Pivoting for LU Factorization

Matthew W. Reid

April 21, 2014

University of Puget Sound
E-mail: mreid@pugetsound.edu

1 Introduction

Pivoting for LU factorization is the process of systematically selecting pivots for Gaussian elimination during the LU factorization of a matrix. The LU factorization is closely related to Gaussian elimination, which is unstable in its pure form. To guarantee the elimination process goes to completion, we must ensure that there is a nonzero pivot at every step of the elimination process. This is the reason we need pivoting when computing LU factorizations. But we can do more with pivoting than just making sure Gaussian elimination completes. We can reduce roundoff errors during computation and make our algorithm backward stable by implementing the right pivoting strategy. Depending on the matrix A, some LU decompositions can become numerically unstable if relatively small pivots are used. Relatively small pivots cause instability because they operate very similar to zeros during Gaussian elimination. Through the process of pivoting, we can greatly reduce this instability by ensuring that we use relatively large entries as our pivot elements. This prevents large factors from appearing in the computed L and U, which reduces roundoff errors during computation. The three pivoting strategies I am going to discuss are partial pivoting, complete pivoting, and rook pivoting, along with an explanation of the benefits obtained from using each strategy.

2 Permutation Matrices

The process of swapping rows and columns is crucial to pivoting. Permutation matrices are used to achieve this. A permutation matrix is the identity matrix with interchanged rows. When these matrices multiply another matrix they swap the rows or columns of the matrix. Left multiplication by a permutation matrix will result in the swapping of rows while right multiplication will swap columns. For example, in order to swap rows 1 and 3 of a matrix A, we right multiply by a permutation matrix P, which is the identity matrix with rows 1 and 3 swapped. To interchange columns 1 and 3 of the matrix A, we left multiply by the same permutation matrix P. At each step k of Gaussian elimination we encounter an optional row interchange, represented by P_k, before eliminating entries via the lower triangular elementary matrix M_k. If there is no row interchange required, P_k is simply the identity matrix. Think of this optional row interchange as switching the order of the equations when presented with a system of equations. For Gaussian elimination to terminate successfully, this process is necessary when encountering a zero pivot. When describing the different pivoting strategies, I will denote a permutation matrix that swaps rows with P_k and will denote a permutation matrix that swaps columns by refering to the matrix as Q_k. When computing the LU factorizations of matrices, we will routinely pack the permutation matrices together into a single permutation matrix. They are simply a matrix product of all the permutation matrices used to achieve the factorization. I will define these matrices here.

When computing $PA = LU$,

$$P = P_k P_{k-1} ... P_2 P_1 \tag{1}$$

where k is the index of the elimination.

When computing $PAQ = LU$,

$$Q = Q_1 Q_2 ... Q_{k-1} Q_k \tag{2}$$

where k is the index of the elimination.

3 Role of Pivoting

The role of pivoting is to reduce instability that is inherent in Gaussian elimination. Instability arises only if the factor L or U is relatively large in size compared to A. Pivoting reduces instablilty by ensuring that L and U are not too big relative to A. By keeping all of the intermediate quantites that appear in the elimination process a managable size, we can minimize rounding errors. As a consequence of pivoting, the algorithm for computing the LU factorization is backward stable. I will define backward stability in the upcoming paragraphs.

3.1 Zero Pivots

The first cause of instability is the situation in which there is a zero in the pivot position. With a zero as a pivot element, the calculation cannot be completed because we encounter the error of division by zero. In this case, pivoting is essential to finishing the computation. Consider the matrix A.

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

When trying to compute the factors L and U, we fail in the first elimination step because of division by zero.

