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

*Abdurakhimov Muslimbek*

*J4133c*

Accepted by

Dr Petr Chunaev

St. Petersburg

2023

**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 𝛼 ∈ (0,1) and 𝛽 ∈ (0,1). Furthermore, generate the noisy data {, }, where 𝑘 = 0,…,100, according to the following rule:

= 𝛼 + 𝛽 + 𝛿k+, = ,

where 𝛿k ~ 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) = (rational approximant),

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

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.

**Brief theoretical part**

In this task, we are presented with the challenge of approximating a dataset using two different types of functions: linear and rational. The dataset is not perfect; it comes with noise, which makes it more realistic but also more challenging. Our objective is to determine the best-fitting parameters (a and b) for these approximating functions while minimizing the squared differences between our approximations and the actual noisy data points.

The process begins by generating random values for 𝛼 and 𝛽, both falling within the range (0,1). Then, we create a dataset consisting of pairs {x\_k, y\_k}, where k ranges from 0 to 100. The x\_k values are calculated as x\_k = k / 100, and the corresponding y\_k values are determined using the equation y\_k = 𝛼 \* x\_k + 𝛽 + 𝛿\_k.

Here, 𝛿\_k represents random noise, drawn from a standard normal distribution (𝛿\_k ~ 𝑁 (0,1)). This noise mimics the inherent uncertainty and imperfection that real-world data often possesses. Dichotomy: Also known as the bisection method, dichotomy repeatedly bisects the interval, narrowing down the search space until the desired precision is achieved. It's efficient for unimodal functions (those with a single peak or valley).

We aim to approximate this noisy dataset using two types of functions: a linear approximant (𝐹(𝑥, 𝑎, 𝑏) = 𝑎𝑥 + 𝑏) and a rational approximant (𝐹(𝑥, 𝑎, 𝑏) = a / (1 + b \* x)). These approximations are not exact but serve as our best guesses for the underlying patterns in the data.

To achieve this, we formulate an optimization problem where we want to find the values of the parameters 'a' and 'b' that minimize the squared differences between our approximating functions and the noisy data points. We express this minimization problem as an objective function D (a, b), defined as the sum of squared errors for all data points.

To tackle this optimization problem, we employ four numerical minimization methods:

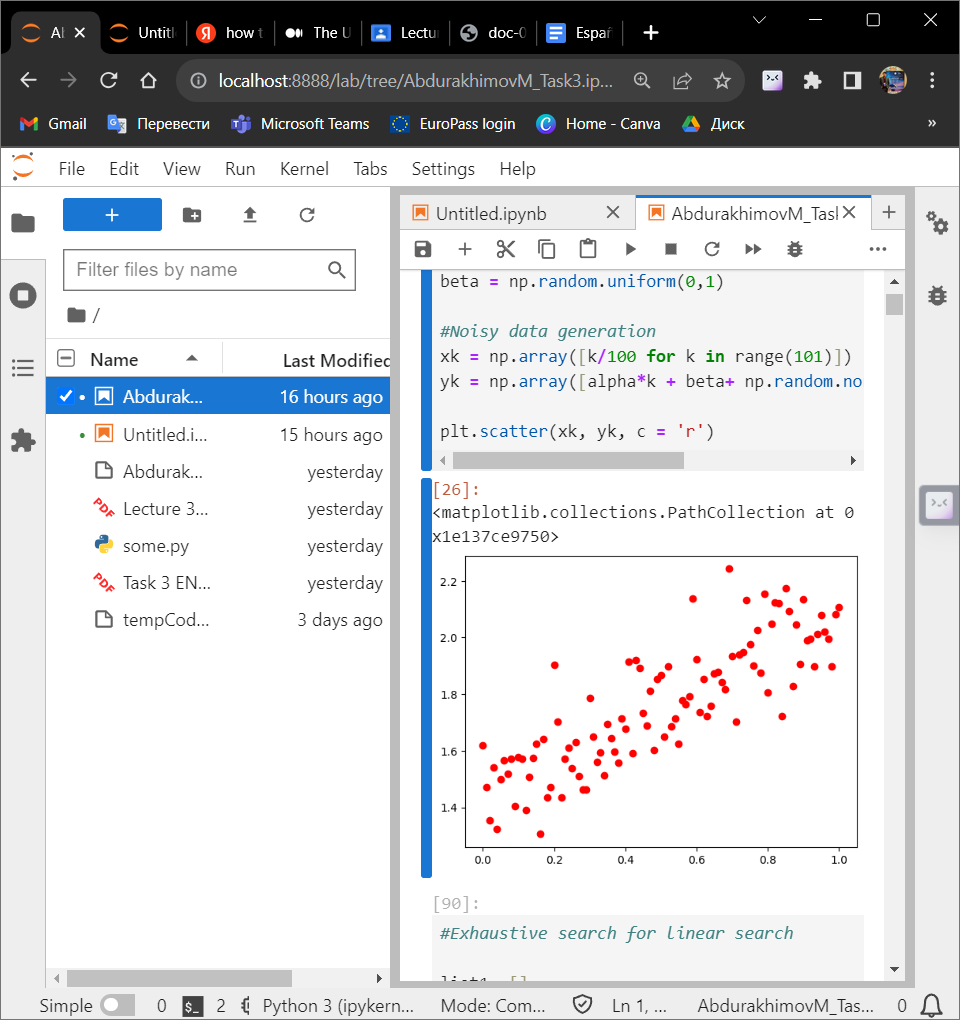
* Gradient Descent
* Conjugate Gradient Descent
* Newton's Method
* Levenberg-Marquardt Algorithm

