**ISE 533 Project 2 REPORT**

**Transshipment Problem**

**Qinhao Chang**

**Shaoqian Chen**

**Wei Li**

**Tianxiao Yu**

**Abstract**

This project focuses on a multi-location transshipment problem. In recent years, research in the area of simulation-optimization continues to attract significant attention. In this project, the group set up the linear program and the dual formulation of the primal problem. Based on the primal linear program, the group solved the primal linear program by using cbc solver. After getting decision variables of primal problem, we find the relation between these dual multipliers and decision variables in the primal problem. The group also uses the SD solver to solve the primal minimization problem. Several assumptions and data sources are made for the project and will be discussed over next chapters. Results generated from two methods give us the broad scope of computational performance of Stochastic Gradient Descent method and SD solver method. Objective value and computational time will be compared.

**Introduction**

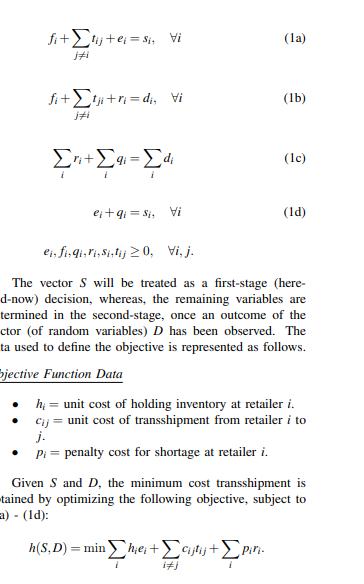
Transshipment is the shipment of goods or containers to an intermediate destination, and then from there to yet another destination. In this project, the group makes several assumptions. 1. We assume there are **7** retailers in total. 2.Assume that initial inventory level is known. 3.Replenishments arrive (based on orders placed in the previous period). 4.The demand (a random variable) is observed at each retailer. 5. Once the demand is observed, the system redistributes its inventory by solving a reallocation problem in which inventory held at one retailer can be transshipped to another, at a cost. 6. Based on the inventory reallocation of step 4, each retailer meets as much demand as possible, and then the inventory, and backlogging quantities for each retailer are updated. We also assume the unit transshipment costs are regardless of distance between each retailer. Throughout the working process, our group decides to select 7 random numbers as demand in the same scenario. In each trial, 10 sets of demand are picked. Among 10 trials, there will be 10 gradients for each supply we give. There will be 10 trials in each iteration and there will be 10 iterations. Our goal is to minimize the long run cost.

**Overview of Models and Methods/Algorithms Used**

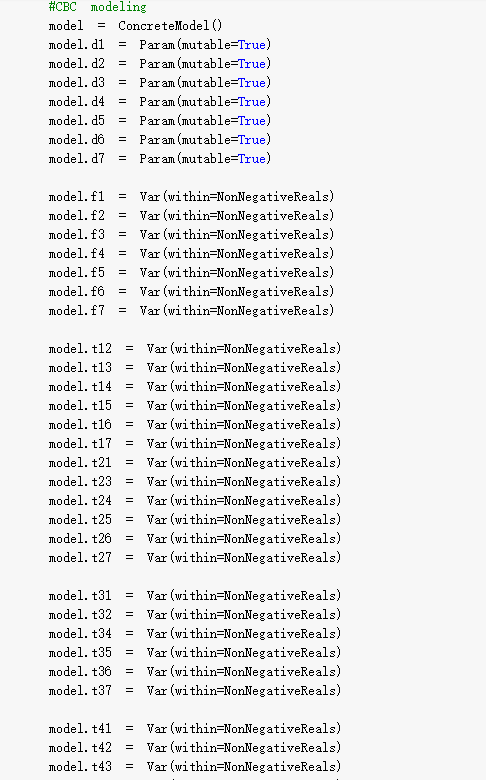
1.Stochastic Gradient Descent Method

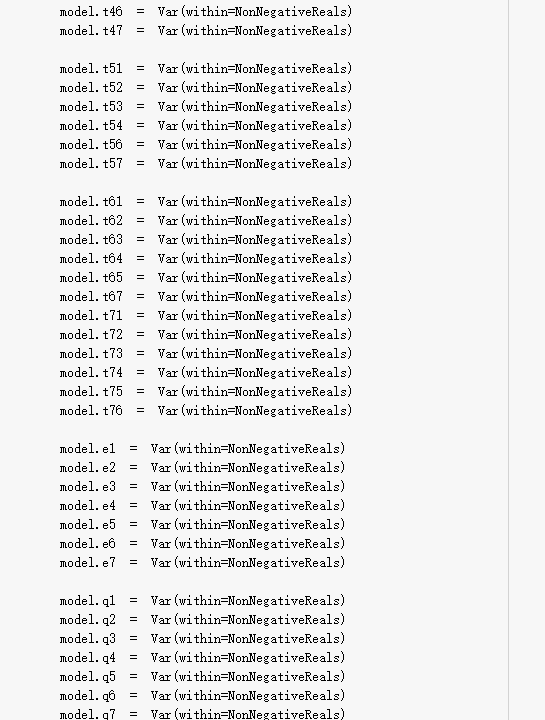
Stochastic gradient descent is an optimization algorithm that estimates the error gradient for the current state of the model .When it comes to solving this project problem. We have to build a linear program. There are 7 types of decision variables. They are: fi:Stock at retailer i used to satisfy demand at retailer i; tij: stock at retailer i used to meet demand at retailer j, using the transshipment option; ei: Ending inventory held at retailer i; qi: Inventory at retailer i increase through replenishment; ri: Amount of shortage met after replenishment at retailer i. Si is the supply of each retailer, di is the demand of each retailer. The formulated linear program is shown in the picture below(figure 3.1)

