# CSC 340 Project 2

14 February 2018

Due: 11:59 PM, 15 March 2018

Name: Y M. Liu (Simone)

1. Eigenvectors and eigenvalues (30 points)
   1. For the class data given in the “eigendata.txt” file, find and report:
      1. The ***mean*** vector and the ***covariance*** matrix. (5 points)

**The mean vector m**:

**The covariance matrix **:

* + 1. The ***trace*** of the covariance matrix. (5 points)

**The trace of **: 0.7816816527823081

* + 1. The ***determinant*** of the covariance matrix. (5 points)

**The determinant of **: 0.02975540054243309

* + 1. All ***eigenvalues*** for the covariance matrix. (5 points)

The characteristic polynomial of ****:   
**|( λI - )| = λ2 - 0.7816816527823081λ + 0.02975540054243308**

**Eigenvalues:**

**λ1** = 0.7415560257077607

**λ2**= 0.040125627074547365

* + 1. A ***unit length*** eigenvector for each of the eigenvalues. (5 points)
* At λ1 = 0.7415560257077607, x1 = 1.919394038864498 x2

**Eigenvector e1** = , **unit length of e1 is 1**

* At λ2 = 0.040125627074547365, x1 = -0.5209977627061892 x2

**Eigenvector e2** = , **unit length of e2 is 1**

* 1. ***One a single chart, plot the data and the class mean*** for the class, as well as ***the eigenvectors*** drawn emanating from (with their tails located at) the class mean and their heads translated (in the mathematical sense) accordingly. You should rescale the eigenvectors to +-convenience lengths so that they can be seen easily in the plot. (5 points)

1. More on eigenvalues and eigenvectors (20 points):   
   Estimate the roots of the polynomial:  
   p\*(x) = 30x5 – 139 x4 – 1689 x3 + 4903 x2 - 2733 x – 756, i.e., find and report all values of r such that p\*(r) = 0. Note that p\*(x) is NOT a monic polynomial. You must:
   * 1. Find a monic polynomial p(x) that has the same roots as p\*(x), and then write the companion matrix **A** for p(x). (5 points) (Hint: This is NOT a programming problem. Simply write down the companion matrix, appropriately labeled.)

**The monic polynomial p(x)**:

**Matrix A**:

* + 1. Use your implementation of ***Leverrier’s algorithm*** to ***find*** the coefficients for and ***report*** the characteristic equation for the matrix **A**. (5 points)

|  |  |
| --- | --- |
|  | **Coefficients** |
| **a5** | -4.633333 |
| **a4** | -56.3 |
| **a3** | 163.433333 |
| **a2** | -91.1 |
| **a1** | -25.2 |

The characteristic equation for the matrix A:

**|( λI - )| = λ5** – 4.633333 **λ4 –** 56.3 **λ3**  + 163.433333 **λ2** – 91.1 **λ**– 25.2

* + 1. Use your implementation of the ***direct power method*** to find an estimate for the largest eigenvalue for the matrix **A**. (5 points)

The estimate for the largest eigenvalue for **A:** 8.99992309439002 ≈ 9

* + 1. Repeat steps i-iii with the deflated polynomials and corresponding companion matrices to find estimates for the other roots of *p*? (5 points)

|  |  |  |  |
| --- | --- | --- | --- |
| **Monic polynomial** |  |  |  |
| **Companion matrix** |  |  |  |
| **Coefficients** | a4 = 4.36666  a3 = -17  a2 = 10.43333  a1 = 2.8 | a3 = -2.63333  a2 = 1.43333  a1 = 0.4 | a2 = -1.13333  a1 = -0.26666 |
| **Characteristic equation** | **λ4 +** 4.36666**λ3** - 17**λ2** + 10.43333**λ**+ 2.8 | **λ3** -2.63333**λ2** +1.43333**λ**+ 0.4 | **λ2** -1.13333**λ**– 0.26666 |
| **Estimated eigenvalues** | -6.999988044802259 ≈ -7 | 1.500215518590002 ≈ 1.5 | • 1.33333462 ≈ 1.3  • -0.2 |

1. **A Traveling Salesperson Problem (50 points)**
   1. **Exhaustive search (10 points)**
      1. Generate all solutions for the given problem instance.
      2. Find and report the mean and standard deviation of this distribution, as well as the length and trip order for both the longest and the shortest trips.

**Mean**: 8.296876318839907

**Standard Deviation**: 0.8444599372840577

**Shortest Route**: [A , F , E , N , D , B , H , K , I , L , M , C , J , G ]

**Shortest distance**: 3.689407122341673

**Longest Route**: [A , K , F , I , E , M , H , G , B , J , D , C , N , L ]

**Longest distance**: 11.503827147907607

* + 1. Collect data for a histogram of this distribution of solutions using at least 100 trip length bins and use some tool, such as Excel®, to ***plot the histogram of the distribution***. (You may actually wait to do this until part “e” of the question.)

Part e

* + 1. How long did the exhaustive search take?

7461964 milliseconds ~ 2.07 hours

* + 1. How long would you expect the algorithm to take if the number of cities, *n*, were to increase by one?

It would take 30 hours if the number of cities were to increase by 1

* + 1. What is the time complexity of the exhaustive search algorithm used?

