MA5233 Computational Mathematics

Lecture 3: Conditioning and Stability

Simon Etter

National University of Singapore

Semester I, AY 2020/2021

Conditioning and Stability

Introduction

Recall from Lecture 1:

- Bad: Most floating-point operations are inaccurate due to rounding.
- Good: Rounding errors are usually harmless.

`rounding_error()` attempts to quantify the "usually" in the above statement using statistics. We observe:

```
P(relative_error < 1e-4) ≈ 0.9999
P(relative_error > 1e-1) ≈ 2.0e-7
```

Thus, rounding errors are indeed very likely to be small, but there is also a nonzero probability for rounding errors to be catastrophically large.

Question addressed in this lecture:
Can we predict when rounding errors will be large?

Spoiler: The answer is yes, and the two key tools are the notions of *conditioning* and *stability*.

Conditioning and Stability

Def: Condition number

The condition number of a function $f : \mathbb{R} \to \mathbb{R}$ at a point $x \in \mathbb{R}$ is given by

$$\kappa(f, x) = \limsup_{\tilde{x} \to x} \frac{|f(\tilde{x}) - f(x)|}{|f(x)|} \frac{|x|}{|\tilde{x} - x|}.$$

Interpretation

The first factor is the relative change in output given the change $x \to \tilde{x}$ in the input. The second factor is 1 / [relative change in input].

The condition number is hence the ratio

$$\frac{\text{[relative change in output]}}{\text{[relative change in input]}}$$

in the limit of infinitely small changes in the input.

Theorem: If $f(x)$ is differentiable, then $\kappa(f, x) = \dfrac{|f'(x)|}{|f(x)|} \, |x|$.

Example: See `condition_number()`.

Conditioning and Stability

Rem: Error relative to 0

The relative error

$$\operatorname{relerr}(f, \tilde{f}, x) = \frac{|\tilde{f}(x) - f(x)|}{|f(x)|}$$

is undefined if $f(x) = 0$. In such cases, we define

$$\operatorname{relerr}(f, \tilde{f}, x) = \limsup_{y \to x} \operatorname{relerr}(f, \tilde{f}, y)$$

whenever possible.

Examples:

- $f(x) = x$, $\tilde{f}(x) = 0.99x \implies \operatorname{relerr}(f, \tilde{f}, x) = \frac{|0.99x - x|}{|x|} = 0.01$;
  hence we set $\operatorname{relerr}(f, \tilde{f}, x) = 0.01$ even for $x = 0$.
- $f(x) = x$, $\tilde{f}(x) = x + 0.01 \implies \operatorname{relerr}(f, \tilde{f}, x) = \frac{|x + 0.01 - x|}{|x|} = \frac{0.01}{|x|}$;
  hence we set $\operatorname{relerr}(f, \tilde{f}, 0) = \lim_{x \to 0} \frac{0.01}{|x|} = \infty$.

The same reasoning also extends to $\kappa(f, x)$:
if $\kappa(f, x)$ is undefined, then we set $\kappa(f, x) = \limsup_{y \to x} \kappa(f, y)$.

Usually these rules boil down to "$c/0 = \infty$ for all $c > 0$".

Conditioning and Stability

Why the condition number is important

Recall from Lecture 1 that computers only operate on finite sets of machine numbers represented by floating-point types like `FloatN` or `BigFloat` (denoted by `T` in the following).

To evaluate a function $f(x)$, computers must hence first round the argument x to the nearest representable machine number $T(x)$.

The condition number is important because it tells us how this rounding affects the accuracy of the output $f(x)$.

Examples: Computation: $\kappa(x \mapsto ax, x) = \frac{|a|\,|x|}{|ax|} = |a|$.

- $f(x) = 1$, $\kappa(f, x) = 0$: output is unaffected by error in input.
- $f(x) = x$, $\kappa(f, x) = 1$: error in output = error in input.
- $f(x) = 1000x$, $\kappa(f, x) = 1000$:
  error in output is 1000x larger than error in input.

The following slides will discuss an example where rounding of the input combined with large condition numbers has nontrivial effects.

Conditioning and Stability

Case study: `sin(pi)`

Consider the function $f(x) = \sin(x)$ evaluated at $x = \pi$.

- In exact arithmetic, we have $\sin(\pi) = 0$.
- In Julia we get `sin(pi) = 1.22e-16`.