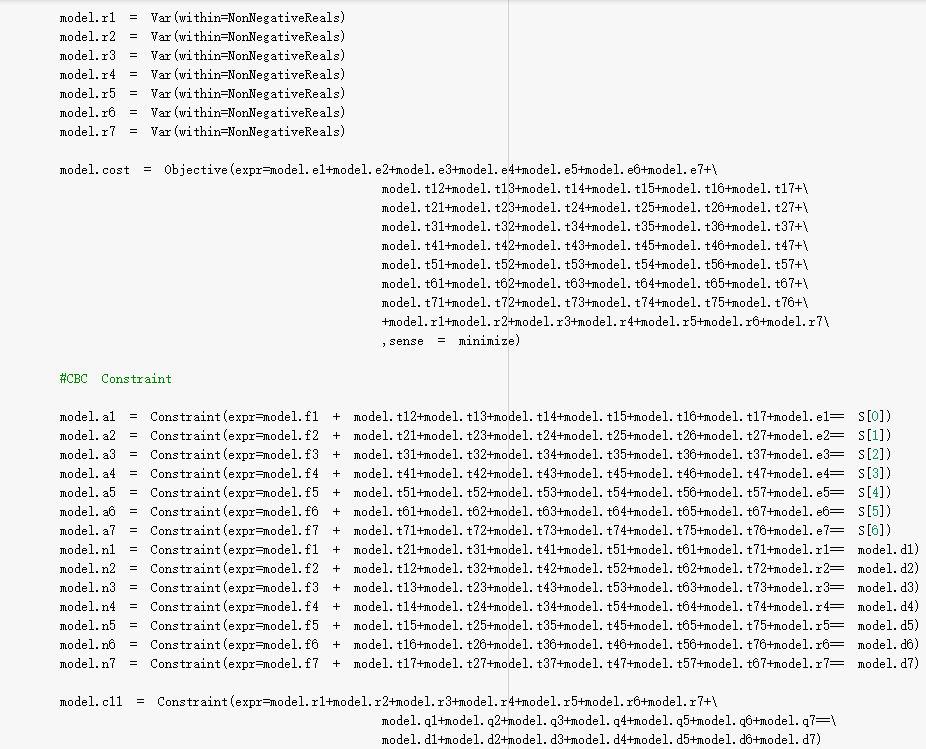
The objective function is to minimize the long run cost which has 3 parts: sum of holding cost of ending inventory of 7 retailers; total transshipment cost between each retailer; sum of penalty cost of 7 retailers.

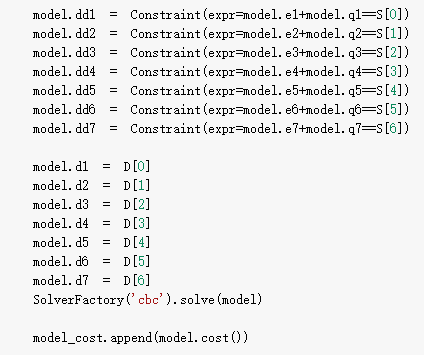
 figure 3.1

Our group programs a cbc solver in python to solve this program. The program is shown by the following screenshot. We will discuss the overview of our code below.

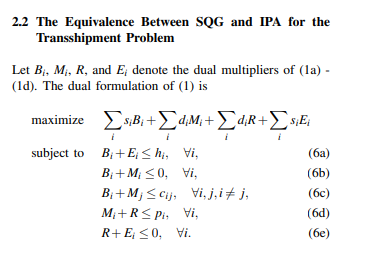




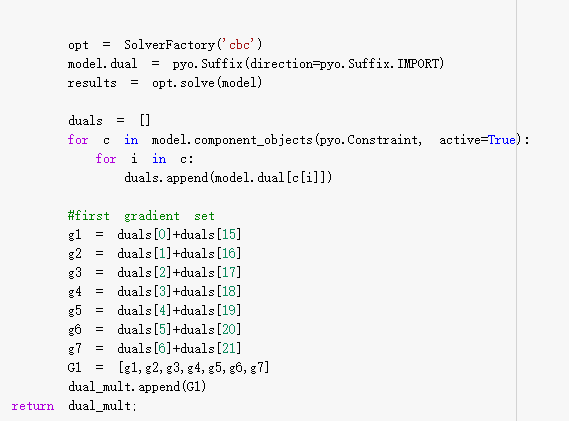




To get the gradient easier, we decide to find dual multipliers of the dual problem of the primal optimization problem. There are four dual multipliers.Bi,Mi,Ei,R. The dual problem is shown below(figure 3.2).

