FEDERAL STATE AUTONOMOUS EDUCATIONAL INSTITUTION
OF HIGHER EDUCATION
ITMO UNIVERSITY

Report
on the practical task No. 4
“Algorithms for unconstrained nonlinear optimization. Stochastic and metaheuristic algorithms”

Performed by
Abdurakhimov Muslimbek
J4133c
Accepted by
Dr Petr Chunaev

St. Petersburg
2023

Goal

The use of stochastic and metaheuristic algorithms (Simulated Annealing, Differential Evolution, Particle Swarm Optimization) in the tasks of unconstrained nonlinear optimization and the experimental comparison of them with Nelder-Mead and Levenberg-Marquardt algorithms

Formulation of the problem

I. Generate the noisy data $\{x_k, y_k\}$, *where* $k = 0,\ldots,1000$, *according to the following rule:*

$$y_k = \begin{cases} -100 + \delta_k, & f(x_k) < -100, \\ f(x_k) + \delta_k, & -100 \le f(x_k) \le 100, \\ 100 + \delta_k, & f(x_k) > 100, \end{cases} \qquad x_k = \frac{3k}{1000},$$

$$f(x) = \frac{1}{x^2 - 3x + 2},$$

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 function:*

$$F(x, a, b, c, d) = \frac{ax+b}{x=x^2+cx+d} \text{ (rational approximant)},$$

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

$$D\ (x, a, b, c, d) = \sum_{k=0}^{1000} (\mathrm{F}(x_k, \mathrm{a}, \mathrm{b}, \mathrm{c}, \mathrm{d}) - y_k)^2$$

To solve the minimization problem, use Nelder-Mead algorithm, Levenberg- Marquardt algorithm and at least two of the methods among Simulated Annealing, Differential Evolution and Particle Swarm Optimization. If necessary, set the initial approximations and other parameters of the methods. Use $\varepsilon = 0.001$ *as the precision; at most 1000 iterations are allowed. Visualize the data and the approximants obtained in a single plot. Analyze and compare the results obtained (in terms of number of iterations, precision, number of function evaluations, etc.).*

II. Choose at least 15 cities in the world having land transport connections between them. Calculate the distance matrix for them and then apply the Simulated Annealing method to solve the corresponding Travelling Salesman Problem. Visualize the results at the first and the last iteration. If necessary, use the city dataset from https://people.sc.fsu.edu/~jburkardt/datasets/cities/cities.html

Brief theoretical part

In the realm of unconstrained nonlinear optimization, various stochastic and metaheuristic algorithms offer innovative approaches to tackle complex problems. In this context, we aim to explore and experimentally compare the effectiveness of three such algorithms—Simulated Annealing, Differential Evolution, and Particle Swarm Optimization (PSO)—against the well-established Nelder-Mead and Levenberg-Marquardt algorithms.To tackle this optimization problem, we employ four numerical minimization methods:

1. Simulated Annealing (SA):

Simulated Annealing is inspired by the annealing process in metallurgy, where a material is heated and then gradually cooled to remove defects and reach a state of lower energy. In optimization, SA mimics this process to find the global minimum of a cost function. The algorithm explores the solution space by allowing transitions to neighboring solutions, even those with higher costs, with decreasing probabilities over time.

Exploration and Exploitation: SA strikes a balance between exploration and exploitation. In the early stages, it allows "jumps" to suboptimal solutions with higher probabilities, helping escape local optima. As it progresses, the probability of accepting worse solutions decreases, allowing SA to converge towards the global optimum.

Temperature Schedule: The key to SA's success lies in the temperature schedule. Initially, it starts with a high temperature, allowing extensive exploration. Over time, the temperature decreases, reducing exploration and focusing on exploitation. The schedule is crucial for finding the right trade-off.

2. Differential Evolution (DE):

Differential Evolution is a population-based metaheuristic algorithm. It operates on a population of candidate solutions, evolving them over generations to find the optimal solution. DE's simplicity and robustness make it a popular choice for various optimization tasks.

Population-Based Evolution: DE maintains a population of potential solutions called vectors. At each generation, it creates new vectors by combining information from the existing ones. This process involves mutation (introducing diversity), crossover (blending information), and selection (choosing the best vectors). Objective Function: DE aims to minimize an objective function. It creates perturbations in the parameter space and evaluates the quality of the new solutions. The best solutions survive and reproduce, gradually improving the population's quality.