$\sin(\pi) = 0$ can be represented exactly in any floating-point type `T`, so `sin(pi) = 1.22e-16` seems to violate the IEEE 754 convention that

$$\texttt{sin}(x) = \texttt{T}(\sin(x)).$$

Strictly speaking, IEEE 754 only applies to `+`,`-`,`*`,`/` and `sqrt`, but it makes sense to expect the same behaviour from functions like sin, exp, log, etc.

However, `sin(pi) = 1.22e-16` is actually consistent with IEEE 754.

To see why, recall that computers only operate on floating-point numbers; hence `sin(pi)` actually computes `sin(Float64(pi))`.

You can see this explicitly by looking at `@edit sin(pi)`.

`Float64(pi)`$\,!\!=\pi$, so Julia correctly returns `sin(Float64(pi))!=0.0`.

In fact, you can check using `BigFloat` that `sin(Float64(pi))` is the correctly rounded result:

```
julia> sin(pi) == Float64(sin(big(Float64(pi))))
true
```

Conditioning and Stability

Case study: `sin(pi)` **(continued)**

Let us now treat `sin(pi) = 1.22e-16` as an approximation to $\sin(\pi) = 0$ with relative error

$$\frac{|\sin(\pi) - \texttt{sin(pi)}|}{|\sin(\pi)|} = \frac{|0 - 1.22 \times 10^{-16}|}{|0|} = \infty.$$

See slide 4 for the reasoning why $c/0 = \infty$ here.

It may seem surprising that even though Julia is trying very hard to do the right thing, it still ends up with an infinitely large relative error.

However, this phenomenon is easily explained by looking at the condition number:

$$\kappa(\sin, x) = \frac{|\cos(x)|}{|\sin(x)|}\,|x| \qquad \Longrightarrow \qquad \kappa(\sin, \pi) = \tfrac{1}{0} \cdot \pi = \infty.$$

This tells us that the small rounding error introduced by replacing π with `Float64(pi)` may be "infinitely times" amplified by $\sin(x)$, which is precisely what we observed.

$\kappa(\sin, \pi) = \infty$ thus imposes a fundamental limit on how accurately we can evaluate $\sin(x)$ for arguments x close to π.

Conditioning and Stability

Case study: `sinpi(1.0)`

Like many fundamental limits, it is often possible to overcome the $\kappa(\sin, \pi) = \infty$ limit in some special cases, and it is very easy to wrongly interpret this victory in one battle as an victory in the overall war.

For example, Julia provides a `sinpi(x)` function which evaluates $\sin(\pi\, x)$ without explicitly rounding π to `Float64`.

For this function, we have `sinpi(1.0) == 0.0` since both the input `1.0` and the exact output $\sin(\pi) = 0.0$ can be represented exactly in `Float64`. At first sight, this seems to solve the problem of evaluating $\sin(\pi)$.

However, the problem reappears as soon as we try to evaluate $\sin(x)$ for arguments x which are very close but not equal to π:

```
julia> f = sinpi
       x_big = 1 + BigFloat(eps())^2   # != 1.0
       x_F64 = Float64(x)              # == 1.0
       abs(f(x_F64)-f(x_big)) / abs(f(x_big))
1.0
```

Conditioning and Stability

Case study: `sinpi(1.0)` **(continued)**

The relative error in this case is 1, which is better than ∞ but it is still useless: the distance between the earth and the moon and your bed and the kitchen is the same up to a relative error of 1.

Thus, while `sinpi(x)` solves the specific problem of evaluating $\sin(\pi)$ exactly, it does not solve the more general problem of evaluating $\sin(x)$ accurately for $x \approx \pi$, because doing so is impossible.

Conditioning and Stability

The following theorem is a bit out-of-place here, but we will need it later and I could not find a better place to introduce it.

Composition theorem for the condition number

$$\kappa(f \circ g, x) = \kappa\big(f, g(x)\big)\, \kappa(g, x)$$

Proof. The proof is straightforward if f and g are differentiable:

$$\begin{aligned}
\kappa(f \circ g, x) &= \frac{|f'(g(x))\, g'(x)|}{|f(g(x))|}\, |x| \\
&= \frac{|f'(g(x))|}{|f(g(x))|}\, |g(x)|\; \frac{|g'(x)|}{|g(x)|}\, |x| \\
&= \kappa\big(f, g(x)\big)\, \kappa(g, x)
\end{aligned}$$

I omit the general proof since it is more complicated but adds no further insight.

