Evolutionary Computation Assignment 1

Xuemeng Li

(1)-Simulated Annealing

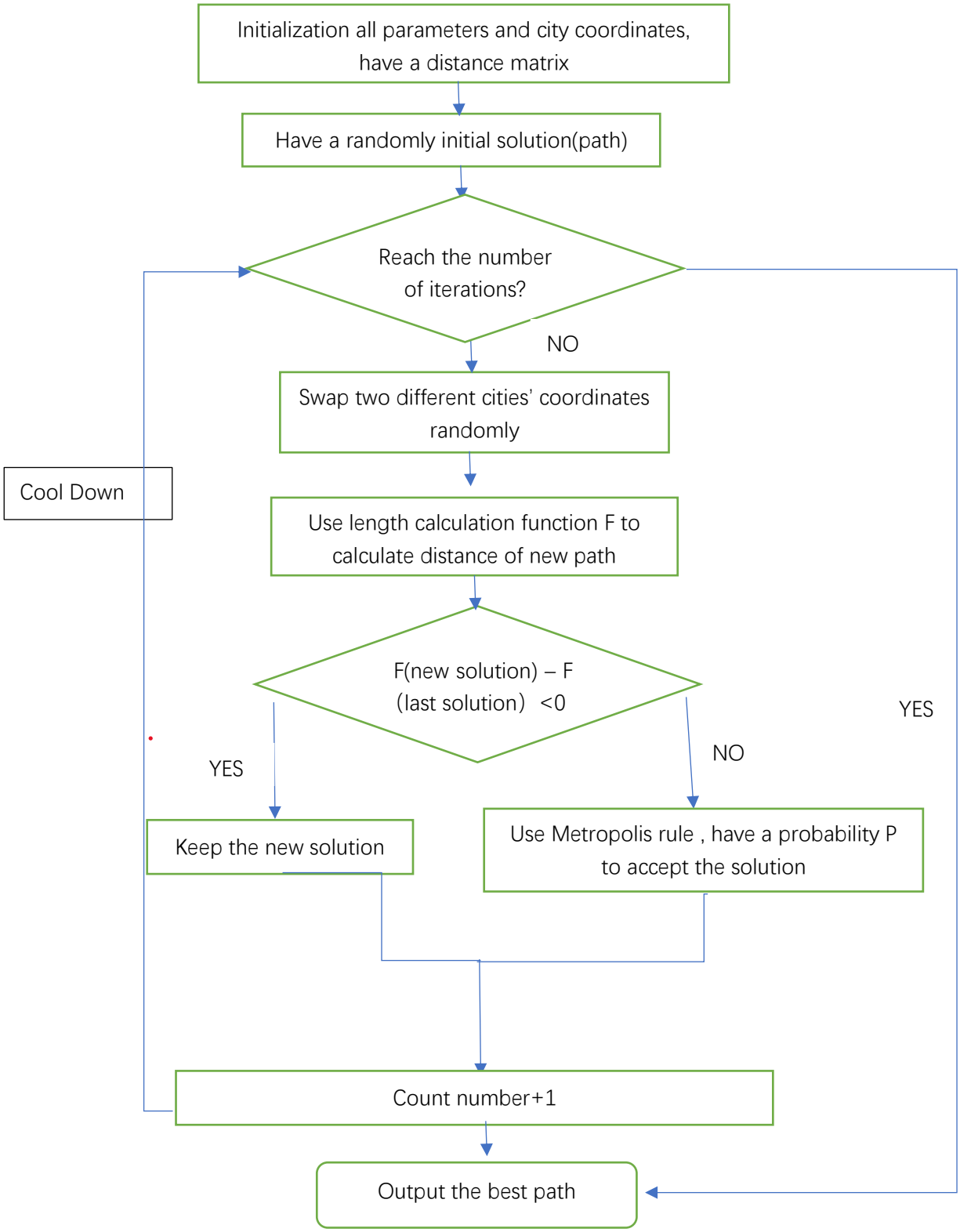
As one of the greedy algorithm and enlightenment algorithm, SA uses random search to expand the local search algorithm. At the beginning, a solution is randomly generated, and then with each cooling, the solution is updated (derived from the solid annealing process), There is a certain probability of accepting a solution with poor effect, so it is possible to jump out of this local optimal solution and achieve the global optimal solution. Finally, when the temperature drops to the minimum temperature, the solution search is stopped.

