Data Pre-processing and Travelling Salesman Problem

Abstract

Data pre-processing focus on making scattered data range into convenient range. We apply standardization, Normalization, and Anomaly detection on a dataset. TSP in which we follow the basic steps require to achieve shortest distance.

Introduction

As AI plays key role in solving more and more complex tasks each day, our dependence on it increases with each day passing. AI helps collect and visualise data in much easier way. Additionally, AI solve complex time taking task fast. This brings us to the two task we will be focusing in the paper.

Task1 will focus on pre-processing algorithm to arrange difficult to distinguish or compare dataset to convenient range. Task1 includes pre-processing of ClimateData and convert using Data Standardization, Normalization, and Anomaly detection.

Task2 consist of Travelling salesman problem which is a very old mathematical problem.In which he must calculate the shortest route.

Pre-processing algorithms

Mostly used to convert data into knowledge. Data gathered could be sometimes be difficult to distinguish which can lead to variety of problems, e.g. impossible data combination, out of range values, etc. Furthermore, making it much difficult to examine and represent by common techniques.

In this report we have focused on three types of pre-processing techniques for process ClimateData.csv which are namely

1) Data Standardization
2) Normalization
3) Anomaly Detection

Data Standardization

Standardisation adjusts the numerical values of different attributes of collection of data into a more convenient range. Standardization is important in cases where distance measure is Euclidean, which is insensitive to different features.

In order to achieve standardization we need to use an algorithm as given

$$x_{std} = \frac{x-\mu}{\sigma}$$

Where, μ is Mean, σ is standard deviation which are not necessarily same for each attribute. Subtracting the mean value from each column and dividing the output by standard deviation.

In the given coursework task1, we do standardization of ClimateData.csv which consist of 5 attributes. Here, as first step we load the ClimateData in the Matlab then calculate the mean and standard deviation respectively .Next, applying for loop up to max attributes i.e. 5 and we do data standardization of each column. Converting the scattered numerical into a more convenient range.

Disadvantages of data standardization are its loss of uniqueness which means if you want to find a least predictable data, it loses to the most predictable data. And if you have outliers in the data set, using data standardization would not make much difference i.e. data aren't bounded unlike normalization.

Normalization

Adjusting the values which are measured on different scales to a notionally common scale is known as normalization. Normalization converts the varying numerical values on different attributes within the intervals of [0, 1].

This is an important pre-processing step, because the clustering of proximity calculations which can be distorted if not normalised.

In order to normalize a data set of varying numerical the algorithm used is as given

$$x_{norm} = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$

Where, x_{min} and x_{max} are the minimum and maximum values of each attribute column respectively. Each operation is performed for each attribute separately.

Techniques used for normalizing data should not generate noise [1]. If our data set has outliers, normalizing the data will certainly scale the data to an interval within [0, 1]. And usually most data have outliers.

In the given coursework, we normalize the given data ClimateData.csv from varying numerals to interval [0, 1].

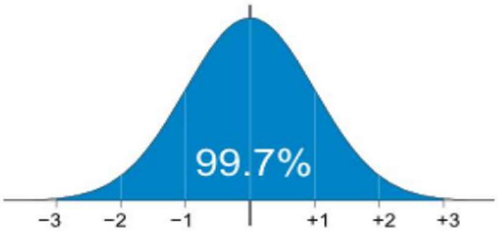
Anomaly Detection

Anomaly detection identify the values that vary greatly when compare to rest. In terms of Ai, getting high quality data is a must for training predictive and classification models.

The so-called “3sigma” rule can be used to identify anomalies, which is based on Chebyshev inequality.

If n=3, it means that vast majority of data (>8/9 in any case or >99.7 follows Gaussian distribution) are within $[-3\sigma, 3\sigma]$.

IF distributed normally (Gaussian): 99.7% data are within so called 3σ region.



In general cases, so-called Chebyshev inequality guarantees >90% of standardised data within [-3, 3].

