Numerical Computation of Inverse Matrices

Danny Philantrope

i.Contents

Contents

The Inverse

In linear algebra, an n-by-n square matrix **A** is called **invertible** if there exists an n-by-n square matrix **B** such that

$$\mathbf{A} \times \mathbf{B} = \mathbf{B} \times \mathbf{A} = \mathbf{I}_n$$

- A matrix inverse is similar to the reciprocal of a number.

The reciprocal of **5** is **1/5**, also denoted as $\mathbf{5^{-1}}$

$$\mathbf{5} \times \mathbf{5^{-1}} = \mathbf{1}$$

<u>similarly,</u>

$$\mathbf{M} \times \mathbf{M^{-1}} = \mathbf{I}$$

Inversion Methods

One of the simplest methods of inverting a matrix is the following formula:

$$A \equiv \begin{bmatrix} a & b \\ c & d \end{bmatrix} \qquad A^{-1} = \frac{1}{|A|}\begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

$$|A| = ad - bc$$

$$A = \begin{bmatrix} 4 & 7 \\ 2 & 6 \end{bmatrix} \qquad A^{-1} = \frac{1}{4*6-7*2}\begin{bmatrix} 6 & -7 \\ -2 & 4 \end{bmatrix} \qquad A^{-1} = \begin{bmatrix} 0.6 & -0.7 \\ -0.2 & 0.4 \end{bmatrix}$$

$$|A| = 10$$

Gauss-Jordan Method

- The original matrix and the identity matrix is stored in an augmented structure.
- The original matrix is reduced to an identity matrix and as a result the identity is reduced to the original's inverse.

*the matrices are reduced using row operations

Row operations involve:

1. Row swapping
2. Rows being multiplied by a factor
3. Multiples of rows being added or subtracted from another row

an example of the procedure follows…

Example

Write the augmented matrix

№	A_1	A_2	A_3	A_1	A_2	A_3
1	3	6	4	1	0	0
2	1	1	4	0	1	0
3	7	2	9	0	0	1

Find the pivot in the 1st column and swap the 2nd and the 1st rows

№	A_1	A_2	A_3	A_1	A_2	A_3
1	1	1	4	0	1	0
2	3	6	4	1	0	0
3	7	2	9	0	0	1

Eliminate the 1st column

№	A_1	A_2	A_3	A_1	A_2	A_3
1	1	1	4	0	1	0
2	0	3	-8	1	-3	0
3	0	-5	-19	0	-7	1

.continued

Make the pivot in the 2nd column by dividing the 2nd row by 3

№	A_1	A_2	A_3	A_1	A_2	A_3
1	1	1	4	0	1	0
2	0	1	-8/3	1/3	-1	0
3	0	-5	-19	0	-7	1

Eliminate the 2nd column

№	A_1	A_2	A_3	A_1	A_2	A_3
1	1	0	20/3	-1/3	2	0
2	0	1	-8/3	1/3	-1	0
3	0	0	-97/3	5/3	-12	1

Make the pivot in the 3rd column by dividing the 3rd row by -97/3

№	A_1	A_2	A_3	A_1	A_2	A_3
1	1	0	20/3	-1/3	2	0
2	0	1	-8/3	1/3	-1	0
3	0	0	1	-5/97	36/97	-3/97

Eliminate the 3rd column

№	A_1	A_2	A_3	A_1	A_2	A_3
1	1	0	0	1/97	-46/97	20/97
2	0	1	0	19/97	-1/97	-8/97
3	0	0	1	-5/97	36/97	-3/97

answer

There is the inverse matrix on the right

№	A_1	A_2	A_3	A_1	A_2	A_3
1	1	0	0	1/97	-46/97	20/97
2	0	1	0	19/97	-1/97	-8/97
3	0	0	1	-5/97	36/97	-3/97

LU Decomposition

The idea is to:

- break a matrix into its L and U factors
- find the inverses of L and U
- then multiply their inverses to obtain the inverse of the original matrix

$$A = L \times U$$

$$U^{-1} \times L^{-1} = A^{-1}$$

L and U are triangular matrix factors of the original matrix.

$$\begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$$

* they can be used to solve linear equations easily with the use of substitution methods.

