AMS 209, Fall 2016
Final Project Type A – Numerical Linear Algebra: Gaussian Elimination with Pivoting for Solving Linear Systems

1. Overview

We are interested in solving a well-defined linear system given as

$$\mathbf{A}\mathbf{x} = \mathbf{b}, \tag{1}$$

where $\mathbf{A}$ is a $n \times n$ *square* matrix and $\mathbf{x}$ and $\mathbf{b}$ are n-vectors.

1.1. Invariant Transformations

1.1.1. Permutation To solve a linear system, we wish to transform the given linear system into an easier linear system where the solution $\mathbf{x} = \mathbf{A}^{-1}\mathbf{b}$ remains unchanged. The answer is that we can introduce any nonsingular matrix $\mathbf{M}$ and multiply from the left both sides of the given linear system:

$$\mathbf{M}\mathbf{A}\mathbf{x} = \mathbf{M}\mathbf{b}. \tag{2}$$

We can easily check that the solution remains the same. To see this, let $\mathbf{z}$ be the solution of the linear system in Eqn. (2). Then

$$\mathbf{z} = (\mathbf{M}\mathbf{A})^{-1}\mathbf{M}\mathbf{b} = \mathbf{A}^{-1}\mathbf{M}^{-1}\mathbf{M}\mathbf{b} = \mathbf{A}^{-1}\mathbf{b} = \mathbf{x}. \tag{3}$$

Example: A permutation matrix $\mathbf{P}$, a square matrix having exactly one 1 in each row and column and zeros elsewhere – which is also always a nonsingular – can always be multiplied without affecting the original solution to the system. For instance,

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{4}$$

permutes $\mathbf{v}$ as

$$\mathbf{P}\begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}\begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix} = \begin{bmatrix} v_3 \\ v_1 \\ v_2 \end{bmatrix}. \tag{5}$$

□

1.1.2. Row scaling Another invariant transformation exists which is called *row scaling*, an outcome of a multiplication by a diagonal matrix $\mathbf{D}$ with nonzero diagonal entries $d_{ii}, i = 1, \ldots n$. In this case, we have

$$\mathbf{D}\mathbf{A}\mathbf{x} = \mathbf{D}\mathbf{b}, \tag{6}$$

by which each row of the transformed matrix $\mathbf{DA}$ gets to be scaled by d_{ii} from the original matrix $\mathbf{A}$. Note that the scaling factors are cancelled by the same scaling factors introduced on the right hand side vector, leaving the solution to the original system unchanged.

Note: The column scaling does not preserve the solution in general. □

1.2. LU factorization by Gaussian elimination

Consider the following system of linear equations:

$$x_1 + 2x_2 + 2x_3 = 3, \tag{7}$$
$$-4x_2 - 6x_3 = -6, \tag{8}$$
$$-x_3 = 1. \tag{9}$$

We know this is easily solvable since we already know $x_3 = -1$, which gives $x_2 = 3$, therefore recursively arriving a complete set of solution with $x_1 = -1$. When putting these equations into a matrix-vector form, we have

$$\begin{bmatrix} 1 & 2 & 2 \\ 0 & -4 & -6 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 3 \\ -6 \\ 1 \end{bmatrix}, \tag{10}$$

where the matrix has a form of (upper) triangular.

Therefore, our strategy then is to devise a nonsingular linear transformation that transforms a given general linear system into a triangular linear system. This is a key idea of *LU factorization (or LU decomposition)* or also known as *Gaussian elimination*.

The main idea is to find a matrix $\mathbf{M}_1$ such that the first column of $\mathbf{M}_1\mathbf{A}$ becomes zero below the first row. The right hand side $\mathbf{b}$ is also multiplied by $\mathbf{M}_1$ as well. Again, we repeat this process in the next step so that we find $\mathbf{M}_2$ such that the second column of $\mathbf{M}_2\mathbf{M}_1\mathbf{A}$ becomes zero below the second row, along with applying the equivalent multiplication on the right hand side, $\mathbf{M}_2\mathbf{M}_1\mathbf{b}$. This process is continued for each successive column until all of the subdiagonal entries of the resulting matrix have been annihilated.

