Pivoting (§3.6)

We saw that in order to avoid dividing by zero (or, near-zero) values in the LU decomposition, we may need to resort to partial or full pivoting.

- Partial pivoting permutes rows, such that the pivot element in the k-th iteration is the largest number in the $(n-k+1)$ lower entries of the k-th column
  It is written, in the context of LU decomposition, as

$$PA = LU \qquad (P = \text{permutation})$$

- Full pivoting selects the pivot element in the k-th iteration as the largest element of the $(n-k+1)\times(n-k+1)$ lower-rightmost submatrix of A. It operates by permuting rows AND columns and leads to a LU decomposition of.

$$PAQ = LU.$$

However, there are certain categories of matrices for which we can safely use Gauss elimination or LU decomposition without the need for pivoting (i.e. the pivot elements will never be problematically small).

Def A matrix A is called diagonally dominant by columns if the magnitude of every diagonal element is larger than the sum of magnitudes of all other entries in the same column, i.e.

$$\text{For every } i=1,2,\dots,n \qquad |a_{ii}| > \sum_{j\neq i} |a_{ji}|$$

If the diagonal element exceeds in magnitude the sum of magnitudes of all other elements in its row, i.e.

$$\text{for } i=1,2,\dots,n \qquad |a_{ii}| > \sum_{j\neq i} |a_{ij}|$$

then the matrix is called diagonally dominant by rows.

Def

A symmetric matrix $A \in \mathbb{R}^{n\times n}$ is called positive definite (in short: SPD: for "symmetric positive definite"), if for any $\underline{x}\in\mathbb{R}^n$, $\underline{x}\neq 0$

we have $\underline{x}^T A \underline{x} > 0$

If for any $\underline{x}\in\mathbb{R}$ $\underline{x}\neq 0$ we have $\underline{x}^T A \underline{x} \geq 0$, the matrix is called positive semi-definite.

If the respective properties are $x^T A x < 0$ (or $x^T A x \leq 0$) the [illegible] (semi-) definite

Def the k-th *leading principal minor* of a matrix $A \in \mathbb{R}^{n\times n}$ is the determinant of the top-leftmost $k\times k$ submatrix of A. Thus if we denote this minor by M_k:

$$M_1 = |a_{11}| \qquad M_2 = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} \quad \cdots \quad M_k = \begin{vmatrix} a_{11} & \cdots & a_{1k} \\ \vdots & & \vdots \\ a_{k1} & \cdots & a_{kk} \end{vmatrix}$$

Thm If *all* leading principal minors (i.e. for $k=1,2,3,\ldots,n$) of the symmetric matrix A are positive, then A is positive definite

* If $M_k < 0$ for k = odd and $M_k > 0$ for k = even, then A is negative definite:

Thm Pivoting is *not* necessary when A is diagonally dominant by columns, or symmetric and positive- (or negative-) definite.

These "special" classes of matrices (which appear quite often in engineering and applied sciences) not only make LU decomposition more robust, but also open some additional possibilities for solving $A\underline{x} = \underline{b}$.

Iterative methods for linear systems.

The general idea is similar to the philosophy of iterative methods we saw for nonlinear equations, i.e. we proceed as follows:

→ We write a (matrix) equation

$$\underline{x} = T\underline{x} + \underline{c}$$

in such a way that this equation is equivalent to $Ax = b$.

→ We start with an initial guess $\underline{x}^{(0)}$ for the solution of $Ax = b$

→ We iterate

$$\underline{x}^{(k+1)} = T\underline{x}^{(k)} + \underline{c}$$

→ If properly designed the sequence $x^{(0)}, x^{(1)}, \ldots, x^{(k)}, \ldots$ converges to $\underline{x}^*$, which satisfies $\underline{x}^* = T\underline{x}^* + c$ and subsequently $Ax^* = b$.

General design approach

We decompose $A = D - L - U$

- D: diagonal
- L: lower triangular
- U: upper triangular

e.g.

