IFN680 Assignment 1 – Particle Filtering Search

**Author:**

*Linni Qin n9632981  
Jiuzeng Rong*

**Report Title:**

*The Impact of Varying Sample size of Particle Filters Search on Estimating the Pose of 2D Objects: An Experimental Report*

**Outline:**

1. Introduction  
   1.1 Background statement: a kind of particle filter search for finding the pose of 2D triangle that has 4-DOF  
   1.2 Explain Briefly the applied Particle Filter Algorithm in the assignment   
   1.3 Explain briefly the importance of the size of particles  
   1.4 The purpose of this experimental report
2. Experimental Method  
   2.1 Experiment Environment ***Rong’s Laptop System Information***  
   2.2 Finding a computational budget of the size of particles  
   2. 3 Investigate the balance of the number of individuals and the number of generation
3. Experimental Results

Boxplot  
Trends

1. Conclusion

Reference

1. **Introduction**

**1.1 Background Statement**

While applying the machine learning algorithm to detect the pose of targeted objects, edge-based method requires cheaper computation than other pose estimation approaches does, such as keypoints-based approaches (Frederic, August, 2017). In the context of this report, a variant of particle filter algorithm is implemented efficiently as the searching method to estimate the pose vector of a 2-dimensional geometry shape. This particular particle filtering algorithm will be explained in the next paragraph. In the meantime, the targeted object is the larger triangle displayed in the left image **imf** of *Figure 1*. Just like other patterns, it has 4 degree of freedoms of pose vector, which are x cords, y cords, theta and scale. The other known factor about this triangle is the distance and location of the pixels for its three edges as indicated in the right float image **imd** of *Figure 1*.

|  |  |
| --- | --- |
| (Imf) | (Imd) |
| *Figure 1 imf* | *imd* |

**1.2 Particle Filter Search Algorithm**

*“# Particle filter search   
Initialize population* ***W*** *with random guesses of pose vectors****Loop*** *until computational budget exhausted* ***evaluate*** *the cost* ***C[i]*** *of each* ***W[i,:]******re-sample*** *the population according to* ***exp(C[i])******mutate*** *each new individual* ***W[i,:]*** *by adding some noise”  
 (Frederic, August, 2017)*

The particle filter generates initially a collection of hypotheses (***W*** population) from a random distribution. Each of them (***W[i,:]***) has four state variables as discussed above, including x cords, y cords, theta and scale. The initial particles with random poses are displayed all over the image in green colour as shown as the Figure 2.

(insert Figure 2: Initialized population W)

The program will iterate until the number of generation runs out. During each loop, the distance of certain particle to the target vector (100, 30, pi/3, 40) is recorded. ***C[i]*** denotes the weight of the particle ***W[i,:]***. Then, the program generates new particles as the re-sample procedure. However, they are not randomly distributed over the entire map starting from this step. They are generated based on previous particles with respect to their weights.

After the re-sample, mutation is proceeded on each individual with equal probability with moving 1 degree in different direction. Meanwhile, a noisy measurement of an objects location is added to figure out where the object really is. The best solution (*self*.best\_w) and the minimum weight (*self*.best\_cost) will be the results that look as similar as the patterns shown in Figure 3.

|  |  |
| --- | --- |
| Best solution | Best cost |
| *Figure 3 Best solution* | *Best cost* |

**1.3 The Importance of Sample Size**

**1.4 Report Purpose**

This report is going to explore the performance of the above particle filter search with varying finite sample sizes (*self*.W). The approaches of testing the different size of the samples, likewise, the related results and patterns obtained from the experiment will be discussed as well in the following section two (Experimental Method) and section three (Experimental Results). Finally, the conclusion with a rational suggestion will be offered about the optimal balance among the total population size, the number of generation and the scale of the particle in each iteration.

