**Problem Description**

Consider a matrix **A** of size , where each non-overlapping d×d block of the matrix, , is a diagonal matrix. So the matrix consists of such blocks. An example of such a matrix is shown below:

Construct an efficient data structure to represent such matrices and devise algorithms to perform matrix operations, such as matrix multiplications and matrix inverse, on the data structure you designed. Provide a technical write-up of your solution along with associated code implementing your solution.

**Solved**

**Solved in python 3.6 using numpy version 1.14.3**

I thought it would be best, if I save only the data. Since the sub matrixes are diagonals. After some thinking, I induced that I can save only the data without any problems and also I would be able to do matrix operations on it. I used an example to get some intuition about the problem.

I will attach the picture at the end of this document.

After I found out that I can multiply a row in a column and get the result I did it using numpy. Therefore, my order would be like n^2d+d which is time of loops in number of multiplication which numpy does fast.

After completing this task it hit me that maybe sparse matrix representation would be a better answer to this problem. However, since it won’t be a sparse matrix unless our d is more than 3, also the sparse algorithms would take more space and time I decided to continue this approach.

More, in the process I found that there is something which is called fast sparse multiplication, which I believe in this case it won’t get better results than this approach.

Solving this problem also was rewarding in another way, because it was an engaging task, which made me go through new stuff.