3.2 Small Pivots

Another cause of instability in Gaussian elimination is when encountering relatively small pivots. Gaussian elimination is guaranteed to succeed if row or column interchanges are used in order to avoid zero pivots when using exact calculations. But, when computing the LU factorization numerically, this is not necessarily true. When a calculation is undefined because of division by zero, the same calculation will suffer numerical difficulties when there is division by a nonzero number that is relatively small.

$$A = \begin{bmatrix} 10^{-20} & 1 \\ 1 & 2 \end{bmatrix}$$

When computing the factors L and U, the process does not fail in this case because there is no division by zero.

$$L = \begin{bmatrix} 1 & 0 \\ 10^{20} & 1 \end{bmatrix}, U = \begin{bmatrix} 10^{-20} & 1 \\ 0 & 2 - 10^{-20} \end{bmatrix}$$

When these computations are performed in floating point arithmetic, the number $2 - 10^{-20}$ is not represented exactly but will be rounded to the nearest floating point number which we will say is -10^{-20}. This means that our matrices are now floating point matrices L' and U'.

$$L' = \begin{bmatrix} 1 & 0 \\ 10^{20} & 1 \end{bmatrix}, \; U' = \begin{bmatrix} 10^{-20} & 1 \\ 0 & -10^{20} \end{bmatrix}$$

The small change we made in U to get U' shows its significance when we compute $L'U'$.

$$L'U' = \begin{bmatrix} 10^{-20} & 1 \\ 1 & 0 \end{bmatrix}$$

The matrix $L'U'$ should be equal to A, but obviously that is not the case. The 2 in position $(2,2)$ of matrix A is now 0. Also, when trying to solve a system such as $A\mathbf{x} = \mathbf{b}$ using the LU factorization, the factors $L'U'$ would not give you a correct answer. The LU factorization was a stable computation but not backward stable.

Many times we compute LU factorizations in order to solve systems of equations. In that case, we want to find a solution to the vector equation $A\mathbf{x} = \mathbf{b}$, where A is the coefficient matrix of interest and $\mathbf{b}$ is the vector of constants. Therefore, for solving this vector equation specifically, the definition of backward stability is as follows.

Definition 3.1. An algorithm is stable for a class of matrices C if for every matrix $A \in C$, the computed solution by the algorithm is the exact solution to a nearby problem. Thus, for a linear system problem

$$A\mathbf{x} = \mathbf{b}$$

an algorithm is stable for a class of matrices C if for every $A \in C$ and for each $\mathbf{b}$, it produces a computed solution $\hat{\mathbf{x}}$ that satisfies

$$(A + E)\hat{\mathbf{x}} = \mathbf{b} + \delta\mathbf{b}$$

for some E and $\delta\mathbf{b}$, where $(A + E)$ is close to A and $\mathbf{b} + \delta\mathbf{b}$ is close to $\mathbf{b}$.

The LU factorization without pivoting is not backward stable because the computed solution $\hat{\mathbf{x}}$ for the problem $L'U'\hat{\mathbf{x}} = \mathbf{b}$ is not the exact solution to a nearby problem $A\mathbf{x} = \mathbf{b}$. The computed solution to $L'U'\hat{\mathbf{x}} = \mathbf{b}$, with $\mathbf{b} = \begin{bmatrix} 1 \\ 3 \end{bmatrix}$, is $\hat{\mathbf{x}} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$. Yet the correct solution is $\mathbf{x} \approx \begin{bmatrix} 1 \\ 1 \end{bmatrix}$.

4 Partial Pivoting

The goal of partial pivoting is to use a permutation matrix to place the largest entry of the first column of the matrix at the top of that first column. For an $n \times n$ matrix B, we scan n rows of the first column for the largest value. At step k of the elimination, the pivot we choose is the largest of the $n - (k + 1)$ subdiagonal entries of column k, which costs $O(nk)$ operations for each step of the elimination. So for a $n \times n$ matrix, there is a total of $O(n^2)$ comparisons. Once located, this entry is then moved into the pivot position A_{kk} on the diagonal of the matrix. So in the first step the entry is moved into the (1,1) position of matrix B. We interchange rows by multiplying B on the left with a permutation matrix P. After we multiply matrix B by P we continue the LU factorization and use our new pivot to clear out the entries below it in its column in order to obtain the upper triangular matrix U. The general factorization of a $n \times n$ matrix A to a lower triangular matrix L and an upper triangular matrix U using partial pivoting is represented by the following equations.