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$$

$$D = \begin{bmatrix} a_{11} & & & \\ & a_{22} & & \\ & & \ddots & \\ & & & a_{nn} \end{bmatrix} \qquad L = \begin{bmatrix} 0 & & & \\ -a_{21} & 0 & & \\ -a_{31} & -a_{32} & \ddots & \\ -a_{n1} & -a_{n2} & \cdots & -a_{n,n-1} \; 0 \end{bmatrix}$$

$$U = \begin{bmatrix} 0 & -a_{12} & -a_{13} & \cdots & -a_{1n} \\ & \ddots & -a_{23} & & -a_{2n} \\ & & \ddots & & \\ & & & \ddots & -a_{n-1,n} \\ & & & & 0 \end{bmatrix}$$

The Jacobi method

$$A = D - L - U$$

$$Ax = b$$

$$(D - L - U)x = b$$

$$Dx = (L+U)x + b$$

$$x = \underbrace{D^{-1}(L+U)}_{T}x + \underbrace{D^{-1}b}_{c} \qquad (x = Tx + c)$$

Iteration:

$$\underline{x}^{(k+1)} = D^{-1}(L+U)\,\underline{x}^{(k)} + D^{-1}\underline{b} \qquad \underline{\text{or}}$$

$$\boxed{D\,\underline{x}^{(k+1)} = (L+U)\,\underline{x}^{(k)} + \underline{b}}$$

The Jacobi iteration formula.

→ Solution: easy, since we need to solve a linear system of equations with diagonal coefficient matrix

$$\begin{bmatrix} d_1 & & & \\ & d_2 & & \\ & & \ddots & \\ & & & d_n \end{bmatrix} \begin{bmatrix} x_1^{(k+1)} \\ \vdots \\ x_n^{(k+1)} \end{bmatrix} = \begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix} \quad \rightarrow \quad x_i^{(k+1)} = \frac{1}{d_i} c_i$$

→ Convergence: The Jacobi method is guarranteed to converge when A is diagonally dominant by rows

→ Complexity: Each iteration has a cost associated with

→ Solving $Dx^{(k+1)} = c$ ⇒ n divisions

→ Computing $c = (L+U)x^{(k)} + b$ ⇒ as many additions and multiplications as nonzero entries in A

worst case $= O(n^2)$, but could be $O(n)$ for sparse matrices

→ Stopping criterion: $\| b - Ax^{(k)} \| < \varepsilon$

or $\| x^{(k+1)} - x^{(k)} \| < \varepsilon$.

There are 3 forms we will see this algorithm, for different purposes:

(i) Matrix form (for proofs) $Dx^{(k+1)} = (L+U)x^{(k)} + b$

(ii) Algorithm form (without in-place update)

Each row of $Dx^{(k+1)} = (L+U)x^{(k)} + b$:

$$a_{ii} x_i^{(k+1)} = b_i - \sum_{j \neq i} a_{ij} x_j^{(k)}$$

$$\Rightarrow x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j \neq i} a_{ij} x_j^{(k)} \right)$$

for $k = 1, 2, \ldots, \langle max \rangle$

  for $i = 1, 2, \ldots, n$

    $x_i^{(k+1)} \leftarrow \frac{1}{a_{ii}} \left(b_i - \sum_{j \neq i} a_{ij} x_j^{(k)} \right)$

  end

  <check for convergence

end

iii) In-place algorithm

(replaces $\underline{x}$ with a better estimate)

$\underline{x} \leftarrow$ initial guess

for $k = 1, 2, \ldots, <\text{max}>$

  for $i = 1, 2, \ldots, n$

$$x_i^{new} \leftarrow \frac{1}{a_{ii}} \left(b_i - \sum_{j \neq i} a_{ij} x_j \right)$$

  end

  $\underline{x} \leftarrow \underline{x}^{new}$

  check for convergence

end