# Matrix inverse with LU decomposition in C++

Comp Sci · KWKWII · Oct 11, 2014

**Oct 11, 2014** #1

**KWKWII**

## Homework Statement

the problem is to find the inverse of a 3x3 matrix using LU Decomposition with C++ command, with the numbers designated. in my case, my numbers for the matrix are
'306
410
780'

Code:

```
#include <stdio.h>
#include <iostream>
#include <stdlib.h>
#include <math.h>
using namespace std;

int main(void)
{
    int e, i, j, k, y;
    float A[3][4] = {{3,0,6,1},{4,1,0,2},{7,8,0,3}};
    float x[3][3], c, sum;
    float L[3][4] = {{1,0,0,0},{0,1,0,0},{0,0,1,0}};
        for(j=0; j<=2; j++)
        {
            for(i=0; i<=2; i++)
            {
                if(i>j)
                {
```

```
                    c=A[i][j]/A[j][j];
                    L[i][j] = c;
                    for(k=0; k<=2; k++)
                    {
                        A[i][k]=A[i][k]-c*A[j][k];
                    }
                }
            }
        }

        for(y=0; y<=2; y++)
        {
            if(y==0)
            {
                    L[0][3] = 1;
                L[1][3] = 0;
                L[2][3] = 0;
            }
            else if (y==1)
            {
                    L[0][3] = 0;
                L[1][3] = 1;
                L[2][3] = 0;
            }
            else if (y==2)
            {
                    L[0][3] = 0;
                L[1][3] = 0;
                L[2][3] = 1;
            }
            A[0][3]=L[0][3]/L[0][0];
            for(i=1; i<=2; i++)
            {
                sum=0;
                for(j=i-1; j>=0; j--)
                {
                    sum=sum+L[i][j]*A[j][3];
                }
                A[i][3]=(L[i][3]-sum)/L[i][i];
            }
                x[2][y]=A[2][3]/A[2][2];
                for(i=1; i>=0; i--)
                {
```

```
                    sum=0;
                    for(j=i+1; j<=2; j++)
                    {
                        sum=sum+A[i][j]*x[j][y];
                    }
                    x[i][y]=(A[i][3] - sum)/A[i][i];
                }
        }
    cout<<"Matriks X"<<endl;
    for(int i=0;i<=2;i++)
    {
        for(int j=0;j<=2;j++)
        {
            cout<< x[i][j]<<"  ";
        }
    cout<<endl;
    }
    return(0);
}
```

## Homework Equations

## The Attempt at a Solution

the result from the program will show most numbers correct, except the numbers on $X_{11}$ and $X_{21}$

it's supposed to be
$A_{11} = A_{21} = 0$
but it became
$A_{11}$ = -3.97364e-08
$A_{21}$ = 1.19209e-07

with loop, i can find out the numbers from each matrix, and all of them are correct

only the calculations from the X matrix seems wrong
need help in making $A_{11}$ and $A_{21}$ become 0

Reply

## Answers and Replies

**Oct 11, 2014** #2

**Mark44**

Mentor

Is this the same as the one you posted yesterday? If so, do not repost a problem just because no one has yet responded.

KWKWII said:

# Homework Statement

the problem is to find the inverse of a 3x3 matrix using LU Decomposition with C++ command, with the numbers designated. in my case, my numbers for the matrix are
'306
410
780'

Code:

```
#include <stdio.h>
#include <iostream>
#include <stdlib.h>
#include <math.h>
using namespace std;

int main(void)
{
    int e, i, j, k, y;
    float A[3][4] = {{3,0,6,1},{4,1,0,2},{7,8,0,3}};
    float x[3][3], c, sum;
    float L[3][4] = {{1,0,0,0},{0,1,0,0},{0,0,1,0}};
        for(j=0; j<=2; j++)
        {
```

```
            for(i=0; i<=2; i++)
            {
                if(i>j)
                {
                    c=A[i][j]/A[j][j];
                    L[i][j] = c;
                    for(k=0; k<=2; k++)
                    {
                        A[i][k]=A[i][k]-c*A[j][k];
                    }
                }
            }
        }

        for(y=0; y<=2; y++)
        {
            if(y==0)
            {
                    L[0][3] = 1;
                L[1][3] = 0;
                L[2][3] = 0;
            }
            else if (y==1)
            {
                    L[0][3] = 0;
                L[1][3] = 1;
                L[2][3] = 0;
            }
            else if (y==2)
            {
                    L[0][3] = 0;
                L[1][3] = 0;
                L[2][3] = 1;
            }
            A[0][3]=L[0][3]/L[0][0];
            for(i=1; i<=2; i++)
            {
                sum=0;
                for(j=i-1; j>=0; j--)
                {
                    sum=sum+L[i][j]*A[j][3];
                }
                A[i][3]=(L[i][3]-sum)/L[i][i];
```

```
            }
                x[2][y]=A[2][3]/A[2][2];
                for(i=1; i>=0; i--)
                {
                    sum=0;
                    for(j=i+1; j<=2; j++)
                    {
                        sum=sum+A[i][j]*x[j][y];
                    }
                    x[i][y]=(A[i][3] - sum)/A[i][i];
                }
        }
    cout<<"Matriks X"<<endl;
    for(int i=0;i<=2;i++)
    {
        for(int j=0;j<=2;j++)
        {
            cout<< x[i][j]<<"  ";
        }
    cout<<endl;
    }
    return(0);
```