--- pseudo-code for SA:

```
/*
* J(y)：the value of the merit function at state y
* Y(i)：current state
* Y(i+1)：new state
* r：  control the speed of cooling
* T：  initial temperature(should be high)
* T_min ：The lower limit of the temperature, if T reaches T_min, stop the search(not needed for this assignment)
*/

dE = J( Y(i+1) ) - J( Y(i) ) ;
if ( dE >= 0 )  //accept the solution or not
Y(i+1) = Y(i) ;
else
{
if ( exp( dE/T ) > random( 0 , 1 ) )
Y(i+1) = Y(i) ;
}
T = r * T ; //cool down
 /*
*/
i ++ ;
</r
```

---The flow chart is as below (not including the steps for plotting figures):


Initialization all parameters and city coordinates, have a distance matrix
Have a randomly initial solution(path)
Reach the number of iterations?
NO
YES
Swap two different cities' coordinates randomly
Cool Down
Use length calculation function F to calculate distance of new path
F(new solution) – F (last solution) <0
YES
NO
Keep the new solution
Use Metropolis rule , have a probability P to accept the solution
Count number+1
Output the best path

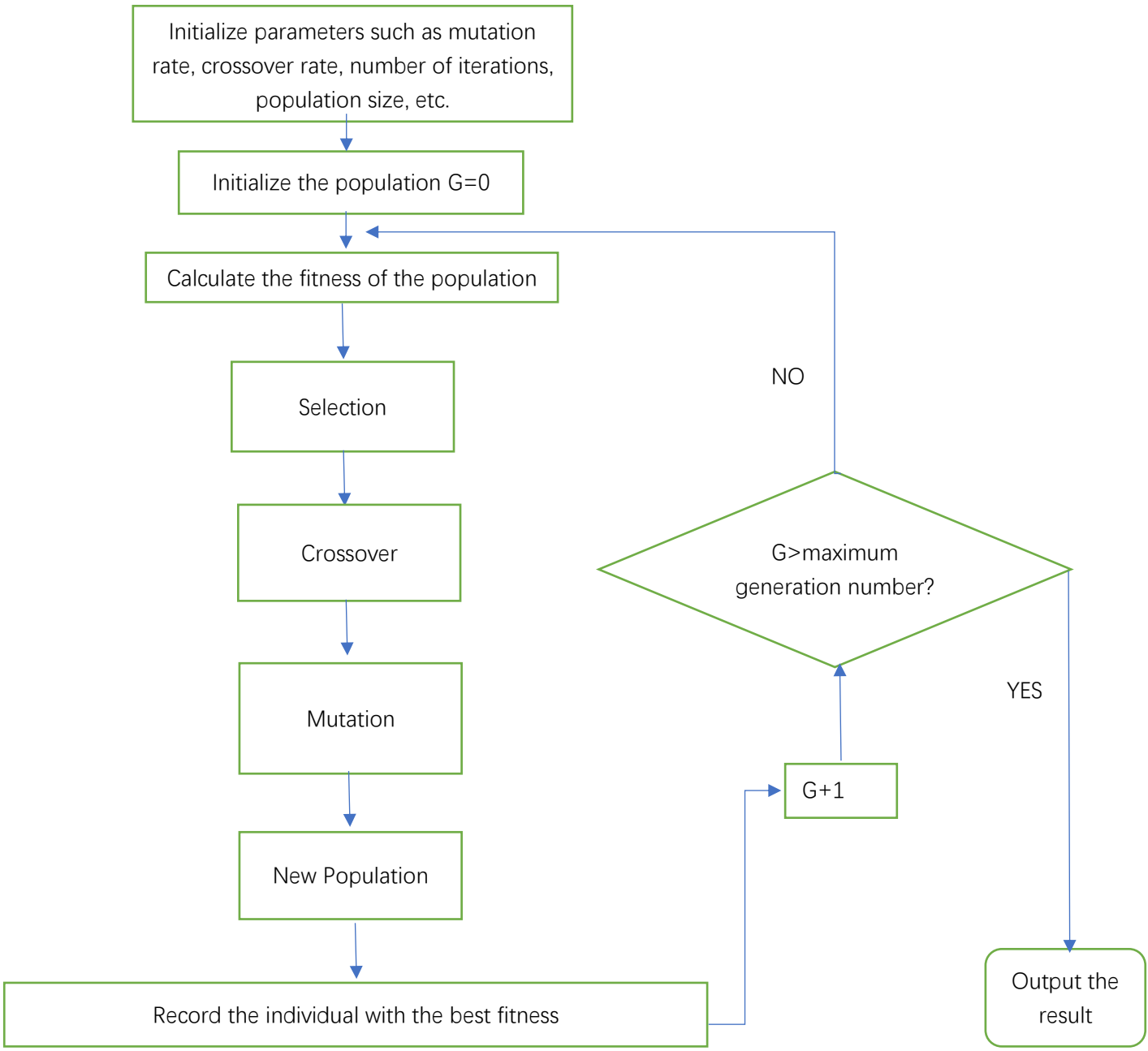
-Genetic Algorithm

GA is a method to search for the optimal solution by simulating the natural evolution process. On the basis of encoding the parameter space to be solved, the genetic algorithm searches for the population with the most fitness through the operations of selection, crossover and mutation. The optimization speed and accuracy of the genetic algorithm are related to the size of the population and the number of iterations. The larger the population, the more iterations, and the higher the optimization accuracy, but the slower the speed.

--- pseudo-code for GA:

```
/*
    initialize P(0);
    t = 0;          //generation, 1st,2nd…..
    while(t <= T) do
        for i = 1 to M  do
            Evaluate fitness of P(t);  //calculate each individual's fitness in P(t)
        end for
        for i = 1 to M  do
            Select operation to P(t);    //Select
        end for
        for i = 1 to M/2  do
            Crossover operation to P(t);   //crossover
        end for
        for i = 1 to M  do
            Mutation operation to P(t);    //mutation
        end for
        for i = 1 to M  do
            P(t+1) = P(t);       //the next population