figure 3.2

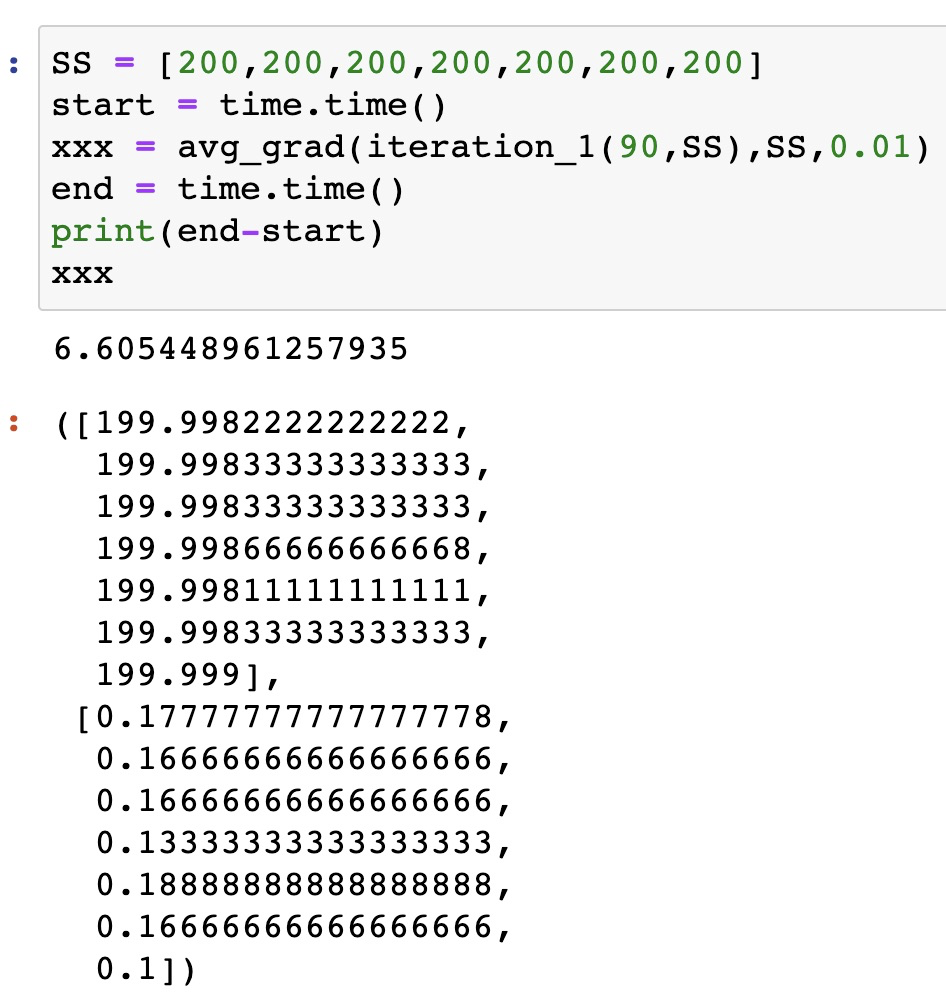
By drawing the matrix of the primal problem and dual problem,the primal problem has 70 decision variables and 22 constraints in total, so it is a 22x70 matrix. By transposing, the matrix for the dual problem is a 70\*22 matrix, which means there are 22 decision variables and 70 constraints. After drawing both matrices we find the corresponding relationship between the primal linear program and the dual linear program. Due to the dual linear program, the gradient of each supply is B+E. We write a code to output dual multipliers after we solve the primal linear program. We can get gradients for each supply by dual multipliers.



This is one trial of our experiment. Each trial a list of gradients will be generated. We have 10 trails in each iteration and have 90 iterations. We take the average value of each trial for each supply. Then we can update our supply. Sample average gradients for each supply and updated supply after 90 iterations are shown below.

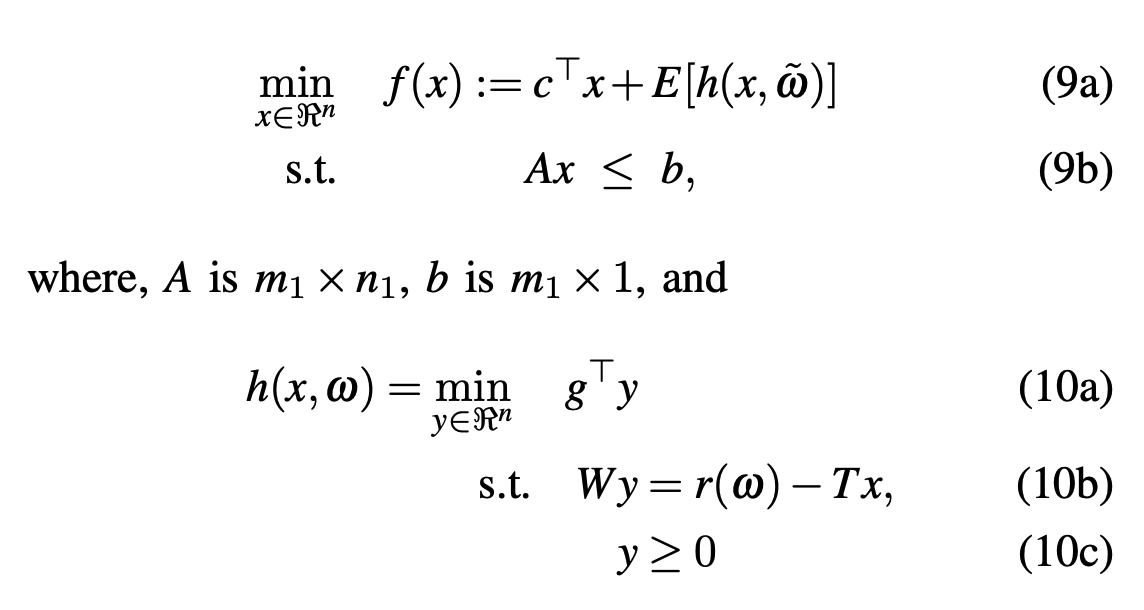
After gradients generated, we can update the supply by subtracting learning rate \* gradients from original supply for each supply index. We set up an initial learning rate 0.1.

Our goal for Stochastic Gradient Descent is that when the gradient becomes smaller and smaller, as Supply updates after gradients generate, gradients are smaller than alpha, which is a small number, then we stop the loop and calculate the objective value.



