

Solving Travelling Salesman Problem using Stochastic Local Search algorithms

Student name:Binjie Zhang

Student ID:1870958

1. Introduction

The main purpose of this assignment is to obtain the results of the Ulysses 22 cities travelling salesman problem by Simulated Annealing (SA) and Tabu search algorithms. The script “Read.m” in source code can read the tsp file and save the array in ‘.mat’ form. The dataset is “Ulysses.tsp”, the edge weights are geographical distances, and display is generated from the node coordinates. The node coordinates give the geographical latitude and longitude of the corresponding point on the earth. The script “geo_distance.m” in source code can calculate the geographic distances. In order to verify the correctness of the function, using the optimal tour to test, and the test results are correct which the optimal distance is 7013.

1.1 Simulated annealing algorithm

Simulated annealing algorithm is a generic heuristic algorithm for optimisation problems. The name and inspiration come from annealing in metallurgy, in annealing it can find a state of lower thermodynamic free energy for metal. The simulation of annealing can be used to find an approximation of a global minimum for a function with a large number of variables. The script “sa.m” in source code is the implement of simulated annealing algorithm. The script “Temperature.m” is the Temperature function. The pseudo-code is shown as Table 1.1 and the flowchart is shown as Fig 1.1.

Table 1.1 The pseudo-code of simulated annealing algorithm

Algorith: Simulated annealing algorithm

Input: Initial temperature T_0 , city_position, Max iterations k_{max}

Output: Best distance

Initial solution current_tour, current_distance

Initial “best” solution best_tour=current_tour, best_distance=current_distance

While $k < k_{max}$

 Randomly change route get new_tour, new_distance

$\Delta = \text{current_distance} - \text{new_distance}$

 if new_distance < current_distance then

 current_distance = new_distance

 current_tour = new_tour

 else if $\exp((\Delta)/\text{Temperature}(T_0, k, k_{max})) > \text{rand}()$ then