Conditioning and Stability

Discussion

Rule-of-thumb summary of our insights so far:

$$\kappa(f, x) \text{ is large} \quad \Longrightarrow \quad f(x) \text{ cannot be evaluated reliably.}$$

The converse is also true:

$$\kappa(f, x) \text{ is small} \quad \Longrightarrow \quad f(x) \text{ can be evaluated reliably.}$$

However, note the "can"! It is not true that if $\kappa(f, x)$ is small, then any algorithm for evaluating $f(x)$ is automatically reliable.

The next slide provides a concrete counterexample.

Example: Conditioning vs. stability

Consider $f(x) = \sqrt{1+x} - 1$. This function is well-conditioned at $x = 0$:

$$\kappa(f,0) = \lim_{x\to 0} \frac{\frac{1}{2}\,|\sqrt{1+x}|^{-1}\,|x|}{|\sqrt{1+x}-1|} = \lim_{x\to 0} \frac{\frac{1}{2}\,|x|}{|1+x-\sqrt{1+x}|}$$

$$\text{(L'Hôpital's rule)} = \lim_{x\to 0} \frac{\frac{1}{2}}{\left|1 - \frac{1}{2}\sqrt{1+x}^{\,-1}\right|} = 1.$$

Nevertheless, relative errors become large for $x \to 0$, see `stability()`.

Explanation: $f = g \circ h$ is the composition of

$$g(y) = y - 1, \qquad h(x) = \sqrt{1+x},$$

and $g(y)$ is ill-conditioned at $y = h(0) = 1$,

$$\kappa(g,1) = \lim_{y\to 1} \tfrac{|y|}{|y-1|} = \infty.$$

Thus, the rounding that is applied to the output of $h(x)$ may lead to an arbitrarily large error in $f(x) = g(h(x))$.

Conditioning and Stability

Example: Conditioning vs. stability (continued)

$\kappa(f, x) = 1$ suggests that there should be a way to evaluate $f(x)$ numerically up to a small relative error.

One such way can be obtained as follows:

$$f(x) = \sqrt{1+x} - 1 \;\; \frac{\sqrt{1+x}+1}{\sqrt{1+x}+1} = \frac{(1+x)-1}{\sqrt{1+x}+1} = \frac{x}{\sqrt{1+x}+1}. \qquad (1)$$

You may check that the last expression is a composition of only well-conditioned elementary functions.

Consequently, the relative errors remain bounded for $x \to 0$ if $f(x)$ is evaluated according to this formula, see `stability()`.

Remark

There is no general recipe for turning any arbitrary formula into a numerically stable one. In particular, it was not obvious that introducing the extra factor in the second expression of (1) would eventually lead to a stable formula.

In particular, it was not obvious that introducing the extra factor in the second expression of (1) would eventually lead to a stable formula.

Continued on next slide.

Remark (continued)

However, it was obvious that a stable formula must exist because $\kappa(f,x) = 1$. This further demonstrates the power of the condition number: if you had seen this example without knowing about $\kappa(f,x)$, you might have concluded that $\sqrt{1+x}-1$ with $x \approx 0$ is like $\sin(x)$ with $x \approx \pi$ and simply cannot be evaluated accurately. It is the knowledge that $\kappa(f,x)$ is small which encouraged us to look for a numerically stable formula, and had I not presented the solution to you, then you could have tried different formulas until you found one which is numerically stable.

Finding a numerically stable algorithm for evaluating a given function is thus similar to finding antiderivatives: there are simple criteria for determining that a stable formula / an antiderivative exists, but actually finding one is a matter of educated guessing.

Furthermore, just like existence of an antiderivative does not imply that this antiderivative can be expressed in terms of elementary functions, existence of stable formulae does not imply that they can always be easily written down. You will see an example of such a function in Assignment 1.

Conditioning and Stability

Discussion

The above example shows that we should distinguish between mathematical functions and numerical algorithms for evaluating such functions. I will do so by writing

- $f : \mathbb{R} \to \mathbb{R}$ for the "exact" function, and
- $\tilde{f} : \texttt{FloatN} \to \texttt{FloatN}$ for a numerical approximation.

The example further shows that two conditions must be satisfied for the numerical evaluation of $f(x)$ to be accurate.

