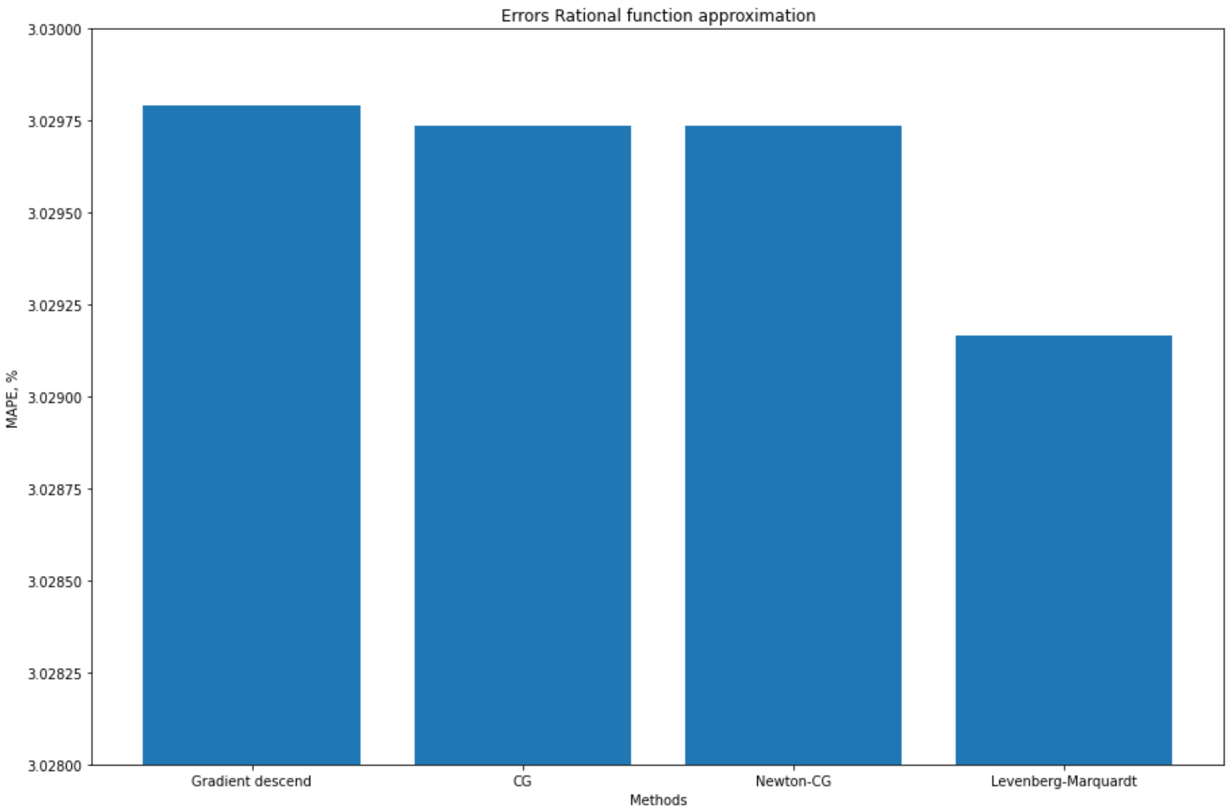


Results table:

Method	*a*	*b*	*Number of iterations*	*Number of function calls*	*Number of gradient calls*	*Number of Hessian calls*
Original linear function	*0.59*	*0.80*	*-*	*-*	*-*	*-*
Gradient descent	*0.89*	*-0.39*	*3167*	*19002*	*3167*	*0*
Non-linear conjugate gradient descent	*0.89*	*-0.39*	*12*	*69*	*23*	*0*
Newton's method	*0.89*	*-0.39*	*8*	*12*	*38*	*0*
Levenberg-Marquardt algorithm	*0.89*	*-0.39*	*105*	*105*	*105*	*0*

