- L-U factorization: specialized way of factoring a matrix:

$$\underline{\underline{A}} = \underline{\underline{L}}\,\underline{\underline{U}}$$

- This enables very efficient solution of system $\underline{\underline{A}}\,\underline{x} = \underline{b}$

$$(\underline{\underline{L}}\,\underline{\underline{U}})\underline{x} = \underline{b} \quad \text{e.g.} \quad \underline{\underline{L}}^{-1}\underline{\underline{L}}\,\underline{\underline{U}}\,\underline{x} = \underline{\underline{L}}^{-1}\underline{b}$$

$$\rightsquigarrow \quad \underline{\underline{U}}\,\underline{x} = \underline{\underline{L}}^{-1}\underline{b} \quad \text{rewrite as} \quad \boxed{(*)\ \underline{\underline{U}}\,\underline{x} = \underline{b}'} \quad (\underline{b}' = \underline{\underline{L}}^{-1}\underline{b})$$

$$\boxed{(*)\quad \underline{\underline{L}}\,\underline{b}' = \underline{b}} \longleftarrow \text{LU-factorization algorithm is}$$

solve (1) $\underline{\underline{L}}\,\underline{b}' = \underline{b}$

(2) $\underline{\underline{U}}\,\underline{x} = \underline{b}'$

- Solution efficiently computed by solving (1) via fwd sub.; (2) solved via backsubstitution. Gauss elim is $\mathcal{O}(n^3)$, e.g. backsub is $\mathcal{O}(n^2)$
- Only useful insofar as $\underline{\underline{L}}\,\underline{\underline{U}}$ factorization is efficient. In fact, one can do a $\underbrace{\text{fwd elim}}_{\mathcal{O}(n^3)}$ in order to factorized (Doolittle algorithm)
- Note that

$\underline{\underline{L}}\,\underline{\underline{U}}$ factorization tends to faster for multiple RHS (only need to factor matrix once).

- Eig calculation of an inverse :

$$(*) \quad \underline{\underline{A}}\,\underline{\underline{A}}^{-1} = \underline{\underline{I}} \qquad \left[\underline{x}_1\ \underline{x}_2\ \underline{x}_3\right] \equiv \underline{\underline{A}}^{-1}$$

$$(1) \quad \underline{\underline{A}}\,\underline{x}_1 = \underline{b}_1 \qquad \left[\underline{b}_1\ \underline{b}_2\ \underline{b}_3\right] = \underline{\underline{I}}$$

$$(2) \quad \underline{\underline{A}}\,\underline{x}_2 = \underline{b}_2$$

$$(3) \quad \underline{\underline{A}}\,\underline{x}_3 = \underline{b}_3$$

- "Modern" $\underline{\underline{L}}\,\underline{\underline{U}}$ solvers :
  - LAPACK (full matrix)
  - MUMPS (sparse, unsymmetric)
  - umfpack (sparse, ")

- Lots of specialized algorithms for banded matrices ...

- Tridiagonal matrices (3 bands)
- Efficient elimination approach :

$$\underline{\underline{T}} = \begin{bmatrix} T_{11} & \boxed{T_{12}} & & & & 0 \\ T_{12} & T_{22} & T_{23} & & & \\ & T_{23} & T_{33} & T_{34} & & \\ & & T_{34} & T_{44} & T_{45} & \\ & & & & \ddots & \\ 0 & & & & T_{n-1,n} & T_{nn} \end{bmatrix}$$

$$R_2 \to R_2 - \frac{T_{21}}{T_{11}} R_1 \quad \left(\begin{matrix}\text{eliminates} \\ T_{12}\end{matrix}\right)$$

- Due to matrix structure fwd-elim only requires one modification :

(1) $$\boxed{T_{i,i} = T_{i,i} - \frac{T_{i,i-1}}{T_{i-1,i-1}} T_{i-1,i}}$$

fwd elim

$\forall i$

$i \in \{1, n\}$ $\rightsquigarrow$ converts into $\underline{U}$

(2) $$\boxed{b_i = b_i - \frac{T_{i,i-1}}{T_{i-1,i-1}} b_{i-1}}$$

(3) backsub : $$\boxed{x_n = b_n / U_{nn} \qquad x_i = \frac{1}{U_{i,i}} \left(b_i - U_{i,i+1} x_{i+1} \right)}$$

Thomas Algorithm

- Elimination methods can be sensitive to noise (conditioning)

characterized by:

matrix condition #

- A norm is a way of characterizing magnitude of $\underline{\underline{A}}$, satisfying the following:

(a) $\|\underline{\underline{A}}\| \geq 0$; $\|\underline{\underline{A}}\| = 0$ iff $\underline{\underline{A}} = \underline{\underline{0}}$

(b) $\|\alpha \underline{\underline{A}}\| = |\alpha| \|\underline{\underline{A}}\|$

(c) $\|\underline{\underline{A}} + \underline{\underline{B}}\| \leq \|\underline{\underline{A}}\| + \|\underline{\underline{B}}\|$ ("triangle inequality")

(d) $\|\underline{\underline{A}}\,\underline{\underline{B}}\| \leq \|\underline{\underline{A}}\| \|\underline{\underline{B}}\|$ (Schwarz inequality)

