Particle Swarm Optimization for Non-negative Square Matrix Factorization (UV Decomposition)

Abstract

Swarm Intelligence is a field in computer science that is interested in simulating the nature behavior to solve some of computer science real problems. To be more specific, we focus on studying how some creatures behave to solve their problems, such as finding their food, hiding and seeking, and self-organization. Ants and birds are good examples for swarm systems. You always see that the birds are grouped in flocks during their traveling or immigration process in a specific shape (called **V-shape**). They organize themselves by that way to lower the wind resistance so that they can conserve their energy which enable them to go through large distances. If you translate what they do in computer science terms, then we shall say that they "**optimized**" their formation to make the wind resistance **as low as** possible.

By studying such behavior and other creatures behaviors, we managed to solve some real problems in computer science that are considered to be NP-complete problems. Travelling Salesman Problem (**TSP**) is good example for problems that could be solved using Swarm algorithms. The task of **TSP** is that given a list of cities and the distances between each pair of cities, the output should be the shortest possible route that visits each city exactly once and returns to the origin city. Such problem could be approximately solved using Ant-based algorithm (**Ant Colony Optimization**) and Bird-based algorithm (**Particle Swarm Optimization**)

On this project, you are asked to apply a swarm optimization algorithm to solve the **Matrix Factorization** problem (**UV Decomposition**)

Problem Definition

Square Matrix Factorization is a mathematical problem that is under the domain of Linear Algebra. Given a square matrix **M** of size **n** x **n**, we need to find the matrices **U** and **V** where $M = UV^T$.

On this problem, you will assume that **M** contains only positive numbers and there are no constraints in choosing the size of the matrices **U** and **V** as long as their multiplication (dot product) would produce the matrix which has the same size of matrix **M.** Your target is to implement the **Particle Swarm Optimization** algorithm to find the values of the matrices **U** and **V** so that you produce a matrix **M'** where the difference (error) between the original matrix **M** and the output matrix **M'** is as minimum as possible.

Your error estimation metric (cost function) would be the **square root of the Mean Squared Error**.

Sample Input:

M =

```
[[5 3 1 1 4]
 [3 1 2 4 1]
 [2 0 3 1 4]
 [2 5 4 3 5]
 [4 4 5 4 0]]
```

Sample Output:

U =

```
[[ 0.58741345  1.53313444]
 [-0.48794005  1.8272795 ]
 [ 1.52291788  0.70364137]
 [ 1.8841015   1.13883695]
 [ 1.35367934  2.01717879]]
```

V =

```
[[ 0.15787364  2.06881824]
 [ 1.86330028  1.00026496]
 [ 1.38459138  1.17708564]
 [ 0.09807387  1.75916916]
 [ 2.01406271  1.32920976]]
```

$M' = UV^T$

```
[[ 3.26451361  2.62806832  2.61795814  2.75465274  3.2209448 ]
 [ 3.7032763   0.91858484  1.47526688  3.16663957  1.4460959 ]
 [ 1.69613468  3.54148113  2.93686511  1.38718264  4.00253909]
 [ 2.65349662  4.64978557  3.94921931  2.18818796  5.30845177]
 [ 4.38688656  4.54002437  4.24868492  3.68131927  5.40764882]]
```

Note: *You can just use the above sample as your only test sample. You don't have to make the code generic for different matrices shapes (It is a plus)*

Studying Materials

- Error Estimation (loss function): https://en.wikipedia.org/wiki/Mean_squared_error

-The concept of Metaheuristic optimization http://www.scholarpedia.org/article/Metaheuristics

- PSO Overview with Pseudo-code: https://en.wikipedia.org/wiki/Particle_swarm_optimization
- Detailed tutorial for PSO: http://www.cs.armstrong.edu/saad/csci8100/pso_tutorial.pdf
- PSO for simple function optimization using C#
https://visualstudiomagazine.com/articles/2013/11/01/particle-swarm-optimization.aspx

Deliverables

- The source code written in any programming language from the the following list [C++, Java, C#, Python]. The code should be clean and well-commented.
- A well-designed project is a plus
- A project documentation is a plus
- Handling different matrices shapes is a plus

Important Notes

- You can use any studying materials from the internet. The above materials are just suggestions
- If you have any previous experience with Swarm Optimization algorithms, then you can any of them to solve it. You don't have to implement PSO
- You can solve the previous problem using Genetic Algorithm if you have a background with it instead of PSO
- If you exceeded the submission deadline, just submit what you did even if you may haven't finished the implementation yet