Jaiden Atterbury ID#: 2028502

- **Exercise 1. (cs1.6):**

Suppose U is an m x (m+1) augmented, upper triangular matrix that represents a linear system with m equations and m unknowns. Assuming the entries along the main diagonal of U are nonzero, we can perform back substitution on U to obtain the unique solution to the linear system. In MATLAB, the code that performs back substitution is given below.

```
% Back substitution on a m by m+1 upper triangular matrix U.
[m, n] = size(U);
x = U(:, m+1); % Initialize column vector of unknowns (to be updated).
x(m) = U(m, m+1)/U(m, m); % solve last equation of augmented matrix.
for i = m-1:-1:1 % counts down from m-1 to 1 in intervals of 1.
    SUM = 0;
    for j = i+1:m
        SUM = SUM + U(i,j) * x(j);
    end
    x(i) = (U(i,n) - SUM) / U(i,i); % updates the ith entry of x
end
x % x is the solution of the linear system.
```

(a.) How many flops occur per pass through the inner for-loop?

The only line in the inner for-loop that contains arithmetic operations is SUM = SUM + U(i,j) * x(j);, as can be seen, there is 1 addition and 1 multiplication for a total of 2 flops per pass through the inner for-loop.

(b.) How many times does one pass through the inner for-loop? Using your answer to the previous question, what is the total flop count of the inner for-loop? Note: Your answers should depend on the index i.

The inner for loop goes from i+1 to m, thus it runs m - (i+1) + 1 = m - i - 1 + 1 = m - i times. Using the answer from part a, the total flop

count of the inner for-loop is $2(m-i)$ flops.

(c.) How many flops are needed to update the ith entry of the vector x? Using this answer and your answer to the previous question, how many flops occur per pass through the outer for-loop?
note: Your answer for the flops per pass of the outer for-loop should still depend on the index i.

The line that updates the ith entry of x is `x(i) = (U(i,n) - sum)/U(i,i);`. as can be seen from this line, there is 1 subtraction and 1 division for a total of 2 flops to update the ith entry of x. Using this and the answer from part b, we can see that there is $2(m-i)+2 = 2(m-i+1)$ flops per pass through the outer for-loop.

(d.) Show that the total number of flops that occur within the outer for-loop is m^2+m-2.
Hint: Recall the useful identity $\sum_{i=1}^{N} i = \frac{N(N+1)}{2}$

As described in lecture, the total flops of back substitution is given by: total flops = (# flops, $i=m-1$) + (# flops, $i=m-2$) + $\cdots$ + (# flops, $i=1$). Using the associativity of addition we can use our result from part c to write the above expression for total flops in summation form:

$$\begin{aligned}
\text{total flops} &= \sum_{i=1}^{m-1} 2(m-i+1) \\
&= \sum_{i=1}^{m-1} 2(m+1) - 2i \\
&= 2(m+1)(m-1) - 2\sum_{i=1}^{m-1} i \\
&= 2(m^2+m-m-1) - 2\left(\frac{(m-1)\cdot(m)}{2}\right) \quad \text{(Using the above hint)} \\
&= 2m^2 - 2 - (m^2-m) \\
&= m^2+m-2
\end{aligned}$$

thus we have shown that the total number of flops that occur within the outer for-loop is m^2+m-2.

(e.) Solving for $x(m)$ requires one additional flop so that the total flop count of back substitution is m^2+m-1. For very large matrices, what is the asymptotic flop count of back substitution?

Since the asymptotic flop count removes all terms below the term of the highest order, we can see that the asymptotic flop count of back substitution is m^2. To see this, let $f(m) = m^2+m-1$ and $g(m) = m^2$, then $\lim_{m\to\infty} \frac{f(m)}{g(m)} = \frac{m^2+m-1}{m^2} = 1$. Thus by Definition 3, $m^2+m-1 \sim m^2$, and thus m^2 is the asymptotic flop count.

