PHY 4000W
Computational Physics
Tutorial 1

Boitshoko Moetaesi

Abstract

The Aim of this doc is to compare the performance differences between matrix operations implemented using arrays and loops with a dedicated matrix library. This was done for a few matric operations discussed below. Execution times as a function of n, up to execution times of a few seconds where measured and compared to the execution times of the array implementation with a dedicated numpy matrix library.

Discussion And Results

Dot Product of two vectors

Figure 1 below shows the relationship between execution time and the size of two vectors. This was done firstly by using two one dimensional arrays of length n and doing the product using a loop.in index notation we wound have $\mathbf{X}^t \bullet \mathbf{X} = X_i X_i$. Therefore, number of floating-point operations is given by n multiplications if vector x has n elements plus (n-1) additions which equals 2n-1 FLOPs. Therefore, we get a linear between vector size and execution time with an array implementation. NumPy package performance barely changes as vector get larger.

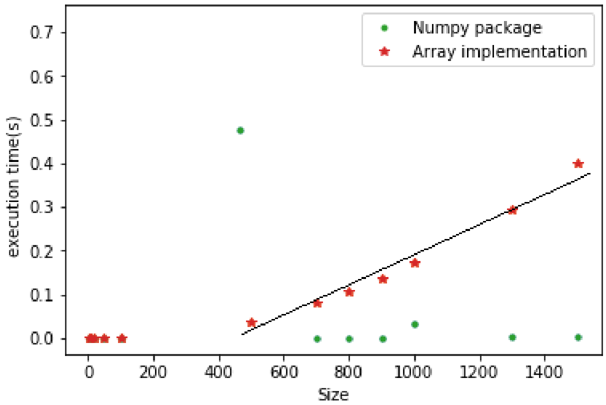

Figure 1: Dot product execution time as vector size increases. The Plot also show performance differences between numpy and an array Implementation of dot product.

Multiplication of a matrix with a vector

Figure 2 shows the relationship between size n and excecution time of this matrix $\mathbf{M}_{n\times n}\ \mathbf{x}_n$ multiplication .This multiplication corresponds to applying inner product rule $\mathbf{M}_{jxi}\mathbf{x}_i$,here j is the jth row in which runs from 1 to n and hence the are n of such product .Thus we have n*n multiplications and n(n-1) summations implying $2n^2$-n FLOPs.Using big o notation we get that the number of FLOPs to be of the order $O(n^2)$ matrix when using arrays and loops to do matrix multiplications.Which is what is observed.As with the previous case numpy perfomes better.

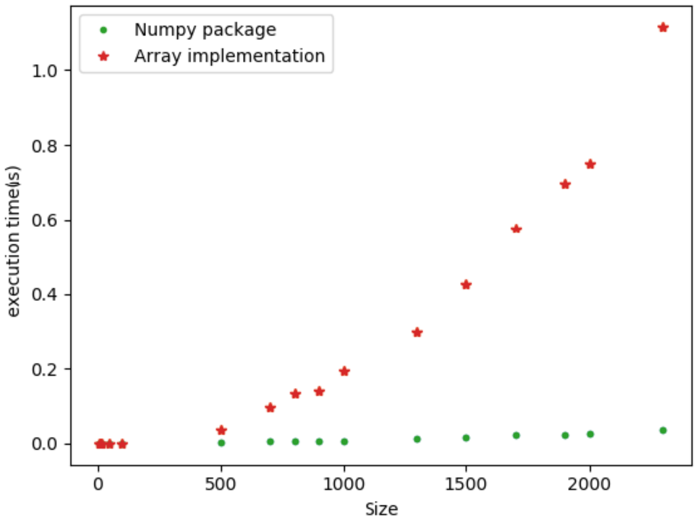

Figure 2: multiplication of a matrix with a vector execution time as vector size increases. The Plot also show performance differences between numpy and an array Implementation of multiplication of a matrix with a vector.

