Statistical Theories for Brain and Parallel Computing

Assignment No.5

17M18819

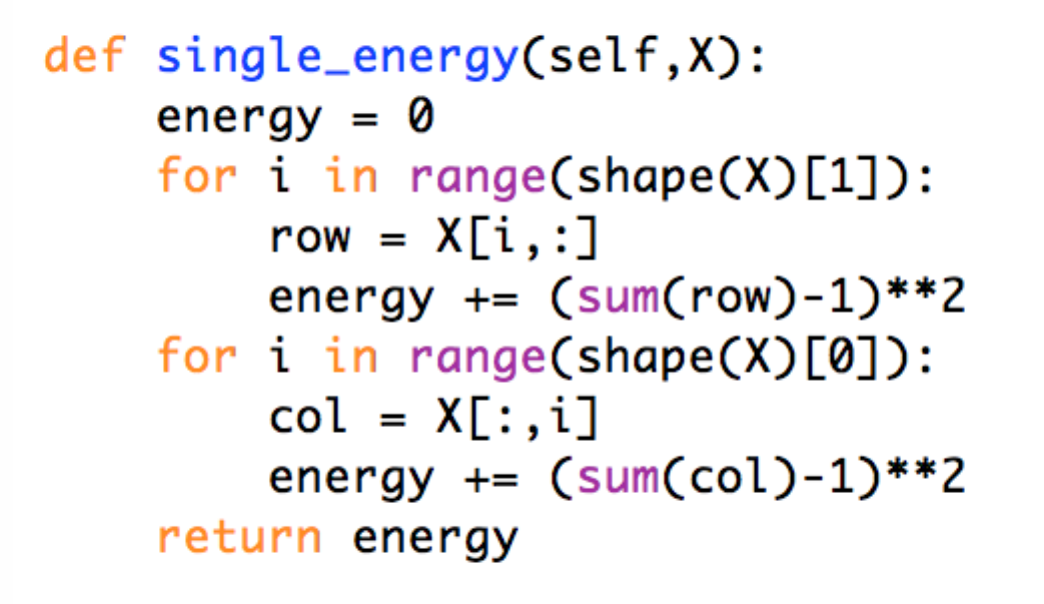
ZHU LICHENG

## Description of Assignment

In assignment 5, I need to use both deterministic and probabilistic and binary model to solve the 4-queen problem, which is a more complicated problem than simulated equation. In addition, observe whether the number of occurrence of most frequent results obeys Boltzman’s theory.

## Solution

First I would like to go to the details of the common parts of both deterministic and probabilistic program. The following function **single\_energy** is a function that can calculate the energy E of any state of chessboard X ( I described it as a 4\*4 matrix in python).



After knowing the value of every single energy of chessboard, we can calculate out the Weights and Thetas of the corresponding RNN of this question where