By using standardized data, anomalies can easily be identified if

$$IF\left(x_{std} > 3\right) OR \left(x_{std} < -3\right)$$
$$THEN\left(x \, is \, anomalous\right)$$

In the given coursework, the standardized ClimateData is taken and furthermore it detects an anomaly if Xstd is within [-3, 3].

Recursive Density Estimation

RDE is based on Cauchy function which has similar properties as Gaussian, but can update recursively. This means small amount of data is required to be stored and updated. This has huge implications because theoretically an infinite data can be processed exactly in very fast in real time.

For detecting outliers in a data stream, choose a starting point with a set of features, when assuming “invariant” value which represents the normal behaviour of system. Even if “invariant” as a point is not substantially oscillatory but may vary within operating boundaries regime for real system [2].

An important stage of the task is feature extraction procedure, because the selected features represents overall density variation idea.

Principal Components Analyssis

PCA is best known to reduce the number of variables from large or massive data sets while retaining much of information in original dataset [4].

Due to PCA’s successful application in pattern recognition such as face classification because of its very effective approach of extracting features.

From the decision function it is easier to observe that PCA decision function is linear [5].

Genetic Algorithm

Genetic algorithm has been used many times as a technique for overcoming combinatorial optimization problems. Additionally, have also been effective at searching solution for large and complex state in adaptive way, led by mechanisms of reproduction, crossover and mutation [6].

Genetic algorithms are adaptive search techniques

Which are based on principles of survival of fittest and natural selection. They are easy to apply to complex problems and can effectively search problem domains.

Usually GAs are operated by iterative procedure on fixed population size or pool of candidate solution [6]. Furthermore, the fitness of the population is taken, it is the fitness of chromosome which decides the parents for the offerings who are generated through crossovers and mutations.

Here we focus on travelling salesman problem (TSP). In TSP the salesman has to visit every city once and reach the starting point. Here, the target is to find the minimum distance we can travel to reach the starting city after visiting every city exactly once.

The TSA code is focused and made up of the following sequences

1) Population

2) Fitness Score

3) Selection

4) Crossover

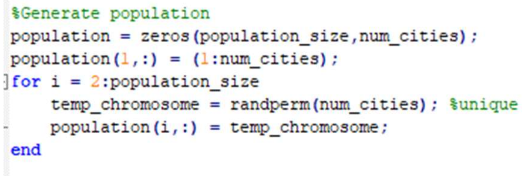
5) Mutation

Population

A Population of individuals is generated at random with an initial test population of 50 members and then increase to 200 members (start with 50 then increase to 100, 150, and 200) which runs at the start for 1000 iteration and then increases to 2000, 4000, 6000, 8000, 10000)

In order, to continue genetic analogy individuals and chromosomes are linked. And variables are analogous to gene. A chromosome composed of two or many genes. Fitness score is allotted to each chromosome based on the individual's ability to compete.

Needed to make multiple design decisions on how to implement algorithm.



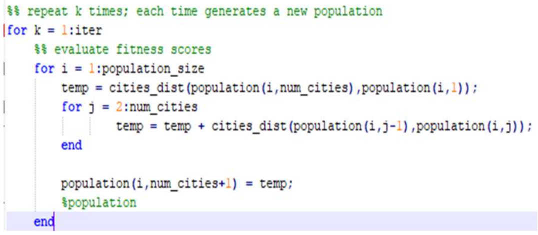
```
%Generate population
population = zeros(population_size,num_cities);
population(1,:) = (1:num_cities);
for i = 2:population_size
    temp_chromosome = randperm(num_cities); %unique
    population(i,:) = temp_chromosome;
end
```

Above in the population code snippet, num_cities = 100 which is the total number of cities on the map. Create 100 population and initialise with zeros. Now, in for loop here we generate random unique population which represents paths take by salesman.

Fitness Score

Fitness score determines which chromosome in the population has the least distance to travel in order to go through each cities and return to 1st point. Fitness calculation changes between different problems.