- $\kappa(f, x)$ must be reasonably small, and
- the numerical algorithm for evaluating $f(x)$ must be "good".

In the example, we characterised "good" algorithms as those involving only well-conditioned elementary functions.

One drawback of this characterisation is that it only applies if $\kappa(f, x)$ itself is small, cf. the composition theorem from slide 10. This is often too restrictive, see the example on the next slide.

Conditioning and Stability

Example

Consider $f(x) = x - 1$. This function is ill-conditioned for $x \approx 1$,

$$\kappa(f, 1) = \lim_{x \to 1} \frac{|x|}{|x - 1|} = \infty.$$

Hence no algorithm for evaluating $f(x)$ can be "good" in the sense discussed above.

However, if both the input and output are to be represented in a floating-point type, then the obvious algorithm

$$\tilde{f}(x) = \mathtt{T}(x) \ominus 1 = \mathtt{T}(\mathtt{T}(x) - 1)$$

is clearly the best possible. It would therefore be nice if we could mark this algorithm as "good" even though its result may be inaccurate when $\kappa(f, x)$ is large.

The standard way to do so are the notions of backward and mixed stability introduced next. (Forward stability is introduced mainly for completeness, see the discussion on slide 19.)

Conditioning and Stability

Def: Stability of algorithms

A numerical algorithm $\tilde{f}(x)$ approximating a function $f(x)$ is called

- *forward stable* if $\dfrac{|\tilde{f}(x) - f(x)|}{|f(x)|} = O(\texttt{eps()})$,
- *backward stable* if there exists a $\tilde{x}$ such that

$$\tilde{f}(x) = f(\tilde{x}) \qquad \text{and} \qquad \frac{|\tilde{x} - x|}{|x|} = O(\texttt{eps()}),$$

- *(mixed) stable* if there exists a $\tilde{x}$ such that

$$\frac{|\tilde{f}(x) - f(\tilde{x})|}{|f(\tilde{x})|} = O(\texttt{eps()}) \qquad \text{and} \qquad \frac{|\tilde{x} - x|}{|x|} = O(\texttt{eps()}).$$

Strictly speaking, $\frac{|\tilde{f}(x) - f(x)|}{|f(x)|} = O(\texttt{eps()})$ means that the relative error in $\tilde{f}(x)$ may at most be proportional to the machine precision if we were to implement $\tilde{f}(x)$ in a sequence of floating-point types with decreasing `eps()`.

However, in practice statements like this one are usually interpreted in the number sense, i.e. we say $c = O(\texttt{eps()})$ if $c \leq 100\,\texttt{eps()}$ or maybe $c \leq 1000\,\texttt{eps()}$.

Conditioning and Stability

Def: Stability of algorithms (continued)

In words:

- $\tilde{f}(x)$ is forward stable if it produces an almost correct output for the correct input.
- $\tilde{f}(x)$ is backward stable if it produces the correct output for an almost correct input.
- $\tilde{f}(x)$ is stable if it produces an almost correct output for an almost correct input.

Note that if $\tilde{f}(x)$ is either forward or backward stable, then it is also stable, but the converse is not true.

Conditioning and Stability

Discussion

I expect that you will have no trouble understanding forward stability; after all, it is precisely the property that we would like all numerical algorithms to have.

Unfortunately, forward stability does not achieve our goal of separating the properties of the numerical algorithm $\tilde{f}(x)$ from those of the mathematical function $f(x)$.

Statement: $\tilde{f}(x)$ can only be forward stable if $f(x)$ is well-conditioned.

"Proof". $\tilde{f}(x)$ must necessarily round its input, and these rounding errors are then amplified by $\kappa(f, x)$. Hence if $\kappa(f, x)$ is large, then so is the relative error in $\tilde{f}(x)$.

The purpose of allowing perturbations in the input in the definition of backward and mixed stability is so we can attribute some of the error in the output to perturbations in the input (e.g. due to rounding) and a potentially large condition number.

The following slides will elaborate further.

Conditioning and Stability

Example

Consider again $f(x) = x - 1$ and its numerical approximation

$$\tilde{f}(x) = \mathtt{T}(x) \ominus 1 = \mathtt{T}(\mathtt{T}(x) - 1).$$

This approximation is

- not forward stable for $x \approx 1$,
- not backward stable for $x \approx 0$, but
- stable for all x.