*Weight(ijnm) = Alpha(nm)+Alpha(ij)-Beta(ijnm)-C*

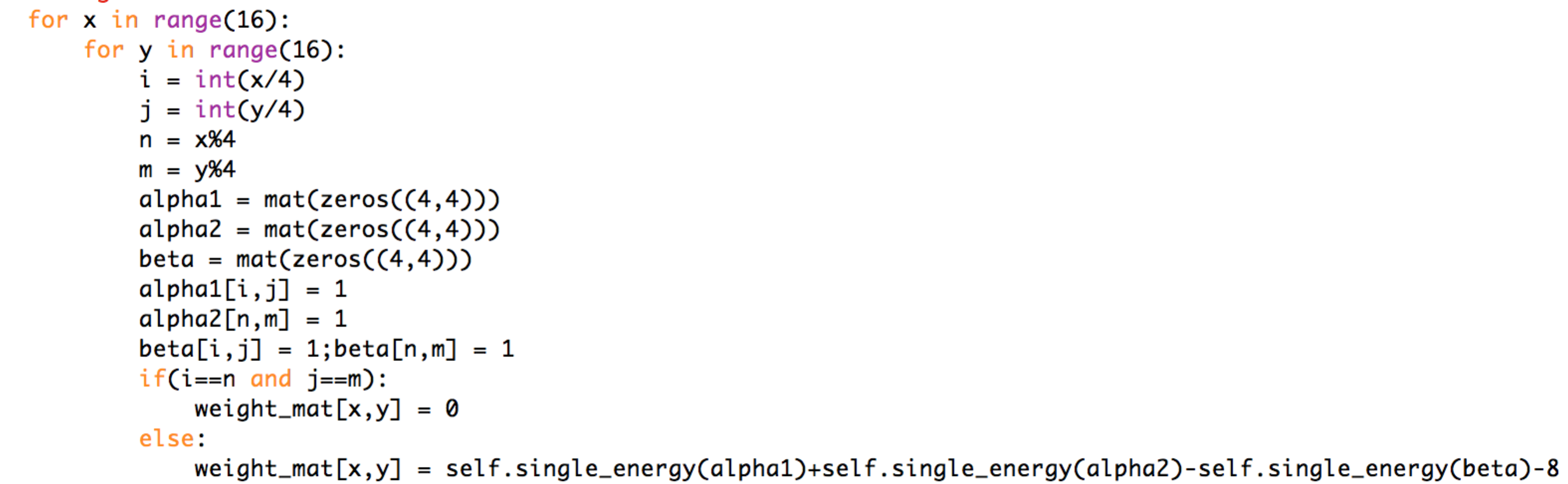
*Theta(ij) = E(ij = 1) – E(ij = 0)*

*Alpha(ij) = E(ij = 1)*

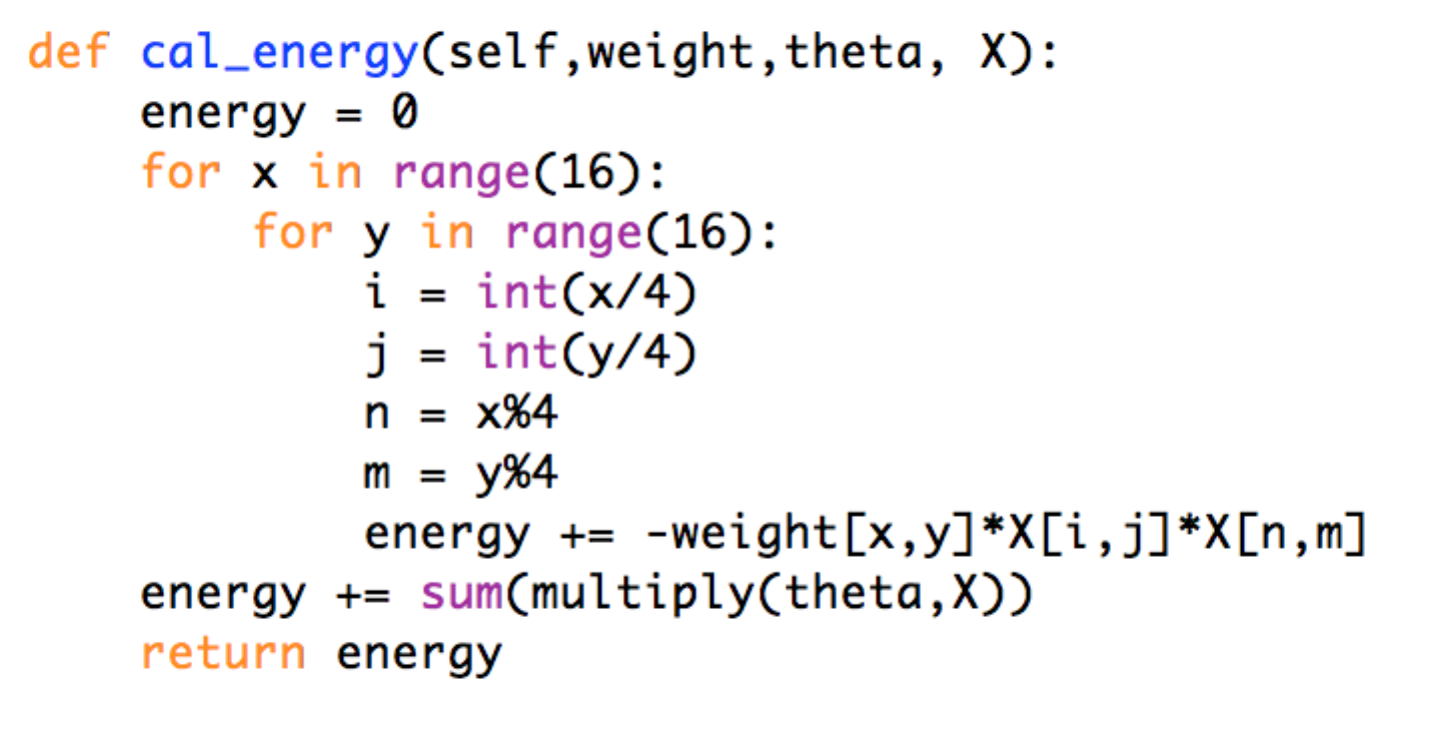
*Beta(ijnm) = E(ij = 1, nm = 1)*

*C is constant.*

The following picture shows the way to calculate the weights.



After we get the weights and thetas, we are able to calculate the energy of every state. The following picture shows how to calculate energy of a certain state.