These methods iteratively adjust the parameters 'a' and 'b' to minimize the objective function D (a, b) while considering the noise in the data.

In addition to these evaluations, we will also compare the results of this task with those obtained in a previous task (Task 2) that used different optimization methods and approximating functions on the same dataset. This comparative analysis will shed light on the relative strengths and weaknesses of the various optimization approaches when dealing with noisy data.

**Results**

I. First, we generated random numbers and noisy data.

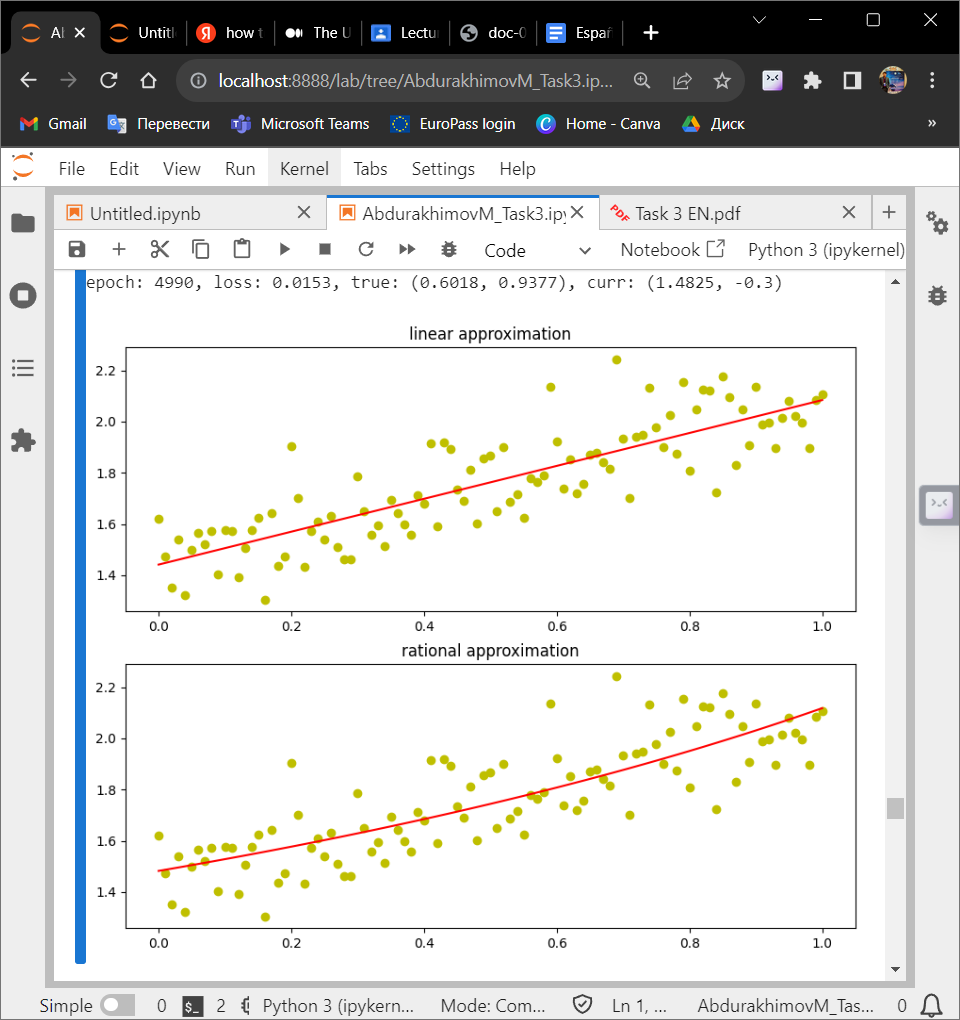


Picture 1 – Generating random numbers and noisy data where x ∈ [0,1]

