Conditioning and stability

Zecheng Zhang

March 31, 2023

In the abstract, we can view a problem as $f : X \to Y$ where X, Y are two spaces. A well-conditioned problem is one with the property that all small perturbation of x lead to only small changes in $f(x)$.

1 Relative condition number

Denote $\delta f = f(x + \delta x) - f(x)$. The relative conditioning number is defined as

$$\kappa(x) = \lim_{\delta \to 0} \sup_{\|\delta x\| \leq \delta} \left(\frac{\|\delta f\|}{\|f(x)\|} \Big/ \frac{\|\delta x\|}{\|x\|}\right). \tag{1}$$

One can assume δx and δf are infinitesimal, then

$$\kappa(x) = \sup_{\|\delta x\|} \left(\frac{\|\delta f\|}{\|f(x)\|} \Big/ \frac{\|\delta x\|}{\|x\|}\right). \tag{2}$$

When f is differentiable, we can express the quantity in terms of the Jacobian of f,

$$\kappa = \frac{\|J(x)\|}{\|f(x)\|/\|x\|}. \tag{3}$$

A problem is well-conditioned if κ is small (e.g., 1, 10, 100), and a problem is ill-conditioned if κ is large (e.g., 10^6 or bigger).

Example 1.1. Consider $x \to x/2$.

Example 1.2. Consider $x \to \sqrt{x}$, $x > 0$.

Example 1.3. Consider $f(x) = x_1 - x_2$.

2 Conditioning of matrix multiplication

Let $A \in \mathbb{R}^{m \times n}$, we consider the problem of computing Ax given a x. We want to know how Ax will change if there is a perturbation in x. The conditioning number of A is defined as,

$$\kappa = \sup_{\delta x} \left(\frac{\|A(x + \delta x) - Ax\|}{\|Ax\|} \Big/ \frac{\|\delta x\|}{\|x\|}\right) = \sup_{\delta x} \frac{\|A\delta x\|}{\|\delta x\|} \Big/ \frac{\|Ax\|}{\|x\|}. \tag{4}$$

Note that sup is over all δx and $\frac{\|Ax\|}{\|x\|}$ is independent with respect to sup, it follows that,

$$\kappa = \frac{\|x\|}{\|Ax\|} \sup_{\delta x} \frac{\|A\delta x\|}{\|\delta x\|} = \|A\| \frac{\|x\|}{\|Ax\|}, \tag{5}$$

where $\|A\|$ is the operator norm (it is L_2 norm if $\|\cdot\|$ is the L_2 vector norm). Note that, the condition number depends both on A and x.

Remark 1. Suppose A is nonsingular square matrix. We have $\|x\| = \|A^{-1}Ax\| \leq \|A^{-1}\|\|Ax\|$, this further implies that,

$$\kappa \leq \|A\|\|A^{-1}\|, \tag{6}$$

or

$$\kappa = c\|A\|\|A^{-1}\|, \tag{7}$$

for some positive constant $c = \frac{\|x\|}{\|Ax\|}/\|A^{-1}\|$.

Theorem 2.1. Let $A \in \mathbb{R}^{m\times n}$ be invertiable and let us consider $Ax = b$. The problem of computing b given x has conditioning number,

$$\kappa = \|A\|\frac{\|x\|}{\|b\|} \leq \|A\|\|A^{-1}\|, \tag{8}$$

with the perturbation in x. The problem of computing x given b has the conditioning number,

$$\kappa = \|A^{-1}\|\frac{\|b\|}{\|x\|} \leq \|A^{-1}\|\|A\|, \tag{9}$$

with the perturbation in b. If we use the L_2 norm, the first equality holds if x is a multiple of a right singular vector of A corresponding to the minimal singular value. The second equality holds if b is a multiple of a left singular vector of A corresponding to the largest singular value.

Definition 2.2. We will call $\kappa(A) = \|A\|\|A^{-1}\|$ the condition of A relative to norm $\|\cdot\|$ and denote it as $\kappa(A) = \|A\|\|A^{-1}\|$. The conditioning number is attached to matrix A not to the problem and x. If $\kappa(A)$ is small, A is called well-conditioned, otherwise, it is called ill-conditioned. If A is singular, we write $\kappa(A) = \infty$.