2. SD solver

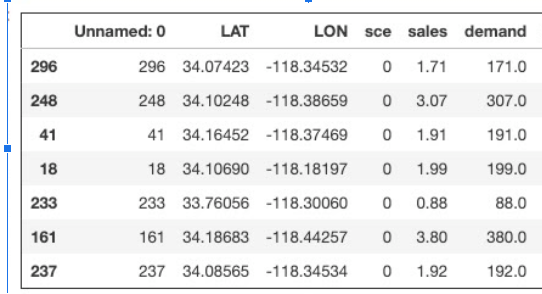
Our group also used SD solver to solve this problem. The Stochastic Decomposition(SD) algorithm was designed to solve optimization problems of the form specified in (9)-(10)

****In essence, it is an extension of Benders’ decomposition, which has come to be known as the L-shaped method in the SP literature. The primary construct in these methods amounts to creating a piecewise linear approximation of the cost-to-go function (which is also known as the recourse function in the SP literature). The main computational difference between SD and the earlier methods (Benders’ decomposition, L-shaped method) is the computational effort necessary to create a new piece (or cut) of the approximation.

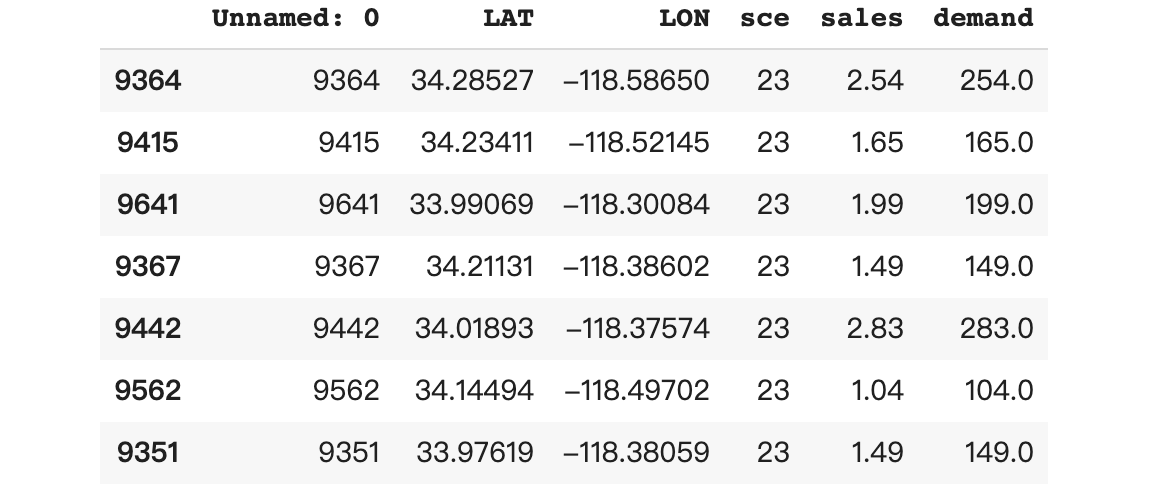
**Data Sources and Data Science**

1.SGD Method

The data we used is the data received from the teaching assistant called “individual\_loc.csv”. It contains thousands of data including sales, location information and scenario number. We use the sales data times 100 as our demand generated in each trial during each iteration because sales data times 100 is more compatible with examples from the professor. Here is the sample of the demand data which are chosen from scenario 0.



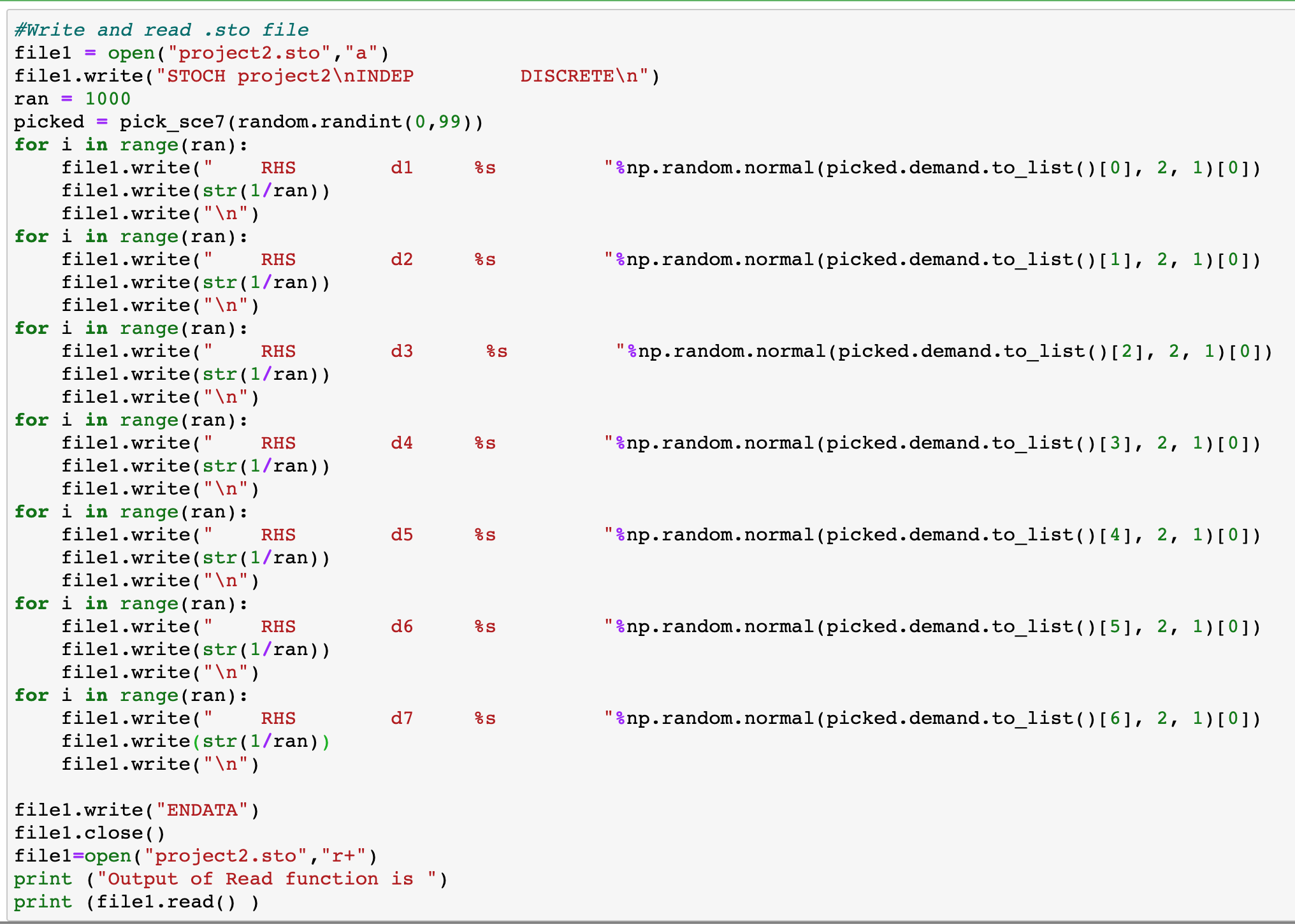
In each trial,we pick 7 random demands from a different scenario than last time.Example shows second trial we pick demands from scenario 23.