- Exercise 2. (cs 2.1-2.2)

What row operations do each of the following elementary matrices E represent? For each E, compute its inverse E^{-1}.

(a.) $\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$

This elementary matrix represents a type 2 row operation. In particular, it represents interchanging row 3 and row 4. By theorem 16, since this matrix is an elementary matrix of the second type it is its own inverse, hence;

$E^{-1} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$.

(b.) $\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -2 & 1 \end{bmatrix}$

This elementary matrix represents a type 1 row operation. In particular, it represents adding -2 times row 2 to row 3. By Definition 8, the inverse of the above matrix is;

$E^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 2 & 1 \end{bmatrix}$.

(c.) $\begin{bmatrix} 1 & 0 & 3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

This elementary matrix represents a type 1 row operation. In particular, it represents adding 3 times row 3 to row 1. By definition 8, the inverse of the above matrix is:

$$E^{-1} = \begin{bmatrix} 1 & 0 & -3 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

- Exercise 3. (cs 2.3):

Elementary matrices can be used to construct more elaborate row operations. For example, suppose we wish to construct a 2x2 matrix A such that multiplying any 2x2 matrix B on the left by A subtracts the first row of B from the second row of B and then swaps the rows of B, then

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix} = \begin{bmatrix} -1 & 1 \\ 1 & 0 \end{bmatrix}$$

For each of the following sequences of row operations, construct a matrix A that executes the row operations in the order they appear, similar to the example above.

(a.) Construct a 3x3 matrix A that swaps the first and third rows and then adds three times the second row to the first row

$$A = \begin{bmatrix} 1 & 3 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 3 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

(b.) Construct a 4x4 matrix A that swaps the first and fourth rows, swaps the second and fourth rows, and then subtracts twice the fourth row from the second row.

$$A = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & -2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & -2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

(c) Construct a 2x2 matrix A that swaps the first and second rows, adds the second row to the first row, and then swaps the first and second rows again. This matrix A is actually elementary. What row operation is equivalent to A?

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$
$$= \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}$$
$$= \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}$$

the matrix A is equivalent to the type 1 row operation of adding the first row to the second row.

- Exercise 4. (cs2.3):

From Olver and Shakiban, complete Exercises 1.3.21 (e) and (f).

Find the LU factorization of the following matrices:

- 1.3.21 (e):

$$\begin{bmatrix} -1 & 0 & 0 \\ 2 & -3 & 0 \\ 1 & 3 & 2 \end{bmatrix} = A$$

① Add 2 times row 1 to row 2:

$$E_1 = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad L_1 = \begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad A = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -3 & 0 \\ 1 & 3 & 2 \end{bmatrix}$$

② Add row 1 to row 3:

$$E_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}, \quad L_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix}, \quad A = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -3 & 0 \\ 1 & 3 & 2 \end{bmatrix}$$

③ Add row 2 to row 3:

$E_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix}$, $L_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix}$, $U = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -3 & 0 \\ 0 & 0 & 2 \end{bmatrix}$

④ Also, $L = L_1 L_2 L_3$

$$= \begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -1 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & -1 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -1 & -1 & 1 \end{bmatrix}$$

⑤ Thus $A = LU$ is

$$A = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -1 & -1 & 1 \end{bmatrix}}_{L} \cdot \underbrace{\begin{bmatrix} -1 & 0 & 0 \\ 0 & -3 & 0 \\ 0 & 0 & 2 \end{bmatrix}}_{U} = \begin{bmatrix} -1 & 0 & 0 \\ 2 & -3 & 0 \\ 1 & 3 & 2 \end{bmatrix}$$

01.3.21 (f):

$$\begin{bmatrix} 1 & 0 & -1 \\ 2 & 3 & 2 \\ -3 & 1 & 0 \end{bmatrix} = A$$

① Add -2 times row 1 to row 2:

$E_1 = \begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$, $L_1 = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$, $A = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 3 & 4 \\ -3 & 1 & 0 \end{bmatrix}$