Remark 2. If $\|\cdot\| = \|\cdot\|_2$, $\|A\| = \sigma_1$ and $\|A^{-1}\| = 1/\sigma_m$, it follows that $\kappa(A) = \frac{\sigma_1}{\sigma_m}$

Remark 3. When $A \in \mathbb{C}^{m\times n}$ of full rank and $m \geq n$. The conditioning number is defined in terms of the pseudo-inverse, i.e.,

$$\kappa(A) = \|A\|\|A^{+}\|, \tag{10}$$

where $A^{+} = (A^{*}A)^{-1}A^{*}$ is called the pseudo-inverse of A.

3 Conditioning of a system of eqautions

We considered the case when A is fixed and perturbed x or b. What if we perturb A? Specifically, b is fixed and let us consider solving x from $Ax = b$ given a small change in A, We have,

$$(A + \delta A)(x + \delta x) = b \tag{11}$$

$$Ax + A\delta x + \delta Ax + \delta A\delta x = b. \tag{12}$$

Using $Ax = b$ and dropping the high order infinitesimal $\delta A\delta x$, it follows that $A\delta x + \delta Ax = 0$, or $\delta x = -A^{-1}\delta Ax$. Taking a norm, $\|\delta x\| \leq \|A^{-1}\|\|\delta A\|\|x\|$, or,

$$\frac{\|\delta x\|}{\|x\|}\Big/\frac{\|\delta A\|}{\|A\|} \leq \|A^{-1}\|\|A\| = \kappa(A). \tag{13}$$

Equality holds when $\|\delta x\| = \|A^{-1}\|\|\delta A\|\|x\|$. It can be shown that for any A and b such δA exists. This leads us to the following result.

Theorem 3.1. Let b be fixed and consider the problem $x = A^{-1}b$, where A is nonsingular. The conditioning number associated with this problem with respect to perturbation in A is:

$$\kappa = \|A\|\|A^{-1}\| = \kappa(A). \tag{14}$$

4 Floating point

Computers use a finite number of bits to represent real numbers, they can only represent only a finite subset of real numbers. This has two limitations. Firstly, the represented number cannot be arbitrarily large or small. Secondly, there must be gaps between them.

In IEEE double precision arithmetic (one way to store numbers/digital representation of number in the computer), the interval $[1, 2]$ is represented by the discrete subset:

$$1, 1 + 1 \times 2^{-52}, 1 + 2 \times 2^{-52}, ..., 2 + 2^{52} \times 2^{-52}. \tag{15}$$

In general, the interval $[2^j, 2^{j+1}]$ is represented by 15 times 2^j. The gap between the two adjacent numbers is never larger than $2^{-52} \approx 2.22 \times 10^{-16}$ in relative sense.

IEEE double precision is an example of an arithmetic system based on a floating-point $\mathbf{F}$ representation of real numbers. Here $\mathbf{F}$ is a discrete subset of real numbers (example, Equation 15) which is used to digitally represent real numbers. Let us now define the machine epsilon ϵ_m. This number is half the distance between 1 and the next larger floating point number. It has the following property.

Property 4.0.1. For all $x \in \mathbb{R}$, ther exists $x' \in F$ such that $|x - x'| \leq \epsilon_m |x|$.

This is in a relative sense since if $x > 0$, $|1 - x'/x| \leq \epsilon_m$.

Let $fl : \mathbb{R} \to F$ be a function giving the closet floating-point approximation to a real number (rounded to one floating number). Then the above property can be stated in terms of ft: for all $x \in \mathbb{R}$, there exists ϵ with $\epsilon < \epsilon_m$, there exists ϵ with $|\epsilon| < \epsilon_m$ such that $fl(x) = x(1 + \epsilon)$.

The difference between a real number and its closest floating-point approximation is always smaller than the machine ϵ_m in a relative sense. Machine epsilon or machine precision is an upper bound on the relative approximation error due to rounding in floating point arithmetic.

5 Stability

A mathematical problem can be formulated as $f : X \to Y$ where X and Y are some spaces. An algorithm can be viewed as another map $g : X \to Y$.

Definition 5.1 (Accuracy)**.** We say an algorithm is accurate if

$$\frac{\|g(x) - f(x)\|}{\|f(x)\|} = O(\epsilon_m), \tag{16}$$

for all $x \in X$.

Loosely speaking, the symbol $\mathcal{O}(\epsilon)$ means "on the order of machine epsilon". This expression applies uniformly to all x.

