8. Problem condition and numerical stability

- vector and matrix norms
- the conditioning of a problem
- the numerical stability of an algorithm
- cancellation

Vector norms

a vector norm on $\mathbf{R}^n$ is a mapping $\|\cdot\| : \mathbf{R}^n \to [0, \infty)$ that satisfies

1. $\|\alpha x\| = |\alpha| \|x\|$ for any $\alpha \in \mathbf{R}$ (homogeneity)
2. $\|x + y\| \leq \|x\| + \|y\|$ (triangle inequality)
3. $\|x\| = 0$ if and only if $x = 0$ (definiteness)

2-norm

$$\|x\|_2 = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2} = \sqrt{x^T x}$$

1-norm

$$\|x\|_1 = |x_1| + |x_2| + \cdots + |x_n|$$

∞-norm

$$\|x\|_\infty = \max_k \{|x_1|, |x_2|, \ldots, |x_n|\}$$

Matrix norms

matrix norm of $A \in \mathbb{R}^{m \times n}$ is defined as

$$\|A\| = \max_{\|x\| \neq 0} \frac{\|Ax\|}{\|x\|} = \max_{\|x\|=1} \|Ax\|$$

also often called **operator norm** or **inducded norm**

properties:

1. for any x, $\|Ax\| \leq \|A\|\|x\|$
2. $\|aA\| = |a|\|A\|$ (scaling)
3. $\|A+B\| \leq \|A\| + \|B\|$ (triangle inequality)
4. $\|A\| = 0$ if and only if $A = 0$ (definiteness)
5. $\|AB\| \leq \|A\|\|B\|$

2-norm or **spectral norm**

$$\|A\|_2 \triangleq \max_{\|x\|_2=1} \|Ax\|_2 = \sqrt{\lambda_{\max}(A^T A)}$$

1-norm

$$\|A\|_1 \triangleq \max_{\|x\|_1=1} \|Ax\|_1 = \max_{j=1,\ldots,n} \sum_{i=1}^{m} |a_{ij}|$$

∞-norm

$$\|A\|_\infty \triangleq \max_{\|x\|_\infty=1} \|Ax\|_\infty = \max_{i=1,\ldots,m} \sum_{j=1}^{n} |a_{ij}|$$

other definitions of matrix norm also exist

Frobenius norm:

$$\|A\|_F = \sqrt{\mathbf{tr}(A^T A)} = \left(\sum_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|^2 \right)^{1/2}$$

Sources of error in numerical computation

example: evaluate a function $f : \mathbf{R} \to \mathbf{R}$ at a given x ($e.g.$, $f(x) = \sin x$)

sources of error in the result:

- x is not exactly known
  - measurement errors
  - errors in previous computations

  $\longrightarrow$ how sensitive is $f(x)$ to errors in x?

- the algorithm for computing $f(x)$ is not exact
  - discretization ($e.g.$, the algorithm uses a table to look up $f(x)$)
  - truncation ($e.g.$, f is computed by truncating a Taylor series)
  - rounding error during the computation

  $\longrightarrow$ how large is the error introduced by the algorithm?

The condition of a problem

sensitivity of the solution with respect to errors in the data

- a problem is **well-conditioned** if small errors in the data produce small errors in the result

- a problem is **ill-conditioned** if small errors in the data may produce large errors in the result

rigorous definition depends on what 'large error' means (absolute or relative error, which norm is used, . . .)

