Recently, I wrote a blog-post on how to perform Gaussian Elimination to reduce a matrix to the echelon form and solve a system of linear equations.
However, it has a few further applications.
Gauss Elimination can be used to :
1. LU decompose a matrix.
2. Find the inverse.
3. Calculate the determinant.

In this post I show you how to calculate the determinant using Gauss elimination.

The process of Gaussian Elimination converts the given matrix into an Upper Triangular matrix U. Now the good thing about triangular matrices is that their determinant is equal to the product of the elements on the diagonal. Another thing to note is that this procedure of Gaussian elimination gives us another matrix L, which is lower triangular and has unit diagonal entries (I will write another post about LU decomposition). So its determinant is effectively 1.
Now the best part is that the product of L and U gives us a permutation of the original matrix A.
What I mean by permutation of A is that the rows are the same as the original matrix A but their order is changed.
Now with all this information the determinant can be easily calculated.
The determinant is simply equal to $det(A) = (-1)^m det(L) * det(U)$ where m is the number of row inter-changes that took place for pivoting of the matrix, during Gaussian elimination. Since the determinant changes sign with every row/column change we multiply by $(-1)^m$.
Also since the L has only unit diagonal entries it's determinant is equal to one.
So all we need is the determinant of U and m.
Therefore,
The following code does all of this and prints the determinant.

CODE:

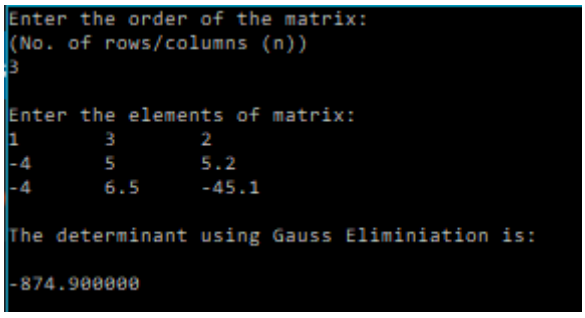
```
/************************************************
******DETERMINANT FROM GAUSS ELIMINATION***********
************************************************/
#include<stdio.h>
#include<math.h>
/*******
Function that calculates the determinant of a square matrix using Gauss-Elimination :
Pass the square matrix as a parameter, and calculate and return the dete
Parameters: order(n),matrix[n][n]
********/
double determinant(int n, double a[n][n]){
        double det=1;
        int i;
        int swapCount=gaussElimination(n,n,a);
        for(i=0;i<n;i++){
                det =det*a[i][i];
        }
        return det*pow(-1,swapCount);
}
/********
Function that perform Gauss Elimination
Pass the square matrix as a parameter, and calculate and store the
upperTriangular(Gauss-Eliminated Matrix) in it
```

```
Parameters: rows(m),columns(n),matrix[m][n]
********/
int gaussElimination(int m, int n, double a[m][n]){
        int i,j,k;
        int swapCount=0;
        for(i=0;i<m-1;i++){
                //Partial Pivoting
                for(k=i+1;k<m;k++){
                        //If diagonal element(absolute vallue) is smaller than any of
the terms below it
                        if(fabs(a[i][i])<fabs(a[k][i])){
                                //Swap the rows
                                swapCount++;
                                for(j=0;j<n;j++){
                                        double temp;
                                        temp=a[i][j];
                                        a[i][j]=a[k][j];
                                        a[k][j]=temp;
                                }
                        }
                }
                //Begin Gauss Elimination
                for(k=i+1;k<m;k++){
                        double  term=a[k][i]/ a[i][i];
                        for(j=0;j<n;j++){
                                a[k][j]=a[k][j]-term*a[i][j];
                        }
                }
        }
        return swapCount;
}
/*******
Function that reads the elements of a matrix row-wise
Parameters: rows(m),columns(n),matrix[m][n]
*******/
void readMatrix(int m, int n, double matrix[m][n]){
        int i,j;
        for(i=0;i<m;i++){
                for(j=0;j<n;j++){
                        scanf("%lf",&matrix[i][j]);
                }
        }
}
/*******
Function that prints the elements of a matrix row-wise
Parameters: rows(m),columns(n),matrix[m][n]
*******/
void printMatrix(int m, int n, double matrix[m][n]){
        int i,j;
        for(i=0;i<m;i++){
                for(j=0;j<n;j++){
                        printf("%lf\t",matrix[i][j]);
```

```
                }
                printf("\n");
        }
}
/*******
Function that copies the elements of a matrix to another matrix
Parameters: rows(m),columns(n),matrix1[m][n] , matrix2[m][n]
*******/
void copyMatrix(int m, int n, double matrix1[m][n], double matrix2[m][n]){
        int i,j;
        for(i=0;i<m;i++){
                for(j=0;j<n;j++){
                        matrix2[i][j]=matrix1[i][j];
                }
        }
}

int main(){
        int n,i,j;
        printf("Enter the order of the matrix:\n(No. of rows/columns (n))\n");
        scanf("%d",&n);
        //Declare a matrix to store the user given matrix
        double a[n][n];
        printf("\nEnter the elements of matrix:\n");
        readMatrix(n,n,a);
        printf("\nThe determinant using Gauss Eliminiation
is:\n\n%lf\n",determinant(n,a));
}
```