$$M'_k = (P_{n-1} \cdots P_{k+1}) M_k (P_{k+1} \cdots P_{n-1}) \tag{3}$$

$$(M_{n-1} M_{n-2} \cdots M_2 M_1)^{-1} = L \tag{4}$$

$$M_{n-1} P_{n-1} M_{n-2} P_{n-2} \cdots M_2 P_2 M_1 P_1 A = U \tag{5}$$

In order to get a visual sense of how pivots are selected, let γ represent the largest value in the matrix B and the bold represent entries of the matrix being scanned as potential pivots.

$$B = \begin{bmatrix} \mathbf{x} & x & x & x \\ \mathbf{x} & x & x & x \\ \boldsymbol{\gamma_1} & x & x & x \\ \mathbf{x} & x & x & x \end{bmatrix} \rightarrow \begin{bmatrix} \gamma_1 & x & x & x \\ 0 & \mathbf{x} & x & x \\ 0 & \mathbf{x} & x & x \\ 0 & \boldsymbol{\gamma_2} & x & x \end{bmatrix} \rightarrow \begin{bmatrix} \gamma_1 & x & x & x \\ 0 & \gamma_2 & x & x \\ 0 & 0 & \mathbf{x} & x \\ 0 & 0 & \boldsymbol{\gamma_3} & x \end{bmatrix} \rightarrow \begin{bmatrix} \gamma_1 & x & x & x \\ 0 & \gamma_2 & x & x \\ 0 & 0 & \gamma_3 & x \\ 0 & 0 & 0 & x \end{bmatrix}$$

After $n - 1$ permutations, the last permutation becomes trivial as the column of interest has only a single entry. As a numerical example, consider the matrix A.

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

In this case, we want to use a permutation matrix to place the largest entry of the first column into the position A_{11}. For this matrix, this means we want 3 to be the first entry of the first column. So we use the permutation matrix P_1,

$$P_1 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \; P_1 A = \begin{bmatrix} 3 & 2 & 4 \\ 2 & 1 & 3 \\ 1 & 2 & 4 \end{bmatrix}$$

Then we use the matrix M_1 to eliminate the bottom two entries of the first column.

$$M_1 = \begin{bmatrix} 1 & 0 & 0 \\ -2/3 & 1 & 0 \\ -1/3 & 0 & 1 \end{bmatrix}, \; M_1P_1A = \begin{bmatrix} 3 & 2 & 4 \\ 0 & -1/3 & 1/3 \\ 0 & 4/3 & 8/3 \end{bmatrix}$$

Then, we use the permutation matrix P_2 to move the larger of the bottom two entries in column two of A into the position a_{22}.

$$P_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \; P_2M_1P_1A = \begin{bmatrix} 3 & 2 & 4 \\ 0 & 4/3 & 8/3 \\ 0 & -1/3 & 1/3 \end{bmatrix}$$

As we did before, we are going to use matrix multiplication to eliminate entries at the bottom of column 2. We use matrix M_2 to achieve this.

$$M_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1/4 & 1 \end{bmatrix}, \; M_2P_2M_1P_1A = \begin{bmatrix} 3 & 2 & 4 \\ 0 & 4/3 & 8/3 \\ 0 & 0 & 1 \end{bmatrix}$$

This upper triangular matrix product is our U matrix. The L matrix comes from the matrix product $(M_2P_2M_1P_1)^{-1}$.

$$L = (M_2P_2M_1P_1)^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 1/3 & 1 & 0 \\ 2/3 & -1/4 & 1 \end{bmatrix}$$

To check, we can show that $PA = LU$, where $P = P_2P_1$.