example: function evaluation

$$y = f(x), \qquad y + \Delta y = f(x + \Delta x)$$

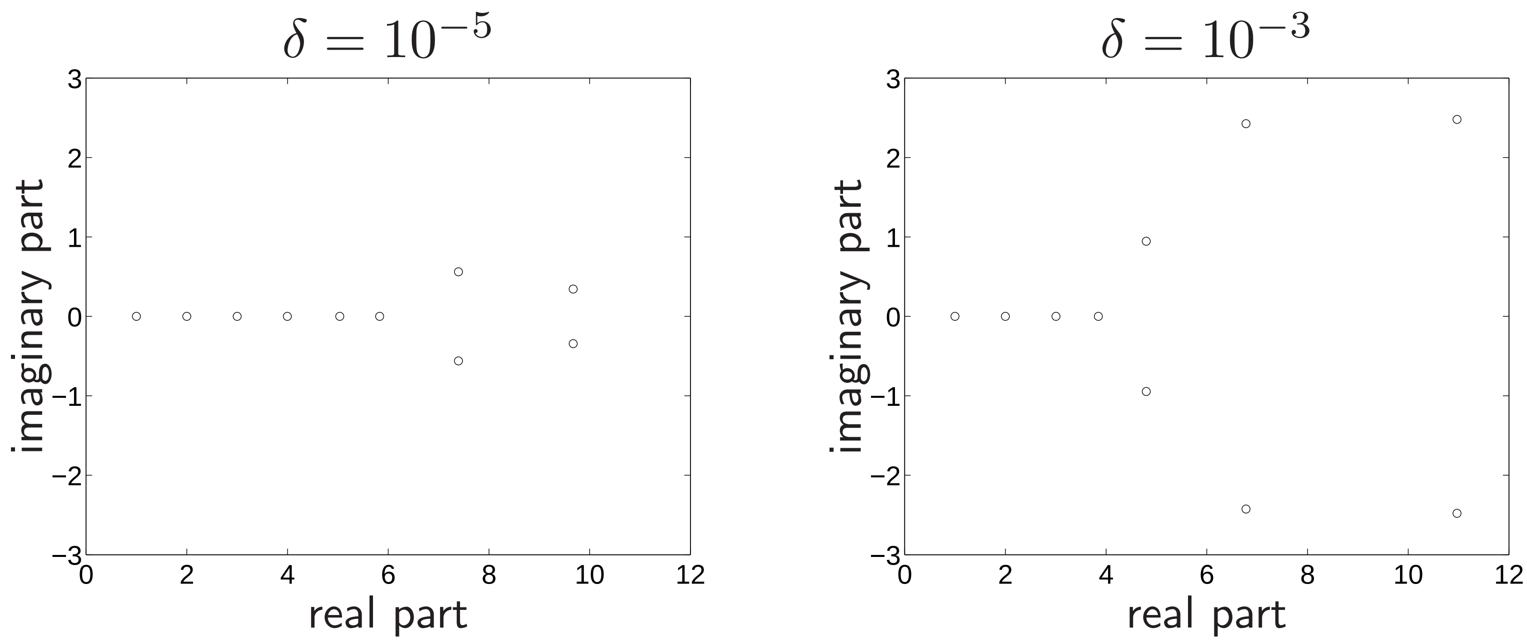
- absolute error
$$|\Delta y| \approx |f'(x)||\Delta x|$$
ill-conditioned with respect to absolute error if $|f'(x)|$ is very large

- relative error
$$\frac{|\Delta y|}{|y|} \approx \frac{|f'(x)||x|}{|f(x)|}\frac{|\Delta x|}{|x|}$$
ill-conditioned w.r.t relative error if $|f'(x)||x|/|f(x)|$ is very large

Roots of a polynomial

$$p(x) = (x-1)(x-2)\cdots(x-10) + \delta \cdot x^{10}$$

roots of p computed by Matlab for two values of δ



roots are very sensitive to errors in the coefficients

Condition of a set of linear equations

assume A is nonsingular and $Ax = b$

if we change b to $b + \Delta b$, the new solution is $x + \Delta x$ with

$$A(x + \Delta x) = b + \Delta b$$

the change in x is

$$\Delta x = A^{-1} \Delta b$$

'condition' of the equations: a technical term used to describe how sensitive the solution is to changes in the righthand side

- the equations are *well-conditioned* if small Δb results in small Δx
- the equations are *ill-conditioned* if small Δb can result in large Δx

Example of ill-conditioned equations

$$A = \frac{1}{2}\begin{bmatrix} 1 & 1 \\ 1+10^{-10} & 1-10^{-10} \end{bmatrix}, \qquad A^{-1} = \begin{bmatrix} 1-10^{10} & 10^{10} \\ 1+10^{10} & -10^{10} \end{bmatrix}$$