Proof. See next slides.

Conditioning and Stability

Example (continued)

Forward stability.

For $x = 1 + \texttt{eps()}/2$, we have $f(x) = \texttt{eps()}/2$ but $\tilde{f}(x) = 0$ because $\texttt{T}(x) = 1$. Hence, the relative forward error is

$$\frac{|\tilde{f}(x) - f(x)|}{|f(x)|} = \frac{|0 - \texttt{eps()}/2|}{|\texttt{eps()}/2|} = 1 \neq O(\texttt{eps()}).$$

This is essentially the same as the earlier conditioning argument.

Backward stability.

For $x = -\texttt{eps()}/2$, we have $\tilde{f}(x) = -1 = f(0)$ because $\texttt{T}(x-1)a = -1$. Hence the relative backward error is

$$\frac{|\tilde{x} - x|}{|x|} = \frac{|0 + \texttt{eps()}/2|}{|\texttt{eps()}/2|} = 1 \neq O(\texttt{eps()}).$$

Conditioning and Stability

Example (continued)

Mixed stability

Set $\tilde{x} = \mathtt{T}(x)$ and observe that

$$\frac{|\tilde{f}(x) - f(\tilde{x})|}{|f(\tilde{x})|} = O(\mathtt{eps()}) \qquad \text{and} \qquad \frac{|\tilde{x} - x|}{|x|} = O(\mathtt{eps()})$$

is satisfied since $\tilde{f}(x) = \mathtt{T}(\mathtt{T}(x) - 1)$ and

$$\mathtt{T}(x) = x\left(1 + O(\mathtt{eps()})\right) \quad \Longleftrightarrow \quad \frac{|\mathtt{T}(x) - x|}{|x|} = O(\mathtt{eps()}).$$

Mixed stability thus achieves our goal of separating the conditioning of $f(x)$ from the quality of $\tilde{f}(x)$, and it remains to show that it is also strong enough to guarantee that $\tilde{f}(x)$ is accurate if $f(x)$ is well conditioned.

Conditioning and Stability

Theorem
Assume $\tilde{f}(x)$ is a backward-stable numerical approximation to $f(x)$. Then,

$$\frac{|\tilde{f}(x) - f(x)|}{|f(x)|} = \kappa(f, x)\, O(\texttt{eps()}).$$

Proof. Since $\tilde{f}(x)$ is backward stable, there exists a $\tilde{x}$ such that

$$\begin{aligned}
\frac{|\tilde{f}(x) - f(x)|}{|f(x)|} &= \frac{|f(\tilde{x}) - f(x)|}{|f(x)|} && \text{(backward stability)} \\
&= \big(\kappa(f, x) + o(1)\big)\, \frac{|\tilde{x} - x|}{|x|} && \text{(condition number)} \\
&= \kappa(f, x)\, O(\texttt{eps()}). && \text{(backward stability)}
\end{aligned}$$

Conditioning and Stability

Theorem

Assume $\tilde{f}(x)$ is a stable numerical approximation to $f(x)$. Then,

$$\frac{|\tilde{f}(x) - f(x)|}{|f(x)|} = \big(1 + \kappa(f, x)\big)\, O(\texttt{eps()}).$$

Proof (not examinable).

Using the backward part of the mixed stability property of $\tilde{f}(x)$ and the theorem from the previous slide, we obtain

$$\frac{|f(\tilde{x}) - f(x)|}{|f(x)|} = \kappa(f, x)\, O\big(\texttt{eps()}\big).$$

It then remains to show that

$$\tfrac{|\tilde{f}(x) - f(\tilde{x})|}{|f(\tilde{x})|} = O\big(\texttt{eps()}\big) \quad \wedge \quad \tfrac{|f(\tilde{x}) - f(x)|}{|f(x)|} = \kappa(f, x)\, O\big(\texttt{eps()}\big)$$
$$\Longrightarrow \quad \tfrac{|\tilde{f}(x) - f(x)|}{|f(x)|} = \big(1 + \kappa(f, x)\big)\, O(\texttt{eps()}).$$

This is achieved by the following lemma.