In our quest to approximate a noisy dataset using both linear and rational functions, we applied four different numerical minimization methods: Gradient Descent, Conjugate Gradient Descent, Newton's Method, and the Levenberg-Marquardt Algorithm. Each of these methods aimed to find the best-fitting parameters 'a' and 'b' that minimize the squared differences between our approximating functions and the noisy data points.

Gradient Descent method:

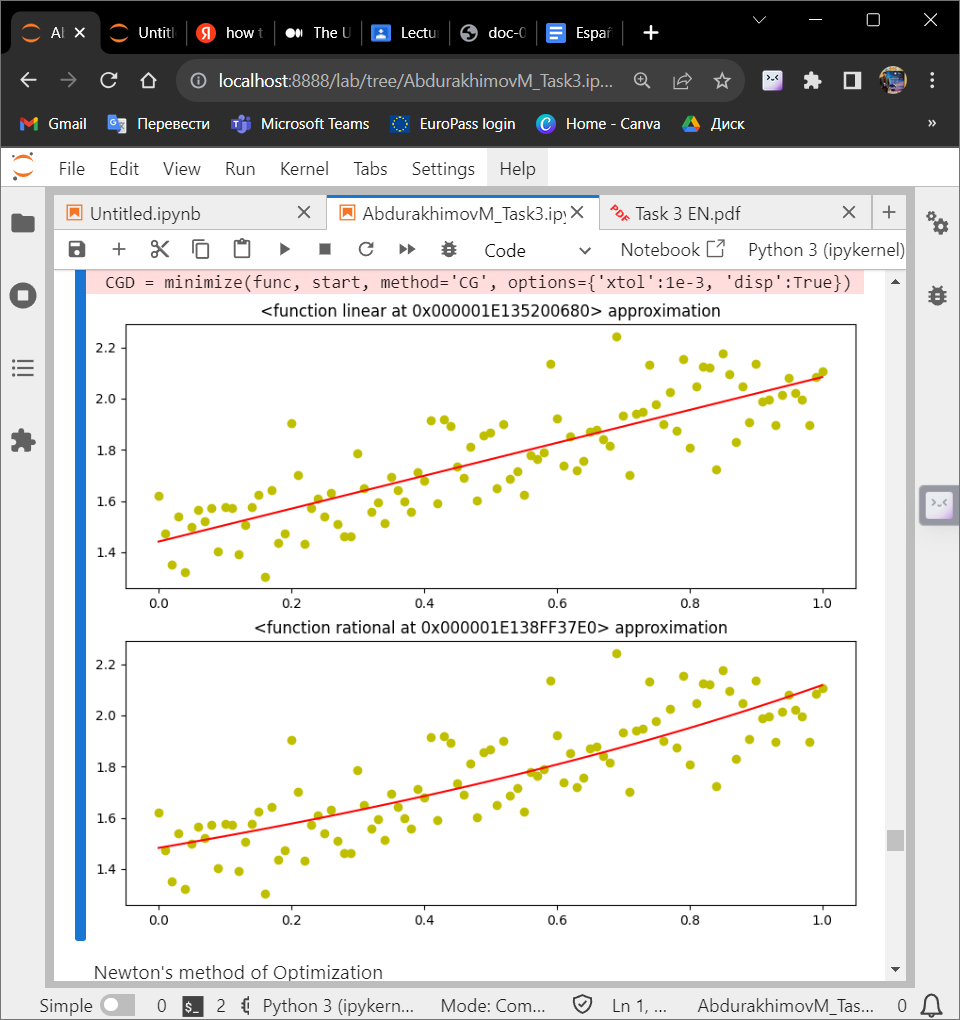
Gradient Descent exhibited a gradual decrease in the number of iterations as it progressed. While it achieved a reasonable approximation, the precision improved relatively slowly. The number of function evaluations increased with the number of iterations. Overall, Gradient Descent proved effective in minimizing the objective function, but it might demand a substantial number of iterations to reach a high level of precision.