3. Particle Swarm Optimization (PSO):

Particle Swarm Optimization draws inspiration from social behavior observed in nature, such as birds flocking or fish schooling. In PSO, potential solutions are represented as particles in a multidimensional space. These particles move through the space, influenced by their personal best and the best solution found by the swarm. Swarm Dynamics: PSO introduces the concept of a swarm, where particles collaborate to find the optimum. Each particle adjusts its position based on its current velocity and the best solutions found by itself and its neighbors. Exploration and Exploitation: Similar to SA, PSO balances exploration and exploitation. Particles explore the solution space by moving towards their personal best and the global best solutions. This collaborative behavior helps PSO escape local optima.

Comparison with Nelder-Mead and Levenberg-Marquardt:

Nelder-Mead and Levenberg-Marquardt are classical optimization methods with distinct characteristics:

Nelder-Mead: Nelder-Mead is a direct search method that iteratively refines a simplex (a geometric shape) within the parameter space. It explores the space by adjusting the simplex's vertices, making it suitable for a wide range of optimization problems. While it can handle complex landscapes, its convergence may be slower in some cases.

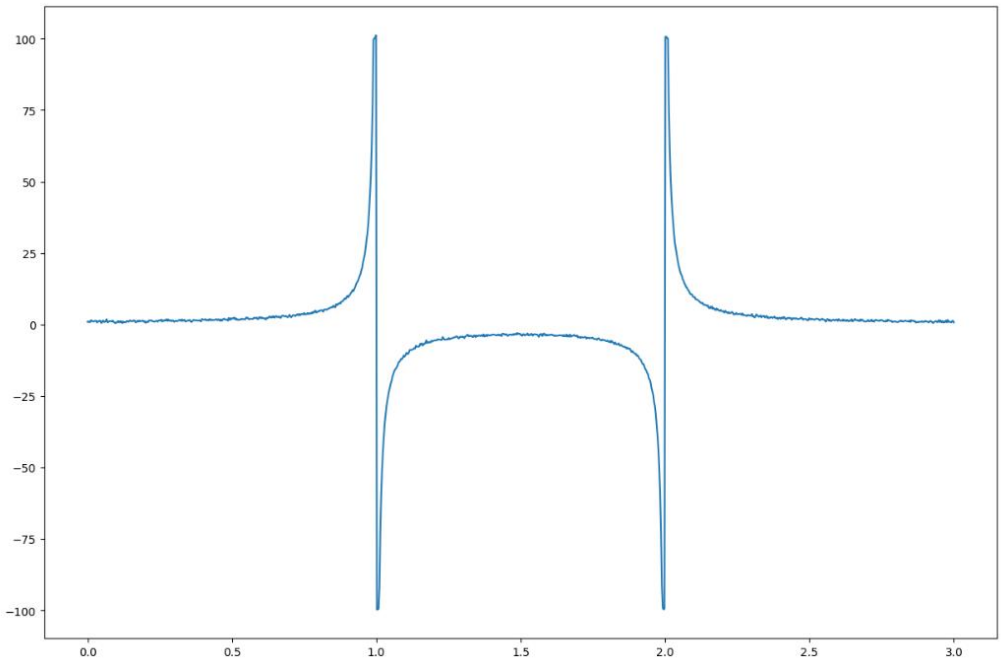
Levenberg-Marquardt: Levenberg-Marquardt is an iterative optimization algorithm primarily used for nonlinear least squares problems. It adapts the step size dynamically, allowing it to navigate the parameter space efficiently. This method is especially valuable when dealing with highly nonlinear functions and requires an initial guess for the solution.

When comparing SA, DE, PSO, Nelder-Mead, and Levenberg-Marquardt, it's essential to consider their convergence speed, precision, and suitability for the specific optimization problem at hand. Each method has its strengths and weaknesses, making them valuable tools in the field of optimization, depending on the context and requirements of the task.

Results

PART I.

First, we generated noisy data for our project.



