Lec 16: Square Linear Systems – Further Analysis

Conditioning

Conditioning of Solving Linear Systems: Overview

- Analyze how robust (or sensitive) the solutions of $A\mathbf{x} = \mathbf{b}$ are to perturbations of A and $\mathbf{b}$.
- For simplicity, consider separately the cases where

1. $\mathbf{b}$ changes to $\mathbf{b} + \delta\mathbf{b}$, while A remains unchanged, that is

$$A\mathbf{x} = \mathbf{b} \quad \longrightarrow \quad A(\mathbf{x} + \delta\mathbf{x}) = \mathbf{b} + \delta\mathbf{b}.$$

2. A changes to $A + \delta A$, while $\mathbf{b}$ remains unchanged, that is

$$A\mathbf{x} = \mathbf{b} \quad \longrightarrow \quad (A + \delta A)(\mathbf{x} + \delta\mathbf{x}) = \mathbf{b}.$$

Sensitivity to Perturbation of RHS

Case 1. $A\mathbf{x} = \mathbf{b} \;\rightarrow\; A(\mathbf{x} + \delta\mathbf{x}) = \mathbf{b} + \delta\mathbf{b}$

- Bound $\|\delta\mathbf{x}\|$ in terms of $\|\delta\mathbf{b}\|$:

$$\begin{aligned} A\mathbf{x} + A\delta\mathbf{x} &= \mathbf{b} + \delta\mathbf{b} \\ A\delta\mathbf{x} &= \delta\mathbf{b} \\ \delta\mathbf{x} &= A^{-1}\delta\mathbf{b} \end{aligned} \qquad \Longrightarrow \qquad \|\delta\mathbf{x}\| \leqslant \|A^{-1}\|\|\delta\mathbf{b}\|.$$

- Sensitivity in terms of relative errors:

$$\frac{\dfrac{\|\delta\mathbf{x}\|}{\|\mathbf{x}\|}}{\dfrac{\|\delta\mathbf{b}\|}{\|\mathbf{b}\|}} = \frac{\|\delta\mathbf{x}\|\,\|\mathbf{b}\|}{\|\delta\mathbf{b}\|\,\|\mathbf{x}\|} \leqslant \frac{\|A^{-1}\|\,\|\delta\mathbf{b}\| \cdot \|A\|\,\|\mathbf{x}\|}{\|\delta\mathbf{b}\|\,\|\mathbf{x}\|} = \|A^{-1}\|\|A\|.$$

Sensitivity to Perturbation of Matrix

Case 2. $A\mathbf{x} = \mathbf{b} \;\rightarrow\; (A + \delta A)(\mathbf{x} + \delta\mathbf{x}) = \mathbf{b}$

- Bound $\|\delta\mathbf{x}\|$ now in terms of $\|\delta A\|$:

$$\begin{aligned} A\mathbf{x} + A\delta\mathbf{x} + (\delta A)\mathbf{x} + (\delta A)\delta\mathbf{x} &= \mathbf{b} \\ A\delta\mathbf{x} &= -(\delta A)\mathbf{x} - (\delta A)\delta\mathbf{x} \\ \delta\mathbf{x} &= -A^{-1}(\delta A)\mathbf{x} - A^{-1}(\delta A)\delta\mathbf{x} \end{aligned} \quad \Longrightarrow \quad \|\delta\mathbf{x}\| \lesssim \|A^{-1}\|\|\delta A\|\,\|\mathbf{x}\|\,.$$

(first-order truncation)

- Sensitivity in terms of relative errors:

$$\frac{\dfrac{\|\delta\mathbf{x}\|}{\|\mathbf{x}\|}}{\dfrac{\|\delta A\|}{\|A\|}} = \frac{\|\delta\mathbf{x}\|\,\|A\|}{\|\delta A\|\,\|\mathbf{x}\|} \lesssim \frac{\|A^{-1}\|\|\delta A\|\,\|\mathbf{x}\| \cdot \|A\|}{\|\delta A\|\,\|\mathbf{x}\|} = \|A^{-1}\|\|A\|.$$

Matrix Condition Number

- Motivated by the previous estimations, we define the **matrix condition number** by

$$\kappa(A) = \|A^{-1}\|\|A\|,$$

  where the norms can be any p-norm or the Frobenius norm.

- A subscript on κ such as 1, 2, ∞, or F(robenius) is used if clarification is needed.

Matrix Condition Number (Cont')

- We can write

$$\frac{\|\delta \mathbf{x}\|}{\|\mathbf{x}\|} \leqslant \kappa(A)\frac{\|\delta \mathbf{b}\|}{\|\mathbf{b}\|}, \qquad \frac{\|\delta \mathbf{x}\|}{\|\mathbf{x}\|} \leqslant \kappa(A)\frac{\|\delta A\|}{\|A\|},$$

  where the second inequality is true only in the limit of infinitesimal perturbations δA.

- The matrix condition number $\kappa(A)$ is equal to the condition number of solving a linear system of equation $A\mathbf{x} = \mathbf{b}$.
- The exponent of $\kappa(A)$ in scientific notation determines the approximate number of digits of accuracy that will be lost in calculation of $\mathbf{x}$.
- Since $1 = \|I\| = \|A^{-1}A\| \leqslant \|A^{-1}\|\|A\| = \kappa(A)$, a condition number of 1 is the best we can hope for.
- If $\kappa(A) > \boxed{\text{eps}}^{-1}$, then for computational purposes the matrix is singular.