$$PA = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 2 & 4 \\ 2 & 1 & 3 \\ 3 & 2 & 4 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1/3 & 1 & 0 \\ 2/3 & -1/4 & 1 \end{bmatrix} \begin{bmatrix} 3 & 2 & 4 \\ 0 & 4/3 & 8/3 \\ 0 & 0 & 1 \end{bmatrix} = LU$$

Through the process of partial pivoting, we get the smallest possible quantites for the elimination process. As a consequence, we will have no quantity larger than 1 in magnitude which minimizes the chance of large errors. Partial pivoting is the most commmon method of pivoting as it is the fastest and least costly of all the pivoting methods.

5 Complete Pivoting

When employing the complete pivoting strategy we scan for the largest value in the entire submatrix $A_{k:n,k:n}$, where k is the number of the elimination, instead of just the next subcolumn. This requires $O(n-k)^2$ comparisons for every step in the elimination to find the pivot. The total cost becomes $O(n^3)$ comparisons for an $n \times n$ matrix. For an $n \times n$ matrix A, the first step is to scan n rows and n columns for the largest value. Once located, this entry is then permuted into the next diagonal pivot position of the matrix. So in the first step the entry is permuted into the $(1,1)$ position of matrix A. We interchange rows exactly as we did in partial pivoting, by multiplying

A on the left with a permutation matrix P. Then to interchange columns, we multiply A on the right by another permutation matrix Q. The matrix product PAQ interchanges rows and columns accordingly so that the largest entry in the matrix is in the (1,1) position of A. With complete pivoting, the general equation for L is the same as for partial pivoting (equation 3 and 4), but the equation for U changes slightly.

$$M_{n-1}P_{n-1}M_{n-2}P_{n-2}\cdots M_2P_2M_1P_1AQ_1Q_2\cdots Q_{n-1} = U \tag{6}$$

Once again, let γ represent the largest value in the matrix B and the bold represent entries of the matrix being scanned.

$$B = \begin{bmatrix} \mathbf{x} & \mathbf{x} & \mathbf{x} & \mathbf{x} \\ \mathbf{x} & \mathbf{x} & \mathbf{x} & \mathbf{x} \\ \mathbf{x} & \mathbf{x} & \boldsymbol{\gamma_1} & \mathbf{x} \\ \mathbf{x} & \mathbf{x} & \mathbf{x} & \mathbf{x} \end{bmatrix} \rightarrow \begin{bmatrix} \gamma_1 & x & x & x \\ 0 & \mathbf{x} & \mathbf{x} & \mathbf{x} \\ 0 & \mathbf{x} & \mathbf{x} & \boldsymbol{\gamma_2} \\ 0 & \mathbf{x} & \mathbf{x} & \mathbf{x} \end{bmatrix} \rightarrow \begin{bmatrix} \gamma_1 & x & x & x \\ 0 & \gamma_2 & x & x \\ 0 & 0 & \mathbf{x} & \mathbf{x} \\ 0 & 0 & \mathbf{x} & \boldsymbol{\gamma_3} \end{bmatrix} \rightarrow \begin{bmatrix} \gamma_1 & x & x & x \\ 0 & \gamma_2 & x & x \\ 0 & 0 & \gamma_3 & x \\ 0 & 0 & 0 & x \end{bmatrix}$$

After $n-1$ permutations, the last permutation becomes trivial as the submatrix of interest is a 1×1 matrix. For a numerical example, consider the matrix A.

$$A = \begin{bmatrix} 2 & 3 & 4 \\ 4 & 7 & 5 \\ 4 & 9 & 5 \end{bmatrix}$$

In this case, we want to use two different permutation matrices to place the largest entry of the entire matrix into the position A_{11}. For this matrix, this means we want 9 to be the first entry of the first column. To achieve this we multiply A on the left by the permutation matrix P_1 and multiply on the right by the permutation matrix Q_1.