 current_distance = new_distance

 current_tour = new_tour

 end if

end if

if new_distance < best_distance then

 best_distance = new_distance

 best_tour = new_tour

end if

k+1

end

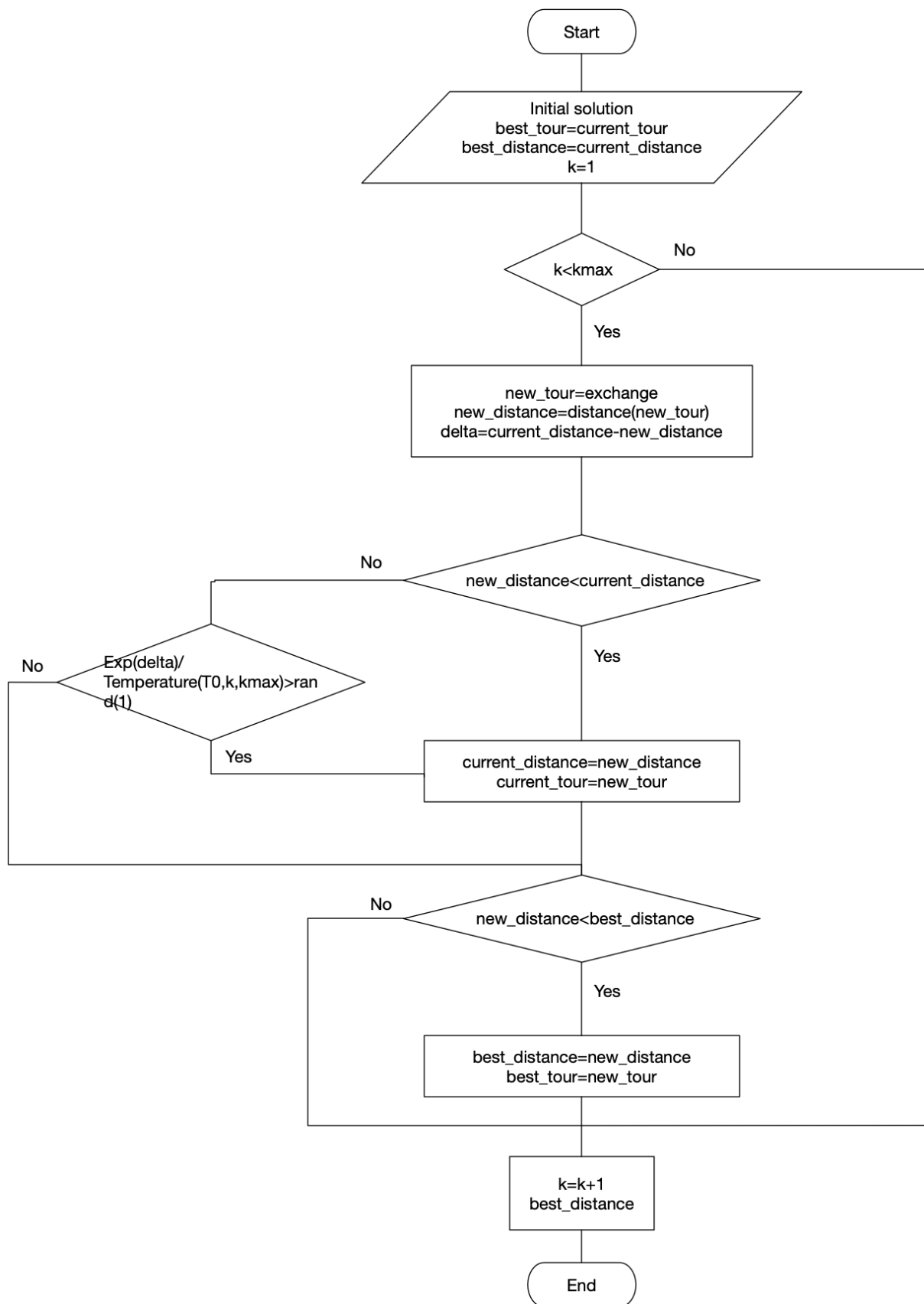


Fig 1.1 The flowchart of simulated annealing algorithm

1.2 Tabu Search algorithm

Tabu search is a metaheuristic search method employing local search methods used for mathematical optimization. ‘Tabu’ means “things that cannot be touched because they are sacred” Main idea of Tabu search is use memory to guide local search process away from local optima, first maintains a memory structure called tabu list that memorises previously visited solutions, then forbids the local search algorithm immediately return to previously visited solutions. The simplest version with the simplest Tabu list which only remembers recently visited solutions. The script “tabu_simple.m” in source code is the implement of simulated annealing algorithm. The pseudo-code is shown as Table 1.2 and the flowchart is shown as Fig 1.2.

Table 1.2 The pseudo-code of Tabu search algorithm

Algorithhe: Tabu search algorithm
Input: city_position, Max iterations
Output: Best distance
Initial solution current_tour, current_distance
Initial “best” solution best_tour=current_tour, best_distance=current_distance
Create Tabu list
While iterations < max_iter
Randomly change route get new_tour, new_distance
if new_tour not in tabu list then
if new_distance < current_distance then
current_distance = new_distance
current_tour = new_tour
put current_tour in Tabu list
end if
end if
best_distance = current_distance
iterations +1
end