1. **Experimental Method**

**Experiment environment**

The experiment is deployed and the testing results were returned within the following computing environment:

*(Rong’s laptop system info)*

**Experiment policy**

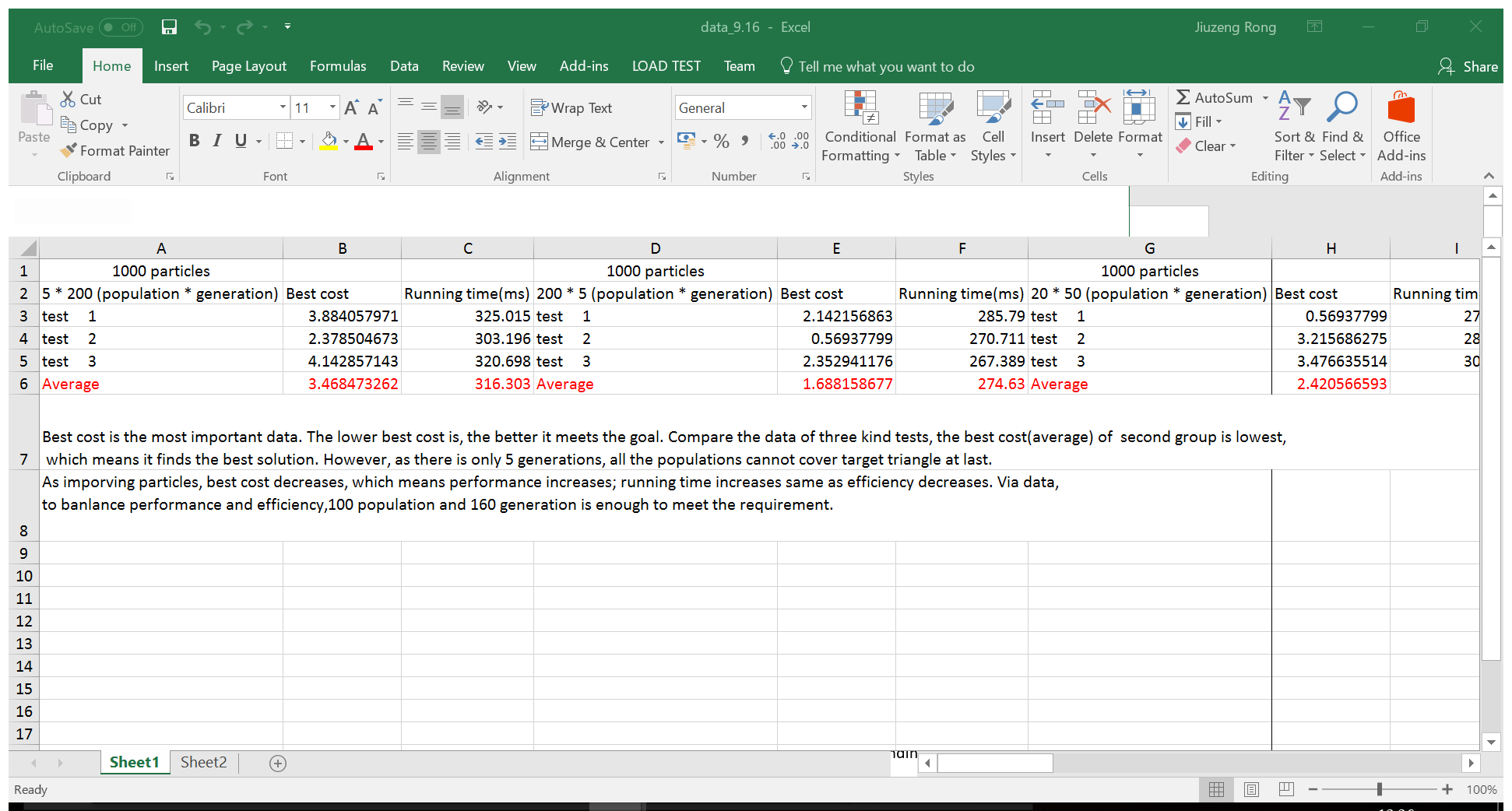
Phase one: Finding the particle budget that is suitable for this problem