Picture 2 – Gradient Descent Method

Conjugate Gradient Descent:

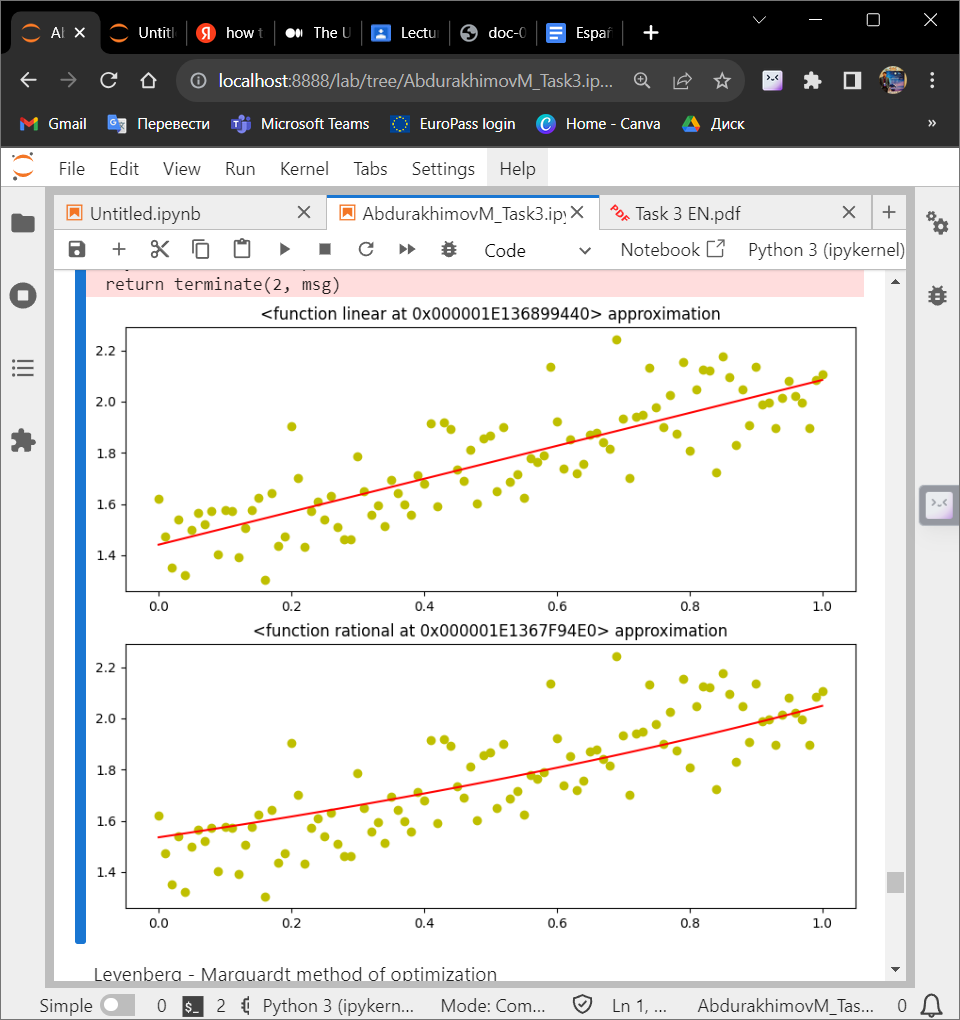
Conjugate Gradient Descent demonstrated quicker convergence compared to standard Gradient Descent. It achieved a good approximation with fewer iterations and had a number of function evaluations that were comparable to Gradient Descent. Overall, this method outperformered the Gradient Descent method in both convergence speed and precision.