Time complexity of the exhaustive search algorithm: O(n!)

* 1. **Random search (10 points)**
     1. Generate data for a histogram of the costs of 1,000,000 ***randomly*** generated solutions for the given TSP problem.
     2. Find and report the mean, extreme values (the maximum and minimum) and trip orders, and standard deviation of this distribution of solutions.

**Mean**: 8.297254226624066

**Standard Deviation**: 0.844408788843778

**Shortest Route**: [L , I , C , J , G , M , A , F , N , E , B , D , H , K ]

**Shortest distance**: 4.2876433025839535

**Longest Route**: [E , C , N , I , F , L , A , K , G , D , M , H , J , B ]

**Longest distance**: 11.313364328956174

* + 1. Organize data for a histogram of this distribution of solutions using at the same 100 bins as in part “a” and use some tool to ***plot the histogram of the distribution***. (You may actually wait to do this until part “e” of the question.)

Part e

* + 1. What is the time complexity of the random search algorithm?

Time complexity of the random search algorithm: O(n2)

* 1. **Genetic algorithm (10 points)**
     1. Create a genetic algorithm to find good solutions for the problem instance.
        + Create a random population list by shuffling the City list
        + Compare the previous trip length with the offspring distance, if the previous trip length is smaller, swap them
        + Create a mutation list by doing sensitive crossover, passing 5 children from parent1 and 9 children from parent2
        + Mutate the population list if the fitness of the offspring is worse than the fitness of the parent
        + Loop through the first 4 steps until 50 good solutions were found (set good distance to less than 4.0)
     2. Find and report the mean, extreme values (the maximum and minimum) and trip orders, and standard deviation of this distribution of solutions.

**Mean:** 7.451856980031576

**Standard Deviation:** 0.9417750486526945

**Shortest Route:** [C , J , G , A , F , E , N , D , B , H , K , I , L , M ]

**Shortest distance:** 3.689407122341673

**Longest Route:** [I , E , J , D , C , H , G , B , M , N , L , F , K , A ]

**Longest distance:** 11.414370379267426

* + 1. Use your genetic algorithm to find and report a histogram for at least 50 solutions for the problem using the same bins as before. (You may actually wait to do this until part “e” of the question.)

Part e

* + 1. What is the time complexity of the genetic algorithm?

Time complexity of the genetic algorithm: O(nlogn)

* 1. **Simulated annealing (10 points)** 
     1. Create a simulated annealing algorithm to find a good solution for the problem instance.
        + Set the cooling down rate to 0.00001 and the temperature to 1000
        + While (solution<50)
          1. Create a Next City list by swapping the 2 elements in the previous city list
          2. Compare the trip length of the previous city list with the Next City list, keep the better one
          3. Append any route that has the distance <=4.0 (solution++)
          4. Reduce the temperature by the cooling down rate
     2. Find and report the mean, extreme values (the maximum and minimum), and standard deviation of this distribution of solutions.

**Mean**: 6.7822596866551494

**Standard Deviation:** 0.5970751943204772

**Shortest Route:** [K , I , C , M , A , J , G , F , N , E , H , B , D , L ]

**Shortest distance:** 5.215523444095629

**Longest Route:** [A , K , G , L , F , E , J , B , D , I , M , H , C , N ]

**Longest distance:** 8.989548874058578

* + 1. Use your simulated annealing algorithm to find and report a histogram for at least 50 solutions for the problem using the same bins as before. (You may actually wait to do this until part “e” of the question.)

Part e

* + 1. What is the time complexity of the simulated annealing algorithm?

Time complexity of the simulated annealing: O(n2)

* 1. ***Compare*** (10 points)
     1. Scale each of the histograms by dividing each count in each bin by the maximum frequency count for that histogram.
     2. ***On a single chart***, plot all four of the scaled histograms.
* The plot of outcomes for Simulated Annealing is the best outcome for each of 50 trials (SASearch.txt)
* The plot of outcomes for Genetic Algorithm is the best outcome for each of 50 trials (GeneticSearch.txt)
* The distributions for Exhaustive Search and Random Search coincide in the plot
  + 1. What fraction of the distribution of possible solutions is better than your best solution by random searching?
    2. What fraction of the distribution of possible solutions is better than your best solution by using the genetic algorithm?
    3. What fraction of the distribution of possible solutions is better than your best solution by using the simulated annealing algorithm?
    4. What are the relative merits of each of the approaches?

# TSP Data

|  |  |  |
| --- | --- | --- |
| TSP |  |  |
| X | Y | Label |
| 0.725228831 | 0.028301616 | A |
| 0.632613331 | 0.879012261 | B |
| 0.085878084 | 0.352754499 | C |
| 0.880437853 | 0.852414005 | D |
| 0.725231388 | 0.382031121 | E |
| 0.74550796 | 0.32391051 | F |
| 0.166612034 | 0.10383117 | G |
| 0.637364579 | 0.962848092 | H |
| 0.136222778 | 0.908730558 | I |
| 0.078296453 | 0.248218346 | J |
| 0.396853572 | 0.962644387 | K |
| 0.390728772 | 0.703886694 | L |
| 0.276887516 | 0.354859612 | M |
| 0.974681576 | 0.344309411 | N |