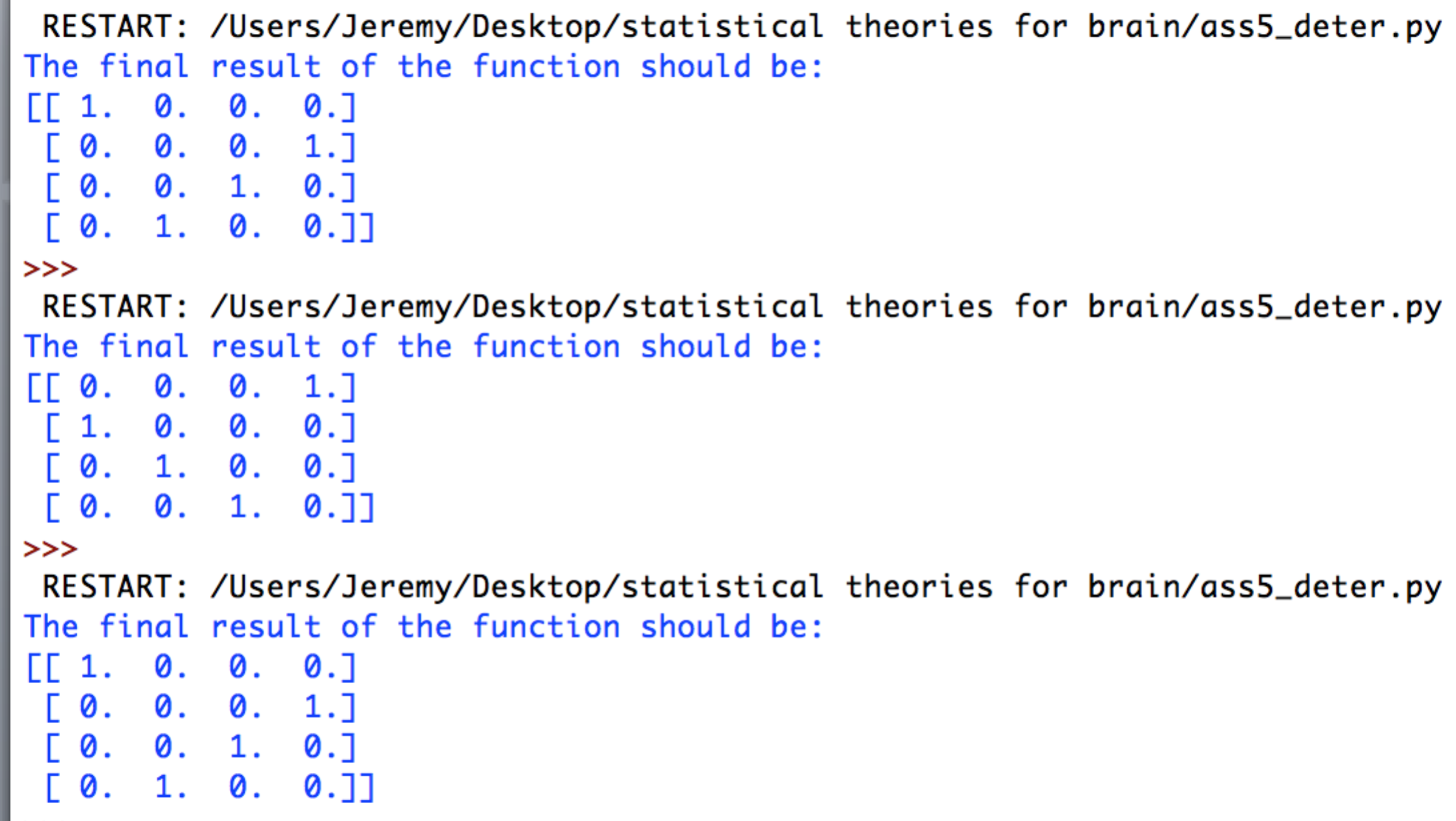


***Deterministic and Binary Model***

In this task, I created one class called **ass5\_deter.py** where we can call the function **deter()** to get a 4-queen solution.

We need to check whether there’re local minima with different initial value.

For the initial value of 16 zeros, I checked many times and proved there’s no local minima in this case. Although the result could be different, the results are all global minima.