.



```
%% repeat k times; each time generates a new population
for k = 1:iter
    %% evaluate fitness scores
    for i = 1:population_size
        temp = cities_dist(population(i,num_cities),population(i,1));
        for j = 2:num_cities
            temp = temp + cities_dist(population(i,j-1),population(i,j));
        end

        population(i,num_cities+1) = temp;
        %population
    end
```

The first loop calculates the distance between 1st and last city and the second loop calculates the distance between every alternate cities present in the row and then the sum of both distance is the fitness score which is stored in the (end+1) column of each rows.

Selection

A population can consist of more than two individuals. In that case a certain number of chromosome must be chosen from the crossover and the selection process is based on fitness calculation.

Selection process should give preference to the chromosomes who are better than other chromosomes in the same population giving them the opportunity to pass their gene into next generation.

The first genetic operation in reproductive phase. Objective of selection is to choose the fitter individuals for next generations.

Here, in TSP we tested three selection process namely roulette, linear, tournament selection we noticed the best results was achieved with tournament selection when compare to other two selection process.

Roulette Wheel

Simple and traditional selection approach. Does selection of chromosomes based on probability. Process is done again and so on until desired number of chromosomes is/are selected.

Drawback is risk of premature convergence.

Linear rank Selection

Roulette selection has problems when fitness differs by a large value.

Drawback is it can lead to slower convergence, So that the best chromosome do not differ too much.

Tournament Selection

Tournament Selection selects the best two chromosomes from the group.

Benefits, efficient to code. Works on parallel architectures.

Drawbacks, not very useful when used in large population.

Crossover

Also known as recombination. After selection chromosomes are recombined. Crossover is performed with a high probability of 0.8;

In TSP, out of K-point crossover, Uniform crossover, Uniform Order based Crossover, Order based. We using Order-based because K-point and uniform crossover are not well suited for search based problems with permutation codes such as used in TSP.

Also Order-based crossover is considered as a variation of uniform Order based crossover.

In the ordered crossover, offspring1 inherits gene between two point from parents 1 and the rest from parents 2 from point2 and fill the offspring with elements present in paqrents2 in sequence.

Mutation

Mutation is designed to overcome problem in order to add diversity to population and make sure that it’s possible to search space.

Mutation can prevent problems with local minima. And mutation is performed in low probability i.e. 0.2.

After several crossover attempts when you expect no solution then mutation comes to play.

In the TSP code, we have used a mix of flip and slide mutation in order to achieve. As use of one mutation type did not yield the desired output.

Conclusion

My objective was to get the normalization, standardization, anomaly detection of the provided data ClimateData.csv .And find the least distance travelled by salesperson for genetic algorithm to solve travelling salesman problems.

Resources

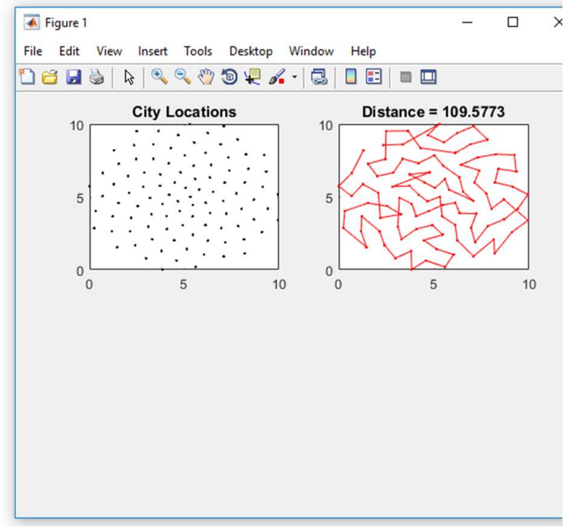
1) Al Shalabi, L. & Shaaban, Z., 2006. Normalization as a Preprocessing Engine for Data Mining and the Approach of Preference Matrix. Dependability of Computer Systems, 2006. DepCos-RELCOMEX '06. International Conference on, pp.207–214.
2) Costa, B.S.J., Angelov, P.P. & Guedes, L.A. J Control Autom Electr Syst (2014) 25: 428. https://doi.org/10.1007/s40313-014-0128-4
3) Jolliffe I. (2011) Principal Component Analysis. In: Lovric M. (eds) International Encyclopedia of Statistical Science. Springer, Berlin, Heidelberg
4) Jolliffe, I.T., 2002. Principal component analysis 2nd ed., New York: Springer.