Forwards & Back Substitution

The substitution process solves an unknown vector, using an upper or lower triangular matrix and a given vector.

ex:

$$\begin{bmatrix} 2 & 3 & 4 \\ 0 & 1 & 5 \\ 0 & 0 & 6 \end{bmatrix} \begin{bmatrix} u_1^{-1} \\ u_2^{-1} \\ u_3^{-1} \end{bmatrix} = \begin{bmatrix} I_1 \\ I_2 \\ I_3 \end{bmatrix}$$

$2\ u_1^{-1} + 3\ u_2^{-1} + 4\ u_3^{-1} = 0$

$u_2^{-1} + 5\ u_3^{-1} = 0$

$6\ u_3^{-1} = 1$

$\mathbf{u_3^{-1}} = 1/6$

$\mathbf{u_2^{-1}} = 0 - 5\ (1/6) = -5/6$

$\mathbf{u_1^{-1}} = [\ 0 - 4\ (1/6) - 3\ (-5/6)\ ] / 2 = 11/12$

The substitution method consists solely of equations, which suggests that it may be slightly faster than Gauss Jordan.

And by that idea we decided to implement it and test the results.

Implementation

We made our program with C++
using:

2-dimensional arrays
square matrix class
identity function
LU decomposition function
inverse functions (LU, Gauss Jordan)