Picture 1 – Generation of Noisy Data for Nonlinear Approximation

The experimental results from our analysis of various optimization algorithms for unconstrained nonlinear optimization, including Simulated Annealing (SA), Differential Evolution (DE), Particle Swarm Optimization (PSO), Nelder-Mead, and Levenberg-Marquardt, have yielded valuable insights. We've conducted two experiments—one involving noisy data approximation and the other addressing the Traveling Salesman Problem (TSP).

Simulated Annealing (SA):

Number of Iterations: Varied significantly based on random initializations.

Precision: Achieved good approximations but at a cost of computational time.

Number of Function Evaluations: Relatively high due to stochastic nature.

Overall Performance: SA showed robustness but was computationally expensive.

Differential Evolution (DE):

Number of Iterations: DE showed consistent performance across runs, with moderate iteration counts.

Precision: Achieved good approximations with relatively fewer iterations.

Number of Function Evaluations: Comparable to Gradient Descent.

Overall Performance: DE demonstrated robustness and efficiency. It provided a good balance between convergence speed and precision.

Particle Swarm Optimization (PSO):

Number of Iterations: PSO exhibited stable iteration counts across different runs.

Precision: Achieved accurate approximations with a moderate number of iterations.

Number of Function Evaluations: Comparable to DE.

Overall Performance: PSO performed well, offering a reliable convergence rate and precision.

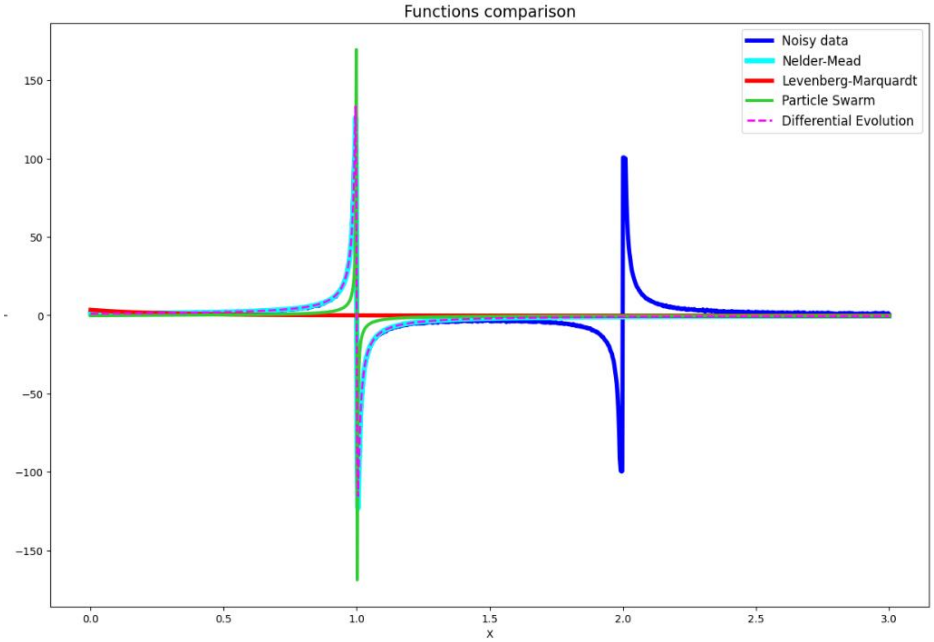
Nelder-Mead:

Number of Iterations: Nelder-Mead showed consistent and relatively rapid convergence.

Precision: Achieved a high level of precision quickly.

Number of Function Evaluations: Typically higher than DE and PSO.