Conditioning and Stability

Lem: Transitivity of relative closeness

$$\frac{|x-y|}{|y|} \le \varepsilon_1 \quad \wedge \quad \frac{|y-z|}{|z|} \le \varepsilon_2 \quad \Longrightarrow \quad \frac{|x-z|}{|z|} = \varepsilon_1 + \varepsilon_2 + \varepsilon_1\,\varepsilon_2$$

Proof. We have

$$\begin{aligned} |y| - |z| \le |y-z| \le \varepsilon_2\,|z| \quad &\Longrightarrow \quad |y| \le |z|\,(1+\varepsilon_2) \\ &\Longrightarrow \quad \frac{1}{|z|} \le \frac{1+\varepsilon_2}{|y|} \end{aligned}$$

and thus

$$\begin{aligned} \frac{|x-z|}{|z|} &\le \frac{|x-y|}{|z|} + \frac{|y-z|}{|z|} \\ &\le \frac{|x-y|}{|y|}\,(1+\varepsilon_2) + \varepsilon_2 \\ &\le \varepsilon_1 + \varepsilon_2 + \varepsilon_1\,\varepsilon_2. \end{aligned}$$

Conditioning and Stability

Conclusion

The discussion so far involved quite a few twists and turns. However, all our insights can be summarised in a single sentence:

$\tilde{f}(x)$ is a reliable method for evaluating $f(x)$
if $f(x)$ is well-conditioned and $\tilde{f}(x)$ is stable.

If you understand this one sentence, then you understand everything there is to be understood from the slides so far.

Remark: Importance of backward stability

Loosely speaking, $f(x) = x - 1$ is not backward stable because $f(0) \neq 0$: $f(0) \neq 0$ means that the output error may be small relative to $f(0)$, but backwards stability requires that we transform this output error into an input error, and any nonzero error is infinitely large relative to $x = 0$.

Most functions studied in computational mathematics satisfy $f(0) = 0$ (e.g. matrix products, linear system solves, etc.), and stable algorithms for such functions are almost always also backward stable.

Backward stability is easier to work with than mixed stability; hence we will be talking about backward stability rather than stability for most of this module.

Conditioning and Stability

Discussion

So far, we exclusively studied functions $f : \mathbb{R} \to \mathbb{R}$, but many important functions are of the form $f : \mathbb{R}^n \to \mathbb{R}^m$.

All that is needed to extend conditioning and stability to such functions is to replace absolute values $|\cdot|$ with norms $\|\cdot\|$.

The following slides will briefly review some key definitions and results for norms and then demonstrate how conditioning and stability apply to the following functions.

- Scalar addition and multiplication: $\mathbb{R}^2 \to \mathbb{R}$.
- Inner products: $\mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$.
- Matrix-vector product (matvec): $\mathbb{R}^{m\times n} \times \mathbb{R}^n \to \mathbb{R}^m$.

Conditioning and Stability

Def: Norm

A function $\|\cdot\| : \mathbb{R}^n \to [0, \infty)$ is called a *norm on* $\mathbb{R}^n$ if for all $a \in \mathbb{R}$ and $u, v \in \mathbb{R}^n$ we have

- $\|u + v\| \leq \|u\| + \|v\|$ (triangle inequality),
- $\|a\, u\| = |a|\, \|u\|$ (absolute homogeneity),
- $\|u\| = 0 \iff u = 0$ (point separation).

Examples

$$\|x\|_1 = \sum_{k=1}^{n} |x[i]|, \qquad \|x\|_2 = \sqrt{\sum_{k=1}^{n} |x[i]|^2}, \qquad \|x\|_\infty = \max_{i \in \{1,\ldots,n\}} |x[i]|.$$

Conditioning and Stability

Thm: Norm equivalence in finite dimensions

For any two norms $\|\cdot\|_a$ and $\|\cdot\|_b$ on $\mathbb{R}^n$, there exist constants $c, C > 0$ depending only on n such that for all $x \in \mathbb{R}^n$ we have

$$c\,\|x\|_a \leq \|x\|_b \leq C\,\|x\|_b.$$

In particular, we have

$$\begin{aligned}
&\|x\|_1 \leq \sqrt{n}\,\|x\|_2, \qquad &&\|x\|_2 \leq \sqrt{n}\,\|x\|_\infty, \qquad &&\|x\|_1 \leq n\,\|x\|_\infty,\\
&\|x\|_2 \leq \|x\|_1, &&\|x\|_\infty \leq \|x\|_2, &&\|x\|_\infty \leq \|x\|_1.
\end{aligned}$$

Proof. See any linear algebra textbook.