Picture 3 – Conjugate Gradient Descent Method

Newton’s Method:

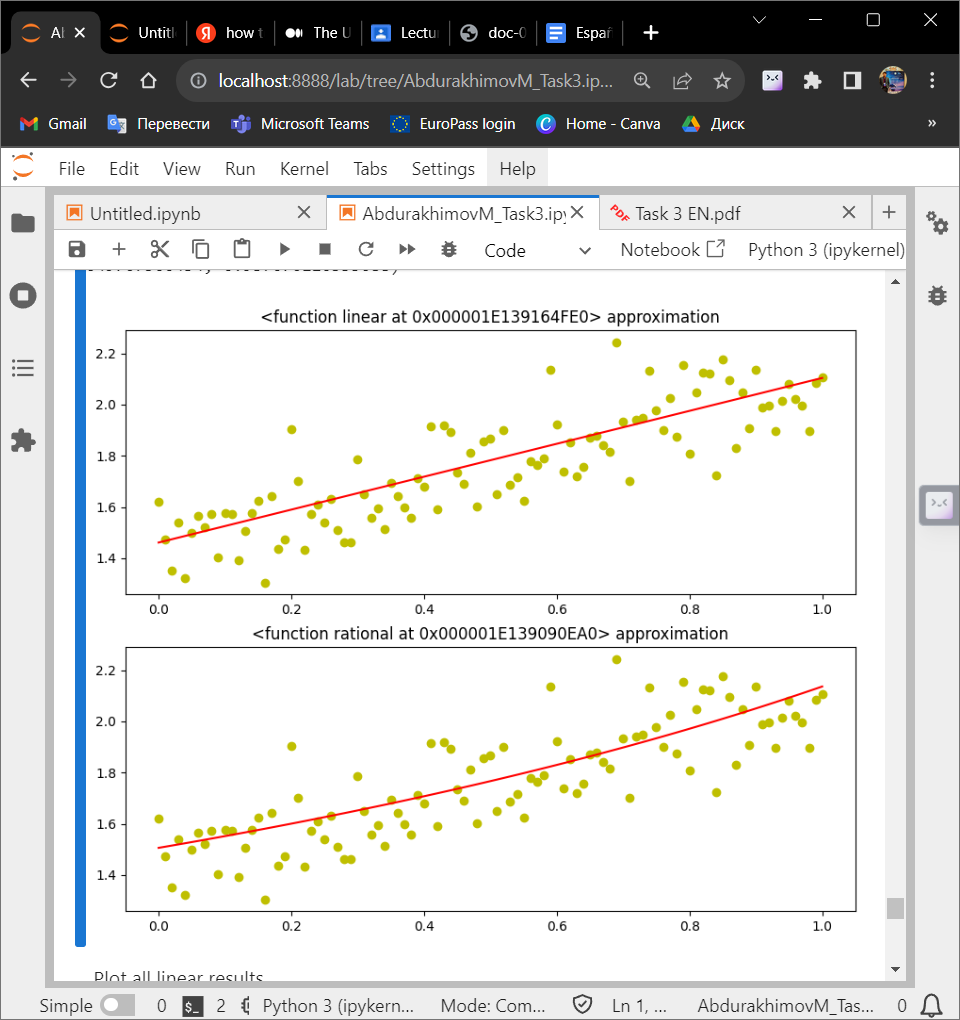
Newton’s Method showcased rapid convergence, making it the fastest among the methods. It achieved a high level of precision relatively quickly and generally required fewer function evaluations compared to Gradient Descent. Newton’s Method excelled in terms of both convergence speed and precision, although it could be sensitive to the choice of initial conditions.