If we define the final matrix $\mathbf{M} = \mathbf{M}_{n-1} \cdots \mathbf{M}_1$, the transformed linear system becomes

$$\mathbf{M}_{n-1} \cdots \mathbf{M}_1\mathbf{A}\mathbf{x} = \mathbf{M}\mathbf{A}\mathbf{x} = \mathbf{M}\mathbf{b} = \mathbf{M}_{n-1} \cdots \mathbf{M}_1\mathbf{b}. \tag{11}$$

Note: As seen in the previous section, we recall that any nonsingular matrix multiplication is an invariant transformation that does not affect the solution to the given linear system.

The resulting transformed linear system $\mathbf{MAx} = \mathbf{Mb}$ is upper triangular which is what we want, and can be solved by back-substitution to obtain the solution to the original linear system $\mathbf{Ax} = \mathbf{b}$.

Example: We illustrate Gaussian elimination by considering:

$$\begin{array}{lllll} 2x_1 & +x_2 & +x_3 & & = 3, \\ 4x_1 & +3x_2 & +3x_3 & +x_4 & = 6, \\ 8x_1 & +7x_2 & +9x_3 & +5x_4 & = 10, \\ 6x_1 & +7x_2 & +9x_3 & +8x_4 & = 1. \end{array} \tag{12}$$

or in a matrix notation

$$\mathbf{Ax} = \begin{bmatrix} 2 & 1 & 1 & 0 \\ 4 & 3 & 3 & 1 \\ 8 & 7 & 9 & 5 \\ 6 & 7 & 9 & 8 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} 3 \\ 6 \\ 10 \\ 1 \end{bmatrix} = \mathbf{b}. \tag{13}$$

The first question is to find a matrix $\mathbf{M}_1$ that annihilates the subdiagonal entries of the first column of $\mathbf{A}$. This can be done if we consider a matrix $\mathbf{M}_1$ that can subtract twice the first row from the second row, four times the first row from the third row, and three times the first row from the fourth row. The matrix $\mathbf{M}_1$ is then identical to the identity matrix $\mathbf{I}_4$, except for those multiplication factors in the first column:

$$\mathbf{M}_1\mathbf{A} = \begin{bmatrix} 1 & & & \\ -2 & 1 & & \\ -4 & & 1 & \\ -3 & & & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & 1 & 0 \\ 4 & 3 & 3 & 1 \\ 8 & 7 & 9 & 5 \\ 6 & 7 & 9 & 8 \end{bmatrix} = \begin{bmatrix} 2 & 1 & 1 & 0 \\ & 1 & 1 & 1 \\ & 3 & 5 & 5 \\ & 4 & 6 & 8 \end{bmatrix}, \tag{14}$$

where we treat the blank entries to be zero entries. At the same time, we proceed the corresponding multiplication on the right hand side to get:

$$\mathbf{M}_1\mathbf{b} = \begin{bmatrix} 3 \\ 0 \\ -2 \\ -8 \end{bmatrix}. \tag{15}$$

The next step would be to annihilate the third and fourth entries from the second column (3 and 4), which will give a next matrix $\mathbf{M}_2$ that has the form:

$$\mathbf{M}_2\mathbf{M}_1\mathbf{A} = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & -3 & 1 & \\ & -4 & & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & 1 & 0 \\ & 1 & 1 & 1 \\ & 3 & 5 & 5 \\ & 4 & 6 & 8 \end{bmatrix} = \begin{bmatrix} 2 & 1 & 1 & 0 \\ & 1 & 1 & 1 \\ & & 2 & 2 \\ & & 2 & 4 \end{bmatrix}, \tag{16}$$

now with the right hand side:

$$\mathbf{M}_2\mathbf{M}_1\mathbf{b} = \begin{bmatrix} 3 \\ 0 \\ -2 \\ -8 \end{bmatrix}. \tag{17}$$