We create a user defined function which a user can set up how many times it will run.

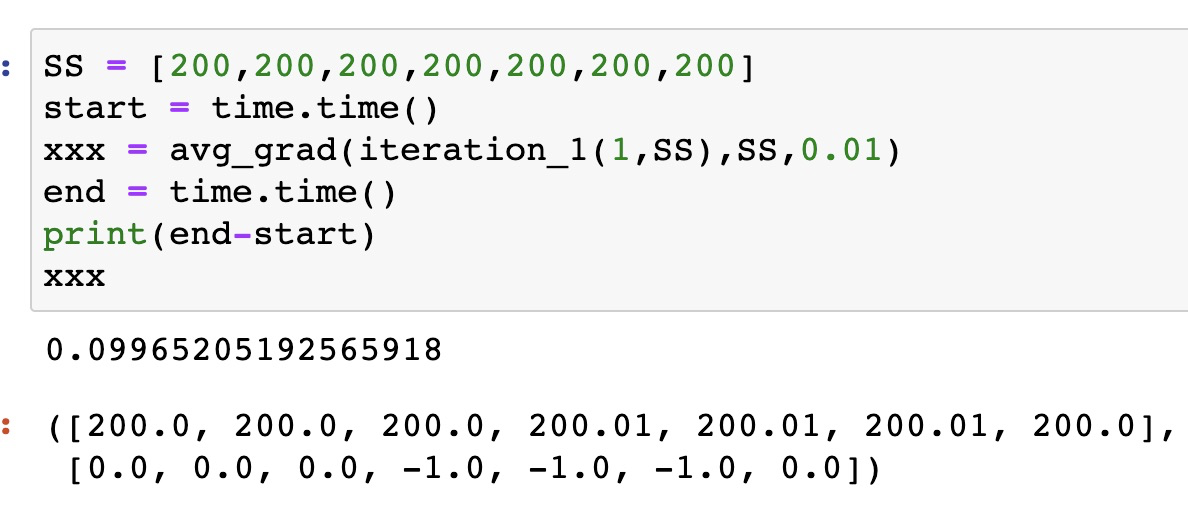
2. SD Solver

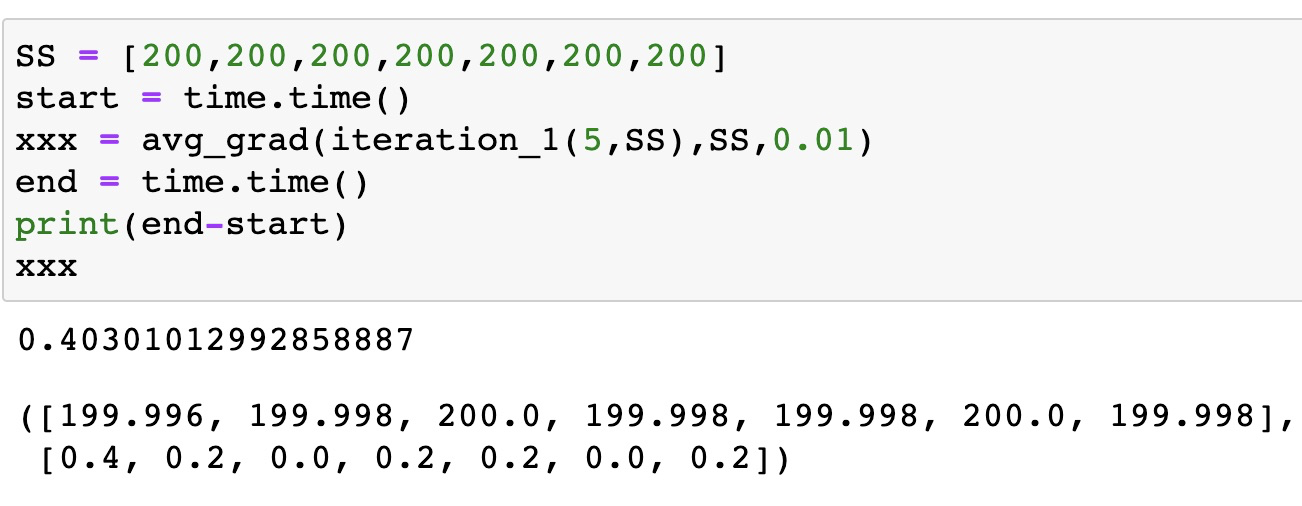
The main data source is the same as the data that we used for the SGD method. More than just that, we wrote a python script to randomly generate the desired amount of iteration and write it into .sto file.