******

However, for some cases, there actually exists local minimum. For example

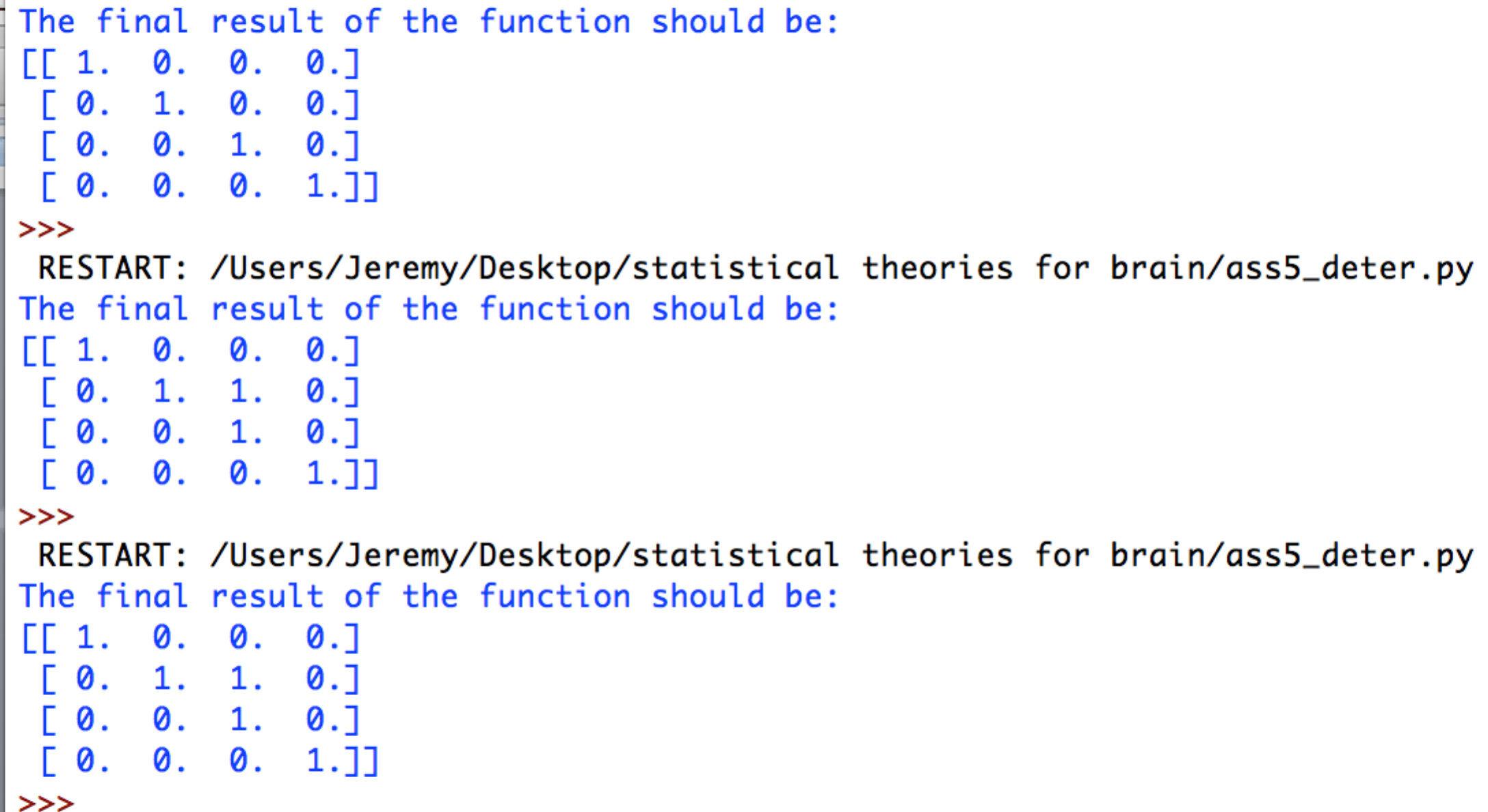
***[[ 1. 0. 0. 0.]***

***[ 0. 1. 1. 0.]***

***[ 0. 0. 1. 0.]***

***[ 0. 0. 0. 1.]]***

This matrix has high possibility to enter a local minimum as the picture shows below. The result is a local minimum for two times in three which is a high possibility.

******

Therefore, we need to apply probabilistic model to overcome this disadvantage of deterministic model.

***Probabilistic and Binary Model***

For probabilistic model, one way to solve it is to apply Gibb’s copies to generate large quantities of copies of neural network, which costs a lot of memory and time. Therefore, I apply the idea of Ergodicity to save the computational cost. In this method, we can just use one RNN and the states generated by it as a sequence while updating.