Conditioning and Stability

Def: Matrix norm induced by vector norms

Let $\|\cdot\|_a$, $\|\cdot\|_b$ be two norms on $\mathbb{R}^n$ and $\mathbb{R}^m$, respectively.
Then, we define $\|\cdot\|_{a\to b} : \mathbb{R}^{m\times n} \to [0,\infty)$ through

$$\|A\|_{a\to b} = \sup_{x\in\mathbb{R}^n\setminus\{0\}} \frac{\|Ax\|_b}{\|x\|_a}.$$

Moreover, we introduce the abbreviation $\|A\|_a = \|A\|_{a\to a}$.

Theorem: $\|\cdot\|_{a\to b}$ is a norm on the vector space of $\mathbb{R}^{m\times n}$ matrices.

Examples

$$\|A\|_1 = \max_{j\in\{1,\dots n\}} \sum_{i=1}^{n} |A[i,j]|, \qquad \|A\|_\infty = \max_{i\in\{1,\dots n\}} \sum_{j=1}^{n} |A[i,j]|.$$

Proof. See any linear algebra textbook.

Conditioning and Stability

Thm: Singular value decomposition

Any matrix $A \in \mathbb{R}^{m \times n}$ can be written as $A = U \Sigma V^T$ where

- $U \in \mathbb{R}^{m \times k}$ and $V \in \mathbb{R}^{n \times k}$ with $k = \min\{m, n\}$ are orthogonal, and
  $U \in \mathbb{R}^{m \times n}$ with $m \geq n$ is called orthogonal if $U^T U = I$.
- $\Sigma \in \mathbb{R}^{k \times k}$ is diagonal with diagonal entries $\sigma_1 \geq \ldots \geq \sigma_k \geq 0$.

Thm: Matrix 2-norm and singular values

$$\|A\|_2 = \sigma_1 \qquad \text{and} \qquad \|A^{-1}\|_2 = \sigma_n^{-1}$$

where for the second identity I assumed that $A \in \mathbb{R}^{n \times n}$ is invertible.

Note that $A \in \mathbb{R}^{n \times n}$ is invertible iff $\sigma_n > 0$, so σ_n^{-1} is well defined in this case.

Proofs. See any linear algebra textbook.

Conditioning and Stability

Def: Condition number

Assume $f : \mathbb{R}^n \to \mathbb{R}^m$, $x \in \mathbb{R}^n$, and $\|\cdot\|_a$, $\|\cdot\|_b$ are norms on $\mathbb{R}^n$ and $\mathbb{R}^m$, respectively. Then, the condition number $\kappa_{a\to b}(f,x)$ is given by

$$\kappa_{a\to b}(f,x) = \lim_{\Delta x \to 0} \frac{\|f(x+\Delta x) - f(x)\|_b}{\|f(x)\|_b} \frac{\|x\|_a}{\|\Delta x\|_a}.$$

As before, we abbreviate $\kappa_a(f,x) = \kappa_{a\to a}(f,x)$.

Theorem

If $f(x)$ is differentiable, then we have

$$\kappa_{a\to b}(f,x) = \frac{\|\nabla f(x+\Delta x)\|_{a\to b}}{\|f(x)\|_b} \|x\|_a$$

Proof. Not quite as obvious as in the $f : \mathbb{R} \to \mathbb{R}$ case, but omitted since more a problem of analysis rather than computational mathematics.

Conditioning and Stability

Showing stability will require the following auxiliary result.

Lemma: Relative perturbation formula

$$\frac{|\tilde{x} - x|}{|x|} = \varepsilon \qquad \Longleftrightarrow \qquad \tilde{x} = x\,(1 \pm \varepsilon).$$

Proof. $\Longleftarrow$: Insert the right-hand side into the left-hand side.
$\Longrightarrow$: We have

$$|\tilde{x} - x| = |x|\,\varepsilon \quad \Longleftrightarrow \quad \tilde{x} = x \pm |x|\,\varepsilon = x \pm x\,\varepsilon = x\,(1 \pm \varepsilon).$$

Conditioning and Stability

Conditioning of addition

We have for all $x, y \in \mathbb{R}$ that

$$\kappa_1\big(+,(x\,y)^T\big) = \frac{\big\|(1\ 1)\big\|_1}{|x+y|}\,\big\|(x\ y)^T\big\|_1 = \frac{|x|+|y|}{|x+y|}$$

$\big\|(1\ 1)\big\|_1 = \big\|(1\ 1)^T\big\|_\infty = 1$ because the first norm is the matrix 1-norm.