- solution for $b = (1,1)$ is $x = (1,1)$
- change in x if we change b to $b + \Delta b$:

$$\Delta x = A^{-1}\Delta b = \begin{bmatrix} \Delta b_1 - 10^{10}(\Delta b_1 - \Delta b_2) \\ \Delta b_1 + 10^{10}(\Delta b_1 - \Delta b_2) \end{bmatrix}$$

small Δb can lead to extremely large Δx

Bound on absolute error

suppose A is nonsingular and $\Delta x = A^{-1}\Delta b$

upper bound on $\|\Delta x\|$

$$\|\Delta x\| \leq \|A^{-1}\|\|\Delta b\|$$

(follows from property 1 on page 8-3)

- small $\|A^{-1}\|$ means that $\|\Delta x\|$ is small when $\|\Delta b\|$ is small
- large $\|A^{-1}\|$ means that $\|\Delta x\|$ can be large, even when $\|\Delta b\|$ is small
- for any A, there exists Δb such that $\|\Delta x\| = \|A^{-1}\|\|\Delta b\|$ (no proof)

Bound on relative error

suppose A is nonsingular, $Ax = b$ with $b \neq 0$, and $\Delta x = A^{-1}\Delta b$

upper bound on $\|\Delta x\|/\|x\|$:

$$\frac{\|\Delta x\|}{\|x\|} \leq \|A\|\|A^{-1}\|\frac{\|\Delta b\|}{\|b\|}$$

(follows from $\|\Delta x\| \leq \|A^{-1}\|\|\Delta b\|$ and $\|b\| \leq \|A\|\|x\|$)

$\kappa(A) = \|A\|\|A^{-1}\|$ is called the **condition number** of A

- small $\kappa(A)$ means $\|\Delta x\|/\|x\|$ is small when $\|\Delta b\|/\|b\|$ is small
- large $\kappa(A)$ means $\|\Delta x\|/\|x\|$ can be large, even when $\|\Delta b\|/\|b\|$ is small
- for any A, there exist b, Δb such that $\|\Delta x\|/\|x\| = \kappa(A)\|\Delta b\|/\|b\|$ (no proof)

Condition number

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

- defined for nonsingular A
- $\kappa(A) \geq 1$ for all A
- A is a *well-conditioned* matrix if $\kappa(A)$ is small (close to 1):

  the relative error in x is not much larger than the relative error in b
- A is *badly conditioned* or *ill-conditioned* if $\kappa(A)$ is large:

  the relative error in x can be much larger than the relative error in b

Iterative refinement

consider the linear system $Ax = b$ with a nonsingular A

- x_c is a computed solution to $Ax = b$ with the residual $r = b - Ax_c$
- x is the true solution

it follows that the solution error $e = x - x_c$ satisfies $Ae = r$

- x_c is deviated from x due to roundoff errors when A is ill-conditioned
- $x = x_c + e$ suggests us to improve the accuracy by an iterative algorithm

Iterative refinement:

---

given initial x, required tolerance $\epsilon > 0$
repeat

1. Compute $r = b - Ax$.
2. Solve $Ae = r$ using the existing LU factorization of A.
3. **if** $\|e\| \leq \epsilon$, **return** x.
4. Compute $x := x + e$.

until maximum number of iterations is exceeded

---

remarks:

- use the original matrix A (not LU) to compute the residual
- compute the residual in a higher precision to avoid the loss of significance

Analysis of iterative refinement

the refinement iteration can be written as

$$x^{(k+1)} = x^{(k)} + B(b - Ax^{(k)}), \quad k \geq 0$$

where B is an *approximate* inverse of A

it can be shown (by induction) that the iteration produces the sequence

$$x^{(n)} = B \sum_{k=0}^{n} (I - AB)^k b, \quad n \geq 0$$

under some condition, this sequence converges to $x = A^{-1}b$

Neumann series

if M is a square matrix such that $\|M\| < 1$ then $I - M$ is invertible and

$$(I - M)^{-1} = \sum_{k=0}^{\infty} M^k$$

Proof. it suffices to show that $(I - M)\sum_{k=0}^{n} M^k \to I$ as $n \to \infty$