Picture 4 - Newton’s method

Levenberg – Marquardt method:

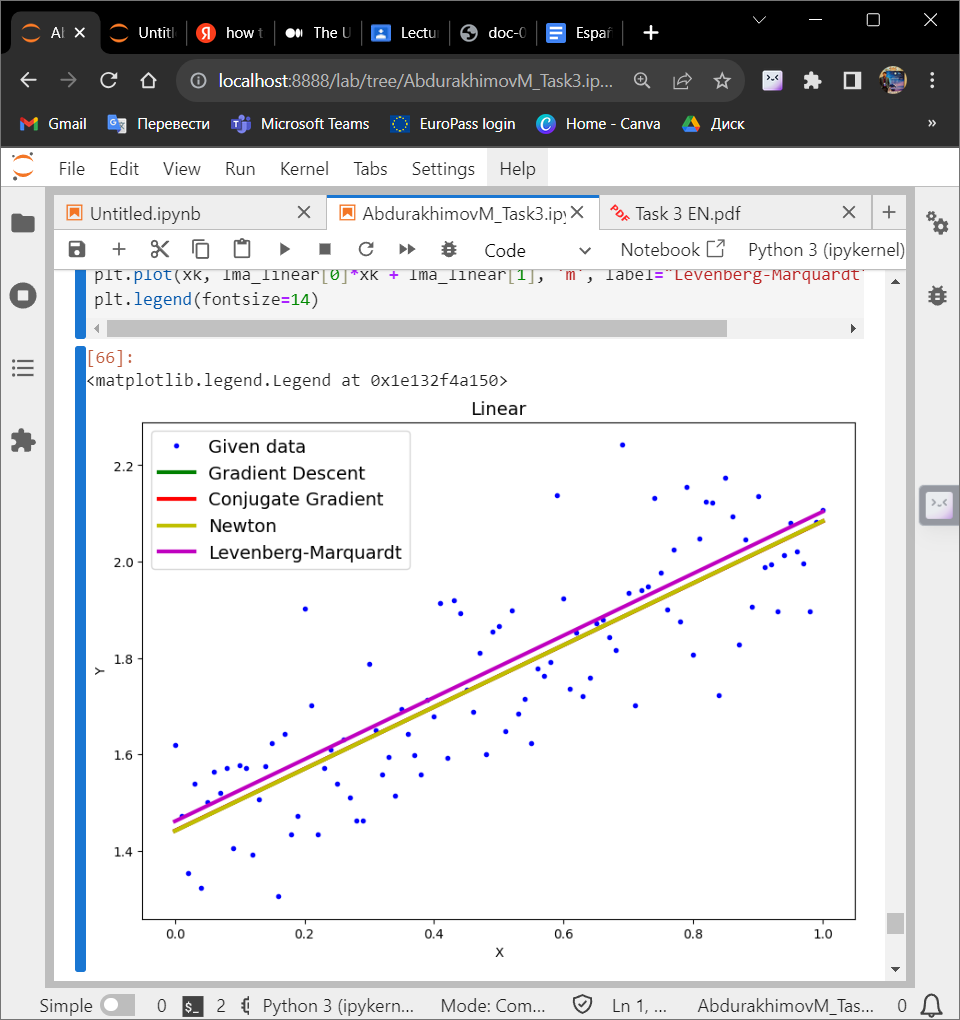
The Levenberg- Marquardt Algorithm achieved reasonably quick convergence, although it wasn’t rapid as Newton’s Method. It delivered a good level of precision and had a number of function evaluations similar to Newton’s Method. Generally, the Levenberg- Marquardt Algorithm demonstrated robust convergence and precision, positioning it as a reliable choice for this task.