The last matrix $\mathbf{M}_3$ will complete the process, resulting an upper triangular matrix $\mathbf{U}$:

$$\mathbf{M}_3\mathbf{M}_2\mathbf{M}_1\mathbf{A} = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & -1 & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & 1 & 0 \\ & 1 & 1 & 1 \\ & & 2 & 2 \\ & & 2 & 4 \end{bmatrix} = \begin{bmatrix} 2 & 1 & 1 & 0 \\ & 1 & 1 & 1 \\ & & 2 & 2 \\ & & & 2 \end{bmatrix} = \mathbf{U}, \tag{18}$$

together with the right hand side:

$$\mathbf{M}_3\mathbf{M}_2\mathbf{M}_1\mathbf{b} = \begin{bmatrix} 3 \\ 0 \\ -2 \\ -6 \end{bmatrix} = \mathbf{y}. \tag{19}$$

We see that the final transformed linear system $\mathbf{MAx} = \mathbf{Ux} = \mathbf{y}$ is upper triangular which is what we wanted and it can be solved easily by back-substitution, starting from obtaining $x_4 = -3$, followed by x_3, x_2, and x_1 in reverse order to find a complete solution

$$\mathbf{x} = \begin{bmatrix} 0 \\ 1 \\ 2 \\ -3 \end{bmatrix}. \tag{20}$$

The full LU factorization $\mathbf{A} = \mathbf{LU}$ can be established if we compute

$$\mathbf{L} = (\mathbf{M}_3\mathbf{M}_2\mathbf{M}_1)^{-1} = \mathbf{M}_1^{-1}\mathbf{M}_2^{-1}\mathbf{M}_3^{-1}. \tag{21}$$

At first sight this looks like an expensive process as it involves inverting a series of matrices. Surprisingly, however, this turns out to be a trivial task. The inverse of $\mathbf{M}_i$, $i = 1, 2, 3$ is just itself but with each entry below the diagonal negated. Therefore, we have

$$\begin{aligned} \mathbf{L} &= \mathbf{M}_1^{-1}\mathbf{M}_2^{-1}\mathbf{M}_3^{-1} \\ &= \begin{bmatrix} 1 & & & \\ -2 & 1 & & \\ -4 & & 1 & \\ -3 & & & 1 \end{bmatrix}^{-1} \begin{bmatrix} 1 & & & \\ & 1 & & \\ & -3 & 1 & \\ & -4 & & 1 \end{bmatrix}^{-1} \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & -1 & 1 \end{bmatrix}^{-1} \\ &= \begin{bmatrix} 1 & & & \\ 2 & 1 & & \\ 4 & & 1 & \\ 3 & & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & \\ & 1 & & \\ & 3 & 1 & \\ & 4 & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & 1 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 1 & & & \\ 2 & 1 & & \\ 4 & 3 & 1 & \\ 3 & 4 & 1 & 1 \end{bmatrix}. \end{aligned} \tag{22}$$

Notice also that the matrix multiplication $\mathbf{M}_1^{-1}\mathbf{M}_2^{-1}\mathbf{M}_3^{-1}$ is also trivial and is just the unit lower triangle matrix with the nonzero subdiagonal entries of $\mathbf{M}_1^{-1}$, $\mathbf{M}_2^{-1}$, and $\mathbf{M}_3^{-1}$ inserted in the appropriate places.

All together, we finally have our decomposition $\mathbf{A} = \mathbf{LU}$:

$$\begin{bmatrix} 2 & 1 & 1 & 0 \\ 4 & 3 & 3 & 1 \\ 8 & 7 & 9 & 5 \\ 6 & 7 & 9 & 8 \end{bmatrix} = \begin{bmatrix} 1 & & & \\ 2 & 1 & & \\ 4 & 3 & 1 & \\ 3 & 4 & 1 & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & 1 & 0 \\ & 1 & 1 & 1 \\ & & 2 & 2 \\ & & & 2 \end{bmatrix}. \tag{23}$$