- write the left-hand side as

$$(I - M)\sum_{k=0}^{n} M^k = \sum_{k=0}^{n}(M^k - M^{k+1}) = M^0 - M^{n+1} = I - M^{(n+1)}$$

- as $n \to \infty$, M^{n+1} goes to 0 because $\|M\| < 1$ which makes

$$\|M^{n+1}\| \leq \|M\|^{n+1} \to 0 \qquad \text{as } n \to \infty$$

Convergence of iterative refinement

quantify the loose term "B is an approximate inverse of A" as

$$\|I - AB\| < 1$$

if the above condition holds, from the Neumann series we have

$$B\sum_{k=0}^{\infty}(I - AB)^k = A^{-1}$$

which means the sequence $x^{(n)}$ converges to $x = A^{-1}b$ as $n \to \infty$

$$\lim_{n\to\infty} x^{(n)} = B\sum_{k=0}^{\infty}(I - AB)^k b = A^{-1}b = x$$

alternative proof: one can write

$$\begin{aligned} x^{(n+1)} - x &= x^{(n)} - x + B(Ax - Ax^{(n)}) \\ &= (I - BA)(x^{(n)} - x) \end{aligned}$$

apply an upper bound of the norm on both sides

$$\|x^{(n+1)} - x\| \leq \|I - BA\| \|x^{(n)} - x\|$$

and iterate the equality so that

$$\|x^{(n+1)} - x\| \leq \|I - BA\|^n \|x^{(0)} - x\|$$

since $\|I - BA\| < 1$, the error $\|x^{(n+1)} - x\|$ goes to 0

example: solving $Ax = b$ where

$$A = \begin{bmatrix} 1 & 1/2 & 1/3 \\ 1/2 & 1/3 & 1/4 \\ 1/3 & 1/4 & 1/5 \end{bmatrix} \qquad b = \begin{bmatrix} 3 \\ 23/12 \\ 43/30 \end{bmatrix}$$