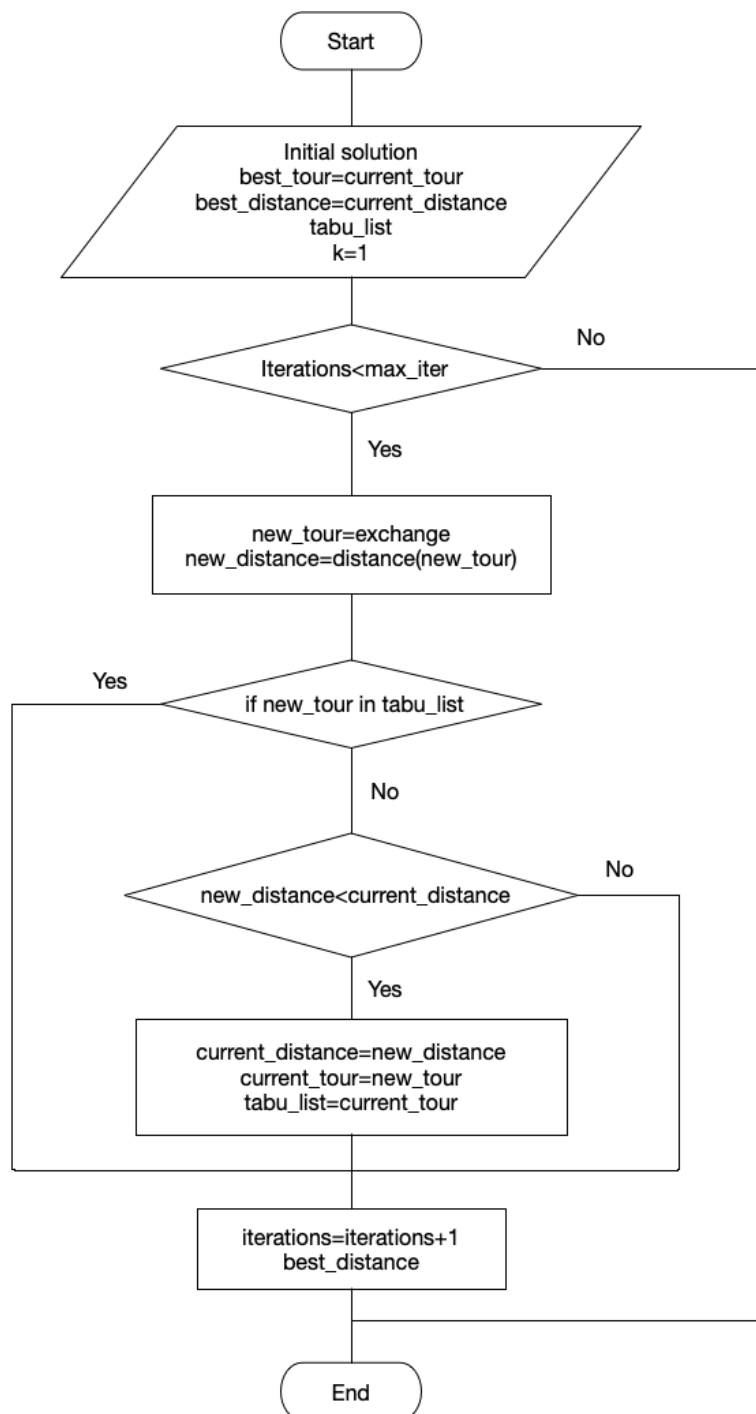
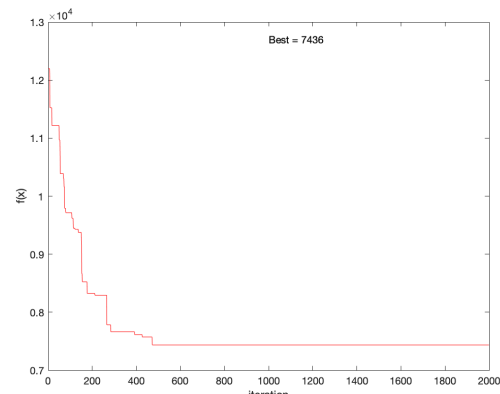
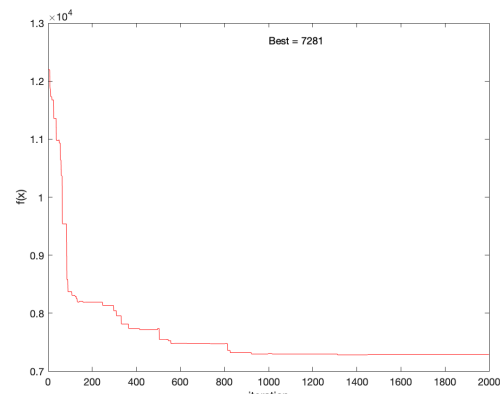
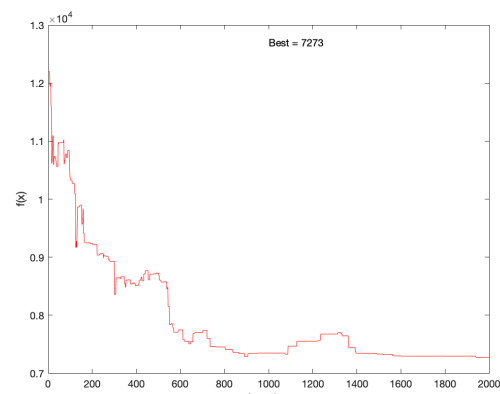