Algorithm Performance

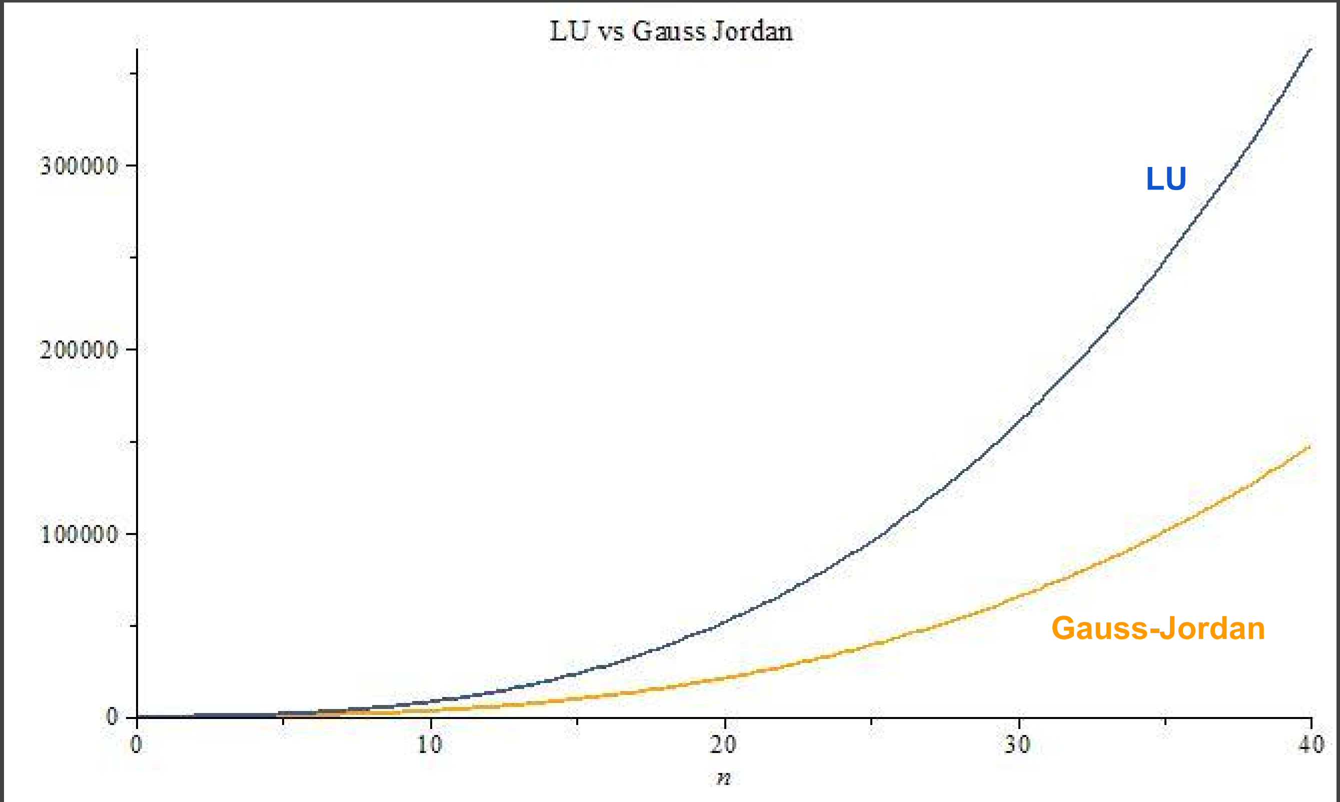
Gauss-Jordan

$$2n^3 + 12n^2 + 9n + 10 \rightarrow O(n^3)$$

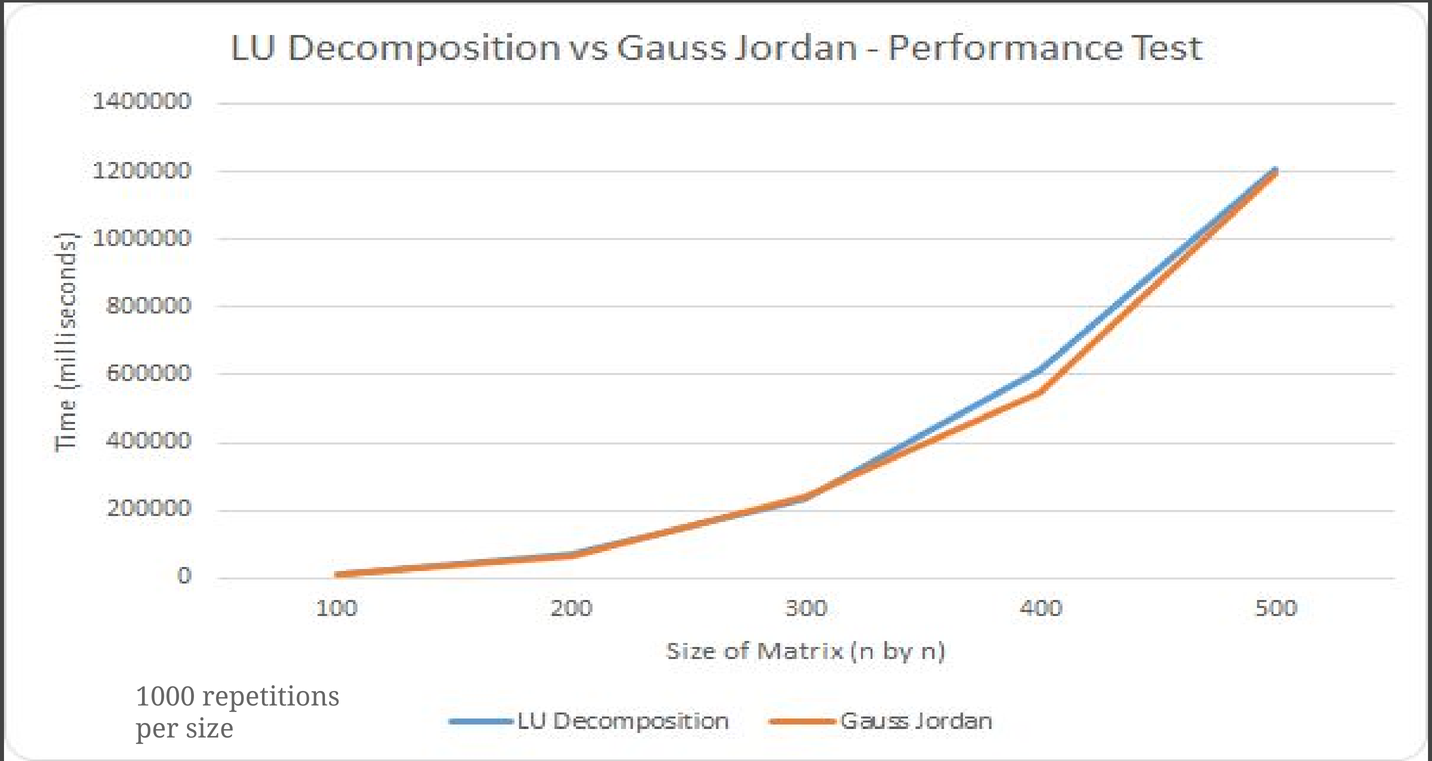
LU Decomposition

$$5n^3 + 26n^2 + 48n + 49 \rightarrow O(n^3)$$

theoretical



actual


LU Decomposition vs Gauss Jordan - Performance Test
1400000
1200000
1000000
800000
600000
400000
200000
0
Time (milliseconds)
100
200
300
400
500
Size of Matrix (n by n)
1000 repetitions
per size
LU Decomposition
Gauss Jordan

Error

We found that on average the code would fail 3.2% of runs

We researched some potential problems that might occur

1. Numerical failure
2. LU inverse does not exist
3. Matrix is singular

Numerical error happens when the amount of digits of a number surpasses the scope of what is allowed by the memory. We can fix this with scaled partial pivoting:

- scaling down the numbers of the rows and columns in a fixed manner
- only dividing by larger numbers so the numbers are accommodated by the memory

A matrix not having an LU decomposition would also be a problem.

It can be fixed by factoring a matrix into LU with a permutation matrix.

$$PA = LU$$

An LU decomposition always exists in this method.

Further Research & Improvements

Scaled partial pivoting with the inverse methods to fix numerical error.

P matrix to overcome LU existence problem.

References

http://mathforcollege.com/nm/simulations/nbm/04sle/nbm_sle_sim_inversecomptime.pdf

https://www.math.ust.hk/~mamu/courses/231/Slides/CH06_2A.pdf