

## Question 1

The functions are defined in the `math425hw2.m` file, under the section `%% Question 1`.

### Part A

See `math425hw2.m`.

### Part B

Using `myPartialPivot`, `myRank` simply counts the number of non-zero pivots in the upper triangular matrix returned from `myPartialPivot`.

Note that, even with partial pivoting, the computation still suffer from the precision issue of floating-point arithmetic. For this reason, in the implementation for `myRank`, I have rounded pivot entries to the tenth decimal place to determine whether they are really zeroes.

For example, consider a matrix

$$A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}.$$

For the first pivot, we would row swap  $R_3$  and  $R_1$  and perform the row eliminations as follows.

$$\begin{pmatrix} 7 & 8 & 9 \\ 4 & 5 & 6 \\ 1 & 2 & 3 \end{pmatrix} \xrightarrow{-\frac{4}{7}R_1+R_2 \rightarrow R_2} \begin{pmatrix} 7 & 8 & 9 \\ 0 & 3/7 & 6/7 \\ 1 & 2 & 3 \end{pmatrix} \xrightarrow{-\frac{1}{7}R_1+R_3 \rightarrow R_3} \begin{pmatrix} 7 & 8 & 9 \\ 0 & 3/7 & 6/7 \\ 0 & 6/7 & 12/7 \end{pmatrix}$$

For the second pivot, we would row swap  $R_3$  and  $R_2$ , and perform the row as follows. This also happens to eliminate the third pivot. Thus, the rank is 2.

$$\begin{pmatrix} 7 & 8 & 9 \\ 0 & 6/7 & 12/7 \\ 0 & 3/7 & 6/7 \end{pmatrix} \xrightarrow{-\frac{1}{2}R_2+R_3 \rightarrow R_3} \begin{pmatrix} 7 & 8 & 9 \\ 0 & 6/7 & 12/7 \\ 0 & 0 & 0 \end{pmatrix}$$

The above should be our upper triangular matrix,  $U$ . However, due floating point error, the result is not quiet zero. In MATLAB, when we display  $U$  using `format rational`, it displays:

$$\begin{array}{ccc} 7 & 8 & 9 \\ 0 & 6/7 & 12/7 \\ 0 & * & * \end{array}$$

where the asterisks denote that the denominator is too large. By using `format long` we can see that the result is pretty much zero:

```

7.0000000000000000    8.0000000000000000    9.0000000000000000
0      0.857142857142857    1.714285714285714
0      0.0000000000000000    0.0000000000000000

```

Using `format longG` we can see that the stored values aren't quiet zero. Hence, I have rounded the digits to ten decimal places.

```

7      8      9
0      0.857142857142857    1.71428571428571
0      5.55111512312578e-17    1.11022302462516e-16

```

## Part C

Yes, the function virtually always returns 3. Since the question did not specify the bounds for the values from which a matrix could be generated, I simply used `rand(5, 3)` and `rand(3, 5)` to generate a  $5 \times 3$  and  $3 \times 5$  matrix, respectively. As such, it is virtually impossible for two or more rows to be identical or be a multiple of each other.

Furthermore, the floating-point precision issue discussed in part (b) would almost certainly ensure that the result appears to be full rank — especially when the values in the matrix are floating-point numbers to begin with — as values that should be zero may instead be interpreted as non-zero.

## Question 2

The relevant snippets are defined in the `math425hw2.m` file, under the section `% Question 2`.

### Part A

Let  $A = \begin{pmatrix} 4 & 1 & 1 & 1 \\ 1 & 4 & 1 & 1 \\ 1 & 1 & 4 & 1 \\ 1 & 1 & 1 & 4 \end{pmatrix}$ .  $A$  is clearly not diagonal and the sum of the off-diagonal entries in

each column is 3, and all the diagonal entries (pivots) are 4. Since the diagonal entries are greater than the sum of the absolute values of the off-diagonal entries in each column,  $A$  is a  $4 \times 4$  strictly column diagonally dominant matrix.

### Part B

In Gaussian elimination with partial pivoting, at each step, the algorithm searches for the largest absolute value in the current column below the diagonal and swaps rows if necessary to ensure that this value is used as the pivot. Row interchanges occur if an off-diagonal element is larger than the diagonal element.

If a matrix  $A$  is a strictly column diagonally dominant matrix, then the absolute value of the entry  $a_{jj}$  for any column  $j$  must be strictly larger than the (sum of the) absolute values of all other entries in that column. Thus, no row interchanges will occur.

## Part C

In `myPartialPivot`, a counter was added to keep track of the number of row interchanges. Sure enough, using the matrix  $A$  in part (a) yields the following output:

The number of row interchanges is 0.

ans =

4	1	1	1
0	15/4	3/4	3/4
0	0	18/5	3/5
0	0	0	7/2

To verify that the function works correctly when row interchanges are required, I tested it using the matrix from part 1(b), which we demonstrated needs two row interchanges. The output for that matrix is as follows:

The number of row interchanges is 2.

ans =

7	8	9
0	6/7	12/7
0	*	*

Again, the asterisks in the second output are due to floating-point precision errors, as discussed in that question.