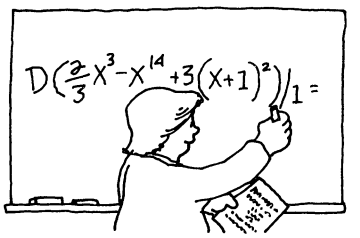
CLASSROOM CAPSULES

EDITOR

Frank Flanigan
Department of Mathematics and Computer Science
San Jose State University
San Jose, CA 95192

ASSISTANT EDITOR

Richard Pfiefer
San Jose State University



A Classroom Capsule is a short article that contains a new insight on a topic taught in the earlier years of undergraduate mathematics. Please submit manuscripts prepared according to the guidelines on the inside front cover to Nazanin Azarnia, Department of Mathematics, Miami University, Hamilton, Hamilton, OH 45011.

Gaussian Elimination in Integer Arithmetic: An Application of the *L-U* Factorization

Thomas Hern, Bowling Green State University, Bowling Green, OH 43403

> I am convinced that some version of Gaussian elimination should be taught. I further believe that students have to work a few examples by hand in order to understand the method. I recommend working three or four small examples. More than that is cruel and unusual punishment!
>
> John Kemeny [7, p. 45]

We will present a convenient, practical method of generating examples of matrices for which the Gaussian elimination process can be done in integer arithmetic, resulting in an echelon form with integer entries. The method can be refined so that the inverse has integer entries. The method is based on the *L-U* factorization, and we will also give an elementary justification for that factorization.

A number of authors have examined the problem of constructing or characterizing *nice* matrices, those with special properties such as having integer eigenvalues [1], [2], [3], [6], [8], [9] or all integer entries in the inverse [1], [4], [5]. For a problem or example, however, we would also like the Gaussian elimination process to be free, or nearly free, of fractions or decimals. A traditional method has been to work the problem backwards from a suitably chosen answer (this method is implied in [1] and [4]). The idea presented here is a variation on that theme, but the method makes the process more convenient, more reliable, and more easily carried out on a computer.

One difficulty with the "work it backwards" method is that a mistake made in the arithmetic at any stage destroys all the "niceness" of the matrix, with disastrous results; and such mistakes are likely to occur when done by hand. The method presented here requires only the selection and multiplication of two matrices. Even this multiplication is prone to error if done by hand, but computer software and hand calculators can take over this task and reduce the likelihood of error. We are free to think about the design of the problem, and can generate matrices easily and accurately.

The Method. To generate a matrix A that reduces to the upper triangular matrix U with operations in integer arithmetic:

1) Choose the desired upper triangular matrix U with integer entries.
2) Choose a lower triangular matrix L with ones on the diagonal, and integer entries.
3) Let $A = LU$.

Example. The matrix generated by

$$LU = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ -2 & -1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 2 & 1 \\ 0 & 3 & 1 \\ 0 & 0 & -2 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 1 \\ 1 & 5 & 2 \\ -2 & -7 & -5 \end{bmatrix}$$

results in the Gaussian elimination

$$\begin{bmatrix} 1 & 2 & 1 \\ 1 & 5 & 2 \\ -2 & -7 & -5 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 2 & 1 \\ 0 & 3 & 1 \\ 0 & -3 & -3 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 2 & 1 \\ 0 & 3 & 1 \\ 0 & 0 & -2 \end{bmatrix}.$$

The row operations should be carried out in the natural order of choosing the next available pivot, and doing no division. It will not be necessary to rescale rows, since the multiples are already designed to be integers. In fact, rescaling, say to have pivots all ones, may disturb the integer arithmetic. It will also not be necessary to permute rows to get a nonzero pivot, since zero pivots will not occur in these matrices. Such permutations may also disturb the integer arithmetic. Granted, this restricts the scope of problems somewhat. More complex issues, such as partial pivoting for numerical stability, can be dealt with later using computers.

Notice that in the example the entries in the matrix L are the negatives of the multipliers used in the row operations. This is no accident, as we will soon see. Thus if the entries of both L and U are chosen to be integers, the multipliers and all entries of A will be integers, so then the arithmetic will be in integers as well, which is what we wanted. The method is based on the L-U factorization, which is now creeping into standard linear algebra textbooks. A good general reference is Strang [**10**] or Tucker [**11**].

