

# Solving Linear Systems

## Gauss Elimination

CS/SE 4X03

Ned Nediakov

McMaster University

September 23, 2021

# Outline

- Linear systems

- Example

- Gauss elimination

  - Algorithm

  - Cost

- Backward substitution

  - Algorithm

  - Cost

- Total cost

# Linear systems

- Given an  $n \times n$  nonsingular matrix  $A$  and an  $n$ -vector  $b$  solve

$$Ax = b$$

The following are equivalent

- $A$  is nonsingular
- The determinant of  $A$  is nonzero,  $\det(A) \neq 0$
- Columns (rows) are linearly independent
- There exists  $A^{-1}$  such that  $A^{-1}A = AA^{-1} = I$ , where  $I$  is the  $n \times n$  identity matrix

## Linear systems cont.

- Dense system:  $A$  may have a small number of nonzeros
- Sparse system: most of the elements are zeros  
See Florida Sparse Matrix Collection
- Direct methods: based on Gauss elimination
- Iterative methods: for large  $A$

## Example

$$Ax = \begin{bmatrix} 1 & -1 & 3 \\ 1 & 1 & 0 \\ 3 & -2 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 11 \\ 3 \\ 3 \end{bmatrix} = b$$

Multiply first row by 1 and subtract from second row, multiply first row by 3 and subtract from third row

$$A|b = \left[ \begin{array}{ccc|c} 1 & -1 & 3 & 11 \\ 1 & 1 & 0 & 3 \\ 3 & -2 & 1 & 3 \end{array} \right] \begin{array}{cc} \times 1 & \times 3 \\ \downarrow & \\ & \downarrow \end{array}$$

$$A|b \leftarrow \left[ \begin{array}{ccc|c} 1 & -1 & 3 & 11 \\ 0 & 2 & -3 & -8 \\ 0 & 1 & -8 & -30 \end{array} \right]$$

## Example cont.

Multiply second row by  $\frac{1}{2}$  and subtract from third row

$$A|b \leftarrow \left[ \begin{array}{ccc|c} 1 & -1 & 3 & 11 \\ 0 & 2 & -3 & -8 \\ 0 & 1 & -8 & -30 \end{array} \right] \quad \begin{array}{c} \times \frac{1}{2} \\ \downarrow \end{array}$$

$$A|b \leftarrow \left[ \begin{array}{ccc|c} 1 & -1 & 3 & 11 \\ 0 & 2 & -3 & -8 \\ 0 & 0 & -6.5 & -26 \end{array} \right]$$

This is Gauss elimination, also called forward elimination

## Example cont.

$$\begin{bmatrix} 1 & -1 & 3 \\ 0 & 2 & -3 \\ 0 & 0 & -6.5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a_{22} & a_{23} \\ 0 & 0 & a_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \begin{bmatrix} 11 \\ -8 \\ -26 \end{bmatrix}$$

$$\begin{aligned} x_3 &= b_3/a_{33} &&= -26/(-6.5) &&= 4 \\ x_2 &= (b_2 - a_{23}x_3)/a_{22} &&= (-8 - (-3) \times 4)/2 &&= 2 \\ x_1 &= (b_1 - a_{12}x_2 - a_{13}x_3)/a_{11} &&= (11 - (-1) \times 2 - 3 \times 4)/1 &&= 1 \end{aligned}$$

This is called backward substitution

## Gauss elimination

## Algorithm

*Algorithm 3.1 (Gauss elimination).*

```

for  $k = 1 : n - 1$                                      % for each row
    for  $i = k + 1 : n$                                      % for each row below  $k$ th
         $m_{ik} = a_{ik} / a_{kk}$                                % multiplier
        % update row
        for  $j = k + 1 : n$ 
             $a_{ij} = a_{ij} - m_{ik} a_{kj}$ 
        end for
         $b_i = b_i - m_{ik} b_k$                                % update  $b_i$ 
    end for

```



## Gauss elimination cont.

### Cost

- We do not count the operations for updating  $b$
- The third nested **for** loop executes  $n - k$  times
  - $n - k$  multiplications
  - $n - k$  additions
- The work per one iteration of the second nested **for** loop is  $2(n - k) + 1$ , the 1 comes from the division
- This loop executes  $n - k$  times
- The total work for the second nested **for** loop is  $2(n - k)^2 + (n - k)$
- The work for the outermost **for** loop is

$$\sum_{k=1}^{n-1} [2(n - k)^2 + (n - k)] = 2 \sum_{k=1}^{n-1} k^2 + \sum_{k=1}^{n-1} k$$

## Gauss elimination cont.

## Cost

Since  $1^2 + 2^2 + 3^2 + \cdots + n^2 = n(n+1)(2n+1)/6$

$$\begin{aligned}\sum_{k=1}^{n-1} k^2 &= (n-1)(n-1+1)(2(n-1)+1)/6 \\ &= n^3/3 - n^2/2 + n/6\end{aligned}$$

Using the above and  $\sum_{k=1}^{n-1} k = \frac{(n-1)n}{2}$ ,

$$\begin{aligned}2 \sum_{k=1}^{n-1} k^2 + \sum_{k=1}^{n-1} k &= 2n^3/3 - 2n^2/2 + 2n/6 + n^2/2 - n/2 \\ &= 2n^3/3 - n^2/2 - n/6 = 2n^3/3 + O(n^2)\end{aligned}$$

Total work for Gauss elimination is  $2/3n^3 + O(n^2)$

# Backward substitution

- After GE, we have

$$\begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & \cdots & a_{1,n} \\ & a_{2,2} & a_{2,3} & \cdots & a_{2,n} \\ & & a_{3,3} & \cdots & a_{3,n} \\ & & & \ddots & \vdots \\ & & & & a_{n-1,n-1} & a_{n-1,n} \\ & & & & & a_{n,n} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_{n-1} \\ b_n \end{bmatrix}$$

- $x_n = b_n / a_{n,n}$
- $a_{n-1,n-1}x_{n-1} + a_{n-1,n}x_n = b_{n-1}$   
 $x_{n-1} = (b_{n-1} - a_{n-1,n}x_n) / a_{n-1,n-1}$
- $x_k = \left( b_k - \sum_{j=k+1}^n a_{k,j}x_j \right) / a_{k,k}$

# Backward substitution

## Algorithm

*Algorithm 4.1 (Backward substitution).*

**for**  $k = n : -1 : 1$

$$x_k = \left( b_k - \sum_{j=k+1}^n a_{k,j} x_j \right) / a_{k,k}$$

# Backward substitution

## Cost

- The work per iteration is
  - $n - k$  multiplications
  - $(n - k - 1) + 1$  additions
  - 1 division
  - total  $2(n - k) + 1$  operations
- Total work is

$$\begin{aligned}\sum_{k=1}^n (2(n - k) + 1) &= 2 \sum_{k=1}^n (n - k) + \sum_{k=1}^n 1 \\ &= 2 \sum_{k=1}^{n-1} k + n = 2 \frac{n(n-1)}{2} + n \\ &= n^2 - n + n = n^2\end{aligned}$$

## Total cost

- GE:  $2n^3/3 - n^2 + n/6$
- Backward substitution:  $n^2$
- Total cost for solving  $Ax = b$  is

$$2n^3/3 + n^2/2 + n/6 = 2n^3/3 + O(n^2) = O(n^3)$$