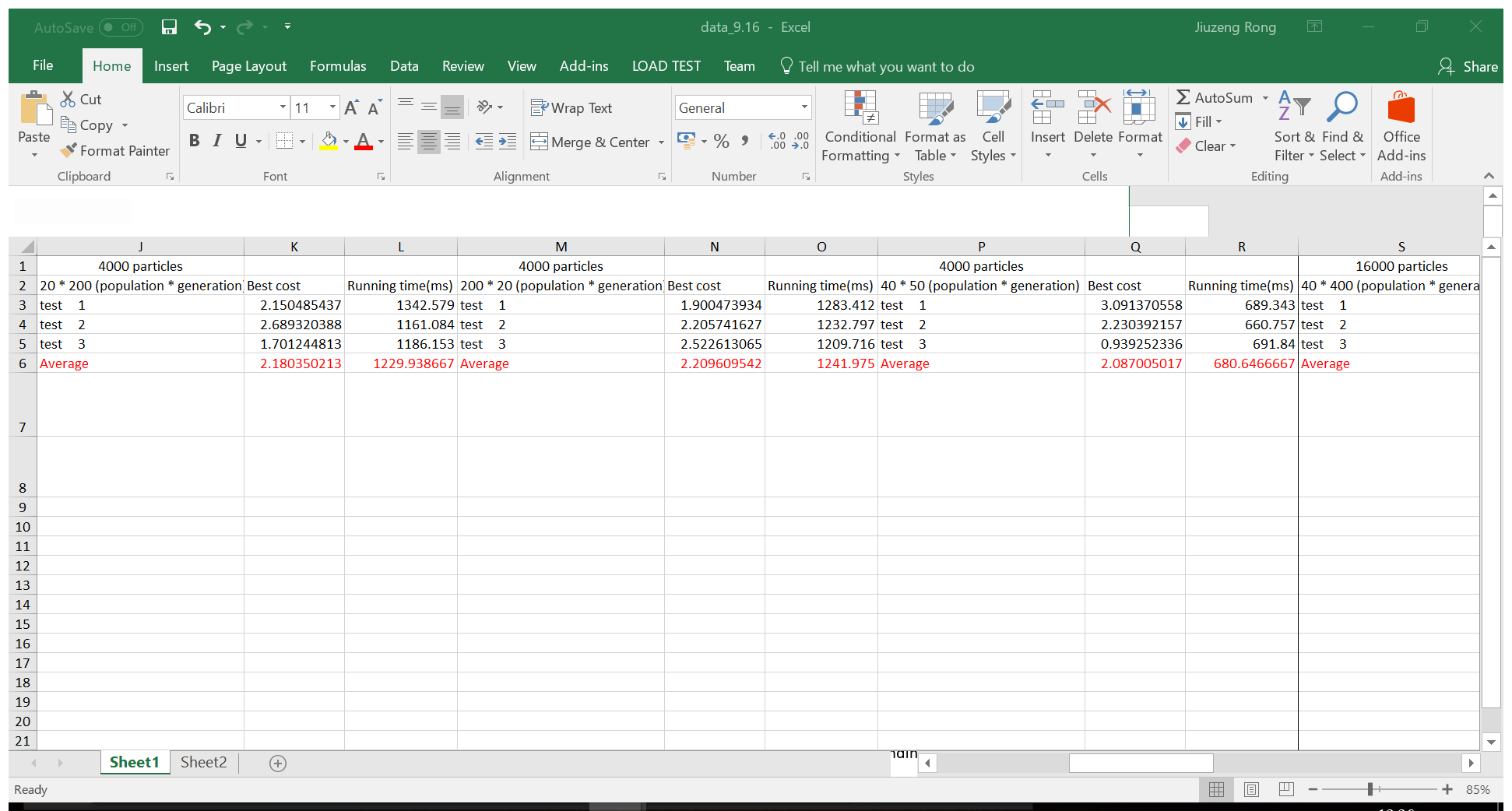
Phase two: Balance the number of generation and the size of the population based on the particle