5) Chen & Zhu, 2004. Subpattern-based principle component analysis. Pattern Recognition, 37(5), pp.1081–1083.

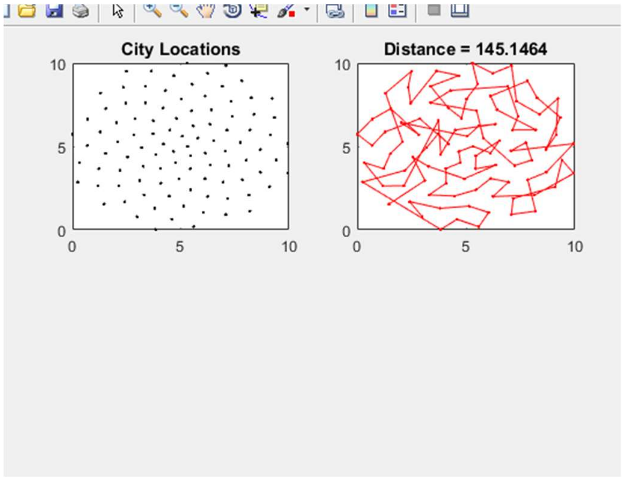
6) Katayama, Sakamoto & Narihisa, 2000. The efficiency of hybrid mutation genetic algorithm for the travelling salesman problem. Mathematical and Computer Modelling, 31(10), pp.197–203.

APPENDIX

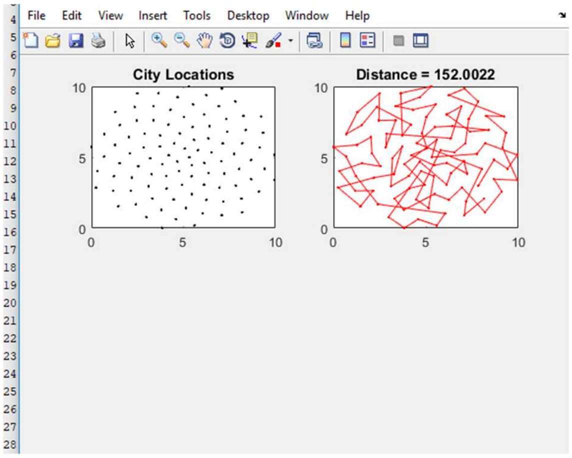
Using Roulette Selection



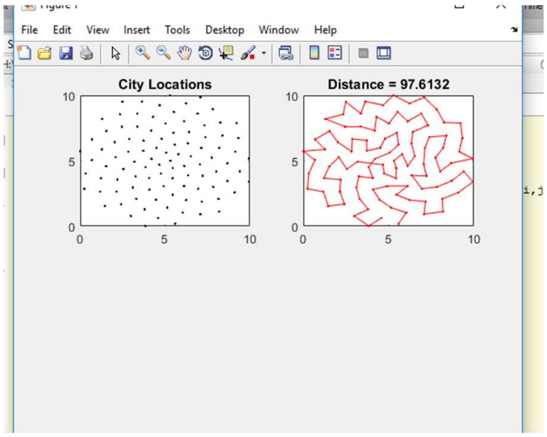
Result with population 50, iteration 1000