L-U Factorization. *So long as no row exchanges are needed in Gaussian elimination, a matrix can be written as a product LU where*:

- *L is a lower triangular matrix with ones on the diagonal. The (i, j) entry in L is the negative of the multiplier l_{ij} in*

$$row(i) \rightarrow row(i) + l_{ij} row(j), \quad i > j$$

  during Gaussian elimination.
- *The matrix U is an upper triangular matrix, the result of Gaussian elimination.*

The following argument is persuasive enough to justify this factorization, and the general case can be imagined easily. For a 3×3 matrix A, writing Gaussian elimination (done in the natural order) in terms of elementary matrices, we get

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & c & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ b & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ a & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} A = U$$

where U is upper triangular. No rescaling or row exchanges are done, just adding a multiple of one row to another. The numbers a, b, and c are the corresponding multiples. Inverting each term and multiplying, we get

$$A = \begin{bmatrix} 1 & 0 & 0 \\ -a & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -b & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -c & 1 \end{bmatrix} U$$

and since

$$\begin{bmatrix} 1 & 0 & 0 \\ -a & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -b & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & -c & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ -a & 1 & 0 \\ -b & -c & 1 \end{bmatrix} = L,$$

we have $A = LU$. (Check the product of the elementary matrices in the reverse order; the behavior is not as nice.)

If there are zeros in the pivot positions, the necessary permutation of rows can be done first (a fact that is not quite obvious); and the full result is that *there is a permutation matrix P such that PA has an L-U factorization*. We do not use this general version here.

Depending on the type of linear system desired, the upper triangular matrix U may be nonsquare and it may include rows of zeros. The number of row operations may be reduced, to save solution time, by keeping the number of nonzero entries in L small.

Example.

$$LU = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 0 & 3 & 1 \end{bmatrix}\begin{bmatrix} 1 & 2 & 3 & 0 \\ 0 & 0 & 3 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 2 & 3 & 0 \\ 2 & 4 & 9 & 1 \\ 0 & 0 & 9 & 3 \end{bmatrix} = A$$

and

$$A = \begin{bmatrix} 1 & 2 & 3 & 0 \\ 2 & 4 & 9 & 1 \\ 0 & 0 & 9 & 3 \end{bmatrix} \to \begin{bmatrix} 1 & 2 & 3 & 0 \\ 0 & 0 & 3 & 1 \\ 0 & 0 & 9 & 3 \end{bmatrix} \to \begin{bmatrix} 1 & 2 & 3 & 0 \\ 0 & 0 & 3 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} = U.$$

We can also see that to keep the entries in the final matrix A from getting too large, the matrices L and U should not have all positive (or all negative) entries. A regular alternating pattern of positive and negative entries may be counterproductive. A random sprinkling of a few minus signs seems to work well. By thinking of the entries of A as inner products of the rows of L and the columns of U, we can spot and correct difficulties. If an entry of A does turn out to be large, change an entry in the corresponding row of L or column of U, and multiply again.

If the matrix U is square we can continue the elimination process to get the inverse of A. There is, yet, no guarantee that the additional operations will be integral. However we can ensure that the entries of A^{-1} are also integers by the usual method of making the determinant equal to ± 1, since

$$A^{-1} = \frac{1}{\det A}\text{adjoint } A = \frac{1}{\det A}\left[\text{cofactor } a_{ij}\right]^t$$

and the cofactors are also determinants that do not involve division. Since U and

A have the same determinant, we can make the determinant of A equal to ± 1 by having each entry on the diagonal of U equal to 1 or -1. There will now be no division needed to complete the elimination as the diagonal entries of U are ± 1, so we get:

The Method for Inverses. The remaining row operations to obtain the inverse will also be in integers if we change step 1 of the method to:

1′) *Choose the matrix U to be upper triangular with integer entries, and ± 1 entries on the diagonal.*

Example. Let

$$L = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 0 & 3 & 1 \end{bmatrix} \quad \text{and} \quad U = \begin{bmatrix} 1 & 2 & 0 \\ 0 & 1 & 3 \\ 0 & 0 & -1 \end{bmatrix}.$$

Then we get

$$A = \begin{bmatrix} 1 & 2 & 0 \\ 2 & 5 & 3 \\ 0 & 3 & 8 \end{bmatrix} \quad \text{and} \quad A^{-1} = \begin{bmatrix} -31 & 16 & -6 \\ 16 & -8 & 3 \\ -6 & 3 & -1 \end{bmatrix}$$

and all steps involve only integer arithmetic.

We can generalize this to a matrix with any integer-valued determinant. Let the diagonal elements of U be any integer values. Their product will be the determinant. By ensuring that each column contains values above the diagonal that are multiples of the diagonal in that column, no fractions will be necessary. One way of doing this is to multiply U on the *right* by a diagonal matrix with integer entries. If we allow only one or two of the leftmost columns to violate this condition, we will get fractions; but the few remaining calculations will be manageable at this late stage.