Condition Numbers in MATLAB

- Use `cond` to calculate various condition numbers:

```
cond(A)              % the 2-norm; or  cond(A, 2)
cond(A, 1)           % the 1-norm
cond(A, Inf)         % the infinity-norm
cond(A, 'fro')       % the Frobenius norm
```

- A condition number estimator (in 1-norm)

```
condest(A)           % faster than cond
```

- The fastest method to estimate the condition number is to use `linsolve` function as below:

```
[x, inv_condest] = linsolve(A, b);
fast_condest = 1/inv_condest;
```

Special Matrices

Symmetric Matrices – LDLT Factorization

Let $A \in \mathbb{R}^{n \times n}$ be **symmetric**, that is, $A^{\mathrm{T}} = A$.

- The Gaussian elimination process without pivoting on this symmetric matrix yields
$$A = LDL^{\mathrm{T}},$$
where L is unit lower triangular and D is diagonal. (LDL$^{\mathrm{T}}$ Factorization)
- This factorization takes $\sim \frac{1}{3}n^3$ *flops*.
- Row pivoting is needed to keep LDL^{T} stable, but it is tedious.

Symmetric Positive Definite Matrices – Cholesky Factorization

Let $A \in \mathbb{R}^{n\times n}$.

- We say that A is **positive definite** if the *quadratic form* is positive, *i.e.*, $\mathbf{x}^{\mathrm{T}}A\mathbf{x} > 0$ for all $\mathbf{x} \neq \mathbf{0}$, *i.e.*,

$$\mathbf{x}^{\mathrm{T}}A\mathbf{x} = \sum_{i=1}^{n}\sum_{j=1}^{n} A_{ij}x_i x_j > 0 \quad \text{for } \mathbf{x} \neq \mathbf{0}.$$

- We say that A is **symmetric positive definite** (SPD) if A is symmetric and A is positive definite.
- **Useful.** A symmetric matrix is positive definite if and only if all its eigenvalues are real positive number[1].

[1] It follows that any SPD matrix is invertible.

Cholesky Factorization – Connection to LDLT

Let $A \in \mathbb{R}^{n\times n}$ be a SPD matrix.

- Symmetry implies $A = LDL^{\mathrm{T}}$.
- Positive definiteness implies $\mathbf{x}^{\mathrm{T}}A\mathbf{x} = \mathbf{x}^{\mathrm{T}}LDL^{\mathrm{T}}\mathbf{x} > 0$ for any $\mathbf{x} \neq \mathbf{0}$.

Consequently, the diagonal element d_{kk} of D is positive for all $k \in \mathbb{N}[1, n]$, which allows

$$A = LDL^{\mathrm{T}} = (LD^{1/2})(D^{1/2}L^{\mathrm{T}}) \equiv R^{\mathrm{T}}R,$$

where $R = D^{1/2}L^{\mathrm{T}}$ is an upper triangular matrix whose diagonal entries are positive.

Cholesky Factorization

Cholestky factorization: $A = R^{\mathrm{T}} R$

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{12} & a_{22} & a_{23} & \cdots & a_{2n} \\ a_{13} & a_{23} & a_{33} & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{14} & a_{2n} & a_{3n} & \cdots & a_{nn} \end{bmatrix}$$

$$= \underbrace{\begin{bmatrix} r_{11} & & & & \mathbf{0} \\ r_{12} & r_{22} & & & \\ r_{13} & r_{23} & r_{33} & & \\ \vdots & \vdots & \vdots & \ddots & \\ r_{1n} & r_{2n} & r_{3n} & \cdots & r_{nn} \end{bmatrix}}_{R^{\mathrm{T}}} \underbrace{\begin{bmatrix} r_{11} & r_{12} & r_{13} & \cdots & r_{1n} \\ & r_{22} & r_{23} & \cdots & r_{2n} \\ & & r_{33} & \cdots & r_{3n} \\ & & & \ddots & \vdots \\ \mathbf{0} & & & & r_{nn} \end{bmatrix}}_{R}$$

Cholesky Factorization – Implementation

The decomposition of a SPD matrix $A = R^{\mathrm{T}} R$ is called the **Cholesky factorization**.

- The calculation of R takes $\sim \frac{1}{3}n^3$ *flops*.
- Once R is obtained, $R^{\mathrm{T}} R\mathbf{x} = \mathbf{b}$ can be solved by forward elimination and backward substitution in $\sim 2n^2$ *flops*.
- **General Formula for** $R = [r_{jk}]$: For derivation, see Section 10.3.

$$r_{jj} = \left(a_{jj} - \sum_{i=1}^{j-1} r_{ij}^2 \right)^{1/2}$$

$$r_{jk} = \left(a_{jk} - \sum_{i=1}^{j-1} r_{ij} r_{ik} \right) \Big/ r_{jj} \quad \text{for } k = j+1,\, j+2,\, \ldots,\, n.$$

- In MATLAB, R is computed by

```
R = chol(A)
```

Banded Matrices

We say that $A \in \mathbb{R}^{n \times n}$ has

- **upper bandwidth** b_u if $A_{ij} = 0$ for $j - i > b_u$;
- **lower bandwidth** b_ℓ if $A_{ij} = 0$ for $i - j > b_\ell$.

The **total bandwidth** of A is $b_u + b_\ell + 1$.

Remarks.

- If no row pivoting is used, the LU factorization preserves the lower and upper bandwidths of A. (Why?)
- Since the zeros appear predictably, the factorization and the triangular substitutions can be done with much less operations. ($O(b_u b_\ell n)$)
- Use `sparse` function so that MATLAB can take advantage of the structure, *e.g.*,

```
[L, U, P] = lu( sparse(A) );
```