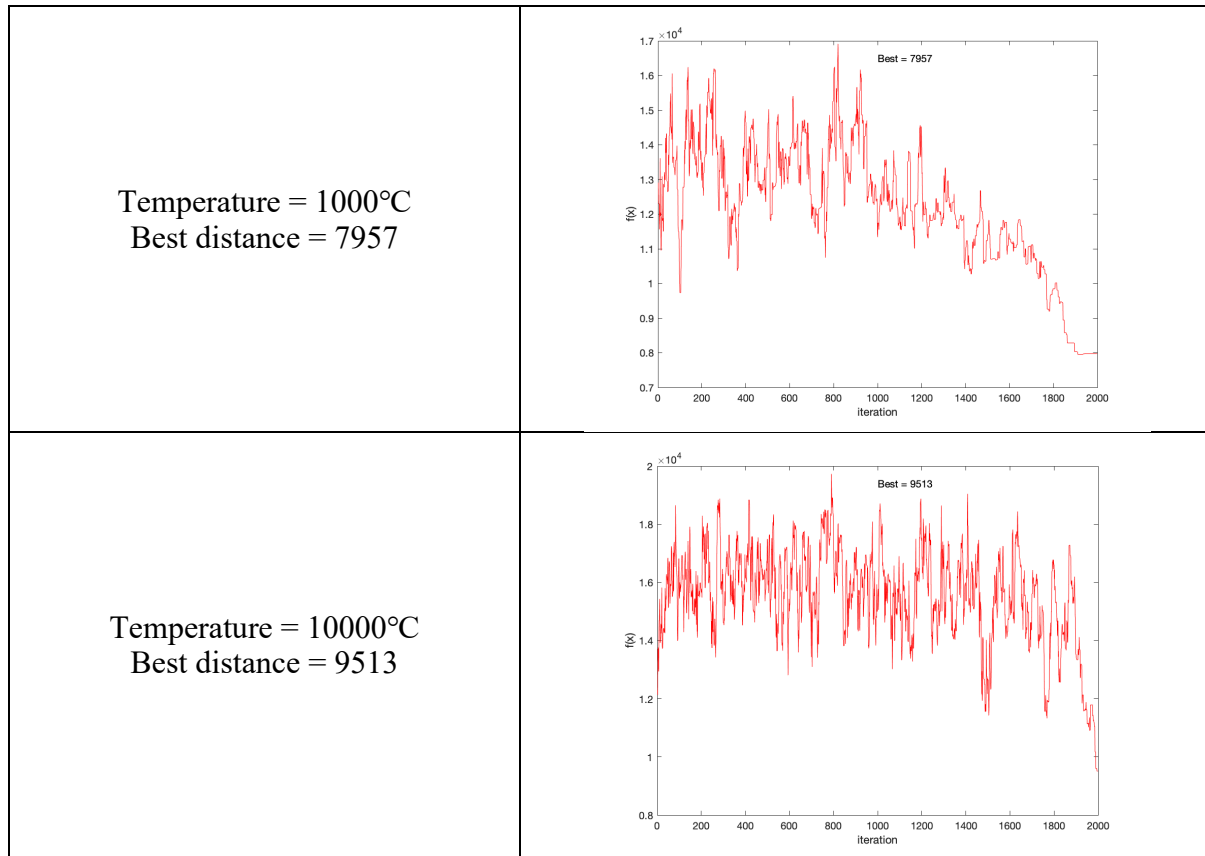
Fig 1.2 The flowchart of Tabu search algorithm