$$P_1 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \mathrm{Q}_1 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} P_1AQ_1 = \begin{bmatrix} 3 & 2 & 4 \\ 2 & 1 & 3 \\ 1 & 2 & 4 \end{bmatrix}$$

Then in computing the LU factorization, the matrix M_1 is used to eliminate the bottom two entries of the first column.

$$M_1 = \begin{bmatrix} 1 & 0 & 0 \\ -7/9 & 1 & 0 \\ -1/3 & 0 & 1 \end{bmatrix}, M_1P_1AQ_1 = \begin{bmatrix} 9 & 4 & 5 \\ 0 & 8/9 & 10/9 \\ 0 & 2/3 & 7/3 \end{bmatrix}$$

Then, the permutation matrices P_2 and Q_2 are used to move the largest entry of the submatrix $A_{2:3;2:3}$ into the position A_{22}.

$$P_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}, Q_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}, P_2M_1P_1AQ_1Q_2 = \begin{bmatrix} 9 & 5 & 4 \\ 0 & 7/3 & 2/3 \\ 0 & 10/9 & 8/9 \end{bmatrix}$$

As we did before, we are going to use matrix multiplication to eliminate entries at the bottom of the column 2. We use matrix M_2 to achieve this,

$$M_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -10/21 & 1 \end{bmatrix}, \; M_2P_2M_1P_1AQ_1Q_2 = \begin{bmatrix} 3 & 2 & 4 \\ 0 & 7/3 & 2/3 \\ 0 & 0 & 4/7 \end{bmatrix}$$

This upper triangular matrix product is the U matrix. The L matrix comes from the inverse of the product of M and P matrices.

$$L = (M_2P_2M_1P_1)^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ 1/3 & 1 & 0 \\ 7/9 & 10/21 & 1 \end{bmatrix}$$

To check, we can show that $PAQ = LU$, where $P = P_2P_1$ and $Q = Q_1Q_2$.

$$PAQ = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 2 & 4 \\ 2 & 1 & 3 \\ 3 & 2 & 4 \end{bmatrix} \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1/3 & 1 & 0 \\ 7/9 & 10/21 & 1 \end{bmatrix} \begin{bmatrix} 9 & 5 & 4 \\ 0 & 7/3 & 2/3 \\ 0 & 0 & 4/7 \end{bmatrix} = LU$$

Complete pivoting requires both row and column interchanges. In theory, complete pivoting is more reliable than partial pivoting because you can use it with singular matrices. When a matrix is rank deficient complete pivoting can be used to reveal the rank of the matrix. Suppose A is a $n \times n$ matrix such that $r(A) = r < n$. At the start of the $r+1$ elimination, the submatrix $A_{r+1:n,r+1:n} = 0$. This means that for the remaining elimination steps, $P_k = Q_k = M_k = I$ for $r+1 \leq k \leq n$. So after step r of the elimination, the algorithm can be terminated with the following factorization.

$$PAQ = LU = \begin{bmatrix} L_{11} & 0 \\ L_{21} & I_{n-r} \end{bmatrix} \begin{bmatrix} U_{11} & U_{12} \\ 0 & 0 \end{bmatrix}$$

In this case, L_{11} and U_{11} have size $r \times r$ and L_{21} and U_{12}^T have size $(n-r) \times r$. Complete pivoting works in this case because we are able to identify that every entry of the submatrix $A_{r+1:n,r+1:n}$ is 0. This would not work with partial pivoting because partial pivoting only searches a single column for a pivot and would fail before recognizing that all remaining potential pivots were zero. But in practice, we know that encountering a rank deficient matrix in numerical linear algebra is very rare. This is the reason complete pivoting does not consistently produce more accurate results than partial pivoting. Therefore, partial pivoting is usually the best pivoting strategy of the two.