□

Quick summary: Gaussian elimination proceeds in steps until a upper triangular matrix is obtained for back-substitution:

$$\begin{bmatrix} * & * & * & * \\ * & * & * & * \\ * & * & * & * \\ * & * & * & * \end{bmatrix} \xrightarrow{\mathbf{M}_1} \begin{bmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & * & * & * \\ 0 & * & * & * \end{bmatrix} \xrightarrow{\mathbf{M}_2} \begin{bmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \\ 0 & 0 & * & * \end{bmatrix} \xrightarrow{\mathbf{M}_3} \begin{bmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \\ 0 & 0 & 0 & * \end{bmatrix} \tag{24}$$

□

Algorithm: LU factorization by Gaussian elimination:

```
for k = 1 to n − 1
    #[loop over column]
    if a_kk = 0 then
        stop
        #[stop if pivot (or divisor) is zero]
    endif
    for i = k + 1 to n
        m_ik = a_ik / a_kk
        #[compute multipliers for each column]
    endfor
    for j = k + 1 to n
        for i = k + 1 to n
            a_ij = a_ij − m_ik a_kj
            #[transformation to remaining submatrix]
        endfor
    endfor
endfor
```

The above algorithm yields both **U** and **L**:

- the subdiagonal entries of **L** are given by $\ell_{i,k} = m_{i,k}$.
- the operations in the algorithm computes new entries $u_{i,j}$ of **U**, ranging $2 \le i, j \le n$. Note that the first row of **U** is the same as the first row of the original **A**.
- Make sure you also perform the similar operations to the right hand side vector **b**.

1.3. Pivoting

1.3.1. Need for pivoting We obviously run into trouble when the choice of a divisor – called a *pivot* – is zero, whereby the Gaussian elimination algorithm breaks down. As illustrated in **Algorithm** above, this situation can be easily checked and avoided so that the algorithm stops when one of the diagonal entries become singular.

The solution to this singular pivot issue is almost equally straightforward: if the pivot entry is zero at state k, i.e., $a_{kk} = 0$, then one interchange row k of *both* the matrix and the right hand side vector with some subsequent row whose entry in column k is nonzero and resume the process as usual. Recall that permutation does not alter the solution to the system.

This row interchanging process is called *pivoting*, which is illustrated in the following example.

Example: Pivoting with permutation matrix can be easily explained as below:

$$\begin{bmatrix} * & * & * & * \\ 0 & 0 & * & * \\ 0 & * & * & * \\ 0 & * & * & * \end{bmatrix} \xrightarrow{\mathbf{P}} \begin{bmatrix} * & * & * & * \\ 0 & * & * & * \\ 0 & * & * & * \\ 0 & 0 & * & * \end{bmatrix} \tag{25}$$

where we interchange the second row with the fourth row using a permutation matrix $\mathbf{P}$ given as

$$\mathbf{P} = \begin{bmatrix} 1 & & & \\ & & & 1 \\ & & 1 & \\ & 1 & & \end{bmatrix}. \tag{26}$$

□

Note: The potential need for pivoting has nothing to do with the matrix being singular. For example, the matrix

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{27}$$

is nonsingular, yet we can't process LU factorization unless we interchange rows. On the other hand, the matrix

$$\mathbf{A} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \tag{28}$$

can easily allow LU factorization

$$\mathbf{A} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} = \mathbf{LU}, \tag{29}$$

while being singular. □

1.3.2. Partial pivoting There is not only zero pivots, but also another situation we must avoid in Gaussian elimination – a case with *small* pivots. The problem is closely related to computer's finite-precision arithmetic which fails to recover any numbers smaller than the machine precision ϵ. Recall that we have $\epsilon \approx 10^{-7}$ for single precision, and $\epsilon \approx 10^{-16}$ for double precision.

