STDSR-2023-Assignment 2

Mirna Alnoukari - B19-RO-01

m.alnoukari@innopolis.university

Github

Task 1

In a research program on human health risk from recreational contact with water contaminated with pathogenic microbiological material, the National Institute of Water and Atmosphere (NIWA) instituted a study to determine the quality of NZ stream water at a variety of catchment types. This study is documented in McBride et al. (2002) where n = 116 one-liter water samples from sites identified as having a heavy environmental impact from birds (seagulls) and waterfowl. Out of these samples, x = 17 samples contained Giardia cysts. Let θ denote the true probability that a one-liter water sample from this type of site contains Giardia cysts.

1. The conditional distribution of X given θ is a binomial distribution with parameters n and θ. We can denote this as X ~ Bin(n, θ).

   In this case, n = 116 (the total number of one-liter water samples) and x = 17 (the number of samples containing Giardia cysts). Therefore, the distribution of X given θ is:

$$X|\theta \; Bin(116, \theta)$$

$$\frac{116!}{x!(116-x)!}\theta^x(1-\theta)^{116-x}$$

2. We are given that the prior distribution of θ follows a β distribution. Let α and β be the parameters of this beta distribution. We are also given the prior mean and standard deviation of θ as follows:

$$Prior\ mean = E(\theta) = \frac{\alpha}{\alpha+\beta} = 0.2$$

$$Prior\ standard\ deviation = \sigma = \sqrt{\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}} = 0.16$$

$$Var(\theta) = \sigma^2$$

   We can solve these equations to get two equations in two unknowns:

$$\frac{\alpha}{\alpha+\beta} = 0.2$$

$$\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} = 0.16^2$$

   Solving this system of equations we get:

$$\alpha = 1, \beta = 4$$

(rounded to the nearest integer).

3. Using Bayes' theorem and the prior and conditional distributions derived above, we can find the posterior distribution of θ:

$$h(\theta \mid X) \propto f(X \mid \theta) g(\theta)$$
$$\propto \theta^x (1-\theta)^{n-x} \theta^{\alpha-1} (1-\theta)^{\beta-1}$$
$$\propto \theta^{x+\alpha-1} (1-\theta)^{n-x+\beta-1}$$

Thus, the posterior distribution of θ is a Beta distribution with parameters x + α and n - x + β:

$$h(\theta \mid X) \sim Beta(x+\alpha, n-x+\beta) h(\theta \mid X) \sim Beta(x+\alpha, n-x+\beta)$$