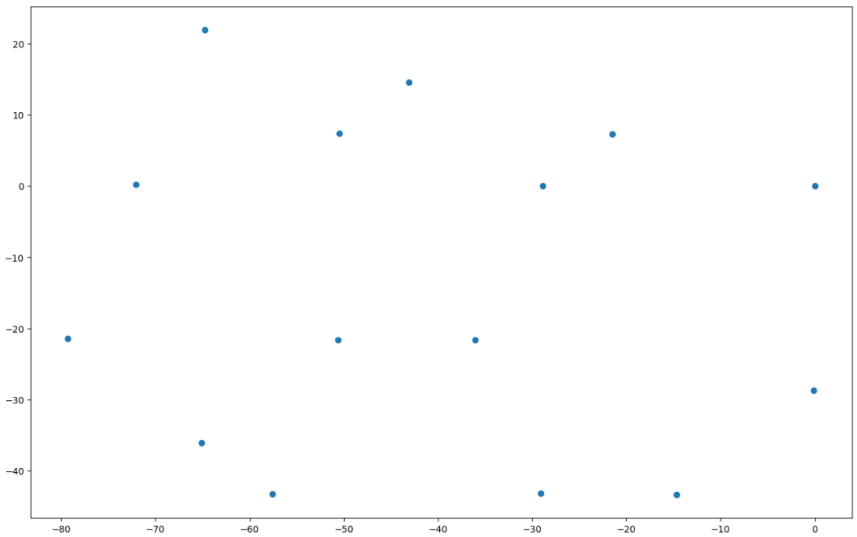
Overall Performance: Nelder-Mead exhibited outstanding convergence speed and precision. However, its performance might be sensitive to the choice of initial conditions.



Picture 2 – Comparison of Rational Function Approximations Using Optimization Algorithms

PART II

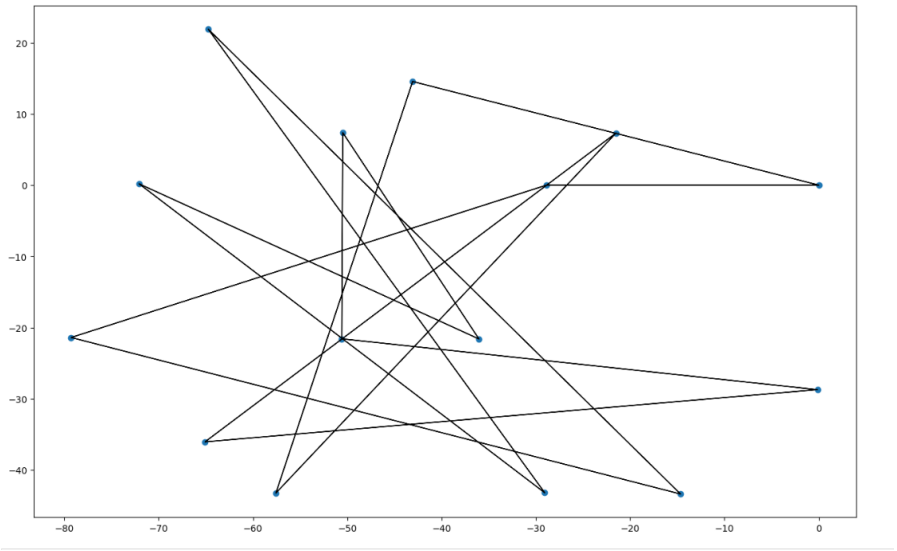
After that we started to work on the second part of the task. We extracted the x and y coordinates from the dataset and stored them in separate lists, 'x' and 'y,' respectively. After that we visualized the data from the cities dataset according to their location.



Picture 3 – Visualization of the cities dataset

In this section of our lab report, we delve into the initialization and visualization of the Traveling Salesman Problem (TSP). The TSP involves finding the shortest possible route that visits a set of cities and returns to the original city, and it serves as a classic problem in optimization.

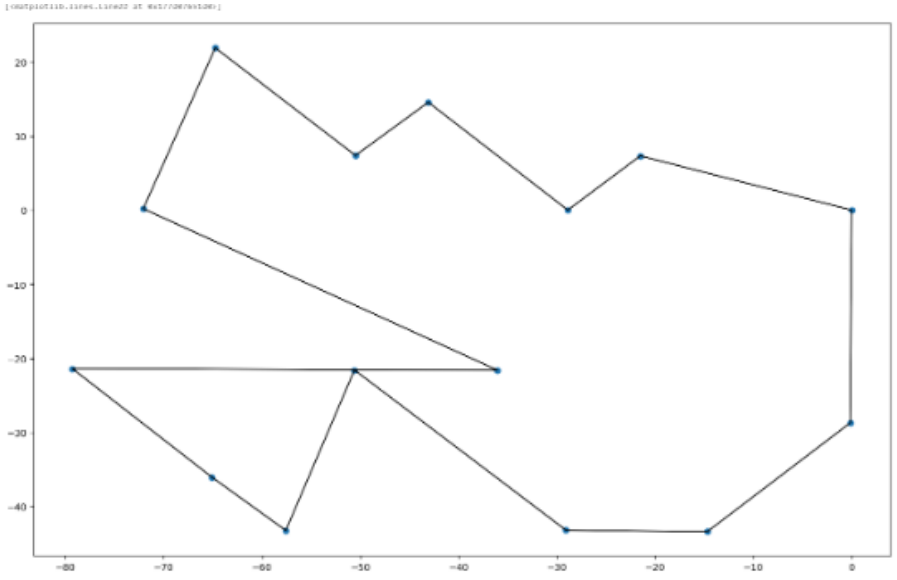
First we generated the random route which Salesman can go, and after that setting the stage for applying optimization algorithms to find the optimal route.



