Code examples and exercises: Stochastic local search for TSP

In this lab session, we will use the USA 48 city TSP problem. The optimal solution is:

opttour = [1 8 38 31 44 18 7 28 6 37 19 27 17 43 30 36 46 33 20 47 21 32 39 48 5 42 24 10 45 35 4 26 2 29 34 41 16 22 3 23 14 25 13 11 12 15 40 9]

The total distance of this optimal solution is **10628**.

Note: If you have not finished the random search for TSP problem, please continue. You will need this random search algorithm for the exercises.

Exercises

1. Evaluate performance of the **Random Search Algorithm** on the USA 48 city TSP problem. You need to execute the algorithm for 30 independent runs with the maximum number of interactions of 100. Record the results and calculate the average and standard deviation of the results.
2. Complete the code in function twoopt.m. Evaluate performance of the **Simple Hill Climbing Search Algorithm** on the USA 48 city TSP problem. You need to execute the algorithm for 30 independent runs. Record the results and calculate the average and standard deviation of the results.
3. Construct a **Simple Hill Climbing Search Algorithm with random restart**. Evaluate its performance on the USA 48 city TSP problem. You need to execute the algorithm for 30 independent runs. Record the results and calculate the average and standard deviation of the results.
4. Modify the Simple Hill Climbing Search Algorithm code to construct a **Stochastic Hill Climbing Search Algorithm**. **Hint**: instead of rejecting worse solutions (immediate neighbours), accept them with a very small probability, e.g., 0.001. You also need to specify the maximum iteration, otherwise your algorithm will never terminate.
5. Now you can try to implement the **Simulated Annealing algorithm** by introducing the annealing schedule `temperature()` into the Stochastic Hill Climbing Search Algorithm. One suggestion is a linear annealing schedule, which is defined as `temperature(t0, k, kmax) = k*t0 * (1 - k/kmax)`. You need to tune the parameters, e.g., `t0` and `kmax`.