The procedure of this program:

1. Use one RNN and assign initial value

**[[ 0. 0. 0. 0.]**

**[ 0. 0. 0. 0.]**

**[ 0. 0. 0. 0.]**

**[ 0. 0. 0. 0.]]**

1. Update the neurons one by one using the probabilistic updating rule, where I choose one of the five neurons to update each time with an equal probability.
2. Among the sequence of states generated by updating ,I will count the number of each states that appeared and output it in the command window.

I used python to implement this task. I created one class called **ass5\_prob.py** where we can call the function **ass5\_prob().prob\_rec()** to calculate the result of 4-queen problem.

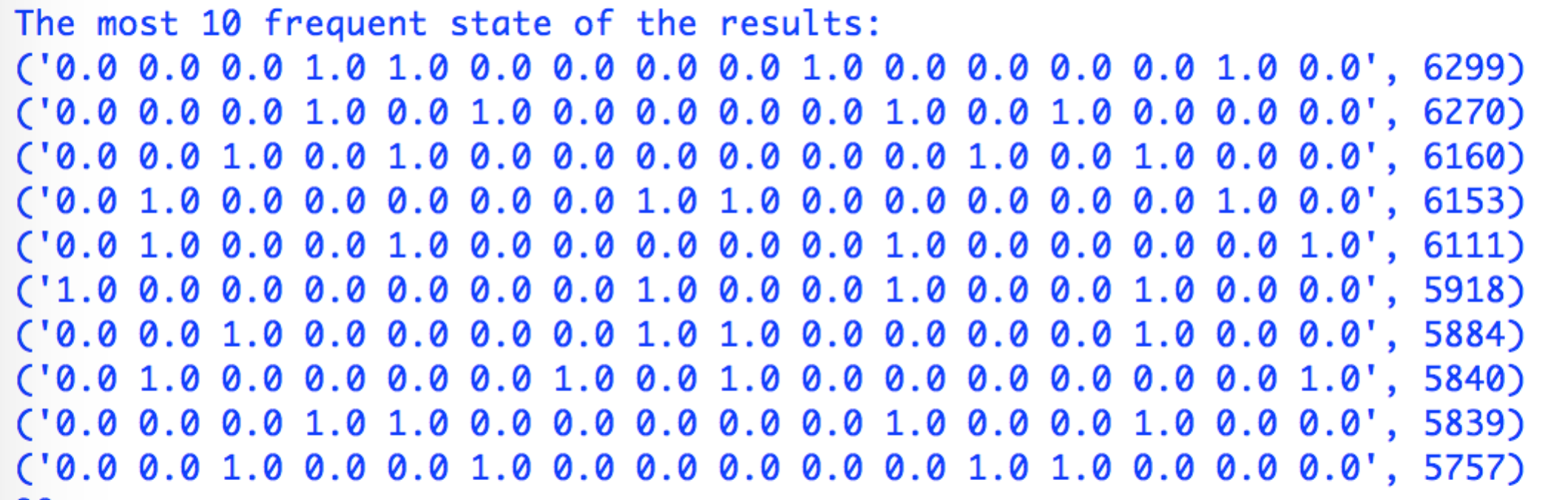
Most part of this program is the same as simultanious equation problem except that I need to update the loop for updating for 1,000,000 times instead 10,000 times.