Example: Let us now consider a matrix $\mathbf{A}$ defined as

$$\mathbf{A} = \begin{bmatrix} \tilde{\epsilon} & 1 \\ 1 & 1 \end{bmatrix}, \tag{30}$$

where $\tilde{\epsilon} < \epsilon \approx 10^{-16}$, say, $\tilde{\epsilon} = 10^{-20}$. If we proceed without any pivoting (i.e., no row interchange) and take $\tilde{\epsilon}$ as the first pivot element, then we obtain the elimination matrix

$$\mathbf{M} = \begin{bmatrix} 1 & 0 \\ -1/\tilde{\epsilon} & 1 \end{bmatrix}, \tag{31}$$

and hence the lower triangular matrix

$$\mathbf{L} = \begin{bmatrix} 1 & 0 \\ 1/\tilde{\epsilon} & 1 \end{bmatrix} \tag{32}$$

which is correct. For the upper triangular matrix, however, we see an incorrect floating-point arithmetic operation

$$\mathbf{U} = \begin{bmatrix} \tilde{\epsilon} & 1 \\ 0 & 1 - 1/\tilde{\epsilon} \end{bmatrix} = \begin{bmatrix} \tilde{\epsilon} & 1 \\ 0 & -1/\tilde{\epsilon} \end{bmatrix}, \tag{33}$$

since $1/\tilde{\epsilon} >> 1$. But then we simply fail to recover the original matrix $\mathbf{A}$ from the factorization:

$$\mathbf{LU} = \begin{bmatrix} 1 & 0 \\ 1/\tilde{\epsilon} & 1 \end{bmatrix} \begin{bmatrix} \tilde{\epsilon} & 1 \\ 0 & -1/\tilde{\epsilon} \end{bmatrix} = \begin{bmatrix} \tilde{\epsilon} & 1 \\ 1 & 0 \end{bmatrix} \neq \mathbf{A}. \tag{34}$$

Using a small pivot, and a correspondingly large multiplier, has caused an unrecoverable loss of information in the transformation.

We can cure the situation by interchanging the two rows first, which gives the first pivot element to be 1 and the resulting multiplier is $-\tilde{\epsilon}$:

$$\mathbf{M} = \begin{bmatrix} 1 & 0 \\ -\tilde{\epsilon} & 1 \end{bmatrix}, \tag{35}$$

and hence

$$\mathbf{L} = \begin{bmatrix} 1 & 0 \\ \tilde{\epsilon} & 1 \end{bmatrix} \text{ and } \mathbf{U} = \begin{bmatrix} 1 & 1 \\ 0 & 1 - \tilde{\epsilon} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \tag{36}$$

in floating-point arithmetic. We therefore recover the original relation:

$$\mathbf{LU} = \begin{bmatrix} 1 & 0 \\ \tilde{\epsilon} & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ \tilde{\epsilon} & 1 \end{bmatrix} = \mathbf{A}, \tag{37}$$

which is the correct result after permutation. □

The foregoing example is rather extreme, however, the principle in general holds to find the largest pivot in producing each elimination matrix, by which one obtains a smaller multiplier as an outcome and hence smaller errors in floating-point arithmetic. We see that this process involves repeated use of permutation matrix $\mathbf{P}_k$ that interchanges rows to bring the entry of largest magnitude on or below the diagonal in column k into the diagonal pivot position.

Quick summary: Gaussian elimination with *partial pivoting* proceeds as below. Assume x_{ik} is chosen to be the maximum in magnitude among the entries in k-th column, thereby selected as a k-th pivot:

$$\begin{bmatrix} * & * & * & * \\ & * & * & * \\ & x_{ik} & * & * \\ & * & * & * \end{bmatrix} \xrightarrow{\mathbf{P}_1} \begin{bmatrix} * & * & * & * \\ & x_{ik} & * & * \\ & * & * & * \\ & * & * & * \end{bmatrix} \xrightarrow{\mathbf{M}_1} \begin{bmatrix} * & * & * & * \\ & x_{ik} & * & * \\ & 0 & * & * \\ & 0 & * & * \end{bmatrix} \tag{38}$$

In general, $\mathbf{A}$ becomes an upper triangular matrix $\mathbf{U}$ after $n-1$ steps,

$$\mathbf{M}_{n-1}\mathbf{P}_{n-1}\cdots\mathbf{M}_1\mathbf{P}_1\mathbf{A} = \mathbf{U}. \tag{39}$$

