Math 425 **Applied & Comput. Lin. Algebra** **Fall 2024**
Homework

Homework I

In this homework, you will write some `MATLAB` code and do computations in `MATLAB`. Include all code and computations in a `MATLAB` file named `math425hw1.m`. You will need to submit this file along with a document containing your answers which do not involve `MATLAB`.

1. In this exercise you will implement Gaussian elimination and backward substitution for solving systems of equations in `MATLAB`.
a) Create a function called `myGaussianElimination` which takes as input a regular $n \times (n+1)$ matrix A. The output is an $n \times (n+1)$ upper-triangular matrix U whose diagonals are nonzero. Recall that if a matrix is regular, then during Gaussian elimination no pivot entry will be zero and therefore no row swapping will be needed. Your function should issue an error message if it encounters a pivot that is zero.
b) Create a function called `myBackwardSubstitution` which takes as input an $n \times n$ upper-triangular matrix U with nonzero diagonal entries and a $n \times 1$ column vector c. The output is a $n \times 1$ column vector x which is the solution to $Ux = c$.
c) Finally create a function `myLinearSolution` that will take as input an $n \times n$ regular matrix A and a $n \times 1$ column vector b. The output is a $n \times 1$ column vector which is the solution to $Ax = b$. Use the two functions you have created in the first two parts. Your function should issue the appropriate error messages.

2. In this exercise we will use the following matrix to illustrate how to compute the LU decomposition and how to use it::

$$A = \begin{pmatrix} -8 & -2 & 3 & 1 \\ 1 & -2 & 0 & 2 \\ -4 & -1 & 3 & 2 \\ 4 & 1 & -1 & -1 \end{pmatrix}$$

By Gaussian elimination one can reduce the above A to an upper triangular matrix by using row operations of the type $cR_i + R_j \to R_j$.
a) State the first row operation you would do in order to reduce A to an upper triangular matrix and apply this row operation to obtain the matrix A_1.
b) Find the corresponding elementary matrix E_1. Then compute $E_1 A$. What do you observe ?
c) State and apply the next row operation that needs to be applied to A_1 to obtain A_2.
d) Construct the corresponding elementary matrix E_2. Compute $E_2 A_1$.
e) Now continue to construct E_3 and E_4. Verify $E_4 E_3 E_2 E_1 A = U$ where U is an upper triangular matrix.
f) Compute the inverses of $E_1, \dots, E_4$. State the pattern you see.
g) Argue that $A = LU$ where $L = E_1^{-1} E_2^{-1} E_3^{-1} E_4^{-1}$. What kind of matrix is L ?
We have obtained an LU factorization of A. To check your work use

```
>> [L, U] = lu(A)
```

command in MATLAB to get the LU factorization of A.
h) Suppose we want to solve $Ax = b$ where $b = \begin{pmatrix} -2 & 6 & -5 & 1 \end{pmatrix}^T$. Show that you can do this in two steps once you have the LU factorization of A: first solve $Ly = b$ and then $Ux = y$. Find the solution x using your function `myLinearSolution` and the LU decomposition. Check you get the same answer.

Include your computations in `math425hw1.m`.

3. Suppose in the linear system $Ax = b$ the matrix A is an $n \times n$ regular matrix.
a) Count the number of multiplication/division operations needed to find the solution x by Gaussian elimination and backward substitution. The answer should be a formula involving a polynomial in n. What is the degree and the leading coefficient of this polynomial ?
b) Now count the number of multiplications/divisions to compute A^{-1} and to find x by $x = A^{-1}b$. What is the degree and the leading coefficient of the resulting polynomial in n in your formula ?
c) Based on your answers above which method will you use for large n ? Explain.

4. To appreciate the issue of numerical inaccuracy in the linear algebra computations we will look at the problem of solving a system $Ax = b$ when A is a square invertible matrix. In that case, we know we can compute the unique solution by $x = A^{-1}b$. However, although a matrix is invertible, it could be almost non-invertible. We call such a matrix *ill-conditioned*. If we try to compute A^{-1} of an ill-conditioned matrix numerical rounding errors could accumulate to such a degree that the computed $A^{-1}b$ may not be close to the true solution. Now go through the following steps and include your computations in your file `math425hw1.m`.

There is a well-known class of ill-conditioned matrices, called *Hilbert matrices*. For each positive integer n there is a Hilbert matrix H_n. You can see the general form of H_n in your textbook on page 57.
a) You can generate Hilbert matrices in MATLAB. For instance, type

```
>> hilb(5)
```

to see H_5. Using MATLAB's `inv` command to compute the inverse of a square matrix, compute the inverse of H_5, H_{10}, H_{15}, etc. Look at the output carefully. What is the feature of the inverses that jumps at you ?
b) Now we will solve a system $H_{15}x = b$, by $H_{15}^{-1}b$. First generate a random (column) vector with 15 entries by

```
>> x = rand(15,1)
```

Then we compute

```
>> b = hilb(15)*x
```

Now we know the solution to the system $H_{15}x = b$: it is the random vector you generated in the first place. As a last step, compute the solution to this system using $H_{15}^{-1}b$, and compare with the true solution. Look at the MATLAB output carefully. It gives you more information.

5. Suppose that A is an $n \times n$ matrix that (somehow) you know to be regular. This means that A has an LU decomposition, i.e., $A = LU$ where L is a lower-triangular matrix whose diagonal entries are equal 1 and U is an upper-triangular matrix with nonzero diagonals. Describe how you can compute the entries of L and U **without** the help of Gaussian elimination. How many multiplications/divisions do you need? Compare the answer to your answers in question 3.