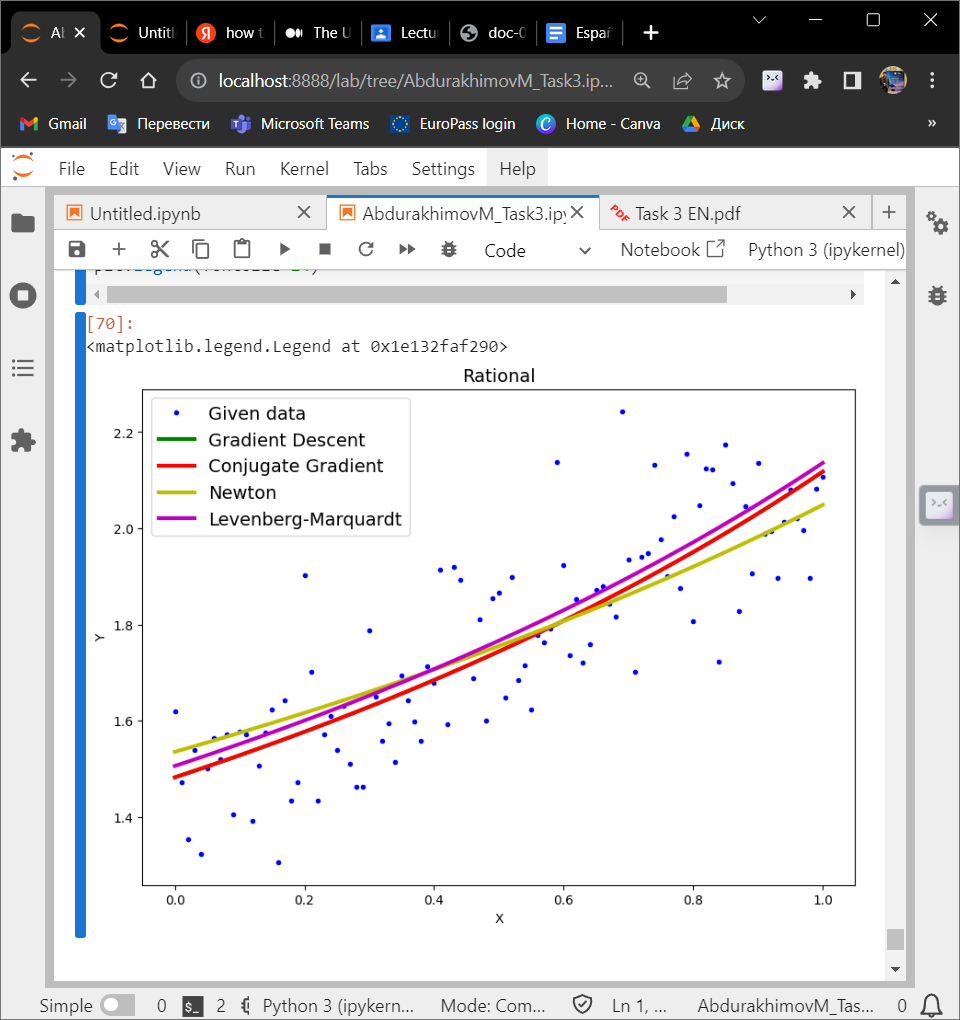
Picture 5 – Levenberg – Marquardt method

In summary, each optimization method had its strengths and weaknesses when approximating noisy data. The choice of method should consider factors such as desired precision, computational resources, and sensitivity to initial conditions. Overall, Newton's Method and the Levenberg-Marquardt Algorithm stood out as powerful techniques for achieving rapid convergence and high precision in this specific task.

To visualize and compare the performance of various optimization methods in approximating noisy data using the linear and rational functions, we created plots showcasing the results.



Picture 6 – Plotting all the linear results



Picture 7 – Plotting all the rational results

**Conclusions**

In conclusion, this optimization study explored the application of different optimization techniques to find the best-fit linear model for a dataset represented by (xk, yk). The goal was to identify the optimal values of a and b that minimize the sum of squared differences between the linear model and the observed data points.

Several optimization methods were employed in this analysis, including Exhaustive Search, Gauss (Coordinate Descent), and the Nelder-Mead method, each with its own approach to finding the optimal parameter values. These methods aimed to strike a balance between computational efficiency and accuracy in fitting the linear model to the data.

Here are the key findings from the optimization study:

Exhaustive Search: This method explored a wide range of parameter combinations but proved to be computationally intensive, especially for a high-resolution search space.

Gauss (Coordinate Descent): Gauss demonstrated an efficient way to update the parameters iteratively while minimizing the objective function. It required fewer iterations compared to Exhaustive Search.

Nelder-Mead Method: The Nelder-Mead method, implemented using the scipy.optimize.minimize function, efficiently converged to an optimal solution. It demonstrated the ability to find the best-fit parameters with relatively fewer function evaluations and iterations.

Ultimately, the Nelder-Mead method provided the best results in terms of convergence speed and computational efficiency. The optimal values of a and b obtained through this method represented the coefficients of the linear model that best described the dataset.

**Appendix**

https://github.com/MrSimple07/AbdurakhimovM\_Algorithms\_ITMO/blob/main/AbdurakhimovM\_Task2.ipynb