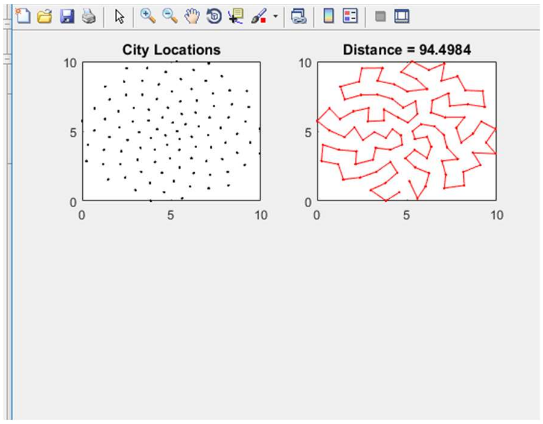
Using swap mutation



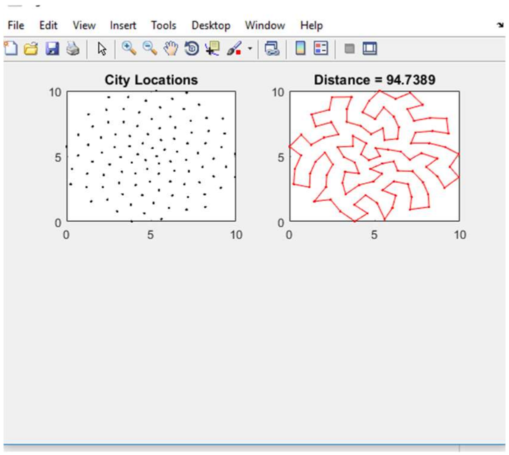
Using Single mutation : slide



Final result (with Flip and slide mutation and crossover)

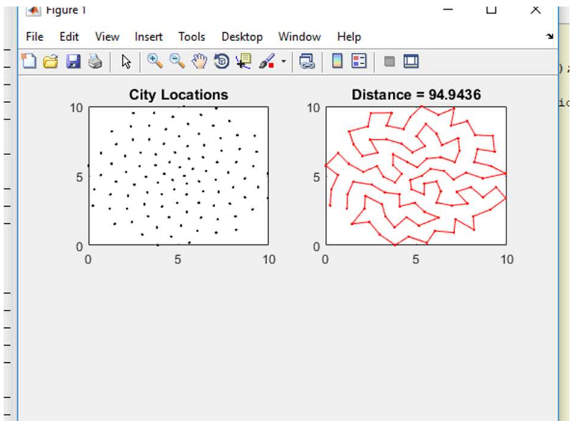


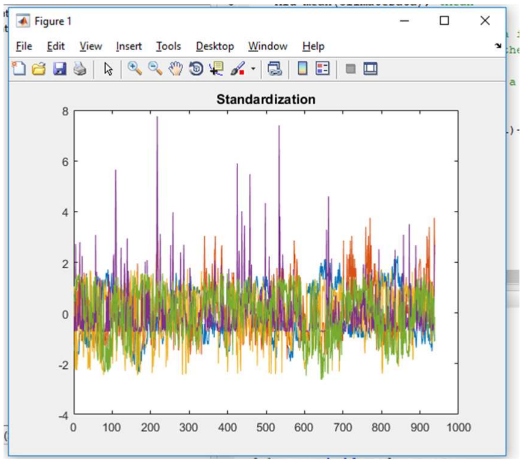
Final result run 1



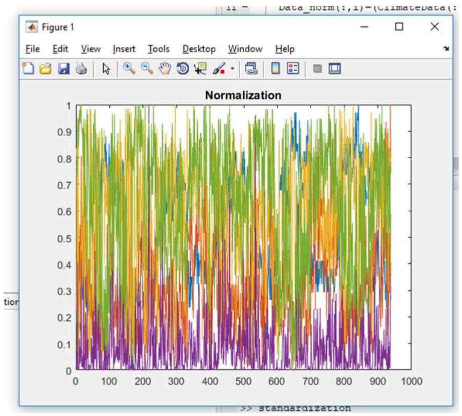
Standardization(all attributes)