2. Parameter tuning

In the simulated annealing algorithm, the initial temperature needs to be tuned to achieve the best results. After literature review, select a temperature range of 1-10000 °C and try different orders of magnitude 1°C, 10°C, 100°C, 1000°C, 10000°C. Table 2.1 shows the best distance under different initial temperatures.

Table 2.1 The best distance under different initial temperatures

<p>Temperature = 1°C Best distance = 7436</p>	
<p>Temperature = 10°C Best distance = 7281</p>	
<p>Temperature = 100°C Best distance = 7273</p>	



Through the analysis of the results, the convergence of the cooling curve is better when the temperature range is between 10°C and 100°C, and the final result is closer to the optimal solution. Therefore, narrow the temperature range and continue to try every 10 °C between 10°C and 100°C. Table 2.2 shows the best distance under different initial temperatures.

Table 2.2 Best distance

Temperature	Best distance
10	7281
20	7384
30	7188
40	7246
50	7426
60	7200
70	7281
80	7259
90	7188
100	7273

Through the analysis of the results, 90 °C is selected as the initial temperature of simulated annealing. At this temperature, the simulated annealing algorithm may obtain a smaller difference from the optimal result, or even the optimal result.

In the Tabu search algorithm, since the simplest Tabu list has only one row, there is no parameter tune.

3. Results

After obtaining good parameters, execute 30 independent runs with 2000 iterations for each algorithm.

For the simulated annealing algorithm, the average result of 30 runs is 7239.7 and the standard deviations is 45.0248. The figure and table are shown below.

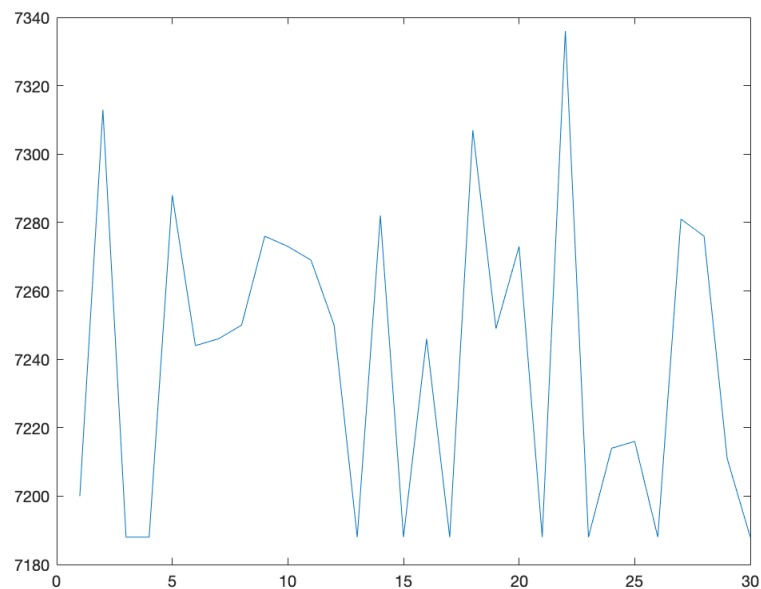


Fig 3.1 The results of simulated annealing algorithm

Table 3.1 The results of simulated annealing algorithm

Run times	Result
1	7200
2	7313
3	7188
4	7188
5	7188
6	7288
7	7244
8	7246
9	7250
10	7276
11	7273
12	7269
13	7250
14	7188
15	7282
16	7188
17	7246
18	7188
19	7307
20	7249
21	7273
22	7188