OUTPUT:



Sample Run

Android Apps:

I've also created a few Android apps that perform various matrix operations and can come in handy to those taking a course on Numerical Methods.
Download: https://play.google.com/store/apps/details?id=com.bragitoff.numericalmethods
Download: https://play.google.com/store/apps/details?id=com.bragitoff.matrixcalculator

References:

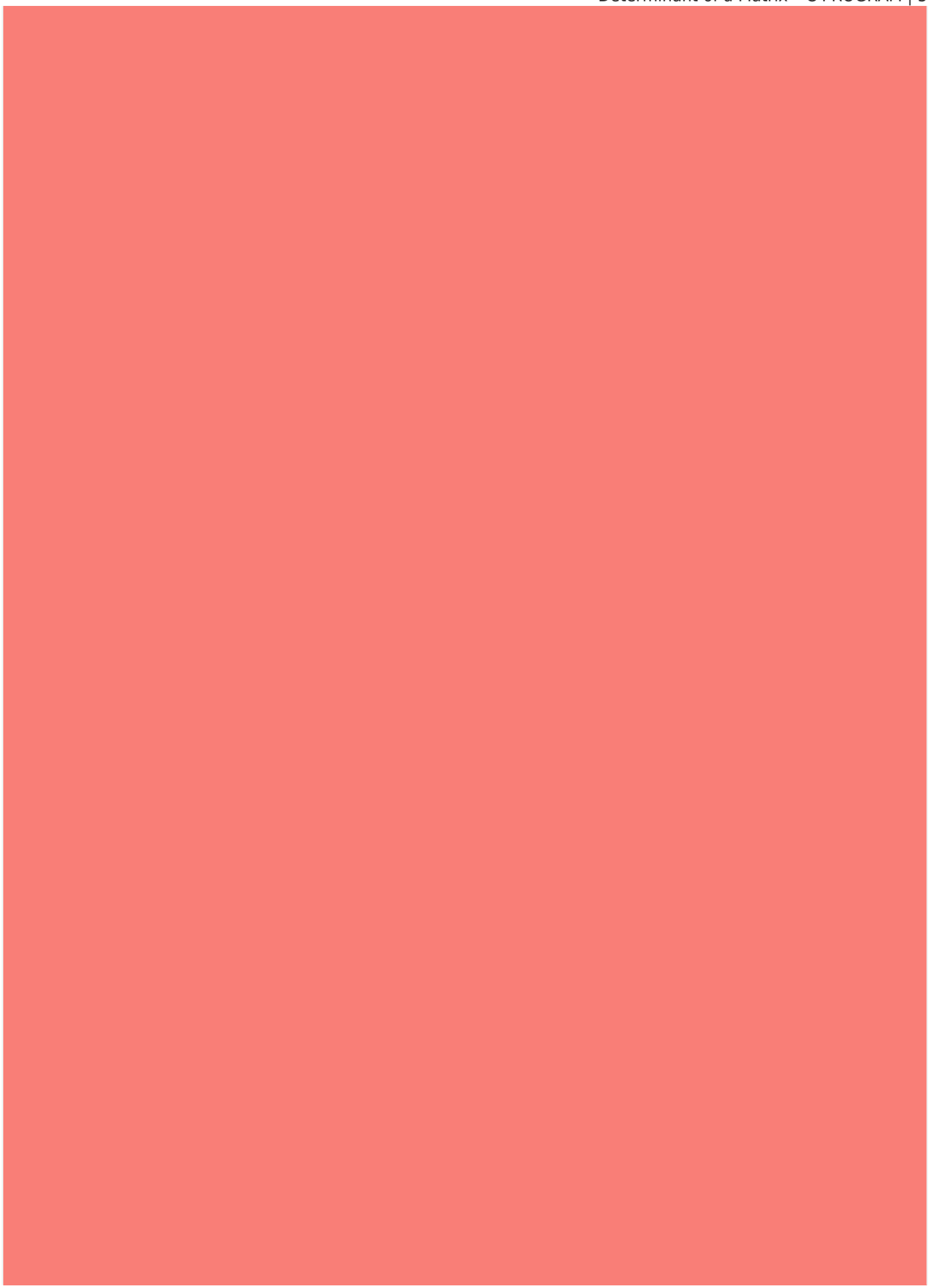
https://en.wikipedia.org/wiki/Gaussian_elimination#Computing_determinants