Final result :After 10 runs

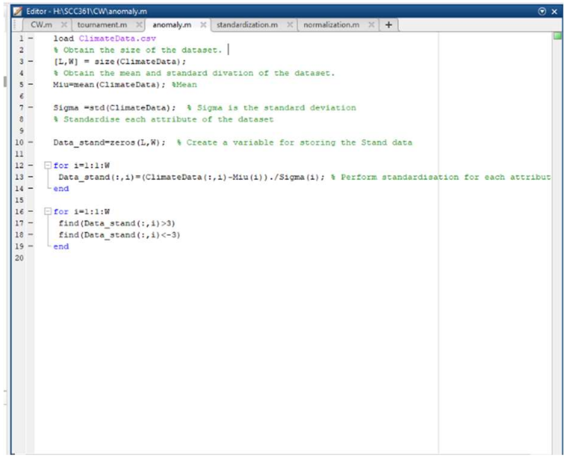




Normalization (all attributes)

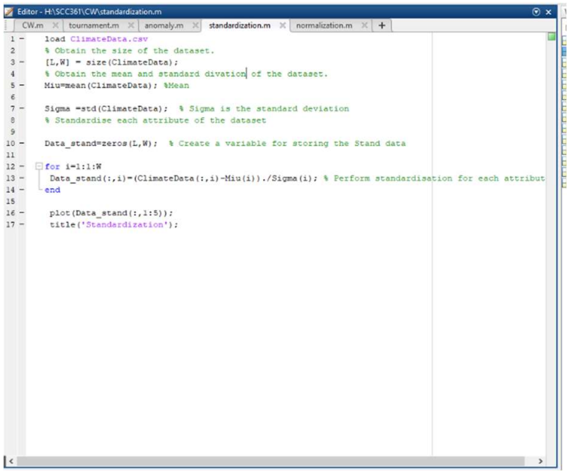


Normalization Code



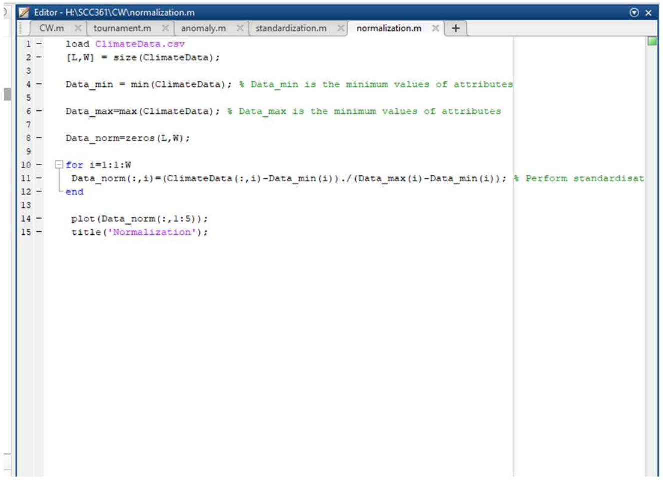
```
    load ClimateData.csv
    % Obtain the size of the dataset.
    [L,W] = size(ClimateData);
    % Obtain the mean and standard divation of the dataset.
    Miu=mean(ClimateData); %Mean

    Sigma =std(ClimateData);  % Sigma is the standard deviation
    % Standardise each attribute of the dataset

    Data_stand=zeros(L,W);  % Create a variable for storing the Stand data

  for i=1:1:W
    Data_stand(:,i)=(ClimateData(:,i)-Miu(i))./Sigma(i); % Perform standardisation for each attribut
  end

  for i=1:1:W
    find(Data_stand(:,i)>3)
    find(Data_stand(:,i)<-3)
  end
```