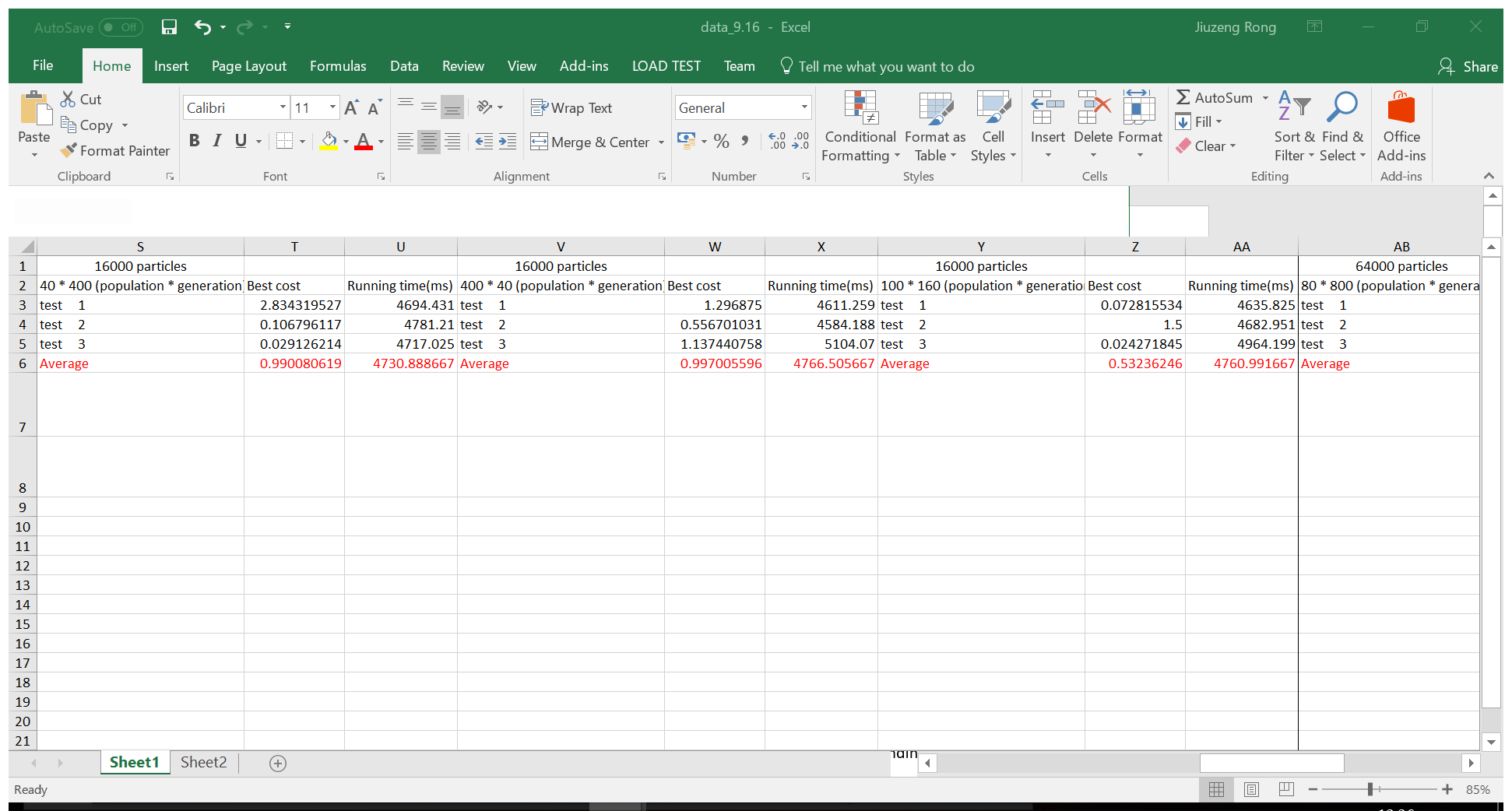
**Experiment method**

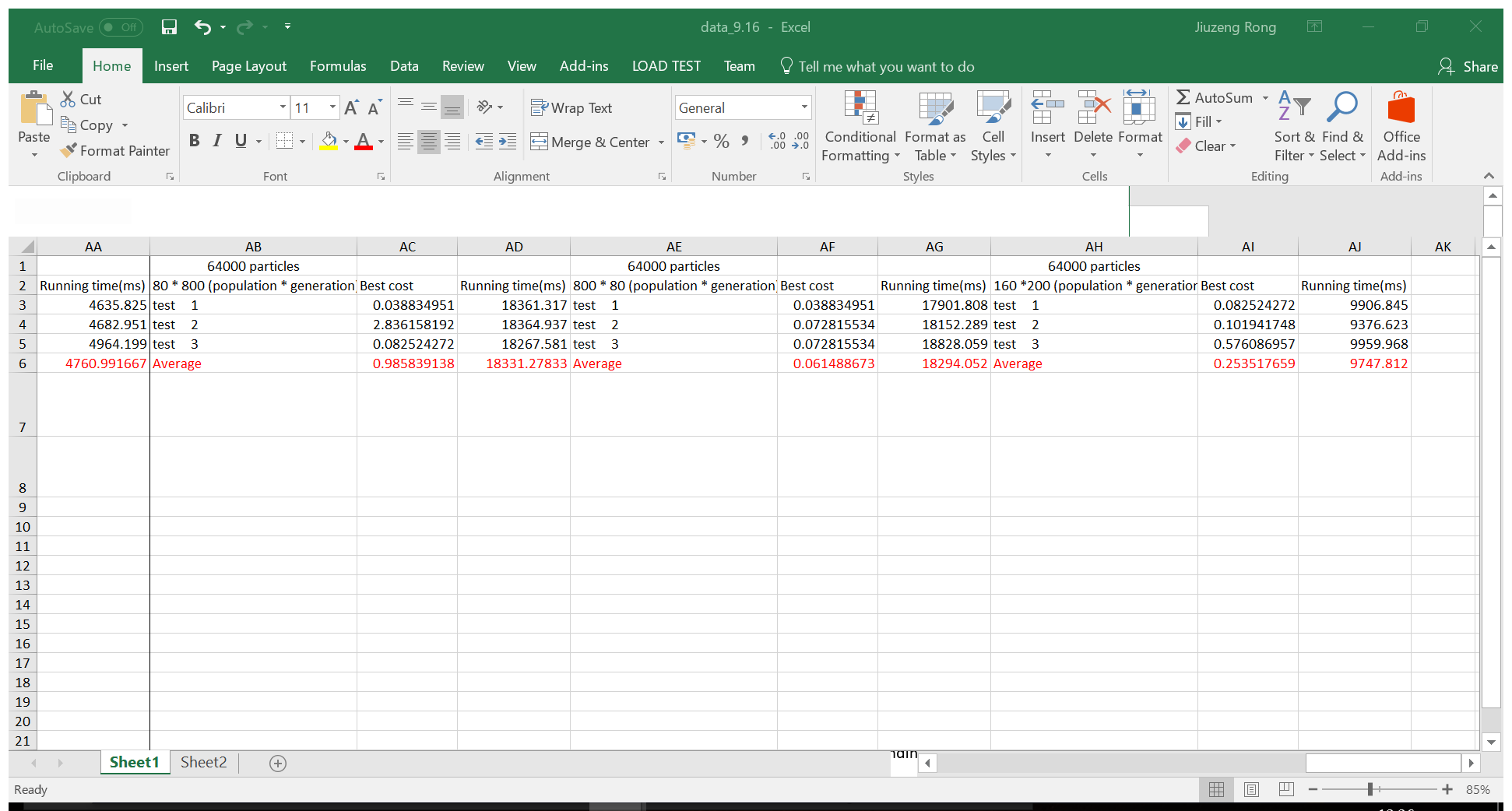
Due to the random sampling of individuals in selection, the population size is set by repeated trial without any applicable theory to guide the choice of the size until the cost appears to be close to zero and the convergence appears to get similar to required gradient descent trend. The data framework is used for the testing are listed as the table below. For each of the total particles, xxx experiments were performed. Each combination of the size of the population and the number of the iteration was replicated xxx times to obtain multiple key values for later averaging.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Total Particles | Number of Generation | Size of Individual Step | Testing Times | Best Cost |
| Small size (## ~##) |  |  |  |  |
| …… |  |  |  |  |
| Medium size (## ~##) |  |  |  |  |
| …… |  |  |  |  |
| Large size (## ~##) |  |  |  |  |
| ……. |  |  |  |  |
|  |  |  |  |  |









