Math 551: Scientific Programming

Lecture 7: Backward substitution and Gaussian Elimination

Weiqi Chu

Department of Mathematics and Statistics, UMass Amherst

Announcements

- Office hours: Wednesday 11-12, Thursday 1-2
- Office LGRT 1529

Today: Section 2.1

- How to solve linear equations when A is in upper triangular form? The algorithm is called **backward substitution**.
- How to transform general linear equations into an upper triangular form? The algorithm is called **Gaussian elimination**.

Linear Algebra Review: Solving a system of linear equations

- How do you solve $Ax = b$ by hand?
- For example,

$$A = \begin{pmatrix} 1 & 3 & -1 \\ 4 & -1 & 0 \\ 2 & 9 & 3 \end{pmatrix}, \qquad b = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$$

Backward Substitution

Let A be an upper triangular matrix and b be a vector

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ & a_{22} & \ddots & a_{2n} \\ & & \ddots & \vdots \\ & & & a_{nn} \end{pmatrix}, \quad b = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$

All elements below the main diagonal are 0 ($a_{ij} = 0$ for all $i > j$).

Backward substitution: Solve unknowns in the order of $x_n, x_{n-1}, \ldots, x_1$

$$x_k = \frac{b_k - \sum_{j=k+1}^{n} a_{kj} x_j}{a_{kk}}, \quad k = n, n-1, \ldots, 1.$$

Backward Substitution

- What is the cost of this algorithm?
- In a simplistic way, we just count each floating point operation as a *flop*
  - a flop is an addition, subtraction, multiplication, or division operation
- The number of flops required in backward substitution is

$$\sum_{k=1}^{n-1} (n-k) + (n-k) + 1 = n^2 \,.$$

Forward Substitution

- A similar situation is when A is a lower triangular matrix

- We use **forward substitution** to solve the system in the order of $x_1, x_2, \ldots, x_n$

$$x_k = \frac{b_k - \sum_{j=1}^{k-1} a_{kj} x_j}{a_{kk}}, \quad k = 1, 2, \ldots, n.$$

Gaussian elimination

Gaussian elimination, also known as *row reduction*, is an algorithm to transform the general system of linear equations to an upper triangular matrix.

$$(A|\mathbf{b}) = \left(\begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} & b_n \end{array}\right)$$

$$(A^{(1)}|\mathbf{b}^{(1)}) = \left(\begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ 0 & a_{22}^{(1)} & \cdots & a_{2n}^{(1)} & b_2^{(1)} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & a_{n2}^{(1)} & \cdots & a_{nn}^{(1)} & b_n^{(1)} \end{array}\right)$$

Gaussian elimination

$$(A^{(2)}|\mathbf{b}^{(2)}) = \left(\begin{array}{ccccc|c} a_{11} & a_{12} & \cdots & \cdots & a_{1n} & b_1 \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & \cdots & a_{2n}^{(1)} & b_2^{(1)} \\ \vdots & 0 & a_{33}^{(2)} & \cdots & a_{3n}^{(2)} & b_3^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & a_{n3}^{(2)} & \cdots & a_{nn}^{(2)} & b_n^{(2)} \end{array}\right)$$

$$(A^{(n-1)}|\mathbf{b}^{(n-1)}) = \left(\begin{array}{ccccc|c} a_{11} & a_{12} & \cdots & \cdots & a_{1n} & b_1 \\ 0 & a_{22}^{(1)} & a_{23}^{(1)} & \cdots & a_{2n}^{(1)} & b_2^{(1)} \\ \vdots & 0 & a_{33}^{(2)} & \cdots & a_{3n}^{(2)} & b_3^{(2)} \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & a_{nn}^{(n-1)} & b_n^{(n-1)} \end{array}\right)$$

Computational cost

- What is the computational cost of Gaussian elimination?
  - We count the number of flops:

$$\sum_{k=1}^{n-1}(n-k)+2(n-k)(n-k+1) \approx \frac{2}{3}n^3 + O(n^2)$$

Reordering rows

- What happens if $a_{kk}^{(k-1)} = 0$ at some point?

$$\left(\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 1 & 1 & 2 & 2 \\ 1 & 2 & 2 & 1 \end{array}\right) \Rightarrow \left(\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{array}\right)$$

Reordering rows

- What happens if $a_{kk}^{(k-1)} = 0$ at some point?

$$\left(\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 1 & 1 & 2 & 2 \\ 1 & 2 & 2 & 1 \end{array}\right) \Rightarrow \left(\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{array}\right) \Rightarrow \left(\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{array}\right)$$

  - Adjust orders of rows to find a nonzero $a_{kk}^{(k-1)}$
  - A related strategy is called "pivoting": we pick the largest element of the column to be $a_{kk}^{(k-1)}$.

Reordering rows

- What happens if $a_{kk}^{(k-1)} = 0$ at some point?

$$\left(\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 1 & 1 & 2 & 2 \\ 1 & 2 & 2 & 1 \end{array}\right) \Rightarrow \left(\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{array}\right) \Rightarrow \left(\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{array}\right)$$

  - Adjust orders of rows to find a nonzero $a_{kk}^{(k-1)}$
  - A related strategy is called "pivoting": we pick the largest element of the column to be $a_{kk}^{(k-1)}$.

- What if we can't find a nonzero $a_{kk}^{(k-1)}$ by rearranging rows?

Reordering rows

- What happens if $a_{kk}^{(k-1)} = 0$ at some point?

$$\left(\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 1 & 1 & 2 & 2 \\ 1 & 2 & 2 & 1 \end{array}\right) \Rightarrow \left(\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{array}\right) \Rightarrow \left(\begin{array}{ccc|c} 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \end{array}\right)$$

  - Adjust orders of rows to find a nonzero $a_{kk}^{(k-1)}$

  - A related strategy is called "pivoting": we pick the largest element of the column to be $a_{kk}^{(k-1)}$.

- What if we can't find a nonzero $a_{kk}^{(k-1)}$ by rearranging rows?

  - Then this Gaussian elimination algorithm fails, but this only happens when the matrix A is singular.

Next time: LU decomposition

- Lower–upper (LU) decomposition factors a matrix into the product of a lower triangular matrix and an upper triangular matrix:

$$A = LU$$

  where L is a lower triangular matrix and U is an upper triangular matrix.

- LU decomposition can be viewed as the matrix form of Gaussian elimination.