Honors Linear Algebra (Spring 2011) — Computer Project

1 Gauss-Jordan Elimination

The process of applying elementary row operations (EROs) to transform a matrix into reduced row echelon form (RREF) is called Gauss-Jordan elimination. If we go only to row echelon form, the process is termed Gaussian elimination. This project asks you to create your own function in MATLAB to perform Gauss-Jordan elimination on any matrix, and count the number of EROs performed in this process.

Details of the required function, which can be called `MyRREF`, are described in the form of a *pseudocode* below. The input matrix A is assumed to be of size $m \times n$. Keywords are shown in `red`, and comments are described in % blue. As used in MATLAB, $A(i,j)$ denotes the (i,j)-th entry, and $A(i,:)$ is the i-th row of matrix A. The matrix A input to the function itself is changed to its reduced row echelon form, and hence output as A in the end. n_e counts the number of EROs performed.

```
function [A, n_e] = MyRREF(A) % input is A, outputs are rref(A) as A itself, and n_e
[m, n] = size(A) % record the number of rows and columns in A
n_e = 0; i = 1; j = 1 % set EROs counter to 0, start at the top-left corner of A
while i ≤ m AND j ≤ n
    % look for a nonzero entry at or below current row i in current column j
    i_1 = i; nzfound = 0 % i_1 is a counter, nzfound is set to 1 when a nonzero entry is found
    while nzfound==0 AND i_1 ≤ m % look till the last row for a nonzero
        if A(i_1, j) ≠ 0
            nzfound = 1 % nonzero found; exit this while loop now
            i_nz = i_1 % store the pivot row index; A(i_nz, j) is the next pivot
        else
            i_1 = i_1 + 1 % pivot not found; check next row
        end
    end
    if nzfound==1 % pivot found; do pivoting
        if i_nz ≠ i % the pivot row is below current row i
            temprow = A(i,:); A(i,:) = A(i_nz,:); A(i_nz,:) = temprow % R_i ⇋ R_{i_nz}
            n_e = n_e + 1 % increase count of EROs by 1
        end
        A(i,:) = A(i,:)/A(i,j), n_e = n_e + 1 % A(i,j) is the pivot now, scale R_i so that pivot is 1
        for i_1 = 1, ..., m, i_1 ≠ i
            if A(i_1, j) ≠ 0
                A(i_1,:) = A(i_1,:) − A(i_1, j) × A(i,:); n_e = n_e + 1 % R_{i_1} − A(i_1, j)R_i
            end
        end
        i = i + 1 % go to next row
    end % matches if nzfound==0
    j = j + 1 % go to next column
end % matches while i ≤ m AND j ≤ n
```

Standard implementations of Gauss-Jordan method often perform the steps somewhat differently so as to ensure numerical stability. For instance, in order to avoid dividing by a number close to zero, one could choose the largest number in the current column (in absolute value) as the pivot, rather than the *first* non-zero number found. But you can implement the function just as described in the pseudocode here.

2 Tasks To Do

1. (60) Write your own function to perform Gauss-Jordan elimination as described in the pseudocode above. You must name your function `MyRREF_firstname_lastname.m`. The function should take as input any matrix A, and give as output the reduced row echelon form of A and the number of EROs used in computing the same.

   Of course, you can check the correctness of your function `MyRREF` by comparing its output with the output given by the inbuilt function `rref` available in MATLAB. One way to do this comparison is as follows.

   `[`A_1, n_1`] = MyRREF(`A`);` $\quad A_2 =$ `rref(`A`);`
   `normdiff = norm(`$A_1 - A_2$`);`

   The function `norm` calculates the norm of a matrix – similar to the length of a vector. The norm of a matrix with all entries equal to zero is zero. If A_1 and A_2 are identical, then $A_1 - A_2 = O$, the zero matrix, and hence `normdiff` $= 0$. If the answer given by your function differs slightly from that given by `rref`, `normdiff` will have a small nonzero value. But if you get large values for `normdiff`, your code is probably doing something wrong.

2. (35) Study how the numbers of EROs required for Gauss-Jordan elimination vary when number of row and number of columns change. Specifically, record the number of EROs needed by `MyRREF` for the following settings. In each case, generate your A matrix randomly using the following function call in MATALB – `A = round(1000*rand(m,n));`.

   (a) Keeping $m = 100$ fixed, run your function `MyRREF` on randomly generated matrices for $n = 50, 75, 100, 200, 400, 600, 800, 1000$. Record the number of EROs *and* the `normdiff` value in each case, a present these numbers in a table with columns for n, number of EROs, and `normdiff`.

   (b) Keeping $n = 100$ fixed, run your function `MyRREF` on randomly generated matrices for $m = 50, 75, 100, 200, 400, 600, 800, 1000$. Record the number of EROs *and* the `normdiff` value in each case, a present these numbers in a table with columns for m, number of EROs, and `normdiff`.

3. (10) Based on your tabulated observations, answer the following questions.

   (a) How does the number of EROs required by `MyRREF` vary when n increases while m remains fixed?

   (b) How does the number of EROs required by `MyRREF` vary when m increases while n remains fixed?

   (c) How does `normdiff` vary when n increases while m remains fixed?

   (d) How does `normdiff` vary when m increases while n remains fixed?

3 Submission

You must **email** me the following items.

- Your MATLAB function file `MyRREF_firstname_lastname.m`.
- A report showing the tables of numbers for Tasks To Do Qestion (2), and your answers to the questions in Tasks To Do Question (3).

You are welcome to submit a hand-written report, but you must email me your MATLAB function file. The total points add up to 105, and you will be graded for 100 points.

These items are due by **5:00 pm, Friday, Apr 29**.