Remark 4. We discuss the order $\mathcal{O}$ here. Let us consider $h(t) = \mathcal{O}(g(t))$. The standard mathematical definition is: there exists a positive constant C such that for all t sufficient close to an understandable limit (for example, 0 or ∞), we have $\|h(t)\| \leq Cg(t)$.

Definition 5.2 (Backward Stability)**.** We say an algorithm g is backward stable if for all $x \in X$,

$$g(x) = f(y), \text{ for some } y \text{ with } \frac{\|x - y\|}{\|x\|} = O(\epsilon_m). \tag{17}$$

$f: X \to Y$ (mathematical problem)

$g: X \to Y$ (algorithm)

For: $x \xrightarrow[f(x)]{} \sin(x)$ (mathematical problem)

$x \xrightarrow[g(x)]{} fl(\sin(x))$ (algorithm)

assume the computer can calculate $\sin(x)$ exactly.

Def: (Accuracy)

We say an algorithm g is accurate if

$$\underbrace{\frac{\|g(x) - f(x)\|}{\|f(x)\|}}_{\text{the forward error.}} = O(\epsilon_m)$$

Remark:

1. $h(t) = O(g(t))$

$\exists\ C > 0$ s.t. for all t sufficient close to an understandable limit, we have $\|h(t)\| \leq C\|g(t)\|$.

2. for ϵ_m, examples of t: matrix size goes to ∞
discretization goes to ∞

def (Backward stability)

We say an algorithm g is backward stable if for all $x \in \bar{X}$

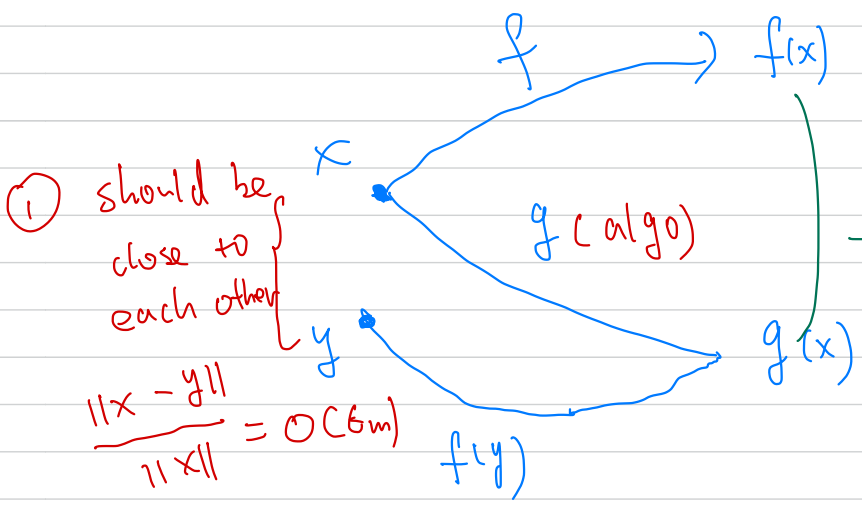
$g(x) = f(y)$, some y with $\frac{\|y-x\|}{\|x\|} = O(\epsilon_m)$

A backward stable algorithm gives the exactly right f answer to nearly the right question (y is close to x)

exact sol of a slightly wrong input y

$f(y)$

= algo with the exact input ($g(x)$)



[exact solution to the right problem]

① should be close to each other

$$\frac{\|x - y\|}{\|x\|} = O(\epsilon_m)$$

→ forward error, accuracy of the algo in the relative sense.

② the difference between x & y is called backward error.

The accuracy (forward error) is not easy, but a backward stable algo is "good" enough.

Thm 5.3 [gives us the relation between accuracy, b/w stable & conditioning of a problem]

Suppose a b/w stable algorithm g is applied to solve a problem f with conditioning $\kappa(x)$. Then the relative error (accuracy) satisfies

$$\frac{\|g(x) - f(x)\|}{\|f(x)\|} = O(\kappa(x)\epsilon_m)$$

accuracy = backward stable "+" well conditioning

Pf: By the definition of the backward stable

$g(x) = f(y)$ for some $y \in \tilde{x}$. with

$$\frac{\|x - y\|}{\|x\|} = O(\epsilon_m)$$

By the definition of conditioning

$$\frac{\frac{\|f(y)-f(x)\|}{\|f(x)\|}}{\frac{\|y-x\|}{\|x\|}}$$

b/w stable

$$=\frac{\frac{\|g(x)-f(x)\|}{\|f(x)\|}}{\underbrace{\frac{\|y-x\|}{\|x\|}}_{O(\epsilon_m)}} \leq \kappa(x)$$

$$\underbrace{\frac{\|g(x)-f(x)\|}{\|f(x)\|}}_{\text{accuracy of the algo } g.} \leq O(\kappa(x)\,\epsilon_m)$$