Manually setting the number of iteration and the size of population needed for each individual step is used to explore the performance of the overall accuracy for pose detection. The relevant codes are extracted from the test\_particle\_filter\_search() from Python file my\_submission as below.

Setting the size of individuals for each generation: pop\_size = ##

Setting the number of the iteration: pop.particle\_filter\_search(##, log=True)

Meanwhile, the computational time is evaluated as well so that the optimal balance of the parameter can be discovered with one more additional reference. The code of evaluating the execution time is added at the beginning and the end of the algorithm.

(inset the date.time code)

Consequently, the most efficient parameters to explore the best solution will be a compromise among the size of the population, the number of the iteration and the computation time. The visible outcomes at the least iteration about the best solution and the final position of the sample convergence will be compared in the section of Experimental Results.

1. **Experimental Results**

If a small W is chosen, there is a high risk of poor performance that causing high loss and non-convergence when reach the maximum of the iteration. The problem climbing is that it is easy to get stuck with local maximum. One idea to overcome this problem is to set a bigger size of population and the larger number of generation.

(inset some figure and table about the small size of population)

Due to the increase in filter size, the cost drops with increasing filter efficiency (XXX, XXx, respectively). Rates ranged from XXX to XXX decreased with the rated efficiency of the filter performance. Effective particle loss rates generally increased as both particle size and rated filter efficiency increased.

(inset some figure and table about the medium size of population)

(inset some figure and table about the large size of population)