□

Note: The expression in Eq. 39 can be rewritten in a way that separates the elimination and the permutation processes into two different groups

$$\mathbf{P} = \mathbf{P}_{n-1}\cdots\mathbf{P}_2\mathbf{P}_1, \tag{40}$$
$$\mathbf{L} = (\mathbf{M}'_{n-1}\cdots\mathbf{M}'_2\mathbf{M}'_1)^{-1}, \tag{41}$$

so that we write the final transformed matrix as

$$\mathbf{PA} = \mathbf{LU}. \tag{42}$$

To do this we first need to find what $\mathbf{M}'_i$ should be. Consider reordering the operations in Eq. 39 in the form, for instance with $n-1=3$,

$$\mathbf{M}_3\mathbf{P}_3\mathbf{M}_2\mathbf{P}_2\mathbf{M}_1\mathbf{P}_1 = \mathbf{M}'_3\mathbf{M}'_2\mathbf{M}'_1\mathbf{P}_3\mathbf{P}_2\mathbf{P}_1 (= \mathbf{L}^{-1}\mathbf{P}). \tag{43}$$

Rearranging operations,

$$\mathbf{M}_3\mathbf{P}_3\mathbf{M}_2\mathbf{P}_2\mathbf{M}_1\mathbf{P}_1 \tag{44}$$
$$= (\mathbf{M}_3)(\mathbf{P}_3\mathbf{M}_2\mathbf{P}_3^{-1})(\mathbf{P}_3\mathbf{P}_2\mathbf{M}_1\mathbf{P}_2^{-1}\mathbf{P}_3^{-1})(\mathbf{P}_3\mathbf{P}_2\mathbf{P}_1) \tag{45}$$
$$\equiv (\mathbf{M}'_3)(\mathbf{M}'_2)(\mathbf{M}'_1)\mathbf{P}_3\mathbf{P}_2\mathbf{P}_1, \tag{46}$$

whereby we can define $\mathbf{M}'_i, i = 1,2,3$ equals to $\mathbf{M}_i$ but with the subdiagonal entries permuted:

$$\mathbf{M}'_3 = \mathbf{M}_3 \tag{47}$$
$$\mathbf{M}'_2 = \mathbf{P}_3\mathbf{M}_2\mathbf{P}_3^{-1} \tag{48}$$
$$\mathbf{M}'_1 = \mathbf{P}_3\mathbf{P}_2\mathbf{M}_1\mathbf{P}_2^{-1}\mathbf{P}_3^{-1} \tag{49}$$

We can see that the matrix $\mathbf{M}'_{n-1} \cdots \mathbf{M}'_2 \mathbf{M}'_1$ is unit lower triangular and hence easily invertible by negating the subdiagonal entries to obtain $\mathbf{L}$. □

Example: To see what is going on, consider

$$\mathbf{A} = \begin{bmatrix} 2 & 1 & 1 & 0 \\ 4 & 3 & 3 & 1 \\ 8 & 7 & 9 & 5 \\ 6 & 7 & 9 & 8 \end{bmatrix}. \tag{50}$$

With partial pivoting, let's interchange the first and third rows with $\mathbf{P}_1$:

$$\begin{bmatrix} & & 1 & \\ & 1 & & \\ 1 & & & \\ & & & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & 1 & 0 \\ 4 & 3 & 3 & 1 \\ 8 & 7 & 9 & 5 \\ 6 & 7 & 9 & 8 \end{bmatrix} = \begin{bmatrix} 8 & 7 & 9 & 5 \\ 4 & 3 & 3 & 1 \\ 2 & 1 & 1 & 0 \\ 6 & 7 & 9 & 8 \end{bmatrix}. \tag{51}$$

The first elimination step now looks like this with left-multiplication by $\mathbf{M}_1$:

$$\begin{bmatrix} 1 & & & \\ -1/2 & 1 & & \\ -1/4 & & 1 & \\ -3/4 & & & 1 \end{bmatrix} \begin{bmatrix} 8 & 7 & 9 & 5 \\ 4 & 3 & 3 & 1 \\ 2 & 1 & 1 & 0 \\ 6 & 7 & 9 & 8 \end{bmatrix} = \begin{bmatrix} 8 & 7 & 9 & 5 \\ & -1/2 & -3/2 & -3/2 \\ & -3/4 & -5/4 & -5/4 \\ & 7/4 & 9/4 & 17/4 \end{bmatrix}. \tag{52}$$