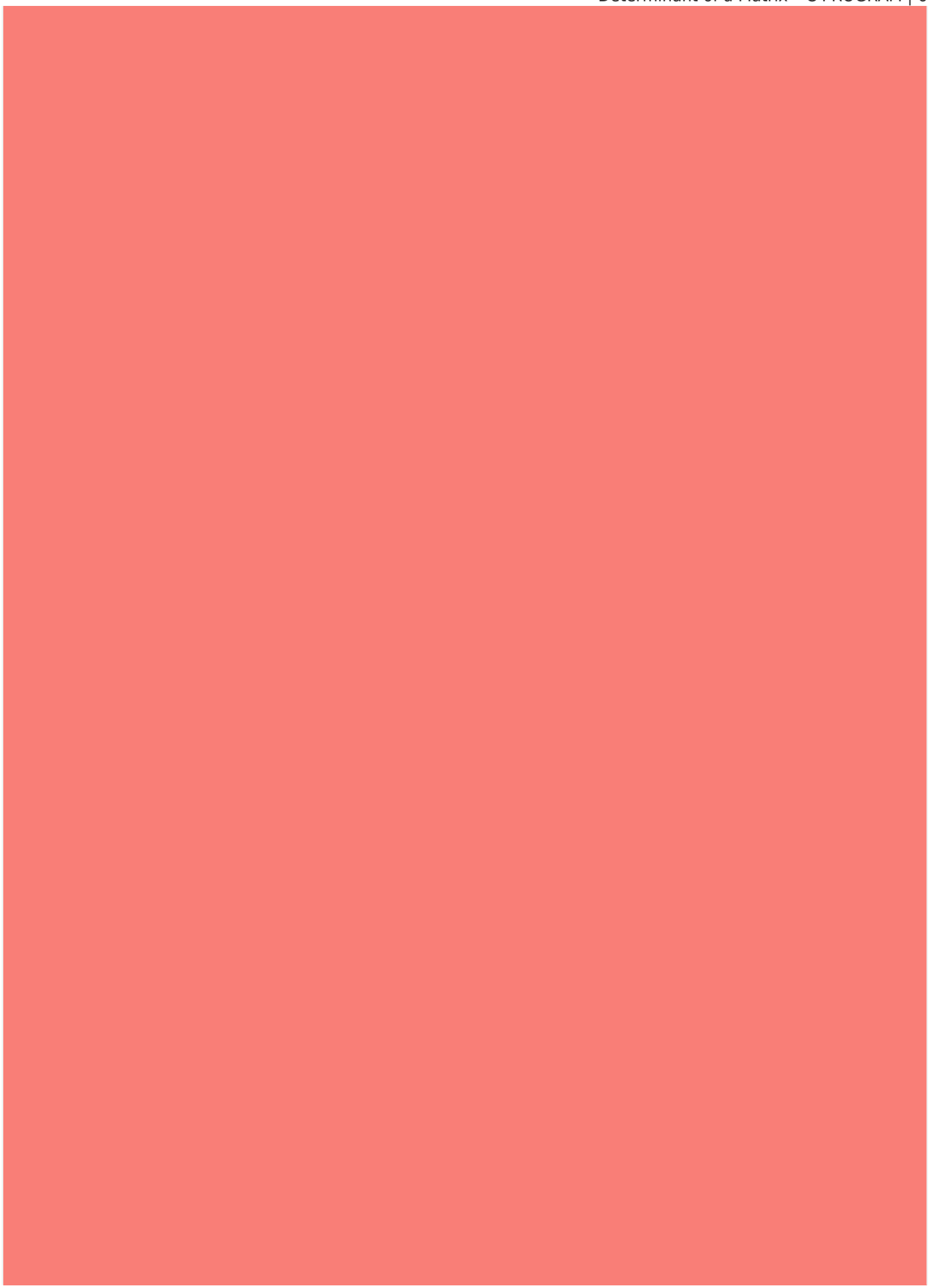
https://en.wikipedia.org/wiki/Gaussian_elimination
http://mathworld.wolfram.com/GaussianElimination.html

Manas Sharma

I'm a physicist specializing in computational material science with a PhD in Physics from Friedrich-Schiller University Jena, Germany. I write efficient codes for simulating light-matter interactions at atomic scales. I like to develop Physics, DFT, and Machine Learning related apps and software from time to time. Can code in most of the popular languages. I like to share my knowledge in Physics and applications using this Blog and a YouTube channel.

manas.bragitoff.com/





Share this:

Click to share on Facebook (Opens in new window)

Click to share on Twitter (Opens in new window)

Click to share on WhatsApp (Opens in new window)

Click to share on Pinterest (Opens in new window)

Click to share on Reddit (Opens in new window)

Click to share on LinkedIn (Opens in new window)

Click to email a link to a friend (Opens in new window)

Click to print (Opens in new window)

Click to share on Tumblr (Opens in new window)

Click to share on Pocket (Opens in new window)

Click to share on Telegram (Opens in new window)

[wpedon id="7041" align="center"]