## Homework Equations

## The Attempt at a Solution

the result from the program will show most numbers correct, except the numbers on $X_{11}$ and $X_{21}$

it's supposed to be
$A_{11} = A_{21} = 0$
but it became
$A_{11}$ = -3.97364e-08
$A_{21}$ = 1.19209e-07

with loop, i can find out the numbers from each matrix, and all of them are correct

only the calculations from the X matrix seems wrong
need help in making $A_{11}$ and $A_{21}$ become 0

All of your matrix variables are type float, which can hold only about 6 or 7 decimal places. You will have a lot more precision if you make them double.

Reply

**Oct 11, 2014** #3

**KWKWII**

> Mark44 said:
>
> do not repost a problem just because no one has yet responded

it is a repost, but i think i need to because when i re-read the guidelines, i must use proper English
which, i think, is not proper on my last post
so i tried to simplify my words on the new post (here)

and by using double for all matrix variables, it indeed became more precise, but also became worse
after changing float -> double
$A_{11}$ = 7.40149e-17
$A_{21}$ = -2.22045e-16

Reply

**Oct 11, 2014** #4

**Mark44**

Mentor

> KWKWII said:
>
> it is a repost, but i think i need to because when i re-read the guidelines, i must use proper English
> which, i think, is not proper on my last post

> so i tried to simplify my words on the new post (here)

Your English was fine in the other post. If you need to edit a post, and you're no longer able to do it, report the post using the Report button, and a mentor will take care of it.

> KWKWII said:
>
> and by using double for all matrix variables, it indeed became more precise, but also became worse
> after changing float -> double
> $A_{11}$ = 7.40149e-17
> $A_{21}$ = -2.22045e-16

Those aren't worse. They're much closer to 0 than the results when you used float variables.

Floating point arithmetic on computers is inherently imprecise due to roundoff and truncation errors.

Reply

**Oct 11, 2014** #5

**KWKWII**

> Mark44 said:
>
> If you need to edit a post, and you're no longer able to do it, report the post using the Report button, and a mentor will take care of it

oh, i thought it wasn't fine...
i see, i'll do so next time

and also
although it is closer to 0, it is not 0
and the actual answer (done manually) is supposed to be 0
tried changing into integer, but it's even worse than using float or double
because the solution become numbers without decimals
is it also because of the rounding and truncation errors?

and i also notice that there's a minus on $A_{21}$ and no minus on $A_{11}$
a weird change considering the previous one is reversed

Reply

**Oct 11, 2014** #6

**Mark44**

Mentor

You can set the precision of the numbers you display using the precision function in the iosbase class. Look at the example code on this page: http://www.cplusplus.com/reference/ios/ios_base/precision/.

Reply

**Oct 11, 2014** #7

**KWKWII**

> Mark44 said:
>
> using the precision function

Code:

```
#include <iomanip>

cout<< setprecision(3) << fixed << x[i][j]<<"  ";
```

(kind of different, but it works the same
i set it to only spell out 3 digits behind . )

using that, the result now becomes

$A_{11}$ = -0.000
$A_{21}$ = 0.000

well, although it's still weird for 0 to have (-) signs
but at least better for it shows the (seemingly) same amount as the actual answer

thanks

Reply

**Oct 11, 2014** #8

**Mark44**

Mentor

The setprecision function is just truncating (or possibly rounding - I don't know) the fractional parts of numbers like 0.000000587 and -0.000342, and displaying only the first three decimal places.

Reply

Write your reply...

LaTeX Guide | BBcode Guide

**Post reply**

## Related Threads on Matrix inverse with LU decomposition in C++

Lu Decomposition in C
Last Post
Replies 1

Comp Sci LU Decomposition in Java
Last Post
Replies 6

Matlab debugging for LU decomposition
Last Post
Replies 1

Inverse of a matrix
Last Post
Replies 4

Comp Sci Matrix diagonalization in c++
Last Post
Replies 2

Matrix inversion with complex numbers? or faster way?
Last Post
Replies 13

Matrix in C program help
Last Post
Replies 4

Comp Sci Fortran77 program to find the inverse of a matrix
Last Post
Replies 1

C matrix multiplication
Last Post
Replies 7

What is the inverse of the 3x3 matrix mod 26
Last Post
Replies 3



## » HOT THREADS

How replace numeral in string 'abc1def' w/ an array element?

Finding out the input impedance in a negative feedback op-amp

AC Circuit Phasors

Electrical and Control Engineering: Transfer Function Reduction problem

Kirchoff's Law Problem about this Induction Generator circuit

## » RECENT INSIGHTS

How to Setup a Raspberry Pi Cluster

An Introduction to Theorema Primum

Python's Sympy Module and the Cayley-Hamilton Theorem