        end for
        t = t + 1;
    end while
end
*/
```

--- The flow chart is as below (not including the steps for plotting figures):


Initialize parameters such as mutation rate, crossover rate, number of iterations, population size, etc.
Initialize the population G=0
Calculate the fitness of the population
Selection
Crossover
Mutation
New Population
Record the individual with the best fitness
G+1
G>maximum generation number?
NO
YES
Output the result

(2) Parameters:

-SA

- T //Initial temperature, which should be large enough to have iterations

The T parameter starts from 1000 and increases continuously, and chooses 5000 and 10000 respectively. As the parameter increases, the total path length shows a decreasing trend.

- L //length of the Markov chain

L just choose 10000, and the path length decreases with the increase of L.

- K // Cooling rate

The K parameter starts from 0.9 and keeps increasing, and chooses 0.85, 0.9, 0.98, etc., as the parameter increases, the total path length shows a decreasing trend. If L is too large, the probability of finding the global optimal solution may be higher, but the search process will be longer. If K is too small, the search process will be very fast, but it may eventually reach a local optimum.

- T0 //Final temperature for finishing

For SA, on my opinion, we have a fixed 10000 times interactions and the number of cities is not too much, so the temperature here is not that important, because when we have the fixed 10000 times, it still has the probability that does not reach the minimum temperature. So the outer while function is not quite important in this assignment

-GA

- popsize //size of population

The popsize starts from 40 and increases continuously, then chooses 50 and 100 respectively. As the parameter increases, the total path length shows a decreasing trend. The larger popsize will also has the larger density of points, which will lead to a better result. But more calculations are needed, and the Robustness will drop. Also, if the popsize is too small, inbreeding will obviously occur, resulting in diseased genes, which will cause congenital deletion of effective allele(a great destructive effect on the existing model).