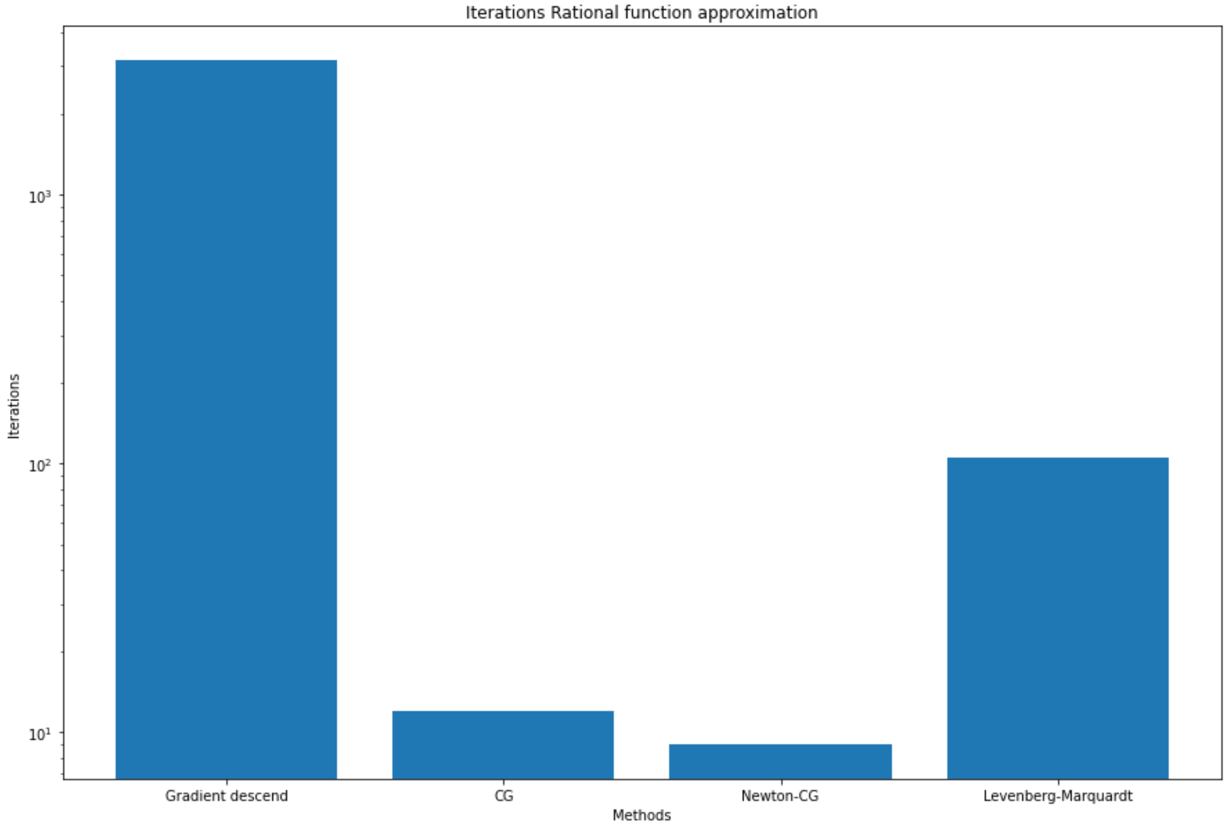
Statistical visualization:

MAPE of Linear function approximation. All algorithms have the same results, but Levenberg-Marquardt a little better than other (on 0.001%)