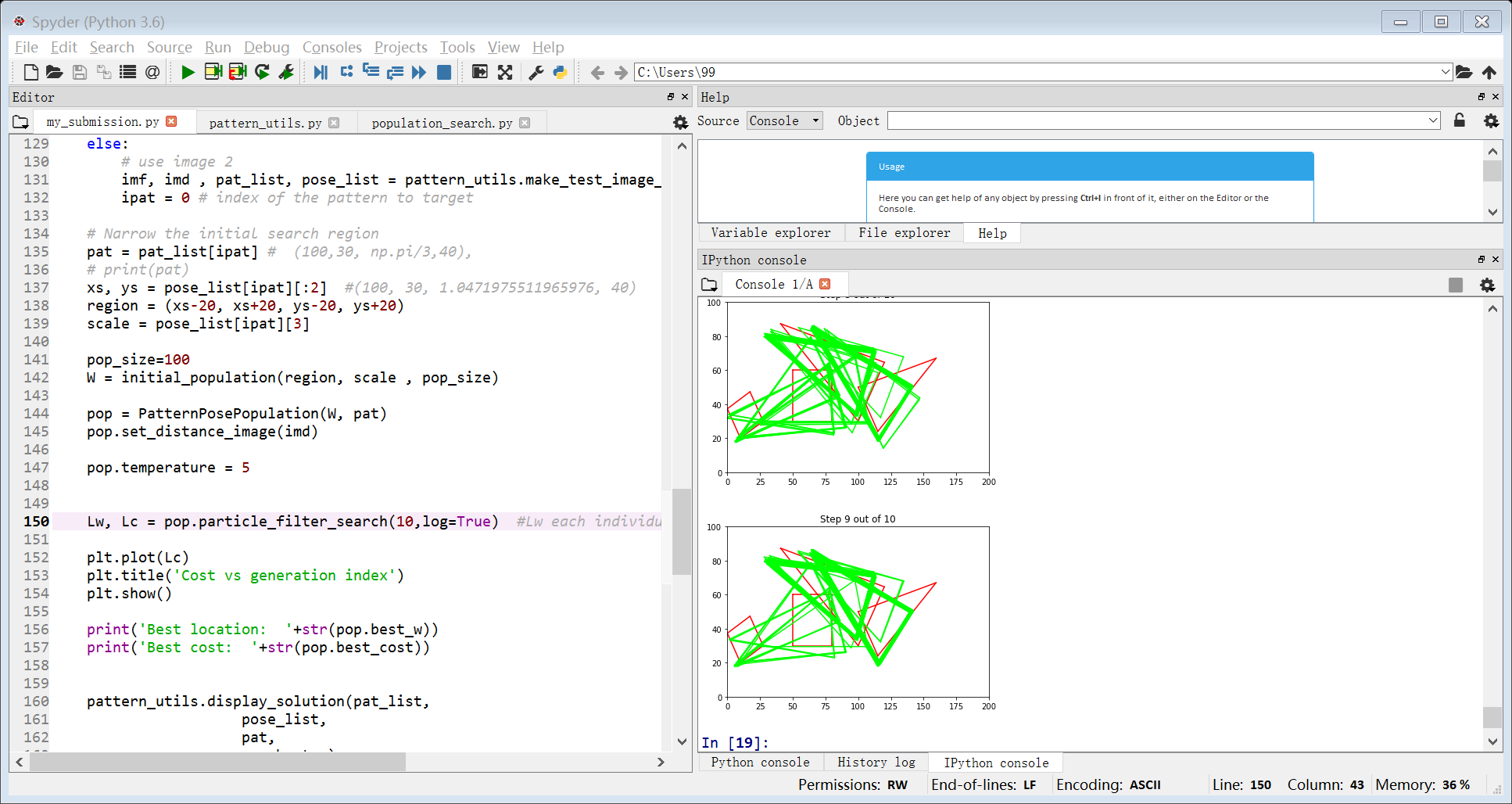
Loss rates are divided into six particle size as indicated by the dashed vertical lines. Boxes represent the center distribution of the loass, whiskers represent the outlier values. (cost distribution for each experiment can be plot with using boxplot)

1. **Conclusion**

In general, larger particles (> 16000 particles) resulted in both lower losing rates (cost) and higher computational efficiency. Small size of population……

This report recommends. ….

1. Generation值过于小，会出现图中的情况



1. 寻找最好的population和generation的比例，先设定其中一个值（population）为定值，去改变generation的值去寻找最佳比例。
2. 当比例找到之后，通过改变整体的size去找最有效的（最小的）值。 40/50 的效果和400/500的效果相差不大，前者的效率要高于后者。
3. Noise对particle的影响

Reference:

Frederic, M. (August, 2017). 2017\_IFN680\_assignment\_1. Retrieved from https://blackboard.qut.edu.au/bbcswebdav/pid-6983471-dt-content-rid-9254318\_1/courses/IFN680\_17se2/2017\_IFN680\_assignment\_1%281%29.pdf