6 Rook Pivoting

Rook pivoting is a third alternative pivoting strategy. For this strategy, in the k^{th} step of the elimination, where $1 \leq k \leq n-1$ for a $n \times n$ matrix A, we scan the submarix $A_{k:n,k:n}$ for values that are the largest in their respective row and column. Once we have found our potential pivots, we are free to choose any that we please. This is because the identified potential pivots are large enough so that there will be a very minimal difference in the end factors, no matter which pivot is chosen. When a computer uses rook pivoting, it searches only for the first entry that is the largest in its respective row and column. The computer chooses the first one it finds because even if there are more, they all will be equally effective as pivots. This allows for some options in our decision making with regards to the pivot we want to use. Rook pivoting is the best of both worlds in that it costs the same amount of time as partial pivoting while it has the same reliability as complete pivoting.

$$B = \begin{bmatrix} \mathbf{x} & \mathbf{x} & \mathbf{x} & \gamma \\ \mathbf{x} & \gamma & \mathbf{x} & \mathbf{x} \\ \gamma & \mathbf{x} & \mathbf{x} & \mathbf{x} \\ \mathbf{x} & \mathbf{x} & \gamma & \mathbf{x} \end{bmatrix} \rightarrow \begin{bmatrix} \gamma_1 & x & x & x \\ 0 & \mathbf{x} & \gamma & \mathbf{x} \\ 0 & \gamma & \mathbf{x} & \mathbf{x} \\ 0 & \mathbf{x} & \mathbf{x} & \gamma \end{bmatrix} \rightarrow \begin{bmatrix} \gamma_1 & x & x & x \\ 0 & \gamma_2 & x & x \\ 0 & 0 & \mathbf{x} & \gamma \\ 0 & 0 & \gamma & \mathbf{x} \end{bmatrix} \rightarrow \begin{bmatrix} \gamma_1 & x & x & x \\ 0 & \gamma_2 & x & x \\ 0 & 0 & \gamma_3 & x \\ 0 & 0 & 0 & x \end{bmatrix}$$

For a numerical example, consider the matrix A.

$$A = \begin{bmatrix} 1 & 4 & 7 \\ 7 & 8 & 2 \\ 9 & 5 & 1 \end{bmatrix}$$

When using the partial pivoting technique, we would identify 8 as our first pivot element. When using complete pivoting, we would use 9 as our first pivot element. When rook pivoting we identify entries that are the largest in their respective rows and columns. So following this logic, we identify 8, 9, and 7 all as potential pivots. In this example we will choose 7 as the first pivot. Since we are choosing 7 for our first pivot element, we multiply A on the right by the permutation matrix Q_1 to move 7 into the position A_{11}. Note that we will also multiply on the left of A by P_1, but P_1 is simply the identity matrix so there will be no effect on A.

$$Q_1 = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \; P_1AQ_1 = \begin{bmatrix} 7 & 4 & 1 \\ 2 & 8 & 7 \\ 1 & 5 & 9 \end{bmatrix}$$

Then we eliminate the entries in row 2 and 3 of column 1 via the matrix M_1.

$$M_1 = \begin{bmatrix} 1 & 0 & 0 \\ -2/7 & 1 & 0 \\ -1/7 & 0 & 1 \end{bmatrix}, \; M_1P_1AQ_1 = \begin{bmatrix} 7 & 4 & 1 \\ 0 & 48/7 & 47/7 \\ 0 & 30/7 & 62/7 \end{bmatrix}$$

If we were employing partial pivoting, our pivot would be $\frac{48}{7}$ and if we were using complete pivoting, our pivot would be $\frac{62}{7}$. But using the rook pivoting strategy allows us to choose either $\frac{48}{7}$ or $\frac{62}{7}$ as our next pivot. In this situation, we do not need to permute the matrix $M_1P_1AQ_1$ to continue with our factorization because one of our potential pivots is already in pivot postion. To keep things interesting, we will permute the matrix anyway and choose $\frac{62}{7}$ to be our next pivot. To achieve this we wil multiply by the matrix P_2 on the left and by the matrix Q_2 on the right.