Substituting the values of n, x, α, and β, we get

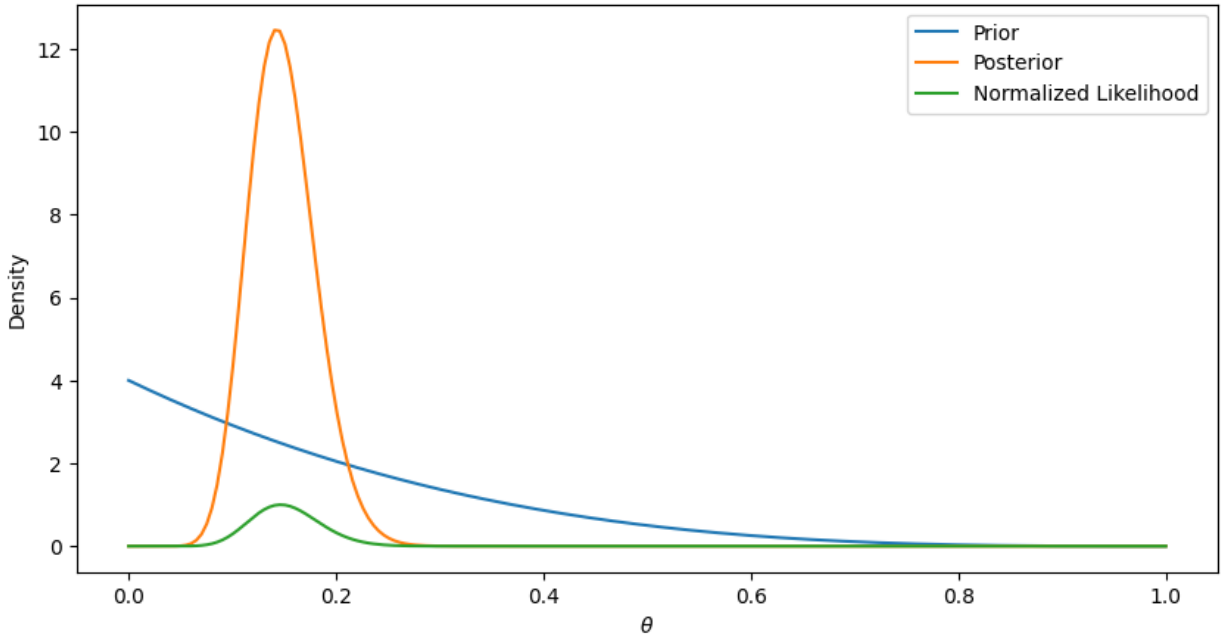
$$h(\theta|X)\ Beta(18, 103)$$

The posterior mean and standard deviation can be calculated as follows:

$$\mu = n + \alpha + \beta x + \alpha = 0.15$$
$$\sigma = (n+\alpha+\beta)^2 (n+\alpha+\beta+1)(x+\alpha)(n-x+\beta) \approx 0.03$$

Therefore, the posterior mean of θ is 0.15 and the posterior standard deviation is approximately 0.03.

4. We can see that the prior distribution has most of its mass between 0 and 0.5. The likelihood distribution is centered around 0.15, which is the maximum likelihood estimate of theta given the data, and has a narrow spread due to the relatively large sample size. The posterior distribution is a combination of the prior and the likelihood, and is centered around 0.16, which is closer to the likelihood than the prior. The posterior distribution is also narrower than the prior, indicating that the data has provided additional information about the parameter.



5. To compute the probability that θ < 0.1 we need to compute the integral on an interval (0, 0.1) of the posterior function:

$$P(\theta < 0.1) = \int_0^{0.1} h(\theta \mid X) d\theta$$

$$P(\theta < 0.1) = \int_0^{0.1} \frac{\theta^{17}(1-\theta)^{102}}{B(18, 103)} d\theta$$

$$P(\theta < 0.1) \approx 0.0528$$

6. To find the central 95% posterior credible interval for θ, we need to find the values of θ for which the area under the posterior distribution is 0.95, i.e., the interval that contains 95% of the posterior probability.

   We can use the `scipy.stats.beta` module in Python to find the credible interval. Specifically, we can use the `beta.interval` function, which returns the endpoints of a credible interval for a Beta distribution based on a given probability level.

   Using this function with a probability level of 0.95 and the parameters of the posterior Beta distribution we found earlier, we get:

```
import scipy.stats as stats

alpha_post = 18
beta_post = 103

credible_interval = stats.beta.interval(0.95, alpha_post, beta_post)
print("The central 95% posterior credible interval for θ is",credible_interval)
```

Which gives:

```
The central 95% posterior credible interval for θ is (0.09138957252823,
0.21710689824337648)
```