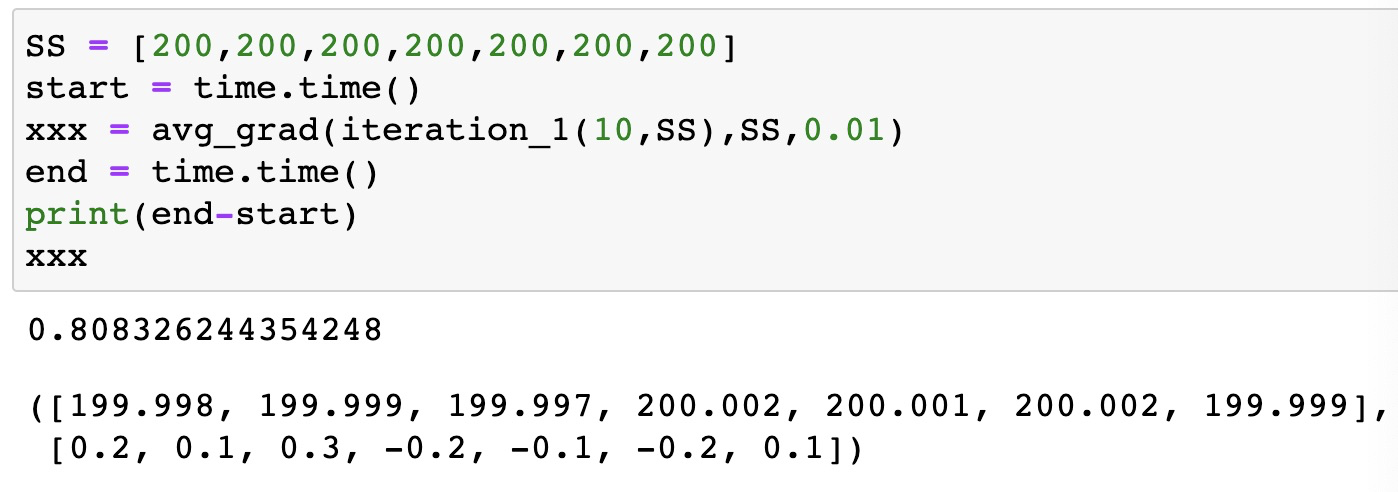


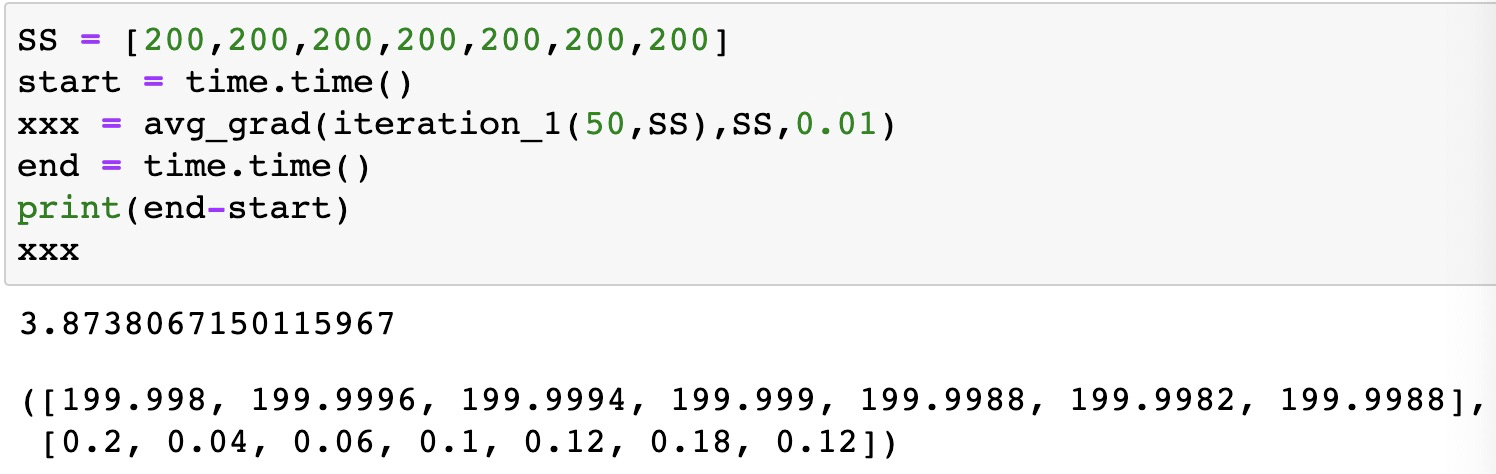
**Discussion of Results**

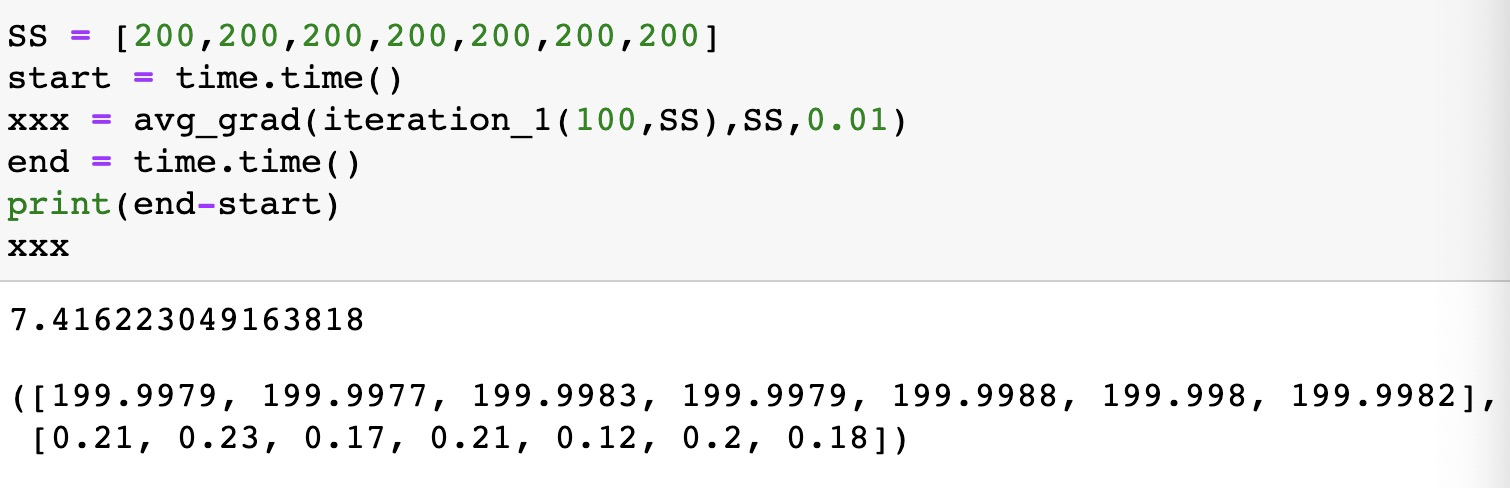
Results of SGD method including computing time of 1 iteration, 5 iteration, 10 iteration, 50 iteration, 90 iteration,100 iteration, 500 iteration, 1000 iteration are shown below.