After updating 1,000,000 times, I count the number of occurrence of the states which appeared during the process.

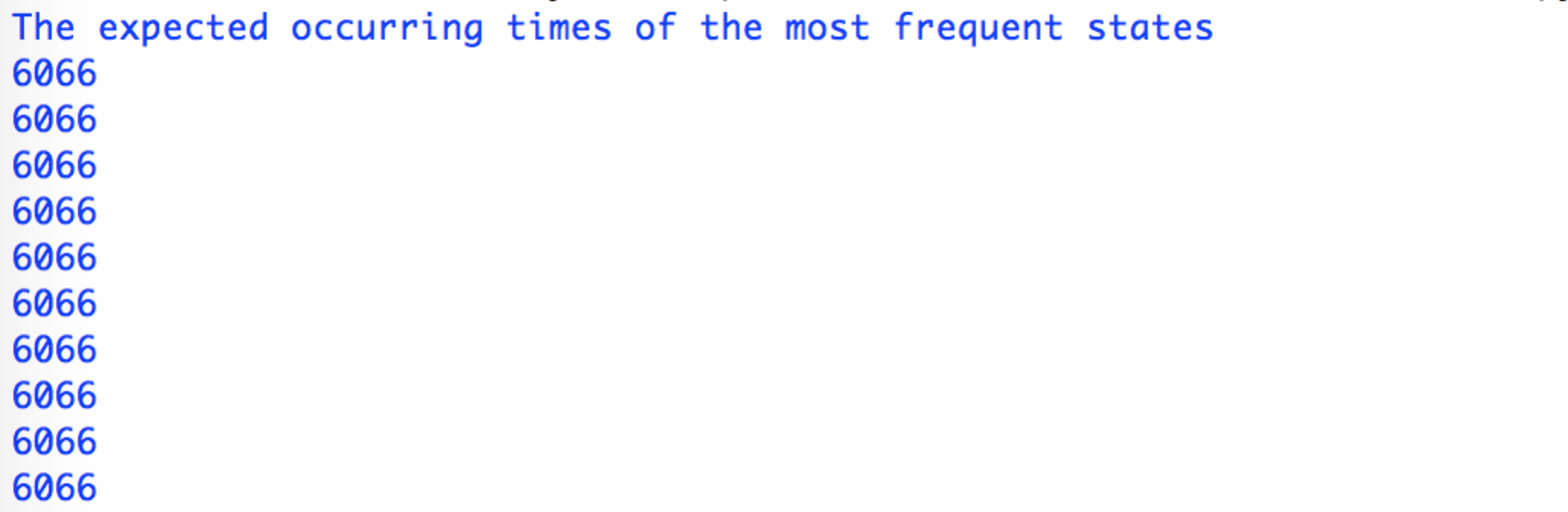
Then, I tried to observe whether the number of occurrence of states obey the Boltzman’s theory where

**N(state) = A\*exp(-alpha\*E(state))**

I tried several times and adjusted the coefficient A to a reasonable value.

**

Since the number of results is quite large, it is hard to write out all the results. Therefore, I just picked out the 10 results which occurred most frequently. The following is the ideal number of occurrence that obeys Boltzman’s Theory.

**

Now we can find that most of the values relate closely with each other which we can roughly say that the number of occurrence of our neural network obeys the boltzman’s theory.

## Running command

**ass5\_deter().deter**

**ass5\_prob().prob\_rec()**