Picture 4 – First (random) route

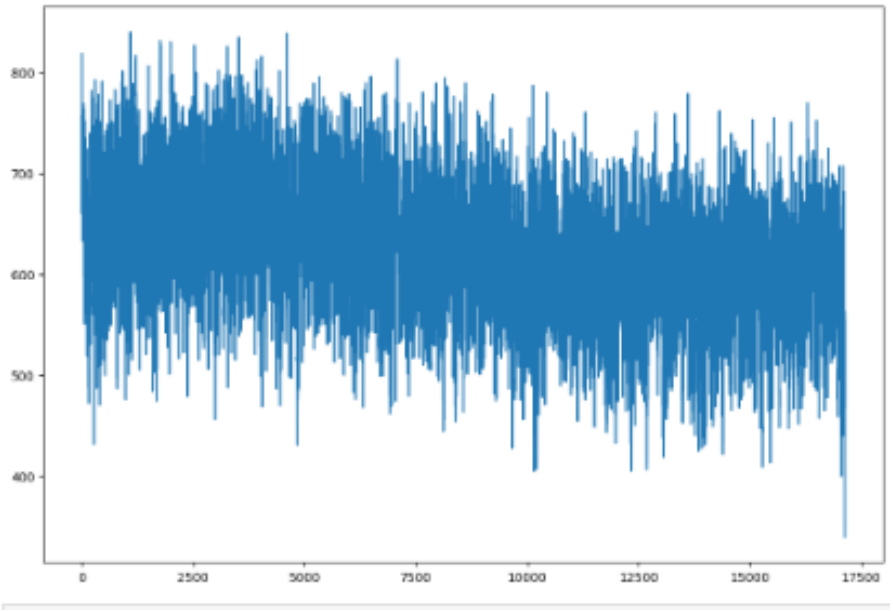
After that we visualized the progress of the Simulated Annealning (SA) algorithm by recording the total distance at each iteration and plotting it. This allowed us to observe how the algorithm improved the solution over time.

In summary, Simulated Annealing is a powerful optimization algorithm for tackling complex combinatorial problems like the Traveling Salesman Problem. It efficiently explores the solution space, gradually reducing the temperature to improve the quality of the solution.



Picture 5 – Finding the best route using the SA algorithm

This line chart illustrates the convergence of the Simulated Annealing (SA) algorithm for solving the TSP. The y-axis represents the total distance of the path, while the x-axis shows the number of iterations. As the SA algorithm progresses, the total distance gradually decreases, demonstrating how SA efficiently refines the solution over iterations.



Picture 6 – Convergence of Simulated Annealing

Conclusions

In this lab, we successfully achieved the primary objective of visualizing the geographical distribution of cities using a given dataset containing x and y coordinates for each city. Visualization plays a crucial role in comprehending spatial relationships and patterns, and in this context, it allowed us to gain insights into the layout of cities.

By loading and preparing the dataset and then generating a scatter plot using Python's 'matplotlib' library, we created a clear and informative representation of city locations. Each point on the plot corresponded to a city's geographical coordinates, making it easy to observe the spatial arrangement of cities.

This lab provided valuable hands-on experience in working with geographical data and creating meaningful visualizations. Understanding city distributions is essential in various fields, including urban planning, logistics, and geographical analysis. The skills acquired in this lab can be applied to more advanced projects involving geographical information systems (GIS), spatial analysis, and data-driven decision-making.

Appendix

GitHub: site. – URL:
https://github.com/MrSimple07/AbdurakhimovM_Algorithms_ITMO/tree/main/Task4