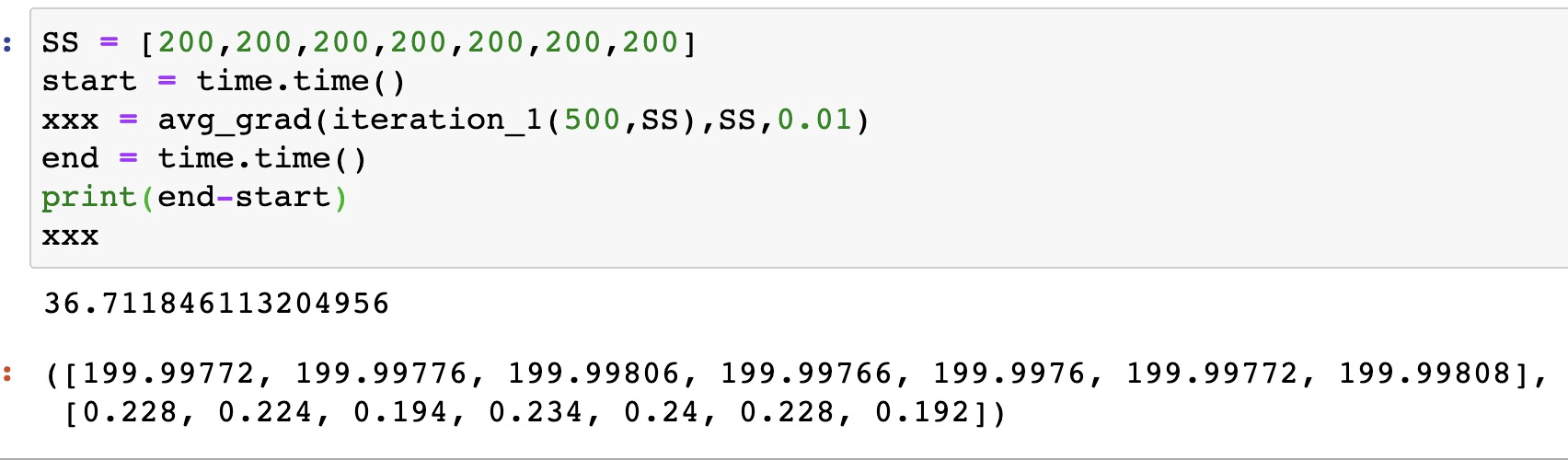


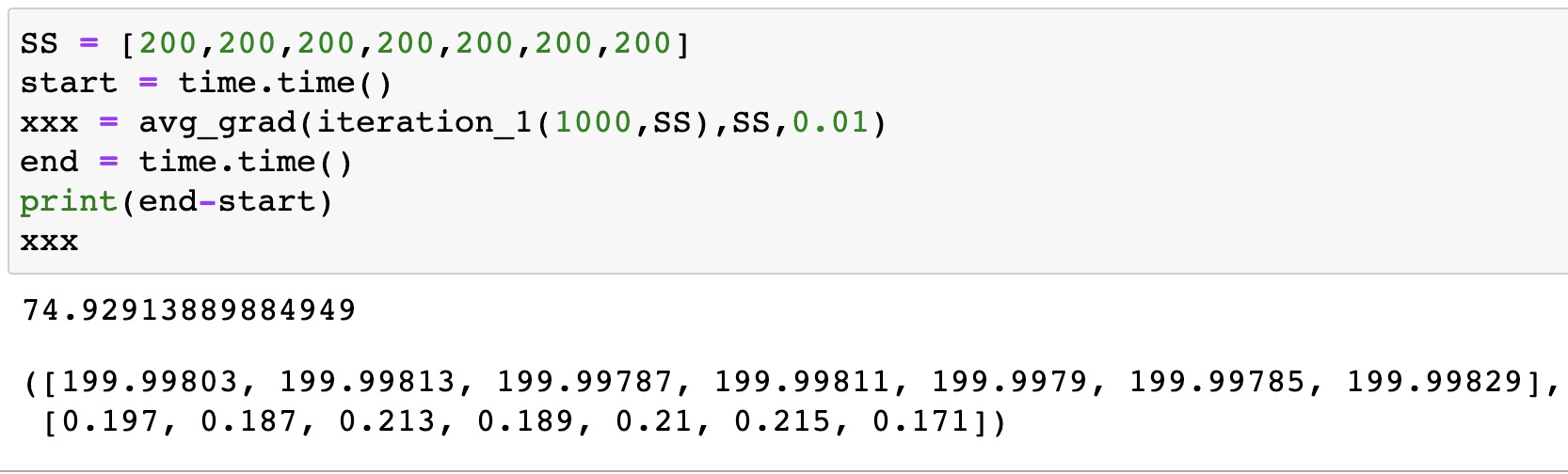


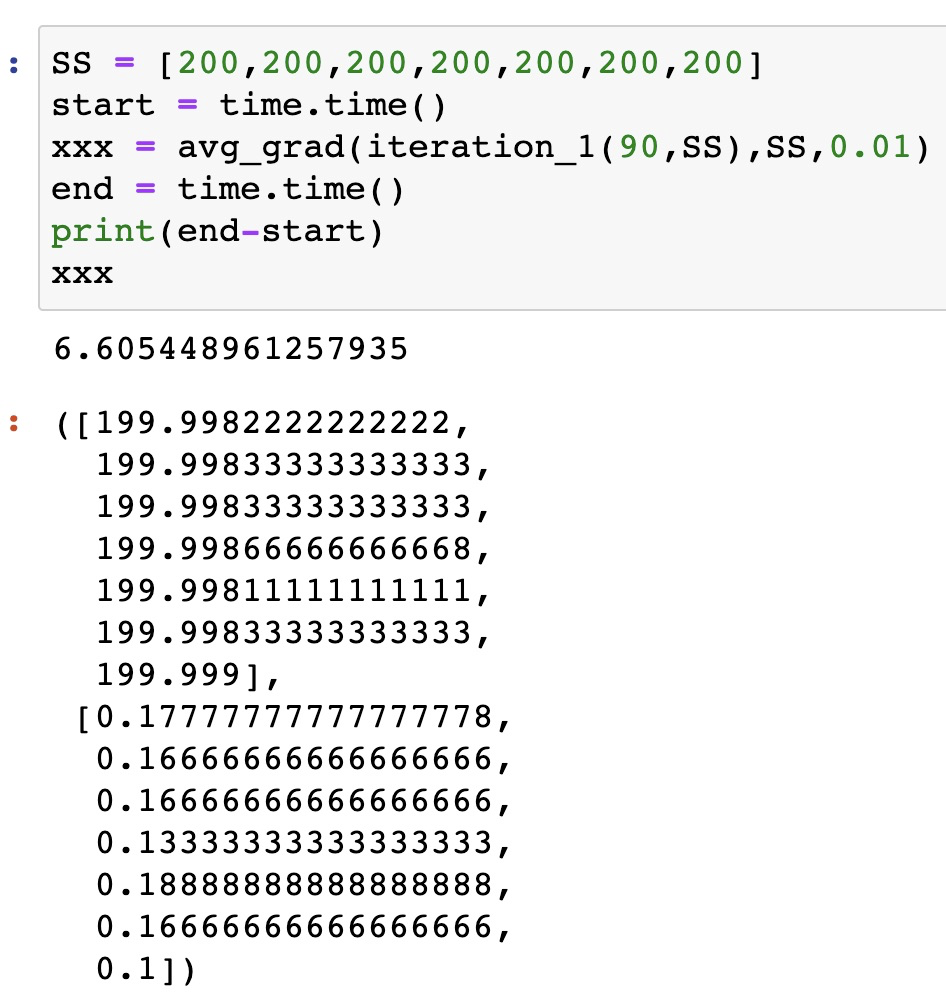


****

****

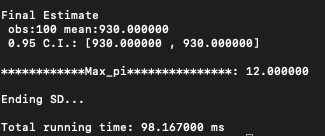
****

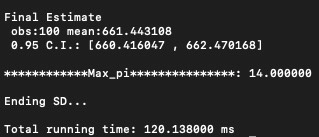
****

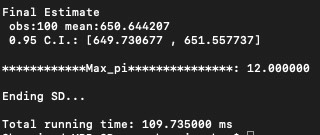
****

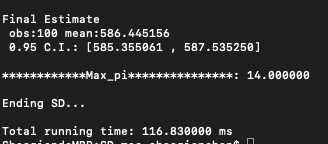
Based on the result, as the number of iterations increases, gradients get smaller that means we are approaching an optimal scenario. But initial Supply will affect the performance. When the initial supply value is far away from an optimized situation, more iterations are needed to achieve the optimization goal. As iteration numbers increase, time needed for computing is getting higher.

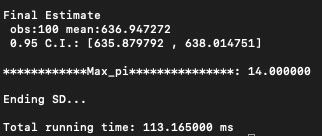
SD solver results of 1,5,10,50,100,500

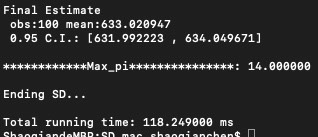
****

****

****

****

****

****

By comparing the computational time, SD solver is faster than SGD method.

**Lessons Learned, and Recommendations for Future Work**

In summary, we learnt a lot through this transshipment problem. First, we strengthen our knowledge of Stochastic gradient descent methods for deep learning. Second, we learn how to use the powerful sd solver to solve stochastic programming which speed is faster than SGD method. We learn the way to change sto,cor,tim files in sd solver file using python. We change the file through jupyter notebook to edit those files.

There are still many rooms for us to improve our design. To SGD Method, we take the way in which we set up different numbers of iterations to achieve the desired goal. We aim to write the third loop which ends the loop when the gradients are less than alpha, which is a small number to make the program clean and easier to run. To SD solver, we have to find out how to run replications over 1000 without error. Overall, we have learnt a lot from this project.