Now the second and fourth rows are interchanged with $\mathbf{P}_2$:

$$\begin{bmatrix} 1 & & & \\ & & & 1 \\ & & 1 & \\ & 1 & & \end{bmatrix} \begin{bmatrix} 8 & 7 & 9 & 5 \\ & -1/2 & -3/2 & -3/2 \\ & -3/4 & -5/4 & -5/4 \\ & 7/4 & 9/4 & 17/4 \end{bmatrix} = \begin{bmatrix} 8 & 7 & 9 & 5 \\ & 7/4 & 9/4 & 17/4 \\ & -3/4 & -5/4 & -5/4 \\ & -1/2 & -3/2 & -3/2 \end{bmatrix}. \tag{53}$$

With multiplication by $\mathbf{M}_2$ the second elimination step looks like:

$$\begin{bmatrix} 1 & & & \\ & 1 & & \\ & 3/7 & 1 & \\ & 2/7 & & 1 \end{bmatrix} \begin{bmatrix} 8 & 7 & 9 & 5 \\ & 7/4 & 9/4 & 17/4 \\ & -3/4 & -5/4 & -5/4 \\ & -1/2 & -3/2 & -3/2 \end{bmatrix} = \begin{bmatrix} 8 & 7 & 9 & 5 \\ & 7/4 & 9/4 & 17/4 \\ & & -2/7 & 4/7 \\ & & -6/7 & -2/7 \end{bmatrix} \tag{54}$$

Interchanging the third and fourth rows now with $\mathbf{P}_3$:

$$\begin{bmatrix} 1 & & & \\ & 1 & & \\ & & & 1 \\ & & 1 & \end{bmatrix} \begin{bmatrix} 8 & 7 & 9 & 5 \\ & 7/4 & 9/4 & 17/4 \\ & & -2/7 & 4/7 \\ & & -6/7 & -2/7 \end{bmatrix} \begin{bmatrix} 8 & 7 & 9 & 5 \\ & 7/4 & 9/4 & 17/4 \\ & & -6/7 & -2/7 \\ & & -2/7 & 4/7 \end{bmatrix}. \tag{55}$$

The final elimination step is obtained with $\mathbf{M}_3$:

$$\begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & -1/3 & 1 \end{bmatrix} \begin{bmatrix} 8 & 7 & 9 & 5 \\ & 7/4 & 9/4 & 17/4 \\ & & -6/7 & -2/7 \\ & & -2/7 & 4/7 \end{bmatrix} = \begin{bmatrix} 8 & 7 & 9 & 5 \\ & 7/4 & 9/4 & 17/4 \\ & & -6/7 & -2/7 \\ & & & 2/3 \end{bmatrix}. \tag{56}$$

□

Remark: The name "partial" pivoting comes from the fact that *only the current column* is searched for a suitable pivot. A more exhausting pivoting strategy is *complete pivoting*, in which the entire remaining unreduced sub matrix is searched for the largest entry, which is then permuted into the diagonal pivot position. □

Algorithm: LU factorization by Gaussian elimination with Partial Pivoting:

```
for k = 1 to n − 1
    #[loop over column]
    Find index p such that
        |a_pk| ≥ |a_ik| for k ≤ i ≤ n
        #[search for pivot in current column]
    if p ≠ k then
        interchange rows k and p
        #[interchange rows if needed]
    endif
    if a_kk = 0 then
        continue with next k
        #[skip current column if zero]
    endif
    for i = k + 1 to n
        m_ik = a_ik / a_kk
        #[compute multipliers for each column]
    endfor
    for j = k + 1 to n
        for i = k + 1 to n
            a_ij = a_ij − m_ik a_kj
            #[transformation to remaining submatrix]
        endfor
    endfor
endfor
```