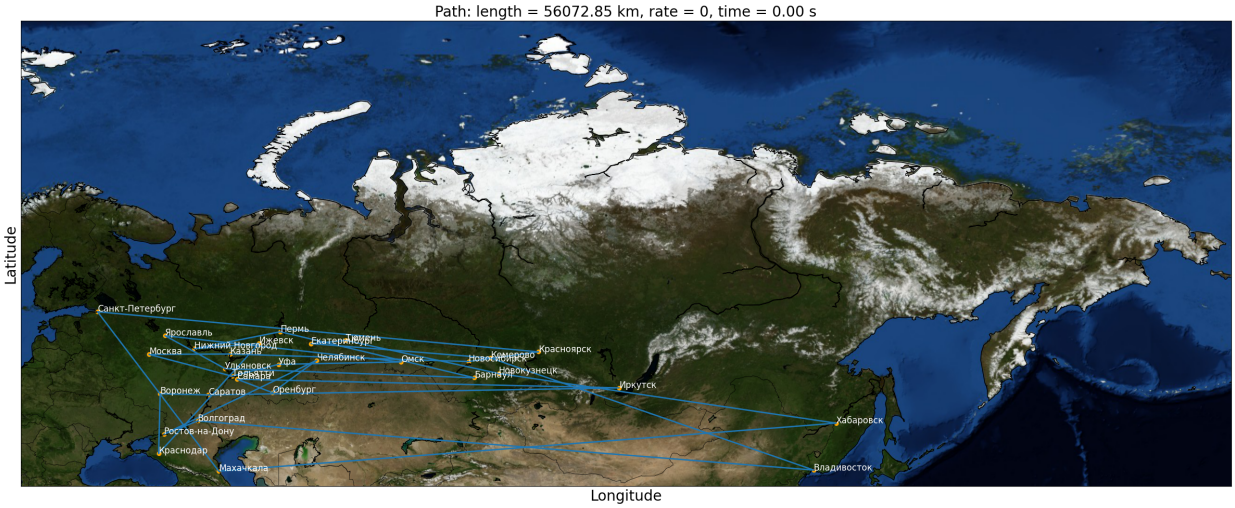
Task 2

Introduction

The goal of this task is to find the optimal path for the salesman problem for the 30 most populated cities in Russia using the Simulated Annealing (SA) algorithm. To achieve this, a CSV file containing city data is read into a pandas DataFrame, from which the 30 most populated cities are selected. The names of these cities are then extracted and stored in a list. Next, the longitude and latitude of each city are extracted from the DataFrame, and a dictionary is created that maps each city name to its coordinates. Additionally, a function to calculate the distance between two cities using their longitude and latitude is defined.

Visualizing the Data

To visualize the cities and their positions on a map, Basemap library is used, and Russia coordinates are loaded from a file. The city coordinates are plotted on the map, and the initial random path is also plotted.



Simulated Annealing

Implementing the simulated annealing optimization technique to solve the Traveling Salesman Problem (TSP).

The inputs are:

- `initial_coords`: a list of coordinates representing the cities to visit
- `start_temp`: the initial temperature for the annealing process
- `end_temp`: the final temperature for the annealing process
- `cooling_rate`: the cooling rate for the annealing process

The algorithm works by first initializing the current path as the initial path, and its length as the current path length. It also initializes an array to store all intermediate paths and their lengths for animation purposes.

It then enters a loop where it decreases the temperature gradually until it reaches the final temperature. Within the loop, the algorithm generates a new path by swapping two random cities in the current path. It then calculates the acceptance probability of the new path based on the change in the path length and the current temperature. If the random number generated is smaller than the acceptance probability, the new path is accepted and becomes the current path. Otherwise, the current path remains the same.

The temperature is then decreased according to the cooling rate, and the current path and its length are stored in the arrays for animation purposes.

Finally, the algorithm returns the arrays of all intermediate paths and their lengths, as well as an array of execution times for each step.

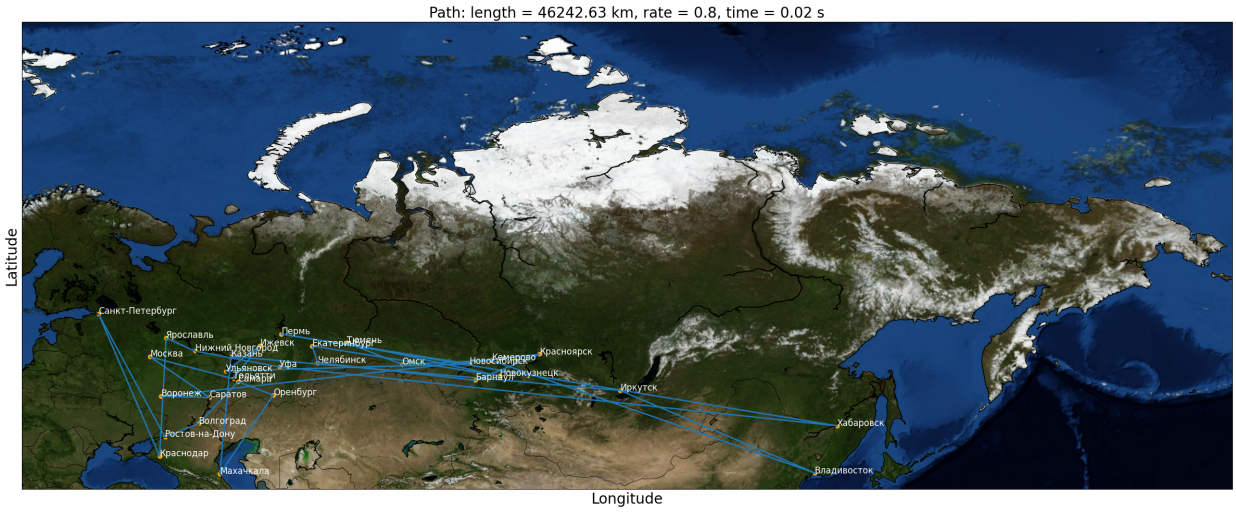
Different values of the annealing rate

We are asked to track the speed of convergence for three different values of the annealing rate.