Eg. 5.4. $f(x) = \sin(x)$, $x = \frac{\pi}{2} - \delta$, δ is small.

algo g can evaluate $\sin(x)$ exactly however we need to round off the floating point arithmetic.

$$g(x) = fl(\sin(x))$$

Assume the algo is b/w stable.

$\exists\ y$ close enough to x, s.t. $f(y) = g(x)$.

However. $g(x) = f(y) = \sin(y)$

Taylor's expansion

$$= \sin(x) + (y-x)\sin'(x) + \text{error}$$

$$\overset{(*)}{=} \sin(x) + (y-x)\cos\left(\frac{\pi}{2} - x\right) + \text{err}$$

$$\cos(a-b) = \cos(a)\cos(b) + \sin(a)\sin(b)$$

$$\simeq \sin(x) + \delta(y-x) + \text{error}$$

$$(**) \rightarrow g(x) - f(x) \simeq fl(\sin(x)) - \sin(x) \simeq O(\epsilon_m)$$

The above is true b/c ϵ_m definition.

$$\frac{|fl(\sin(x)) - \sin(x)|}{|\sin(x)| \leq 1} = O(\epsilon_m)$$

$$\Rightarrow \quad \delta(y-x) \overset{(*)}{=} O(\epsilon_m) \overset{**}{}$$

$$y - x = O(\epsilon_m/\delta) \cdot O(\tfrac{\pi}{2})$$

As $\delta \to 0$, $y-x$ cannot be so close to each other

↯

Intuitively, a backward stable algorithm gives exactly the right answer to nearly the right question. $f(y)$ is "the exact solution (calculated by f) of a slightly wrong input (y which is closed to x)" and is exact to the algorithm with the exact input.

To repeat, conditioning is intrinsic to the problem. Stability is a property of an algorithm. Thus we will never say, "this problem is backward stable" or "this algorithm is ill-conditioned". We can say, "this problem is ill/well-conditioned", or "this algorithm is/isn't (backward) stable".

Theorem 5.3. Suppose a backward stable algorithm g is applied to solve a problem f with conditioning number κ. Then the relative error satisfies:

$$\frac{\|g(x) - f(x)\|}{\|f(x)\|} = O(\kappa(x)\epsilon_m). \tag{18}$$

Proof. By the definition of backward stability, we have $g(x) = f(y)$ for $y \in X$ satisfying

$$\frac{\|x - y\|}{\|x\|} = \mathcal{O}(\epsilon_m). \tag{19}$$

By the definition of conditioning number,

$$\frac{\|f(x) - g(y)\|}{\|f(x)\|} \Big/ \frac{\|x - y\|}{\|x\|} \leq \kappa(x). \tag{20}$$

It follows that,

$$\frac{\|f(x) - g(y)\|}{\|f(x)\|} \leq \kappa(x)\mathcal{O}(\epsilon_m) \approx \mathcal{O}(\kappa(x)\epsilon_m). \tag{21}$$

□

Here is how to interpret the result: If the problem is well-conditioned $O(\kappa) = 1$, this immediately implies good accuracy of the solution! However, otherwise, the solution might have poor accuracy. It is still the exact solution to a nearby problem (due to the backward stability). This is often as good as one can possibly hope for.

Example 5.4. Suppose we evaluate $f(x) = sin(x)$ for $x = \pi/2 - \delta$, δ is small. Suppose we are lucky enough to get as a computed result the exact correct answer, rounded to the floating point system: $g(x) = fl(sin(x))$ (i.e., g is the algorithm).

We want to find y close enough to x such that $g(x) = f(y)$. However, $g(x) = f(y) = f(x) + \delta(y - x) + error$, or $y - x \approx (g(x) - f(x))/\delta$. We have $g(x) - f(x) = fl(sin(x)) - sin(x) = \mathcal{O}(\epsilon_m)$, this implies that $y - x = \mathcal{O}(\epsilon_m/\delta)$. Since δ can be arbitrarily small, the $y - x$ is not of magnitude machine epsilon.