Standardization code



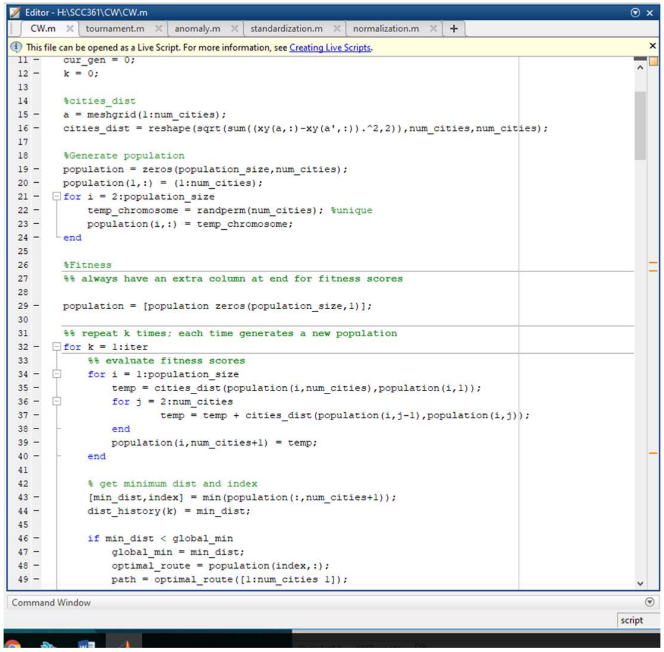
```
    load ClimateData.csv
    % Obtain the size of the dataset.
    [L,W] = size(ClimateData);
    % Obtain the mean and standard divation of the dataset.
    Miu=mean(ClimateData); %Mean

    Sigma =std(ClimateData);  % Sigma is the standard deviation
    % Standardise each attribute of the dataset

    Data_stand=zeros(L,W);  % Create a variable for storing the Stand data

  for i=1:1:W
    Data_stand(:,i)=(ClimateData(:,i)-Miu(i))./Sigma(i); % Perform standardisation for each attribut
  end

     plot(Data_stand(:,1:5));
     title('Standardization');
```

Normalization code



```
    load ClimateData.csv
    [L,W] = size(ClimateData);

    Data_min = min(ClimateData); % Data_min is the minimum values of attributes

    Data_max=max(ClimateData); % Data_max is the minimum values of attributes

    Data_norm=zeros(L,W);

  for i=1:1:W
    Data_norm(:,i)=(ClimateData(:,i)-Data_min(i))./(Data_max(i)-Data_min(i)); % Perform standardisat
  end

     plot(Data_norm(:,1:5));
     title('Normalization');
```

%Fitness TSP


Editor - H:\SCC361\CW\CW.m
CW.m
tournament.m
anomaly.m
standardization.m
normalization.m
+
This file can be opened as a Live Script. For more information, see Creating Live Scripts.
Command Window
script

```
    cur_gen = 0;
    k = 0;

    %cities_dist
    a = meshgrid(1:num_cities);
    cities_dist = reshape(sqrt(sum((xy(a,:)-xy(a',:)).^2,2)),num_cities,num_cities);

    %Generate population
    population = zeros(population_size,num_cities);
    population(1,:) = (1:num_cities);
    for i = 2:population_size
        temp_chromosome = randperm(num_cities); %unique
        population(i,:) = temp_chromosome;
    end

    %Fitness
    %% always have an extra column at end for fitness scores

    population = [population zeros(population_size,1)];

    %% repeat k times; each time generates a new population
    for k = 1:iter
        %% evaluate fitness scores
        for i = 1:population_size
            temp = cities_dist(population(i,num_cities),population(i,1));
            for j = 2:num_cities
                    temp = temp + cities_dist(population(i,j-1),population(i,j));
            end
            population(i,num_cities+1) = temp;
        end

        % get minimum dist and index
        [min_dist,index] = min(population(:,num_cities+1));
        dist_history(k) = min_dist;

        if min_dist < global_min
            global_min = min_dist;
            optimal_route = population(index,:);
            path = optimal_route([1:num_cities 1]);
```