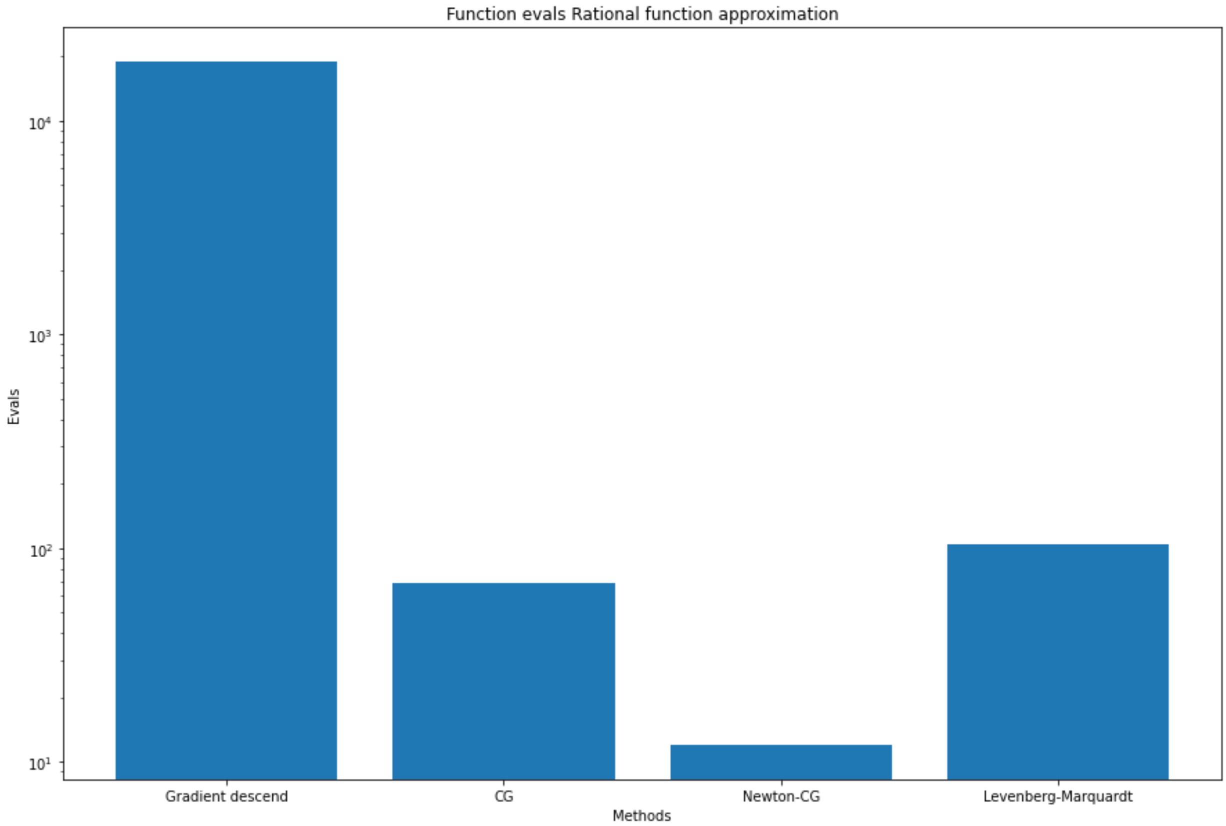
Numbers of iterations

As expected by theory, gradient descent has the most amount of iterations by solving optimization problem in comparing of other algorithms. The least is Newton-CG.



Number of function evals

As expected by theory, gradient descent has the greatest number of functions evals in comparing of other algorithms. The least number is Newton-CG.



Conclusions

From the data obtained, we can see that all 4 methods give almost the same approximation result in each function. Expectedly, the first-order methods converge faster and in fewer calculations than the direct methods, and also the second-order methods need fewer iterations than the first-order methods to reach the local minimum of the function. According the theory we confirmed that gradient descent algorithm requires the greatest number of iterations and evaluating functions. The most productive methods are CG and Newton-CG which is same as our theoretically expectations.

Appendix

https://github.com/AAYamoldin/TrainingPrograms/blob/master/Python/ITMO_algorithms_lab/Task_3/task3_Algorithms_for_unconstrained_nonlinear_optimization_first_and_second_order_methods.ipynb