- gnmax //number of maximum generations

popsize * gnmax = 10000

The gnmax increases continuously from 250, and selects 200 and 100 respectively (to match popsize * gnmax = 10000). As the parameter increases, the total path length shows a decreasing trend. If the gnmax is too small, the algorithm is not easy to converge (the population is not mature enough). On the other side, if the gnmax is too large, the algorithm is already proficient or the population is too precocious, it is impossible to converge more, it will only increase time expenditure.

- pc //rate of crossover

The pc starts from 0.9 and continues to decrease, and 0.8 and 0.7 are selected respectively. If the pc is too large, it is easy to destroy the existing favorable mode, and the randomness increases, then it is easy to miss the optimal individual. If the pc is too small, it will not update the population efficiently.

- pm //rate of mutation

The pm starts from 0.6 and continues to decrease, and 0.5 and 0.4 are selected respectively. For pm, less pm will result in the diversity of the population declines more quickly, each may easily lead to the rapid loss of effective genes and will not easy to be repaired. For larger pm, the probability of higher-order patterns being destroyed will increase.

(3) mean result and standard deviation

-SA

The result for 30 runs is [35688 37298 34198 34952 34480 36096 36111 35238 35076 35279 35090 37455 36067 37253 34562 36515 38073 34466 37106 34717 35400 35118 35400 37852 36556 35818 36670 34840 34442 34456],
mean:35742.37，standard deviation：1124.58。

-GA

The result for 30 runs is [43006 42346 42144 44448 44327 43374 40198 46154 38651 45879 46385 43920 48475 40811 48531 46013 46351 47422 41158 45345 46640 43401 41554 45421 44826 43751 41692 44571 45226 37704],
mean:43984.26，standard deviation：2681.72。

(4) Comparison

Result 1 - *Z*-value

The value of *z* is-4.7821. The *p*-value is < .00001.

The result is significant at $p < .05$.

Result 2 - *W*-value

The value of *W* is 0. The critical value for *W* at $N = 30$ ($p < .05$) is 137.

The result is significant at $p < .05$.

Result Details

W-value: 0
Mean Difference: -6603.6
Sum of pos. ranks: 0
Sum of neg. ranks: 465

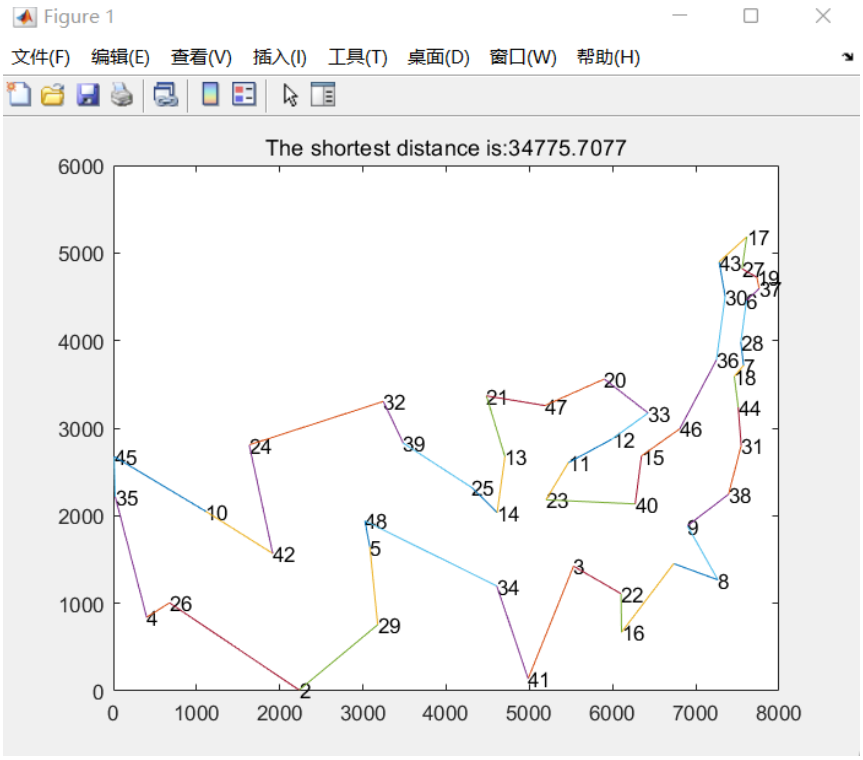
Z-value: -4.7821
Mean (*W*): 232.5
Standard Deviation (*W*): 48.62

Sample Size (*N*): 30

From the results of SA and GA, the SA has a smaller standard deviation and the GA's volatility is larger, which may be related to it does not have enough iterations and the algorithm I wrote have further steps to optimize. The Wilcoxon signed rank test was performed on the 30 results of the two, and the result p is <0.05, which means it reject the hypothesis h0(no significant difference between these two), so that we can say these two algorithms here have a significant difference.