23	7336
24	7188
25	7214
26	7216
27	7188
28	7281
29	7276
30	7211

For the Tabu search algorithm, the average result of 30 runs is 7292.7 and the standard deviations is 86.0619. The figure and table are shown below.

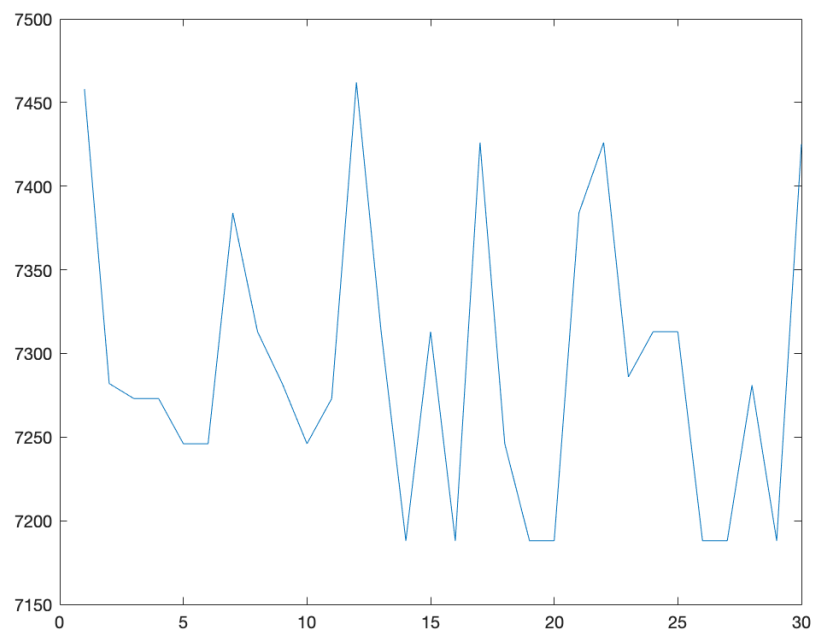


Fig 3.2 The results of Tabu search algorithm

Table 3.2 The results of Tabu search algorithm

Run times	Result
1	7458
2	7282
3	7273
4	7273
5	7246
6	7246
7	7384
8	7313
9	7282
10	7246
11	7273
12	7462
13	7313
14	7188
15	7313
16	7188

17	7426
18	7246
19	7188
20	7188
21	7384
22	7426
23	7286
24	7313
25	7313
26	7188
27	7188
28	7281
29	7188
30	7425

From the average, the results obtained by the two algorithms are not much different. From the standard deviation, the results obtained by the simulated annealing algorithm are more stable.

4. Statistical comparison

The Wilcoxon signed-rank test is a non-parametric statistical hypothesis test used to compare two related samples, matched samples, or repeated measurements on a single sample to assess whether their population mean ranks differ. MATLAB implements this test using "Wilcoxon rank sum test" as `[p,h] = signrank(x,y)` also returns a logical value indicating the test decision. The result `h = 1` indicates a rejection of the null hypothesis, and `h = 0` indicates a failure to reject the null hypothesis at the 5% significance level. The script "Wilcoxon_test.m" can do the Wilcoxon rank sum test. The result is shown below.

```
>> Wilcoxon_test(sa_distance, tabu_distance)

p =

    0.0150

h =

    logical

    1
```

At the default 5% significance level, the value `h = 1` indicates that the test rejects the null hypothesis of zero median, which means the results of simulated annealing algorithm and Tabu search algorithm do not from same set, the performance of these two algorithms is different.