② Add 3 times row 1 to row 3:

$E_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 3 & 0 & 1 \end{bmatrix}$, $L_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -3 & 0 & 1 \end{bmatrix}$, $A = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 3 & 4 \\ 0 & 1 & -3 \end{bmatrix}$

③ Add $-\frac{1}{3}$ times row 2 to row 3:

$E_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -\frac{1}{3} & 1 \end{bmatrix}$, $L_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & \frac{1}{3} & 1 \end{bmatrix}$, $U = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 3 & 4 \\ 0 & 0 & -\frac{13}{3} \end{bmatrix}$

④ Also, $L = L_1 L_2 L_3$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -3 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & \frac{1}{3} & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -3 & \frac{1}{3} & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ -3 & \frac{1}{3} & 1 \end{bmatrix}$$

⑤ Thus $A = LU$ is

$$A = \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ -3 & \frac{1}{3} & 1 \end{bmatrix}}_{L} \cdot \underbrace{\begin{bmatrix} 1 & 0 & -1 \\ 0 & 3 & 4 \\ 0 & 0 & -\frac{13}{3} \end{bmatrix}}_{U} = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 3 & 2 \\ -3 & 1 & 0 \end{bmatrix}$$

- Exercise 5. (CS 2.4):

Read Oliver and Shakiban Example 1.12 (page 28) to learn an alternative (and easier) way to compute a PLU decomposition. Then, complete exercises 1.4.21(b) and (c). Use partial pivoting to determine whether to exchange rows. You only need to compute the PLU decomposition. Do not solve the systems.

- 1.4.21(b):

$$A = \begin{bmatrix} 0 & 0 & -4 \\ 1 & 2 & 3 \\ 0 & 1 & 7 \end{bmatrix}$$

① since row 2 has the highest magnitude entry in column 1 interchange row 1 and row 2.

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 0 & 0 & -4 \\ 0 & 1 & 7 \end{bmatrix}, \quad L = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad P = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

② since row 3 has the highest magnitude entry in column 2 (excluding row 1) we will interchange row 2 and row 3

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & 7 \\ 0 & 0 & -4 \end{bmatrix} = U, \quad L = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad P = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

③ Thus the PLU decomposition is:

$$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 & -4 \\ 1 & 2 & 3 \\ 0 & 1 & 7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 2 & 3 \\ 0 & 1 & 7 \\ 0 & 0 & -4 \end{bmatrix}$$

P A L U

- 1.4.21 (c):

$$A = \begin{bmatrix} 0 & 1 & -3 \\ 0 & 2 & 3 \\ 1 & 0 & 2 \end{bmatrix}$$

① since row 3 has the highest magnitude entry in column 1 interchange row 1 and row 3.

$$A = \begin{bmatrix} 1 & 0 & 2 \\ 0 & 2 & 3 \\ 0 & 1 & -3 \end{bmatrix}, \quad L = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad P = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

② Add $-\frac{1}{2}$ times row 2 to row 3

$$A = \begin{bmatrix} 1 & 0 & 2 \\ 0 & 2 & 3 \\ 0 & 0 & \frac{-9}{2} \end{bmatrix} = U, \quad L = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & \frac{1}{2} & 1 \end{bmatrix}, \quad P = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}$$

③ Thus the PLU decomposition is:

$$\begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 & 1 & -3 \\ 0 & 2 & 3 \\ 1 & 0 & 2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1/2 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 2 \\ 0 & 2 & 3 \\ 0 & 0 & \frac{-9}{2} \end{bmatrix}$$

P A L U

- Exercise 6. (cs2.4):

consider the linear system

$$\begin{cases} \varepsilon x_1 + x_2 = 3 \\ x_1 + 2x_2 = 5 \end{cases}$$

where $|\varepsilon| << 1$ is a very small parameter. In this exercise, we will solve this linear system by regular Gaussian Elimination and Gaussian elimination with partial pivoting. We will see that Gaussian elimination with partial pivoting is much preferred when numerical round-off errors are present.

