NUMERICAL LINEAR ALGEBRA

HW - 1.1
HW - 1.2
HW - 1.3

Murat Çabuk

HW - 1.1 PURPOSE

Writing six different “for loops” for multiplication of two square matrix and testing time speed by changing loop's places in the code.

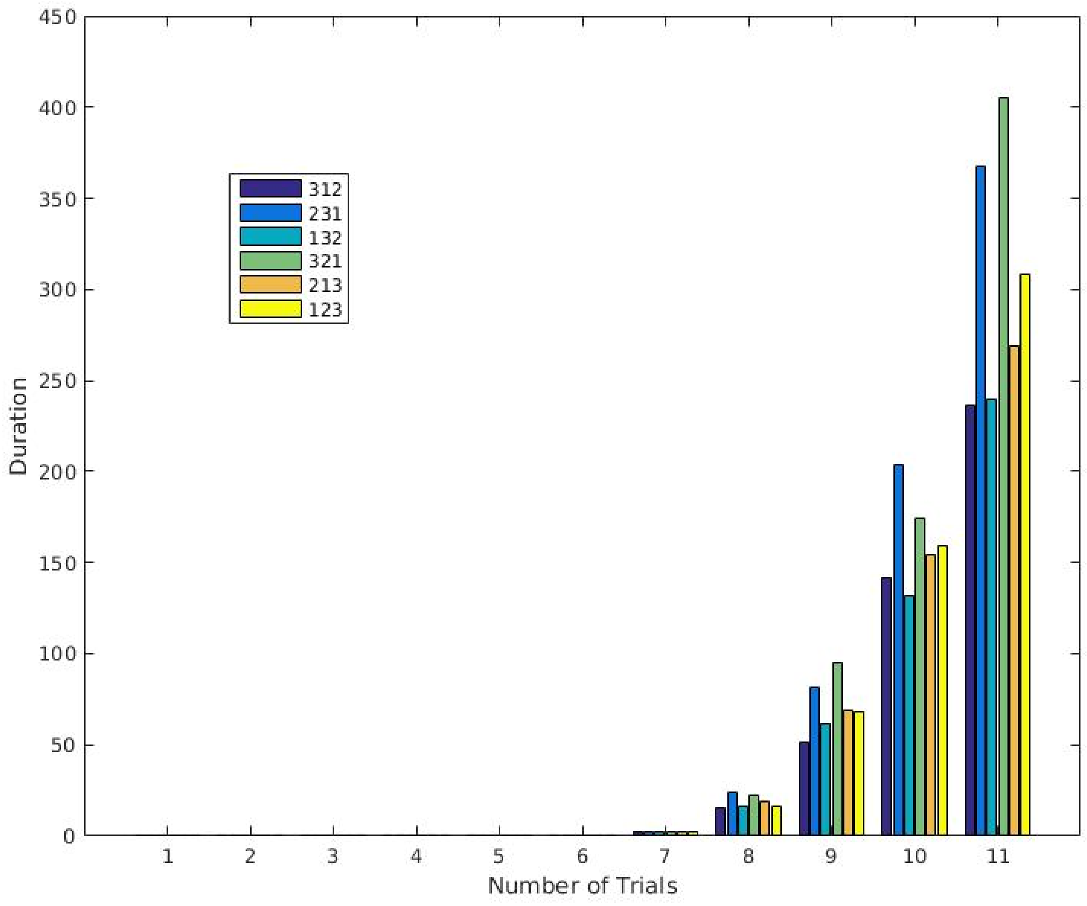
PROCESS

Created square matrices using MATLAP 'rand' function with eleven different sizes.

1. A(10 X 10) * B(10 X 10)
2. A(20 X 20) * B(20 X 20)
3. A(30 X 30) * B(30 X 30)
4. A(40 X 40) * B(40 X 40)
5. A(100 X 100) * B(100 X 100)
6. A(200 X 200) * B(200 X 200)
7. A(500 X 500) * B(500 X 500)
8. A(1000 X 1000) * B(1000 X 1000)
9. A(1500 X 1500) * B(1500 X 1500)
10. A(2000 X 2000) * B(2000 X 2000)
11. A(2500 X 2500) * B(2500 X 2500)

