GAUSSIAN ELIMINATION - REVISITED

Consider solving the linear system

$$\begin{aligned} 2x_1 + x_2 - x_3 + 2x_4 &= 5 \\ 4x_1 + 5x_2 - 3x_3 + 6x_4 &= 9 \\ -2x_1 + 5x_2 - 2x_3 + 6x_4 &= 4 \\ 4x_1 + 11x_2 - 4x_3 + 8x_4 &= 2 \end{aligned}$$

by Gaussian elimination **without** pivoting. We denote this linear system by $Ax = b$. The augmented matrix for this system is

$$[A \mid b] = \left[\begin{array}{rrrr|r} 2 & 1 & -1 & 2 & 5 \\ 4 & 5 & -3 & 6 & 9 \\ -2 & 5 & -2 & 6 & 4 \\ 4 & 11 & -4 & 8 & 2 \end{array} \right]$$

To eliminate x_1 from equations 2, 3, and 4, use multipliers

$$m_{2,1} = 2, \quad m_{3,1} = -1, \quad m_{4,1} = 2$$

To eliminate x_1 from equations 2, 3, and 4, use multipliers

$$m_{2,1} = 2, \quad m_{3,1} = -1, \quad m_{4,1} = 2$$

This will introduce zeros into the positions below the diagonal in column 1, yielding

$$\left[\begin{array}{cccc|c} 2 & 1 & -1 & 2 & 5 \\ 0 & 3 & -1 & 2 & -1 \\ 0 & 6 & -3 & 8 & 9 \\ 0 & 9 & -2 & 4 & -8 \end{array}\right]$$

To eliminate x_2 from equations 3 and 4, use multipliers

$$m_{3,2} = 2, \quad m_{4,2} = 3$$

This reduces the augmented matrix to

$$\left[\begin{array}{cccr|r} 2 & 1 & -1 & 2 & 5 \\ 0 & 3 & -1 & 2 & -1 \\ 0 & 0 & -1 & 4 & 11 \\ 0 & 0 & 1 & -2 & -5 \end{array}\right]$$

To eliminate x_3 from equation 4, use the multiplier

$$m_{4,3} = -1$$

This reduces the augmented matrix to

$$\left[\begin{array}{cccc|c} 2 & 1 & -1 & 2 & 5 \\ 0 & 3 & -1 & 2 & -1 \\ 0 & 0 & -1 & 4 & 11 \\ 0 & 0 & 0 & 2 & 6 \end{array}\right]$$

Return this to the familiar linear system

$$\begin{aligned} 2x_1 + x_2 - x_3 + 2x_4 &= 5 \\ 3x_2 - x_3 + 2x_4 &= -1 \\ -x_3 + 4x_4 &= 11 \\ 2x_4 &= 6 \end{aligned}$$

Solving by back substitution, we obtain

$$x_4 = 3, \quad x_3 = 1, \quad x_2 = -2, \quad x_1 = 1$$

There is a surprising result involving matrices associated with this elimination process. Introduce the upper triangular matrix

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

which resulted from the elimination process. Then introduce the lower triangular matrix

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 \\ m_{2,1} & 1 & 0 & 0 \\ m_{3,1} & m_{3,2} & 1 & 0 \\ m_{4,1} & m_{4,2} & m_{4,3} & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ -1 & 2 & 1 & 0 \\ 2 & 3 & -1 & 1 \end{bmatrix}$$

This uses the multipliers introduced in the elimination process. Then

$$A = LU$$

$$\begin{bmatrix} 2 & 1 & -1 & 2 \\ 4 & 5 & -3 & 6 \\ -2 & 5 & -2 & 6 \\ 4 & 11 & -4 & 8 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 2 & 1 & 0 & 0 \\ -1 & 2 & 1 & 0 \\ 2 & 3 & -1 & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & -1 & 2 \\ 0 & 3 & -1 & 2 \\ 0 & 0 & -1 & 4 \\ 0 & 0 & 0 & 2 \end{bmatrix}$$

In general, when the process of Gaussian elimination without pivoting is applied to solving a linear system $Ax = b$, we obtain $A = LU$ with L and U constructed as above.