(a.) What is the solution of the linear system when $\varepsilon = 0$? We can use this solution as a proxy for the exact solution when $\varepsilon \neq 0$ and $|\varepsilon| << 1$.

If $\varepsilon = 0$, the linear system becomes

$$\begin{cases} x_2 = 3 \\ x_1 + 2x_2 = 5 \end{cases}$$

thus, $x_2 = 3$. Plugging in 3 for x_2 in equation 2 and isolating x_1, we can see that $x_1 + 2(3) = 5 \Rightarrow x = -1$. Thus our proxy solution is $(x_1, x_2) = (-1, 3)$.

(b.) Let's solve the system by regular Gaussian elimination, subject to numerical round-off.

i. Reduce the system to triangular form by regular Gaussian Elimination.

thus we must reduce the following system to triangular form

$$\begin{cases} \varepsilon x_1 + x_2 = 3 \\ x_1 + 2x_2 = 5 \end{cases}$$

Adding $-\frac{1}{\varepsilon}$ times equation 1 to equation 2, we obtain the system in triangular form

$$\begin{cases} \varepsilon x_1 + x_2 = 3 \\ (2 - \frac{1}{\varepsilon})x_2 = 5 - \frac{3}{\varepsilon} \end{cases}$$

ii. Since $|\varepsilon| << 1$, we have $1/|\varepsilon| >> 1$. Therefore, expressions such as $a + b/\varepsilon$, where a and b are real numbers whose magnitudes are small compared to $1/|\varepsilon|$, are often rounded to b/ε on a computer. Apply this approximation to your triangular system, carry out back substitution as usual. Compare your solution with (a).

Using the above approximation rule to our triangular system, we obtain

$$\begin{cases} \varepsilon x_1 + x_2 = 3 \\ -\frac{1}{\varepsilon} x_2 = -\frac{3}{\varepsilon} \end{cases}$$

Hence using back substitution, $\frac{1}{\varepsilon}x_2 = \frac{-3}{\varepsilon} \Rightarrow x_2 = 3$. Also, $\varepsilon x_1 + x_2 = 3 \Rightarrow \varepsilon x_1 + 3 = 3 \Rightarrow x_1 = 0$. Hence the solution is $(x_1, x_2) = (0, 3)$. In comparison to the "exact" solution in part a, the only difference is that x_1 is now 0 instead of -1. Hence, this solution is wrong in terms of the proxy.

(c.) Let's solve the system by Gaussian elimination with partial pivoting, subject to numerical round-off

i. Reduce the system to triangular form by Gaussian elimination with partial pivoting.

Since $|\varepsilon| << 1$, we will interchange row 1 and row 2, our system thus becomes: (by row I mean equation)

$$\begin{cases} x_1 + 2x_2 = 5 \\ \varepsilon x_1 + x_2 = 3 \end{cases}$$

Adding $-\varepsilon$ times equation 1 to equation 2 leads to the following triangular system

$$\begin{cases} x_1 + 2x_2 = 5 \\ (-2\varepsilon + 1)x_2 = -5\varepsilon + 3 \end{cases}$$

ii. Expressions such as $a + b\varepsilon$, where a and b are real numbers whose magnitudes are large compared to $|\varepsilon|$, are often rounded to a on a computer. Apply this approximation to your triangular system, carry out back substitution as usual. Compare your solution with (a).

Using the above rule to our triangular system, we obtain

$$\begin{cases} x_1 + 2x_2 = 5 \\ x_2 = 3 \end{cases}$$

Hence using back substitution, $x_2 = 3$, also, $x_1 + 2(3) = 5 \Rightarrow x_1 = -1$. Hence the solution is $(x_1, x_2) = (-1, 3)$. In comparison to the "exact" solution in part a, there are no differences. Hence, this solution is correct in terms of the proxy. This shows partial pivoting is preferred when round-off errors are present.