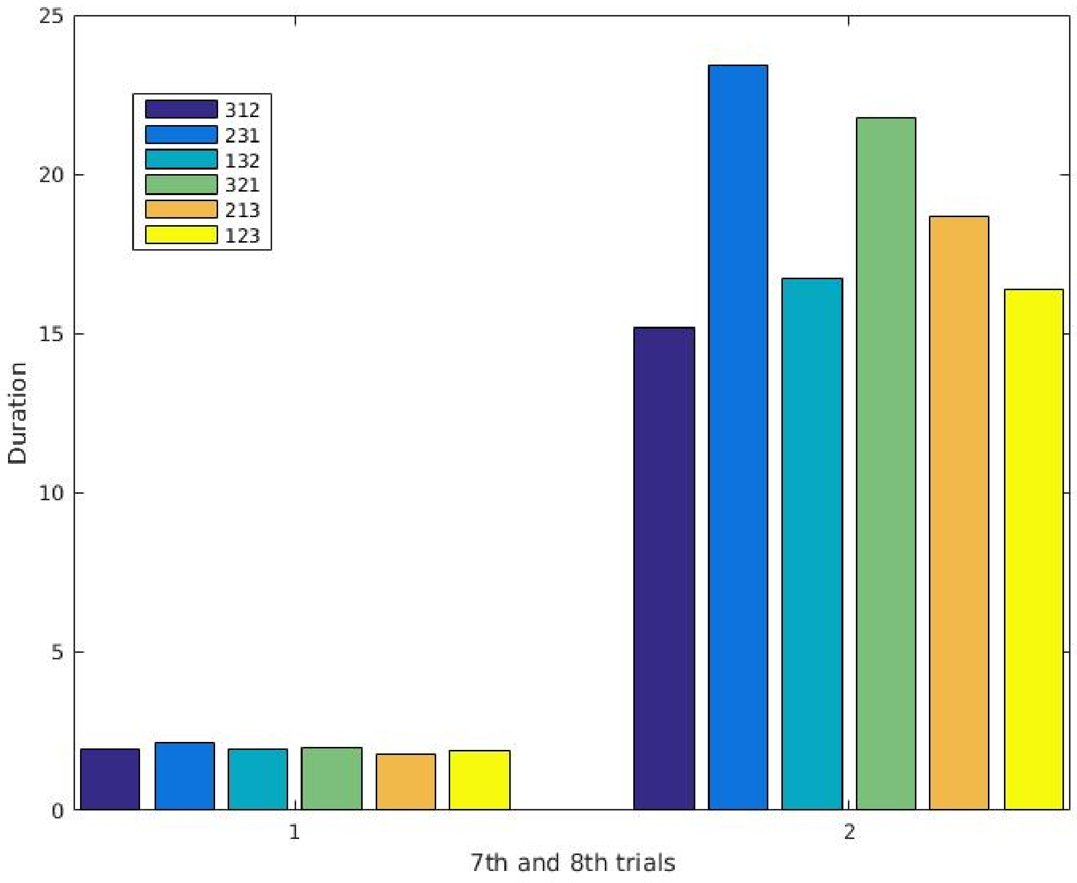
tested all multiplication in six different ways.

RESULT



The colors are on the figure indicates version of “for loops”. Colors' number shows us order of loops. According to this bar graph, there is no visible difference to 8th test.

If we look more closely at the seventh and eighth test results, we can see the differences.



According to graph, almost ten times 8th experiment takes almost ten times more longer time than 7th.

As we know computers have L1 and L2 caches and also RAM. L1-cache is the fastest cache and it usually comes within the processor chip itself. The L1 cache typically ranges in size from 8KB to 64KB and uses the high-speed SRAM (static RAM) instead of the slower and cheaper DRAM (dynamic RAM) used for main memory.

L2 cache comes between L1 and RAM(processor-L1-L2-RAM) and is bigger than the primary cache (typically 64KB to 4MB).

Basically, if our data cannot be stored in L1 and L2 cache, multiplication can take really long time. Matrix size is 500X500 In 7th test, however, In 8th test 1000X1000. According to this explanation, In 8th test, caches cannot store all data in one time during processing. Besides, the durations of each version are different. This reason is that system reads data from memory. After changing the order of “for loop”, Matlab reads the data from different memory addresses.

