In problem 1, I am asked to decide a code which can compute Distance Matrix by using loop and without using loop.

The function "compute\_distance\_native" is the function I use loop to make. I use inner product and T to make a code(function compute\_distance\_smart)

whose values of all index are equal Xi^2-2XiT\*Xj+Xj^2, and then I use square root, so I get ||Xi-Xj||

In problem 2, I am asked to decide a code which can compute Correlation Matrix by using loop and without using loop.

The function "compute\_correlation\_native" is the function I use loop to make. In the function"compute\_correlation\_smart" I use dot and numpyp.linalg.inv

(compute the multiplicative inverse of a matrix.), numpy.diag, to get (Xn,i-ui)^2 from n=0 to n=N-1,and then I get S.σ is equal S^1/2, so I use square root to get σ. The value of R is between 0 to 1; therefore, I use "numpy.round" to round to two decimal places.

After finish the code, I use random to decide N, D, and X. Then I put N, D, X into my function. I compare the two function, use loop and without using, if the result is the same, I know the coding is right.

In problem 3, the data is big; therefore, I find that using loop will need to spend more time.

The function of using loop is easy to make, and not easy to have bug; the function of without using loop is difficult to make, and I spend more to debug and check it. But the function of using loop is Inefficient, because the function search and compute too many times, it computes only one value for each time; therefore it waste much time.

After I finish the code, I write a small code to test two function. The N, D and X are created by random. If smart and naïve have the same result, I know that my idea and my coding is right.