For the case in which partial pivoting is used, we obtain the slightly modified result

$$LU = PA$$

where L and U are constructed as before and P is a *permutation matrix*. For example, consider

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

Then

$$PA = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \\ a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} \\ a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4} \end{bmatrix} = \begin{bmatrix} A_{3,*} \\ A_{1,*} \\ A_{4,*} \\ A_{2,*} \end{bmatrix}$$

$$
\begin{aligned}
PA &= \begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \\ a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} \\ a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4} \end{bmatrix} \\
&= \begin{bmatrix} A_{3,*} \\ A_{1,*} \\ A_{4,*} \\ A_{2,*} \end{bmatrix}
\end{aligned}
$$

The matrix PA is obtained from A by switching around rows of A. The result $LU = PA$ means that the LU-factorization is valid for the matrix A with its rows suitably permuted.

Consequences: If we have a factorization

$$A = LU$$

with L lower triangular and U upper triangular, then we can solve the linear system $Ax = b$ in a relatively straightforward way.

The linear system can be written as

$$LUx = b$$

Write this as a two stage process:

$$Lg = b, \qquad Ux = g$$

The system $Lg = b$ is a lower triangular system

$$\begin{array}{lcl} g_1 & = & b_1 \\ \ell_{2,1}g_1 + g_2 & = & b_2 \\ \ell_{3,1}g_1 + \ell_{3,2}g_2 + g_3 & = & b_3 \\ & \vdots & \\ \ell_{n,1}g_1 + \cdots \ell_{n,n-1}g_{n-1} + g_n & = & b_n \end{array}$$

We solve it by "forward substitution". Then we solve the upper triangular system $Ux = g$ by back substitution.

VARIANTS OF GAUSSIAN ELIMINATION

If no partial pivoting is needed, then we can look for a factorization

$$A = LU$$

without going thru the Gaussian elimination process.

For example, suppose A is 4×4. We write

$$\begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \\ a_{3,1} & a_{3,2} & a_{3,3} & a_{3,4} \\ a_{4,1} & a_{4,2} & a_{4,3} & a_{4,4} \end{bmatrix}$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ \ell_{2,1} & 1 & 0 & 0 \\ \ell_{3,1} & \ell_{3,2} & 1 & 0 \\ \ell_{4,1} & \ell_{4,2} & \ell_{4,3} & 1 \end{bmatrix} \begin{bmatrix} u_{1,1} & u_{1,2} & u_{1,3} & u_{1,4} \\ 0 & u_{2,2} & u_{2,3} & u_{2,4} \\ 0 & 0 & u_{3,3} & u_{3,4} \\ 0 & 0 & 0 & u_{4,4} \end{bmatrix}$$

To find the elements $\left\{\ell_{i,j}\right\}$ and $\left\{u_{i,j}\right\}$, we multiply the right side matrices L and U and match the results with the corresponding elements in A.

Multiplying the first row of L times all of the columns of U leads to

$$u_{1,j} = a_{1,j}, \quad j = 1, 2, 3, 4$$

Then multiplying rows 2, 3, 4 times the first column of U yields

$$\ell_{i,1} u_{1,1} = a_{i,1}, \quad i = 2, 3, 4$$

and we can solve for $\left\{\ell_{2,1}, \ell_{3,1}, \ell_{4,1}\right\}$. We can continue this process, finding the second row of U and then the second column of L, and so on. For example, to solve for $\ell_{4,3}$, we need to solve for it in

$$\ell_{4,1} u_{1,3} + \ell_{4,2} u_{2,3} + \ell_{4,3} u_{3,3} = a_{4,3}$$

Why do this? A hint of an answer is given by this last equation. If we had an $n \times n$ matrix A, then we would find $\ell_{n,n-1}$ by solving for it in the equation

$$\ell_{n,1} u_{1,n-1} + \ell_{n,2} u_{2,n-1} + \cdots + \ell_{n,n-1} u_{n-1,n-1} = a_{n,n-1}$$

$$\ell_{n,n-1} = \frac{a_{n,n-1} - \left[\ell_{n,1} u_{1,n-1} + \cdots + \ell_{n,n-2} u_{n-2,n-1}\right]}{u_{n-1,n-1}}$$

Embedded in this formula we have a dot product. This is in fact typical of this process, with the length of the inner products varying from one position to another.

Recalling §2.4 and the discussion of dot products, we can evaluate this last formula by using a higher precision arithmetic and thus avoid many rounding errors. This leads to a variant of Gaussian elimination in which there are far fewer rounding errors.

With ordinary Gaussian elimination, the number of rounding errors is proportional to n^3. This reduces the number of rounding errors, with the number now being proportional to only n^2. This can lead to major increases in accuracy, especially for matrices A which are very sensitive to small changes.