Multiplication of two vectors and a matrix

Figure 3 shows a the relationship between size n and excecution time of a matrix pf the form $\mathbf{x}_n^T\boldsymbol{M}_{n\times n}\ \mathbf{x}_n$ matrix.The total number of FLOPs for this matrix oparation is $2n^2$+n-1 which is of $O(n^2)$.This plot below shows that the number of FLOPs is of $O(n^2)$ for array implementantion.

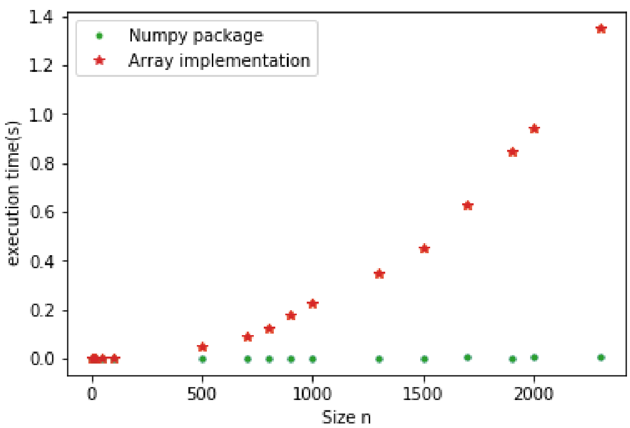

Figure 3: Multiplication of two vectors and a matrix vector execution time as vector size increases. The Plot also show performance differences between numpy and an array Implementation of Multiplication of two vectors and a matrix.

Multiplication of two matric

For the multiplication of n*n matrix which can be represented in index notation as $M_{ij}M_{ji}$ there are are n multiples and n-1 additions. This makes a total of about (2n-1)nn flops, Thus we expect a that the execution time will grow with an odder O(3).Completed to the other figures, in fig 4 execution time grows really fast with increasing .It doesn't look quadratic but its fairly close.

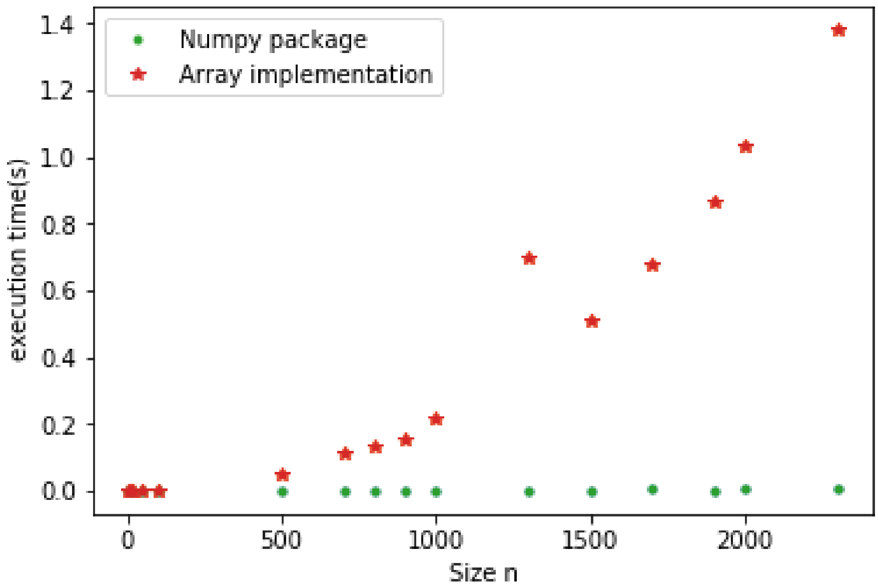

Figure 4: Multiplication of two matrix vector execution time as vector size increases. The Plot also show performance differences between numpy and an array

Implementation of Multiplication of two matrix.

Conclusion

Matrix multiplication can be expensive because of the computational complexity of doing matrix multiplication. NumPy has dedicated libraries that can handle large matrix calculations fairly wel.