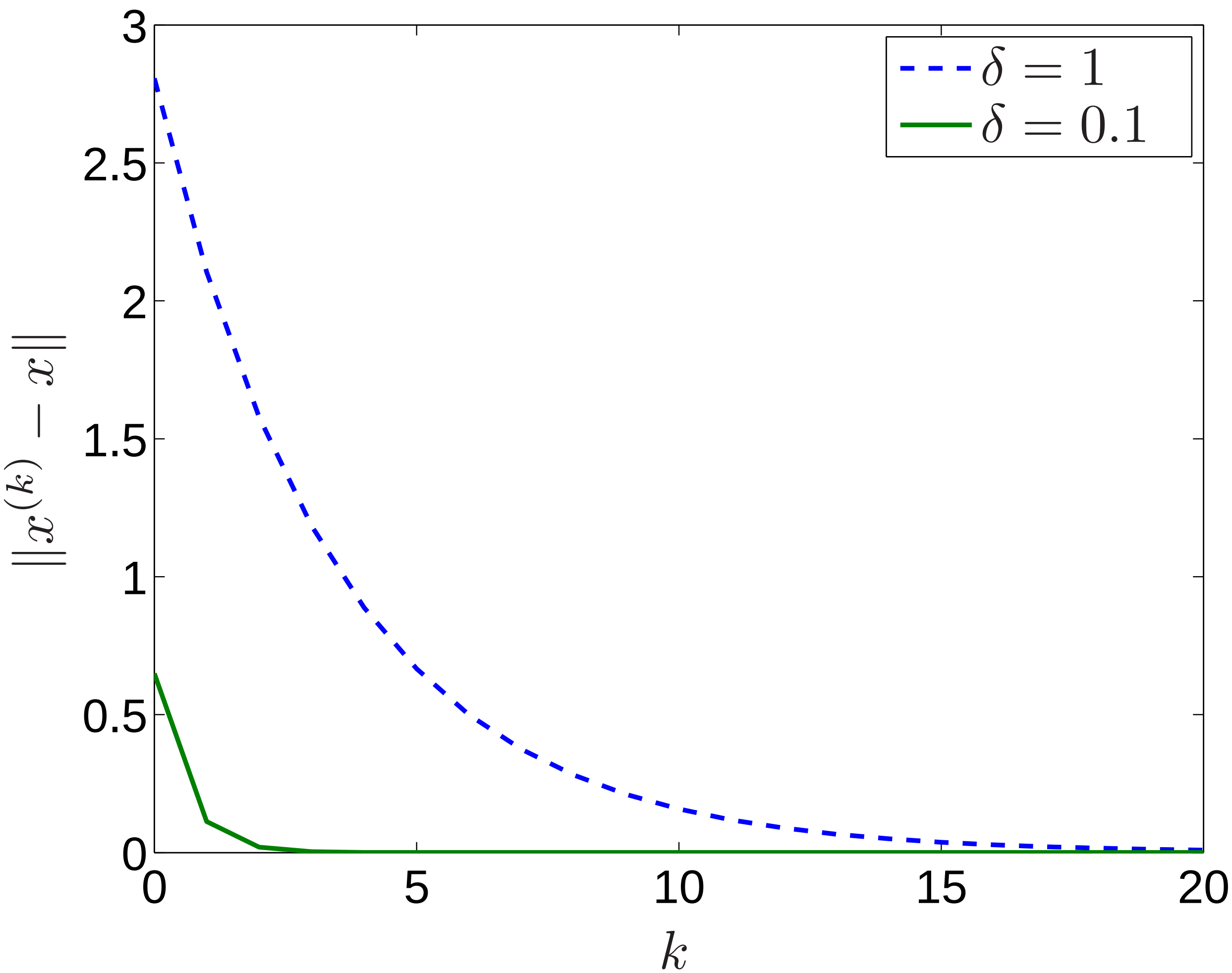
the exact solution is $x = (1, 2, 3)$ and an LU factorization is

$$L = \begin{bmatrix} 1 & 0 & 0 \\ 1/2 & 1 & 0 \\ 1/3 & 1 & 1 \end{bmatrix}, \quad U = \begin{bmatrix} 1 & 1/2 & 1/3 \\ 0 & 1/12 & 1/12 \\ 0 & 0 & 1/180 \end{bmatrix}$$

assume there is an roundoff error in computing L and U

$$L_c = L(1 + \delta), \qquad U_c = U(1 + \delta), \qquad \delta = 1$$

the initial solution is $x^{(0)} = (0.25, 0.5, 0.75)$ and we use L_c, U_c in the iterative refinement the norm of error versus the iteration



- for $\delta = 1$, $\|I - BA\| = 0.75$
- for $\delta = 0.1$, $\|I - BA\| = 0.1736$

Summary

the **conditioning** of a mathematical problem

- sensitivity of the solution with respect to perturbations in the data
- ill-conditioned problems are 'almost unsolvable' in practice (*i.e.*, in the presence of data uncertainty): even if we solve the problem exactly, the solution may be meaningless
- a property of a problem, independent of the solution method

stability of an algorithm

- accuracy of the result in the presence of rounding error
- a property of a numerical algorithm

precision of a computer

- a machine property (usually IEEE double precision, $i.e.$, about 15 significant decimal digits)
- a bound on the *rounding error* introduced when representing numbers in finite precision

accuracy of a numerical result

- determined by: machine precision, accuracy of the data, stability of the algorithm, . . .
- usually much smaller than 16 significant digits

References

Lecture notes on

Problem condition and numerical stability, EE103, L. Vandenberhge, UCLA

Chapter 7 in

J. F. Epperson, *An Introduction to Numerical Methods and Analysis*, John Wiley & Sons, 2007

Chapter 4 in

D. Kincaid and W. Cheney, *Numerical Analysis: Mathematics of Scientific Computing*, 3rd edition, Brooks & Cole, 2002