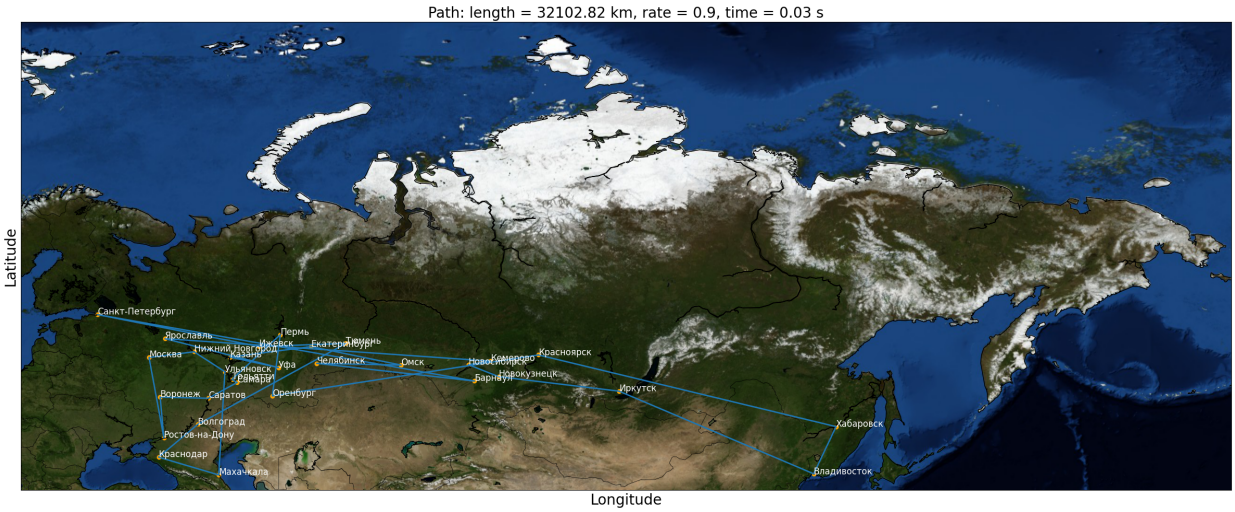
The values are:

- **Fast cooling = 0.8**
- **mid cooling = 0.9**
- **Slow cooling = 0.99**

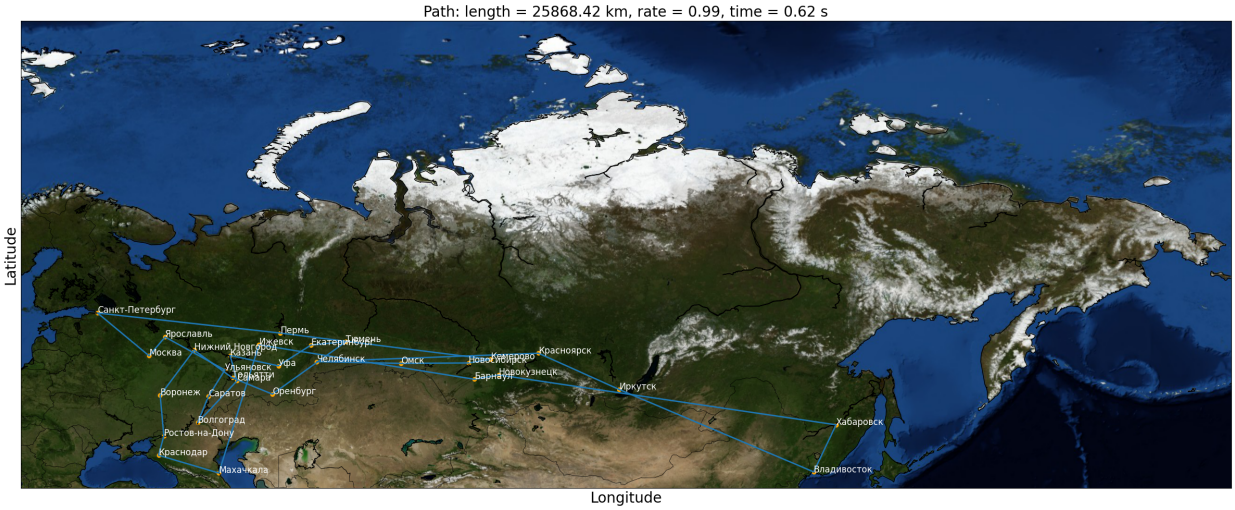
Fast Cooling



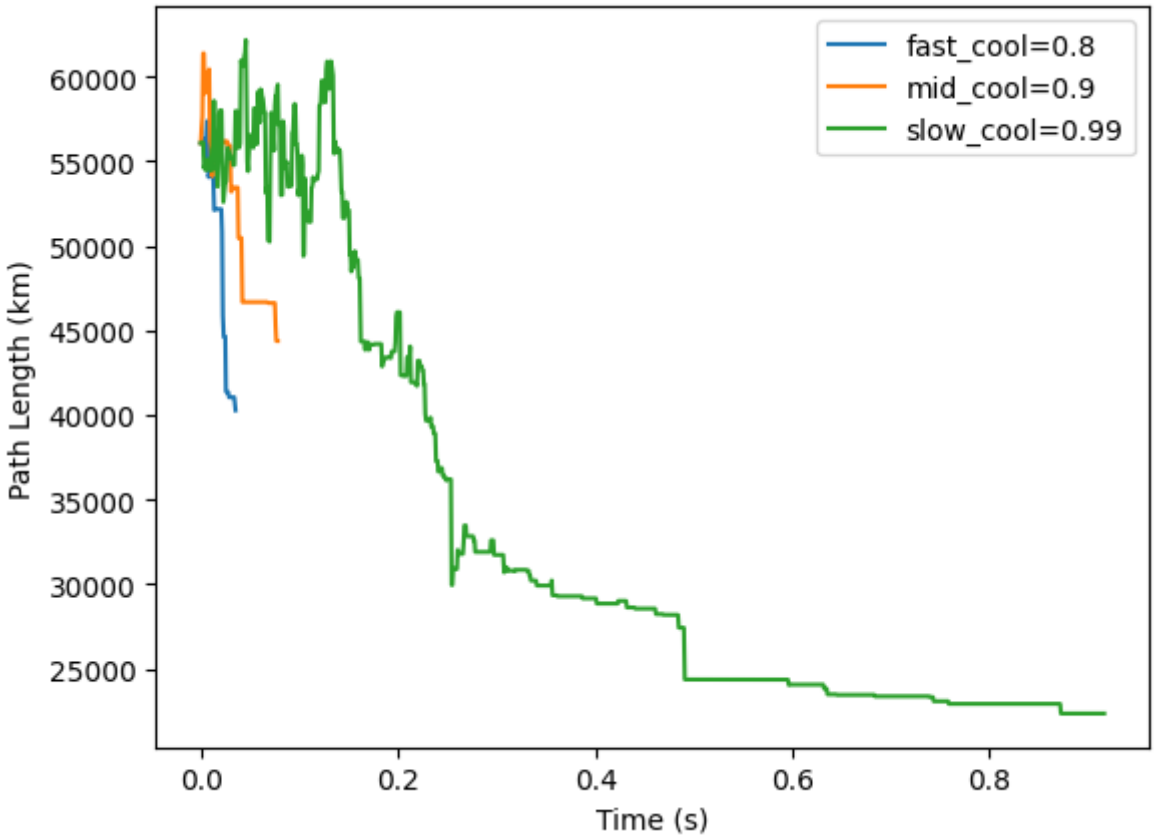
Mid Cooling



Slow Cooling



Comparing the optimization result



From the results, we can conclude that the simulated annealing algorithm's performance depends on the cooling rate. If we choose a high cooling rate, the algorithm explores a large area of the search space, resulting in a low final path length, but the algorithm's execution time is very short. On the other hand, if we choose a small cooling rate, the algorithm explores a smaller area of the search space, resulting in a higher final path length, but the algorithm's execution time is much longer.

In this case, the slow cooling rate gave the best result with the smallest final path length of 22363.01 km, but at the cost of a much longer execution time of 0.92 seconds. The fast cooling rate gave the fastest execution time of 0.04 seconds, but at the cost of a higher final path length of 40280.23 km. The middle cooling rate resulted in a final path length of 44379.65 km with an execution time of 0.08 seconds.

Overall, we can conclude that the choice of cooling rate depends on the specific problem and its requirements in terms of solution quality and runtime