TRIDIAGONAL MATRICES

$$A=\left[\begin{array}{cccccc} b_1 & c_1 & 0 & 0 & \cdots & 0 \\ a_2 & b_2 & c_2 & 0 & & \\ 0 & a_3 & b_3 & c_3 & & \vdots \\ & & & \ddots & & \\ \vdots & & & a_{n-1} & b_{n-1} & c_{n-1} \\ 0 & & \cdots & & a_n & b_n \end{array}\right]$$

These occur very commonly in the numerical solution of partial differential equations, as well as in other applications (e.g. computing interpolating cubic spline functions).

We factor $A = LU$, as before. But now L and U take very simple forms. Before proceeding, we note with an example that the same may not be true of the matrix inverse.

EXAMPLE

Define an $n \times n$ tridiagonal matrix

$$A = \begin{bmatrix} -1 & 1 & 0 & 0 & \cdots & 0 \\ 1 & -2 & 1 & 0 & & \\ 0 & 1 & -2 & 1 & & \vdots \\ & & & \ddots & & \\ \vdots & & & 1 & -2 & 1 \\ 0 & & \cdots & & 1 & -\frac{n-1}{n} \end{bmatrix}$$

Then A^{-1} is given by

$$\left(A^{-1}\right)_{i,j} = \max\{i, j\}$$

Thus the sparse matrix A can (and usually does) have a dense inverse.

We factor $A = LU$, with

$$L = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 \\ \alpha_2 & 1 & 0 & 0 & & \\ 0 & \alpha_3 & 1 & 0 & & \vdots \\ & & & \ddots & & \\ \vdots & & & \alpha_{n-1} & 1 & 0 \\ 0 & & \cdots & & \alpha_n & 1 \end{bmatrix}$$

$$U = \begin{bmatrix} \beta_1 & c_1 & 0 & 0 & \cdots & 0 \\ 0 & \beta_2 & c_2 & 0 & & \\ 0 & 0 & \beta_3 & c_3 & & \vdots \\ & & & \ddots & & \\ \vdots & & & 0 & \beta_{n-1} & c_{n-1} \\ 0 & & \cdots & & 0 & \beta_n \end{bmatrix}$$

Multiply these and match coefficients with A to find $\{\alpha_i, \gamma_i\}$.

By doing a few multiplications of rows of L times columns of U, we obtain the general pattern as follows.

$$\begin{array}{cl} \beta_1 = b_1 & : \text{row 1 of } LU \\ \alpha_2\beta_1 = a_2, \quad \alpha_2 c_1 + \beta_2 = b_2 & : \text{row 2 of } LU \\ \vdots & \\ \alpha_n\beta_{n-1} = a_n, \quad \alpha_n c_{n-1} + \beta_n = b_n & : \text{row } n \text{ of } LU \end{array}$$

These are straightforward to solve.

$$\beta_1 = b_1$$

$$\alpha_j = \frac{a_j}{\beta_{j-1}}, \quad \beta_j = b_j - \alpha_j c_{j-1}, \quad j = 2, ..., n$$

To solve the linear system

$$Ax = f$$

or

$$LUx = f$$

instead solve the two triangular systems

$$Lg = f, \qquad Ux = g$$

Solving $Lg = f$:

$$\begin{aligned} g_1 &= f_1 \\ g_j &= f_j - \alpha_j g_{j-1}, \quad j = 2, ..., n \end{aligned}$$

Solving $Ux = g$:

$$\begin{aligned} x_n &= \frac{g_n}{\beta_n} \\ x_j &= \frac{g_j - c_j x_{j+1}}{\beta_j}, \quad j = n-1, ..., 1 \end{aligned}$$

See the numerical example on page 278.

OPERATIONS COUNT

Factoring $A = LU$.

Additions:	$n-1$
Multiplications:	$n-1$
Divisions:	$n-1$

Solving $Lz = f$ and $Ux = z$:

Additions:	$2n-2$
Multiplications:	$2n-2$
Divisions:	n

Thus the total number of arithmetic operations is approximately $3n$ to factor A; and it takes about $5n$ to solve the linear system using the factorization of A.

If we had A^{-1} at no cost, what would it cost to compute $x = A^{-1}f$?

$$x_i = \sum_{j=1}^{n} \left(A^{-1}\right)_{i,j} f_j, \quad i = 1, ..., n$$

MATLAB MATRIX OPERATIONS

To obtain the *LU*-factorization of a matrix, including the use of partial pivoting, use the Matlab command *lu*. In particular,

$$[L,\ U,\ P] = lu(X)$$

returns the lower triangular matrix L, upper triangular matrix U, and permutation matrix P so that

$$PX = LU$$