
Solving linear systems: row pivoting

We have seen how Gaussian elimination makes use of the row operation

"replace a row by itself minus a multiple of another row"

to reduce the coefficient matrix to upper triangular form. Another useful row operation is

"interchange two rows".

Performing this operation on a matrix A is equivalent to multiplying A on the left by a *permutation matrix* P, where P is obtained by interchanging the corresponding rows of the identity matrix. Recall that an identity matrix is a matrix with ones on its main diagonal and zeros everywhere else. For instance, check that in the first example we looked at, $A\mathbf{x} = \mathbf{b}$ where

$$A = \begin{pmatrix} 4 & 0 & -8 \\ 1 & 2 & 3 \\ -2 & 4 & -1 \end{pmatrix}, \qquad \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}, \qquad \mathbf{b} = \begin{pmatrix} 12 \\ 2 \\ 7 \end{pmatrix},$$

multiplying both A and $\mathbf{b}$ (or the augmented matrix $\big(A \mid \mathbf{b}\big)$) on the left by

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

results in a system with the first and second equations switched.

When row pivoting is necessary

Consider the coefficient matrix

$$A = \begin{pmatrix} 1 & 2 & 3 & -2 \\ 2 & 4 & 1 & 0 \\ 3 & 3 & 2 & 5 \\ -1 & 6 & 2 & 1 \end{pmatrix}$$

What happens after you've done Gaussian elimination on the first column?

Two common pivoting strategies:

- *First non-zero pivot strategy*: If, when trying to do Gaussian elimination on the jth column, the (j, j) diagonal entry is zero, then search the entries below (i.e. those with row number $> j$) for the first non-zero one, and interchange that row with row j.

- *Maximal pivot strategy*, also called *partial pivoting*: Before doing Gaussian elimination on the jth column, search all entries in that column on and below the diagonal (i.e. with row number $\geq j$) for the one of greatest magnitude, and use that entry as the pivot, i.e. interchange that row with row j (if needed).

Using row pivoting to reduce round-off errors

Consider the system $A\mathbf{x} = \mathbf{b}$ where

$$A = \begin{pmatrix} 10 & -7 & 0 \\ -3 & 2.09999 & 6 \\ 5 & -1 & 5 \end{pmatrix}, \qquad \mathbf{b} = \begin{pmatrix} 7 \\ 3.90001 \\ 6 \end{pmatrix}.$$

You can verify that the exact solution is $x_1 = 0,\ x_2 = -1,\ x_3 = 1$.

Now perform step-by-step Gaussian elimination or LU factorization and see what you get as the solution.

Partial (i.e. maximal entry) pivoting aims to avoid division by small numbers and hence reduce the possibility of round-off errors being magnified.

Matlab's built-in LU factorization command "`lu`" automatically employs the partial pivoting strategy:

```
[L,U,P]=lu(A)
```

produces a lower triangular matrix L, an upper triangular matrix U, and a (combined) permutation matrix P such that

$$PA = LU$$

Then, given any right-hand side vector $\mathbf{b}$, we can solve the system $A\mathbf{x} = \mathbf{b}$ as follows:

- Since $PA = LU$, multiplying both sides of $A\mathbf{x} = \mathbf{b}$ on the left by P gives

  $$LU\mathbf{x} = P\mathbf{b}$$

- Let $U\mathbf{x} = \mathbf{y}$; then the equation becomes

  $$L\mathbf{y} = P\mathbf{b}$$

  Since L is lower triangular, this system can be solved by *forward substitution.*

- Once $\mathbf{y}$ has been found, solve

  $$U\mathbf{x} = \mathbf{y}$$

  by backward substitution. This yields the solution $\mathbf{x}$ to the original system $A\mathbf{x} = \mathbf{b}$.

Matlab's backslash operator for solving linear systems basically also relies on LU factorization with partial pivoting:

```
A\b
```

attempts to find the solution of $A\mathbf{x} = \mathbf{b}$ by first checking whether A is some special type of matrix (triangular, symmetric, Hermitian positive definite, Hessenberg, sparse); if so, then an appropriately tailored algorithm is used; if not, then LU factorization with partial pivoting followed by forward and backward subsitution is applied.