$$P_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \; Q_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \; P_2M_1P_1AQ_1Q_2 = \begin{bmatrix} 7 & 1 & 4 \\ 0 & 62/7 & 30/7 \\ 0 & 47/7 & 48/7 \end{bmatrix}$$

At this point, we want to eliminate the last entry of the second column to complete the factorization. We achieve this by matrix M_2.

$$M_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -47/62 & 1 \end{bmatrix},\ M_2P_2M_1P_1AQ_1Q_2 = \begin{bmatrix} 7 & 1 & 4 \\ 0 & 62/7 & 30/7 \\ 0 & 0 & 7/2 \end{bmatrix}$$

After the elimination the matrix is now upper triangular which we recognize as the factor U. The factor L is equal to the matrix product $(L_2Q_2L_1Q_2)^{-1}$. You can use Sage to check that $PAQ = LU$, where $P = P_2P_1$ and $Q = Q_1Q_2$.

$$PAQ = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 4 & 7 \\ 7 & 8 & 2 \\ 9 & 5 & 1 \end{bmatrix} \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 1/7 & 1 & 0 \\ 2/7 & 47/62 & 1 \end{bmatrix} \begin{bmatrix} 7 & 1 & 4 \\ 0 & 62/7 & 30/7 \\ 0 & 0 & 7/2 \end{bmatrix} = LU$$

7 Conclusion

In its most basic form, pivoting guarantees that Gaussian elimination runs to completion by permuting the matrix whenever a zero pivot is encountered. However, pivoting is more beneficial than just ensuring Gaussian elimination goes to completion. When doing numerical computations, we pivot to avoid small pivots in addition to zero pivots. They are the cause of instability in the factorization and pivoting is advatageous in avoiding those small pivots. Pivoting dramatically reduces roundoff error in numerical calcuations by preventing large entries from being present in the factors L and U. This also makes the LU factorization backward stable which is essential when using L and U to solve a system of equations. Partial pivoting is the most common pivoting strategy because it is the cheapest, fastest algorithm to use while sacrifing very little accuracy. In practice, partial pivoting is just as accurate as complete pivoting. Complete pivoting is theoretically the most stable strategy as it can be used even when a matrix is singular, but it is much slower than partial pivoting. Rook pivoting is the newest strategy and it combines the speed of partial pivoting with the accuracy of complete pivoting. That being said, it is fairly new and not as widely used as partial pivoting. Pivoting is essential to any LU factorization as it ensures accurate and stable computations.

References

[1] Biswa Nath Datta *Numerical Linear Algebra and Applications, Second Edition* 2010: Society for Industrial and Applied Mathmatics (SIAM).

[2] Gene H. Golub and Charles F. Van Loan*Matrix Computations 4th Edition* 2013: The John Hopkins University Press.

[3] James E. Gentle *Matrix Algebra: Theory, Computations, and Applications in Statistics* 2007: Springer Science + Business Media, LLC.

[4] Leslie Hogben *Handbook of Linear Algebra* 2007: Taylor and Francis Group, LLC.

[5] Lloyd N. Trefethen and David Bau, III *Numerical Linear Algebra* 1997: Society for Industrial and Applied Mathmatics (SIAM).

[6] Nicholas J. Higman *Accuracy and Stability of Numerical Algorithms, Second Edition* 2002: Society for Industrial and Applied Mathmatics (SIAM).

[7] Nicholas J. Higman *"Gaussian Elimination"; http://www.maths.manchester.ac.uk/ higham/papers /high11g.pdf* 2011: John Wiley and Sons, Inc.

[8] Phillip Gill, Walter Murray, and Margaret H. Wright *Numerical Linear Algebra and Optimization Volume 1* 1991: Addison-Wesley Publishing Company.

[9] Robert Beezer *A Second Course in Linear Algebra* 2013.

[10] Xiao-Wen Chang *Some Features of Gaussian Elimination with Rook Pivoting* 2002: School of Computer Science, McGill University.