- Vector norms:

$$\|\underline{x}\|_1 = \sum_i |x_i| \qquad (L_1\text{-norm})$$

(*) $$\|\underline{x}\|_2 = \sqrt{\sum_i x_i^2} \qquad (L_2\text{-norm})$$

$$\|x\|_\infty = \max\left(|x_i|\right) \qquad (L_\infty\text{-norm})$$

- Matrices:

$$\|\underline{\underline{A}}\|_1 = \max_j \left\{ \sum_i |A_{ij}| \right\} \qquad (\text{max col. sum})$$

$$\|\underline{\underline{A}}\|_\infty = \max_i \left\{ \sum_j |A_{ij}| \right\} \qquad (\text{max row sum})$$

$$\|\underline{\underline{A}}\|_2 = \min\left(\lambda_i\right) \qquad (\text{spectral norm})$$

$$\|\underline{\underline{A}}\|_e = \sqrt{\sum_{ij} A_{ij}^2} \qquad (\text{Euclidean norm})$$

- Matrix condition number is a way to measure sensitivity to small perturbations: $\underline{\underline{A}}\,\underline{x} = \underline{b}$; $\boxed{\|\underline{A}\|\,\|\underline{x}\| \le \|\underline{b}\|}$
- Consider, e.g., system perturbed by $\underline{\delta b}$

$$\underline{\underline{A}}\,\underline{x} + \underline{\underline{A}}\,\underline{\delta x} = \underline{\underline{A}}\left(\underline{x} + \underline{\delta x}\right) = \underline{b} + \underline{\delta b} \Rightarrow \underline{\underline{A}}\,\underline{\delta x} = \underline{\delta b}$$

$$\underline{\delta x} = \underline{\underline{A}}^{-1}\, \underline{\delta b} \quad \Rightarrow \quad \boxed{\|\underline{\delta x}\| \leq \|\underline{\underline{A}}^{-1}\|\, \|\underline{\delta b}\|}$$

* Really concerned w/ relative error in $\underline{x}$, $\underline{b}$, e.g. $\dfrac{\|\underline{\delta x}\|}{\|\underline{x}\|}$

$$\boxed{\|\underline{b}\| \leq \|\underline{\underline{A}}\|\, \|\underline{x}\|} \qquad \|\underline{b}\|\, \|\underline{\delta x}\| \leq \|\underline{\underline{A}}\|\, \|\underline{\underline{A}}^{-1}\|\, \|\underline{x}\|\, \|\underline{\delta b}\|$$

$$\frac{\|\underline{\delta x}\|}{\|\underline{x}\|} = \left(\|\underline{\underline{A}}\|\, \|\underline{\underline{A}}^{-1}\|\right) \frac{\|\underline{\delta b}\|}{\|\underline{b}\|} \qquad \text{(condition \#)}$$

$$\hookrightarrow \boxed{c(\underline{\underline{A}}) \equiv \|\underline{\underline{A}}\| \cdot \|\underline{\underline{A}}^{-1}\|}$$

* large $c(\underline{\underline{A}})$ $\Rightarrow$ small relative change in $\underline{b}$ results in substantial change in $\underline{x}$
* very small $c(\underline{\underline{A}})$ $\Rightarrow$ small change in $\underline{x}$ result in sizeable change in $\underline{b}$
* well-conditioned system : $\boxed{c(\underline{\underline{A}}) \sim 1}$

- Iterative Methods for solving systems: address error accumulation in elimination methods.
- Iterative methods only work with diagonally dominant systems:

$$\boxed{\exists i \quad \text{s.t.} \quad |A_{ii}| \geq \sum_{j, j\neq i} |A_{ij}|}$$

- Many systems do not satisfy this requirement.
- Start w/ initial guess $(\underline{x}^{(0)})$ perform some operation to refine until a convergence criteria is met.
- Define residual:

it. #

$$R_i^{(k)} \equiv b_i - \sum_j A_{ij} x_j^{(k)}$$

(unknown #)

(want to be small)

$\Rightarrow$ "converged"

- Jacobi iteration algorithm:

$$\boxed{x_i^{(k+1)} = x_i^{(k)} + \frac{R_i^{(k)}}{A_{ii}}}$$

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

$$= A_{ii} x_i^{(k)} + b_i - \sum_j A_{ij} x_j^{(k)}$$

$$\boxed{x_i^{(k+1)} = x_i^{(k)} + \frac{b_i - \sum_j A_{ij} x_j^{(k)}}{A_{ii}}}$$

- $\underline{\text{Convergence}}$: $\left\{ \begin{array}{l} \underline{x}^{(k+1)} - \underline{x}^{(k)} < \varepsilon \\ \| \underline{R}^{(k)} \| < \delta \end{array} \right\}$ $\boxed{\underline{\underline{A}}\,\underline{x} \approx \underline{b}}$

- $\underline{\text{Gauss- Seidel iteration}}$: (same update formula)

$$x_i^{(k+1)} = x_i^{(k)} + \frac{R_i^{(k)}}{A_{ii}}$$

$$R_i^{(k)} = b_i - \sum_{j<i} A_{ij} x_j^{(k+1)} - \sum_{j \geq i} A_{ij} x_j^{(k)}$$

- Successive over-relaxation (SOR):

$$\boxed{x_i^{(k+1)} = x_i^{(k)} + \omega \frac{R_i^{(k)}}{A_{ii}}}$$

Stability ($\omega < 2$)

Same $R_i^{(k)}$ as Gauss-Seidel: $R_i^{(k)} = b_i - \sum_{j<i} A_{ij} x_j^{(k+1)} - \sum_{j \geq i} A_{ij} x_j^{(k)}$