Yusif Alakbarzade

Report

Description of Task:

In this project I have compute the **det (A) &** $\boldsymbol{Cond_F(A) = ||A||_F . ||A^{-1}||_F}$ (Determinant of Matrix A and Condition of Matrix A) using **Gaussian Elimination with Partial Pivoting (GEPP),** where A is Hessenberg matrix.

Hessenberg Matrix is a special type of square matrix, it's almost triangular matrix. There are two types of Hessenberg Matrix, Upper and Lower. Upper consists of zero entries below the first sub diagonal where Lower consists of zero entries below the first sub diagonal.

Gaussian Elimination

It's also known as Row Reduction Method used to solve the system of equations mainly. We can find out the determinant and inverse of Matrix using this.

Gaussian Elimination with Partial Pivoting

It's a method same as Gaussian Elimination Method but with Pivoting where **Pivoting** assists to reduce rounding errors, in this case you will not much add or subtract with very small or very large number.

Mathematical Description of Numerical Method:

Hessenberg Matrix

Examples of Hessenberg Matrix is shown below:

$$\begin{matrix} 7 & 0 & 0 \\ 3 & 4 & 0 \\ -1 & 9 & 2 \end{matrix}$$

Gaussian Elimination

Use Row Operations from these following three,

i. Swapping of two rows have a notation ↔, for example, R2 ↔ R3
ii. Multiplying a row by a nonzero number, for example, R1 → iR2 where i donates nonzero number
iii. Adding a multiple of one row to another row, for example, R2 → R2 + 3R1

After it we will obtain a Row Echelon form where matrix is called as reduced row-echelon form when all the leading coefficients equal 1. That form is unique.

Gaussian Elimination with Partial Pivoting

There are some steps of it,

1. Find the Pivot entry, It's the value in the left column with largest absolute value.
2. Go for Row Interchange if needed, keep in mind that pivot should be in the 1^{st} row
3. Now start Gaussian Elimination
4. Find new Pivot
5. Switch rows if needed
6. Repeat Step 3,4 and 5 till we got all the value in upper triangular form or lower triangular form.
7. Go for Back Substitution to find the results.

Numerical Example:

1. I have used above example of Hessenberg Matrix as a Matrix A. It's a Lower Hessenberg matrix of 3rd order.

```
Command Window
New to MATLAB? See resources for Getting Started.

A =

     7     0     0
     3     4     0
    -1     9     2
```

2. I have used another example of 4x4 Matrix. It's an Upper Hessenberg matrix of 4th order

```
Command Window
New to MATLAB? See resources for Getting Started.

A =

     1     2     5     7
     1     5     7     8
     0    -1    -2     6
     0     0     3     5
```

3. I have used another example of 3x3 Matrix. It's an Upper Hessenberg matrix of 3rd order

```
Command Window
New to MATLAB? See resources for Getting Started.

A =

     1     2     8
     1     3     4
     0     3     4
```

4. I have used above example of Hessenberg Matrix as a Matrix A. It’s a Lower Hessenberg matrix of 3rd order.

```
Command Window
New to MATLAB? See resources for Getting Started.

A =

     1     0     0
    -1    -4     0
    -1    -7     2
```

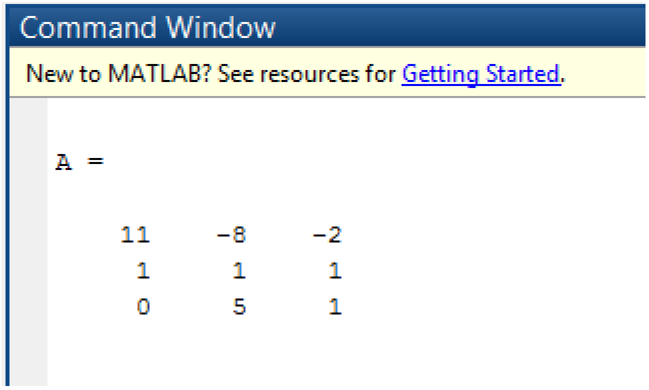
5. I have used another example of 4x4 Matrix. It’s an Upper Hessenberg matrix of 4th order

```
Command Window
New to MATLAB? See resources for Getting Started.

A =

    -3    10     6     7
    -1   -15     8    -8
     0     1    -2     5
     0     0     3     8
```

6. I have used another example of 3x3 Matrix. It’s an Upper Hessenberg matrix of 3rd order



Numerical Results:

I have got the results for each of above example, and it's shown in a table

Parameters	**Example 1**	**Example 2**	**Example 3**
Determinant of A	56	-77	16
Cond of A	13.7820	35.2929	15.2854
Norm of A	10.0261	15.9728	10.7231
Norm of A Inverse	1.3746	2.2096	1.4255
Result	13.7820	35.2929	15.2854

Parameters	**Example 4**	**Example 5**	**Example 6**
Determinant of A	-8	-1468	-46
Cond of A	10.4839	25.0364	18.4886
Norm of A	8.3621	21.8503	14.1133
Norm of A Inverse	1.2537	1.1458	1.3100
Result	10.4839	25.0364	18.4886

Where **Result** is equals to **Norm of A * Norm of A Inverse**

Command Window Results (MATLAB)

1. For Example 1

```
Command Window
New to MATLAB? See resources for Getting Started.

determinant =

    56


cond_of_matrix =

   13.7820


norm_of_matrix =

   10.0261


norm_of_matrix_inverse =

    1.3746


Result =

   13.7820

fx >>
```

2. For Example 2

```
Command Window
New to MATLAB? See resources for Getting Started.

determinant =

   -77


cond_of_matrix =

   35.2929


norm_of_matrix =

   15.9728


norm_of_matrix_inverse =

    2.2096


Result =

   35.2929

fx >>
```

3. For Example 3

```
Command Window
New to MATLAB? See resources for Getting Started.

determinant =

    16


cond_of_matrix =

   15.2854


norm_of_matrix =

   10.7231


norm_of_matrix_inverse =

    1.4255


Result =

   15.2854

fx >>
```

4. For Example 4

```
Command Window
New to MATLAB? See resources for Getting Started.

determinant =

    -8


cond_of_matrix =

   10.4839


norm_of_matrix =

    8.3621


norm_of_matrix_inverse =

    1.2537


Result =

   10.4839

fx >>
```

5. For Example 5

```
Command Window
New to MATLAB? See resources for Getting Started.

determinant =

       -1468


cond_of_matrix =

   25.0364


norm_of_matrix =

   21.8503


norm_of_matrix_inverse =

    1.1458


Result =

   25.0364

fx >>
```

6. For Example 6

```
Command Window
New to MATLAB? See resources for Getting Started.

determinant =

   -46


cond_of_matrix =

   18.4886


norm_of_matrix =

   14.1133


norm_of_matrix_inverse =

    1.3100


Result =

   18.4886

fx >>
```

Conclusion:

In this task we have used A – Hessenberg Matrix, and compute determinant of A and compute $\boldsymbol{Cond_F(A)} = ||\boldsymbol{A}||_{\boldsymbol{F}}.\left|\left|\boldsymbol{A^{-1}}\right|\right|_{\boldsymbol{F}}$

As shown in results for the first example, determinant of matrix is 56, I have calculated using det function as well, its 86 from it.

Cond of A is calculated as 13.7820 and then separately by multiplying the norm of inverse matrix and norm of matrix computes exactly 13.7820.

So, in a nutshell this method is fast, and **pivoting** reduce rounding errors and got the exact results.

For the Example 2 and 3, its exact same. As shown in above Table.