Today’s Agenda

- Review on Gaussian elimination
- Testing the pseudo-code
- Pivoting (motivation and basic idea)
- Error analysis

LU factorization

Gaussian elimination transforms a matrix into the product of a unit lower triangular matrix and an upper triangular matrix, i.e.,

$$A = LU$$

Lower Triangular

$$\begin{bmatrix} A00 & A01 & A02 \\ A10 & A11 & A12 \\ A20 & A21 & A22 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ L10 & 1 & 0 \\ L20 & L21 & 1 \end{bmatrix} \begin{bmatrix} U00 & U01 & U02 \\ 0 & U11 & U12 \\ 0 & 0 & U22 \end{bmatrix}$$

Upper Triangular

Forward elimination

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & & \vdots \\ a_{i1} & a_{i2} & a_{i3} & \cdots & a_{in} \\ \vdots & \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_i \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_i \\ \vdots \\ b_n \end{bmatrix}$$

$$k+1 \leq i \leq n \qquad \begin{cases} a_{ij} \leftarrow a_{ij} - \left(\dfrac{a_{ik}}{a_{kk}}\right) a_{kj} & (k \leqq j \leqq n) \\ b_i \leftarrow b_i - \left(\dfrac{a_{ik}}{a_{kk}}\right) b_k \end{cases}$$

Back substitution

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n = b_1 \\ a_{22}x_2 + a_{23}x_3 + \cdots + a_{2n}x_n = b_2 \\ a_{33}x_3 + \cdots + a_{3n}x_n = b_3 \\ \ddots \; \cdots \quad \cdots \quad \vdots \\ a_{n-1,n-1}x_{n-1} + a_{n-1,n}x_n = b_{n-1} \\ a_{nn}x_n = b_n \end{cases}$$

$$x_i = \left(b_i - \sum_{j=i+1}^{n} a_{ij}x_j \right) / a_{ii} \qquad (i = n-1, n-2, \ldots, 1)$$

Pseudo-code

```
%% forward elimination
for k = 1:n-1
    for i = k+1:n
        Mtp = A(i,k)/A(k,k);
        for j = k:n
            A(i,j) = A(i,j) - Mtp * A(k,j);
        end
        b(i) = b(i) - Mtp * b(k);
    end
end
```

```
%% backward substitution
b(n) = b(n)/A(n,n);
for i = n-1 : -1 : 1
    tmp = b(i);
    for j = i+1:n
        tmp = tmp - A(i,j)*b(j);
    end
    b(i) = tmp/A(i,i);
end
```

Testing the pseudocode

- Polynomial interpolation via Vandermonde
- Suppose the underlying polynomial

$$p(t) = 1 + t + t^2 + \cdots + t^{n-1} = \sum_{j=1}^{n} t^{j-1}$$

- Evaluate

$$p(1+i) = \sum_{j=1}^{n} (1+i)^{j-1} = (1+i)^n - 1/(1+i) - 1 = [(1+i)^n - 1]/i$$

- The roundoff error is propagated and magnified throughout the back substitution phase.

Pivoting

- Recall $a_{ij} \leftarrow a_{ij} - \left(\frac{a_{ik}}{a_{kk}}\right) a_{kj}$
- We must expect all quantities to be infected with roundoff errors.
- The roundoff error in a_{kj} is multiplied by $\left(\frac{a_{ik}}{a_{kk}}\right)$.
- The small pivot elements would lead to large multipliers and to worse roundoff errors.

Naïve Gaussian can fail

- Gaussian elimination would fail if $a_{11} = 0$.

$$\begin{cases} 0x_1 & + & x_2 & = & 1 \\ x_1 & + & x_2 & = & 2 \end{cases} \qquad \begin{cases} \varepsilon x_1 & + & x_2 & = & 1 \\ x_1 & + & x_2 & = & 2 \end{cases}$$

- How about this for a small number $\epsilon \neq 0$?

$$\begin{cases} \varepsilon x_1 + & x_2 & = & 1 \\ & \left(1 - \varepsilon^{-1}\right) x_2 & = & 2 - \varepsilon^{-1} \end{cases}$$

On 8-digit decimal computer

- Consider $\epsilon = 10^{-9} \Rightarrow \epsilon^{-1} = 10^9$.
- To compute $2 - \epsilon^{-1}$, the computer must interpret the numbers as

$$\begin{aligned} \varepsilon^{-1} = 10^9 &= 0.10000\,000 \times 10^{10} &= 0.10000\,00000\,00000\,0 \times 10^{10} \\ 2 &= 0.20000\,000 \times 10^{1} &= 0.00000\,00002\,00000\,0 \times 10^{10} \end{aligned}$$

- Thus $2 - \epsilon^{-1}$ is rounded to ϵ^{-1}.

Remedy

$$\begin{cases} 0x_1 & + & x_2 & = & 1 \\ x_1 & + & x_2 & = & 2 \end{cases} \qquad \begin{cases} \varepsilon x_1 & + & x_2 & = & 1 \\ x_1 & + & x_2 & = & 2 \end{cases}$$

Switch the two rows

$$\begin{cases} x_1 & + & x_2 & = & 2 \\ 0x_1 & + & x_2 & = & 1 \end{cases} \qquad \begin{cases} x_1 & + & x_2 & = & 2 \\ \varepsilon x_1 & + & x_2 & = & 1 \end{cases}$$

- Given

$$\begin{cases} x_1 + x_2 = 2 \\ \varepsilon x_1 + x_2 = 1 \end{cases}$$

- After elimination

$$\begin{cases} x_1 + x_2 = 2 \\ (1-\varepsilon)x_2 = 1-2\varepsilon \end{cases}$$

- Solution

$$x_2 = 1-2\varepsilon/1-\varepsilon \approx 1$$

$$x_1 = 2 - x_2 \approx 1$$

- Consider

$$\begin{cases} x_1 + \varepsilon^{-1}x_2 = \varepsilon^{-1} \\ x_1 + x_2 = 2 \end{cases}$$

- After elimination

$$\begin{cases} x_1 + \varepsilon^{-1}x_2 = \varepsilon^{-1} \\ (1-\varepsilon^{-1})x_2 = 2-\varepsilon^{-1} \end{cases}$$

- Solution

$$x_2 = (2-\varepsilon^{-1})/(1-\varepsilon^{-1}) \approx 1$$

$$x_1 = \varepsilon^{-1} - \varepsilon^{-1}x_2 \approx 0$$

Error analysis

For a linear system $Ax = b$ having the true solution x and a computed solution $\tilde{x}$, we define

- Error vector: $e = \tilde{x} - x$
- Residual vector: $r = A\tilde{x} - b$

For two solutions, how do we evaluate which one is better?

- Look at the residual vector: smaller the better!

Condition number