One of the results:

-SA



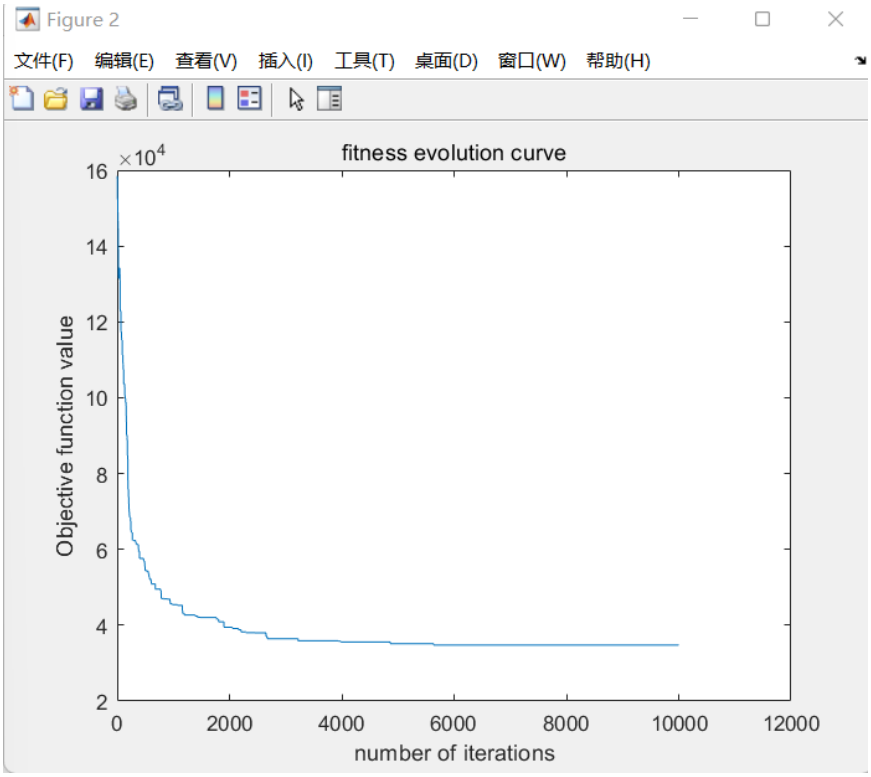


Figure 2
文件(F) 编辑(E) 查看(V) 插入(I) 工具(T) 桌面(D) 窗口(W) 帮助(H)
fitness evolution curve
×10^4
Objective function value
number of iterations
16
14
12
10
8
6
4
2
0
2000
4000
6000
8000
10000
12000