This method generates simple examples that can illustrate the Gaussian elimination algorithm or test student ability to perform it. Not only does it allow us to design the form of the solution, as usual, but we can also control the complexity of the solution process, either by reducing the numbers of steps required or allowing some non-integer arithmetic. Modern computer software, e.g. MATLAB [**12**], as well as some hand calculators, will easily solve linear systems and compute inverses of matrices; and we can later use these tools to examine the numerical properties of the algorithm and to treat more complex problems and applications.

References

1. T. M. Cronin, The construction of matrices with required properties over the integers, *American Mathematical Monthly*, 94 (1987) 656–662.
2. W. P. Galvin, Matrices with "custom-built" eigenspaces, *American Mathematical Monthly*, 91 (1984) 308–309.
3. Richard C. Gilbert, Companion matrices with integer entires and integer eigenvalues and eigenvectors, *American Mathematical Monthly*, 95 (1988) 947–950.
4. Robert Hanson, Integer matrices whose inverses contain only integers, *Two-Year College Mathematics Journal*, 13 (1982) 18–21.
5. _____, Self-inverse matrices, *College Mathematics Journal*, 16 (1985) 190–198.

6. Konrad J. Heuvers, Symmetric matrices with prescribed eigenvalues and eigenvectors, *Mathematics Magazine*, 55 (1982) 106–111.
7. John G. Kemeny, How computers have changed the way I teach, *Academic Computing*, 2 (1988) 44–46, 59–61.
8. J. M. Ortega, Letter to the editor, *American Mathematical Monthly*, 92 (1985) 526.
9. J. C. Renaud, Matrices with integer entries and integer eigenvalues, *American Mathematical Monthly*, 90 (1983) 202–203.
10. Gilbert Strang, *Linear Algebra and Its Applications*, third ed., Academic Press, New York, 1988.
11. Alan Tucker, *A Unified Introduction to Linear Algebra: Methods, Models and Theory*, Macmillan, New York, 1988.
12. MATLAB™ software by The MathWorks, Inc., Natick, MA.

———o———

Rotation Matrices in the Plane without Trigonometry

Arnold J. Insel, Illinois State University, Normal, IL 61761

For a fixed angle θ, the rotation matrix

$$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$

induces a rotation of the plane by the angle θ about the origin. That is, for any column vector (2×1 matrix) $\mathbf{v}$ in $\mathbb{R}^2$, the product of this matrix with $\mathbf{v}$ is the rotation of $\mathbf{v}$ by the angle θ about the origin. Ordinarily, this rotation property is established by a trigonometric argument that relies on the angle addition formulas for the sine and cosine functions. For example see [C. H. Edwards, Jr. and D. E. Penney, *Calculus and Analytic Geometry*, 3rd ed., Prentice Hall, Englewood Cliffs, NJ, 1990, p. 491]. In this capsule we reverse the procedure. We provide a simple and direct justification of the rotation property of rotation matrices that is virtually free of trigonometric arguments, and we then use our results to derive these trigonometric identities.

The rotation operator. We begin with the mapping on $\mathbb{R}^2$ that rotates a vector by a fixed angle θ about the origin so that the rotation is counter-clockwise if θ is positive, and the origin is fixed under the rotation. Let us denote this mapping by T_θ. We argue that T_θ is linear, and then show that its matrix representation with respect to the standard ordered basis for $\mathbb{R}^2$ is the rotation matrix given above. This establishes the rotation property for this matrix. Finally, we use the rotation property to derive the desired trigonometric identities.

In what follows we list certain basic geometric properties about T_θ that are inherent in the idea of rotation:

1. The rotation of a vector by the angle α followed by the rotation by the angle β is equivalent to the rotation of the vector by $\alpha + \beta$. Symbolically: $T_\beta T_\alpha = T_{\alpha+\beta} = T_{\beta+\alpha} = T_\alpha T_\beta$.
2. The angle of intersection of two vectors is preserved under any rotation.
3. The length of a vector is preserved under any rotation.

To show that T_θ is linear, consider any two nonzero vectors $\mathbf{v}$ and $\mathbf{w}$ in $\mathbb{R}^2$. We may think of $\mathbf{v}$ and $\mathbf{w}$ as adjacent sides of a parallelogram intersecting at the origin. Then $\mathbf{v} + \mathbf{w}$ is the diagonal of this parallelogram with the origin as an endpoint. The rotated vectors $T_\theta(\mathbf{v})$ and $T_\theta(\mathbf{w})$ are now adjacent sides of a new (rotated) parallelogram. See the figures below. One can now argue that the rotated diagonal,