HW - 1.2 PURPOSE

Prove that the inverse of an Mi matrix is obtained by negating the off- diagonal elements in the ith column. That is, no operations, except negation, are needed to compute the inverse.

$$M_i^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & -a & 1 & \cdots & 0 \\ 0 & 0 & -b & 0 & \cdots & 0 \\ 0 & 0 & -c & 0 & \cdots & 0 \\ \vdots & & & & & \vdots \\ 0 & 0 & -p & 0 & \cdots & 1 \end{bmatrix}$$

For this purpose we will use following matrix. If the rule is correct, negating the off- diagonal elements should be enough operation to inverse the matrix M1.

$$M_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -2 & 1 & 0 & 0 \\ -3 & 0 & 1 & 0 \\ -4 & 0 & 0 & 1 \end{bmatrix} \qquad M_1^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ 3 & 0 & 1 & 0 \\ 4 & 0 & 0 & 1 \end{bmatrix}$$

Multiblication in Matlab and result is M1 * M2 = `I (Identity Matrix).`

PROOF

First of all, we should explain that

$$X = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ \boxed{x_{21} \quad \mathbf{0} \quad \mathbf{0} \quad \cdots \quad \mathbf{0}} \\ x_{31} & x_{32} & \mathbf{0} & \cdots & \mathbf{0} \\ . & & & & \\ . & & & & \\ \boxed{x_{n1} \quad x_{n2} \quad \cdots \quad \ldots \quad \mathbf{0}} \end{pmatrix} \qquad Y = \begin{pmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ y_{21} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \\ y_{31} & y_{32} & \mathbf{0} & \cdots & \mathbf{0} \\ . & & & & \\ . & & & & \\ y_{n1} & y_{n2} & \cdots & \ldots & \mathbf{0} \end{pmatrix}$$

X*Y always going to be zero matrix.

Lets remember what we said,

$$M^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ m_{21} & 1 & 0 \\ m_{31} & 0 & 1 \end{bmatrix} \qquad \textit{it means} \quad M^{-1} = I + A$$

$$M = \begin{bmatrix} 1 & 0 & 0 \\ -m_{21} & 1 & 0 \\ -m_{31} & 0 & 1 \end{bmatrix} \qquad \textit{it means} \quad M = I - A$$

$M^{-1} * M \quad = (I + A)(I – A)$

$= II – IA + AI - AA$

$= I$

We explained -AA is always going to be zero

– IA + AI = A – A = 0

Replacing M and M^{-1}

$M * M^{-1} \quad = (I - A)(I + A)$

$= II + IA - AI - AA$

$= I$

HW - 1.3 PURPOSE

Prove that the product of Mi and Mj matrices, where i < j, is obtained by simply adding the off diagonal elements. That is, Mi Mj = Mi \ Mii + Mj \ Mjj + I. This means that when two M matrices are multiplied (while the one with bigger index being on the right) the result is simply the addition of the off diagonal elements while keeping the diagonal elements out and adding the identity matrix to the addition of off diagonal elements.

For this purpose we will use following matrices.

$$M_1 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -2 & 1 & 0 & 0 \\ -3 & 0 & 1 & 0 \\ -4 & 0 & 0 & 1 \end{bmatrix} \qquad M_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 2 & 1 & 0 \\ 0 & 2 & 0 & 1 \end{bmatrix}$$

Basically we will sum following two matrices.

```
   1     0     0     0          0     0     0     0
  -2     1     0     0          0     0     0     0
  -3     0     1     0    +     0     2     0     0
  -4     0     0     1          0     2     0     0
```

Result

```
   1     0     0     0
  -2     1     0     0
  -3     2     1     0
  -4     2     0     1
```

We tested M_1 and M_2 multiplication using Matlab and result is the same

M_1 * M_2 =

```
   1     0     0     0
  -2     1     0     0
  -3     2     1     0
  -4     2     0     1
```

PROOF

First of all, we should explain that

$$A_1 = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 \\ \boxed{x_{21} \quad 0 \quad 0 \quad \cdots \quad 0} \\ x_{31} & x_{32} & 0 & \cdots & 0 \\ . & & & & \\ . & & & & \\ \boxed{x_{n1} \quad x_{n2} \quad \cdots \quad \ldots \quad 0} \end{pmatrix} \qquad A_2 = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 \\ y_{21} & 0 & 0 & \cdots & 0 \\ y_{31} & y_{32} & 0 & \cdots & 0 \\ . & & & & \\ . & & & & \\ y_{n1} & y_{n2} & \cdots & \ldots & 0 \end{pmatrix}$$

X*Y always going to be zero matrix.

M_1 and M_2 are lower triangular identity matrix

I = Identity Matrix

$A_1 = M_1 - I$

$A_2 = M_2 - I$

We explained A_1A_2 is always going to be zero

$$M_1 * M_2 = (A_1 + I)(A_2 + I) = A_1A_2 + IA_1 + IA_2 + II$$

$$= 0 + A_1 + A_2 + I$$

$$\mathbf{M_1 * M_2} = A_1 + A_2 + I$$