-GA

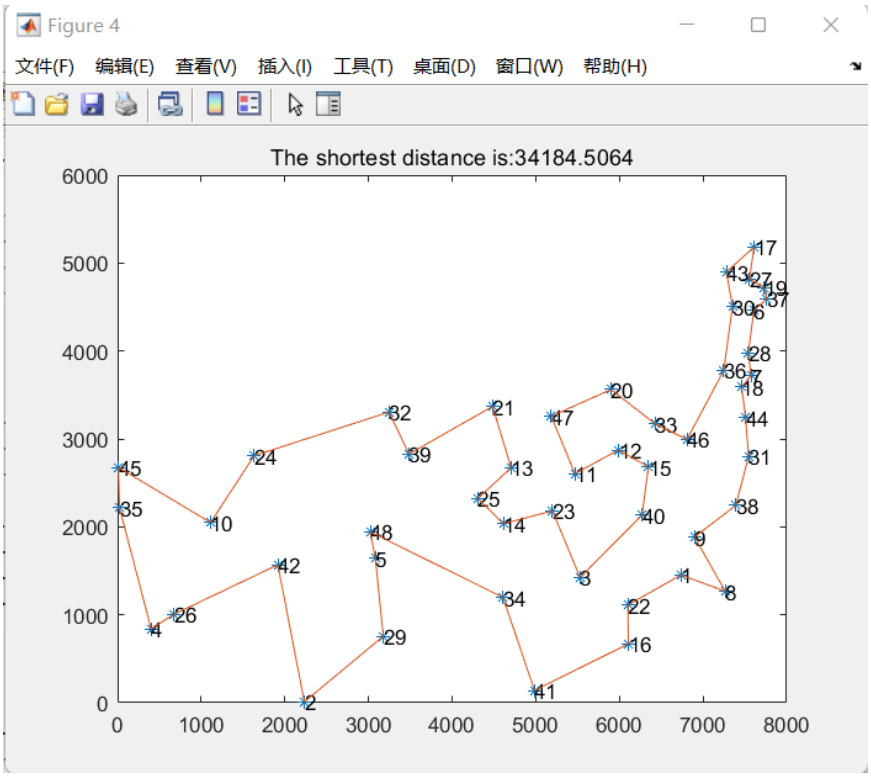


Figure 4
文件(F) 编辑(E) 查看(V) 插入(I) 工具(T) 桌面(D) 窗口(W) 帮助(H)
The shortest distance is:34184.5064
6000
5000
4000
3000
2000
1000
0
0
1000
2000
3000
4000
5000
6000
7000
8000

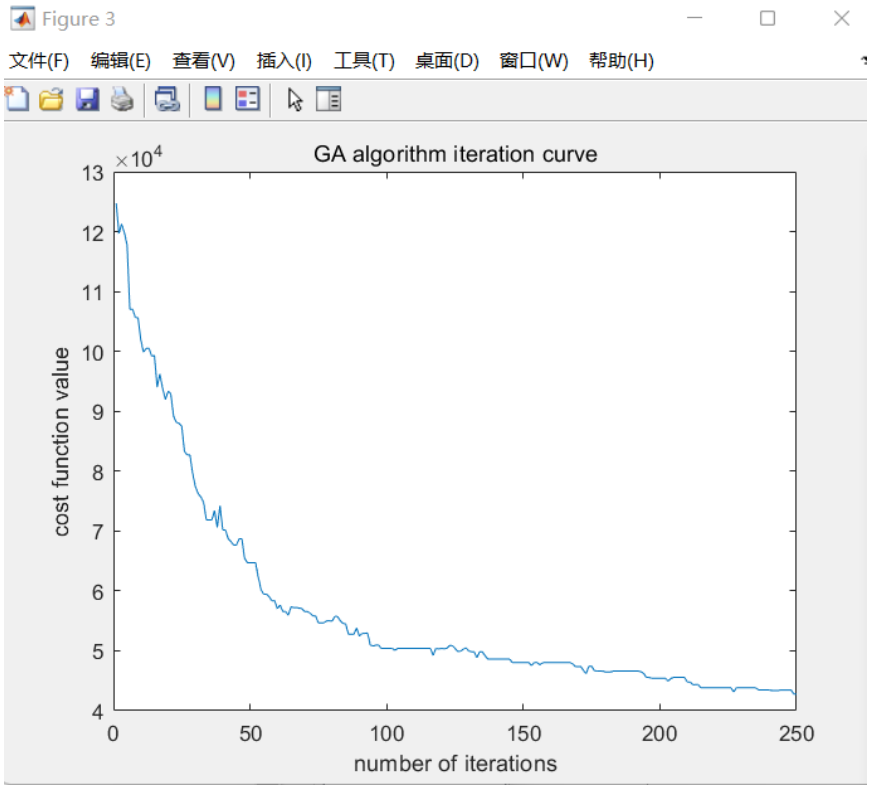


Figure 3
文件(F) 编辑(E) 查看(V) 插入(I) 工具(T) 桌面(D) 窗口(W) 帮助(H)
GA algorithm iteration curve
×10^4
13
12
11
10
9
8
7
6
5
4
cost function value
0
50
100
150
200
250
number of iterations