Table of Contents


```
% Multi-Step Problem

format long;
```

Setup

```
% Change to format long to show more significant digits:
format long;

% Initialize the tridiagonal matrix:
A = [-2, 1, 0, 0, 0;
     1, -2, 1, 0, 0;
     0, 1. -2, 1, 0;
     0, 0, 1, -2, 1;
     0, 0, 0, 1, -2];

% Initialize the vector of knowns:
b = [1; 2; 3; 4; 5].*6^(-3);
```

Part 1.

```
% This section of code, copied from the assignment specification, performs
% Gaussian elimination on the augmented matrix that represents the given
% linear system. At the end, the code will output the upper triangular
% augmented matrix.

M = [A,b];
[m,n] = size(M);
for j = 1:m
    if M(j,j)==0
        error('System cannot be solved by regular Gaussian elimination.');
    end
    for i = j+1:m
        l_ij = M(i,j)/M(j,j);
        M(i,j:n) = M(i,j:n)-l_ij*M(j,j:n);
    end
end
M
```

```
M =
```

```
  Columns 1 through 3

  -2.000000000000000   1.000000000000000                   0
                   0  -1.500000000000000   1.000000000000000
                   0                   0  -1.333333333333333
                   0                   0                   0
                   0                   0                   0

  Columns 4 through 6

                   0                   0   0.004629629629630
                   0                   0   0.011574074074074
   1.000000000000000                   0   0.021604938271605
  -1.250000000000000   1.000000000000000   0.034722222222222
                   0  -1.200000000000000   0.050925925925926
```

Part 2.

```
% This section of code, copied from Exercise 1, performs back substitution
% on the upper triangular augmented matrix obtained in from Part 1. At the
% end, the code will output the resulting solution vector.

U = M;
[m,n] = size(U);
x = U(:,m+1);
x(m) = U(m,m+1)/U(m,m);
for i = m-1:-1:1
    SUM = 0;
    for j = i+1:m
        SUM = SUM + U(i,j)*x(j);
    end
    x(i) = (U(i,n) - SUM)/U(i,i);
end
x


x =

  -0.027006172839506
  -0.049382716049383
  -0.062500000000000
  -0.061728395061728
  -0.042438271604938
```

Part 3

```
% This section of code, copied from Part 1, modifies this copied code to
% compute the LU factorization of the coefficient matrix A of the above
% linear system. In the end this code will display the L and U matrices.
```

```
U = A;
[m,n] = size(U);
L = eye(m);
for j = 1:m
    if U(j,j)==0
        error('System cannot be solved by regular Gaussian elimination.');
    end
    for i = j+1:m
        l_ij = U(i,j)/U(j,j);
        U(i,j:n) = U(i,j:n)-l_ij*U(j,j:n);
        L(i,j) = l_ij;
    end
end
L
U

% Lastly, to check if our L and U are correct, we will perform A-LU and see
% if it equals the zero matrix
A-L*U
```

```
L =

  Columns 1 through 3

   1.000000000000000                   0                   0
  -0.500000000000000   1.000000000000000                   0
                   0  -0.666666666666667   1.000000000000000
                   0                   0  -0.750000000000000
                   0                   0                   0

  Columns 4 through 5

                   0                   0
                   0                   0
                   0                   0
   1.000000000000000                   0
  -0.800000000000000   1.000000000000000


U =

  Columns 1 through 3

  -2.000000000000000   1.000000000000000                   0
                   0  -1.500000000000000   1.000000000000000
                   0                   0  -1.333333333333333
                   0                   0                   0
                   0                   0                   0

  Columns 4 through 5

                   0                   0
                   0                   0
```

```
   1.000000000000000                   0
  -1.250000000000000   1.000000000000000
                   0  -1.200000000000000


ans =

     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
```

Published with MATLAB® R2023a