Hence addition is well conditioned unless

$$|x+y| \ll |x|+|y| \quad \iff \quad x \approx -y.$$

Note that even though our analysis uses the 1-norm, the conclusion hold for all norms due to norm equivalence.

Stability of addition

According to IEEE 754, we have for some $\varepsilon \leq \texttt{eps()}/2$ that

$$x \oplus y = (x+y)\,(1+\varepsilon) = x\,(1+\varepsilon) + y\,(1+\varepsilon).$$

Hence $\oplus$ is backward stable.

I ignore here that $\oplus$ requires the input to be rounded. The implications of input rounding should be clear on an intuitive level, and rigorous statements can easily be worked out whenever needed.

Conditioning and Stability

Discussion: Cancellation

The ill-conditioning of addition is how the large errors arose in the example at the very beginning of this lecture: for `a,b = randn(2)`, there is a small but nonzero probability that $a \approx -b$, and addition will be inaccurate when this happens due to ill-conditioning.

To illustrate, replace `randn()` by `rand()` in `rounding_errors()` and note how the large errors disappear.

This phenomenon is known as "cancellation" and the main culprit for inaccurate floating-point results.

Conditioning and Stability

Conditioning of multiplication

We have for all $x, y \in \mathbb{R}$ that

$$\kappa_1\big(\times, (x\,y)^T\big) = \frac{\big\|(y\ x)\big\|_1}{|xy|}\,\big\|(x\ y)^T\big\|_1 = \frac{\max\{|y|,|x|\}\,(|x|+|y|)}{|xy|}$$
$$= 1 + \frac{\max\{|x|^2,|y|^2\}}{|xy|} = 1 + \max\Big\{\tfrac{|x|}{|y|},\tfrac{|y|}{|x|}\Big\}$$

$\big\|(|y\ x)\big\|_1 = \big\|(y\ x)^T\big\|_\infty = |x|$ because the first norm is the matrix 1-norm.

Hence multiplication is well-conditioned unless one number is much larger than the other.

Stability of multiplication

According to IEEE 754, we have for some $\varepsilon \leq \texttt{eps()}/2$ that

$$x \otimes y = (x \times y)\,(1+\varepsilon) = x \times \big(y\,(1+\varepsilon)\big).$$

Hence $\otimes$ is backward stable.

Conditioning and Stability

Discussion: Ill-conditioning of multiplication

The above result regarding the ill-conditioning of multiplication is somewhat misleading because it assumes that the errors in both x and y are proportional to $\max\{|x|, |y|\}$, but this is often not the case.

Example: The above result predicts that $\mathtt{T}(10^{16}) \otimes \mathtt{T}(\pi)$ should be inaccurate because $10^{16} \gg \pi$, but this is not true: we have for some $\varepsilon_i \le \mathtt{eps()}/2$ that

$$\begin{aligned}
\mathtt{T}(10^{16}) \otimes \mathtt{T}(\pi) &= \mathtt{T}(\mathtt{T}(10^{16}) \times \mathtt{T}(\pi)) \\
&= \Big(\big(10^{16}\,(1+\varepsilon_1)\big) \times \big(\pi\,(1+\varepsilon_2)\big)\Big)\,(1+\varepsilon_3) \\
&= \big(10^{16} \times \pi\big)\,(1+\varepsilon_1)\,(1+\varepsilon_2)\,(1+\varepsilon_3) \\
&= \big(10^{16} \times \pi\big)\,\big(1 + O(\mathtt{eps()})\big),
\end{aligned}$$

i.e. $\mathtt{T}(10^{16}) \otimes \mathtt{T}(\pi)$ is about as accurate as $\mathtt{T}(10^{16} \times \pi)$.

For further illustration, replace `+` by `*` in `rounding_errors()` and note how the large errors disappear.

Conditioning and Stability

Conditioning of inner product

Let $x, y \in \mathbb{R}^n$ and denote their inner product by $x \cdot y = x^T y$. We have

$$\kappa_2(\,\cdot\,, (x; y)) = \frac{\|(y^T \;\; x^T)\|_2}{|x^T y|} \, \|(x; y)\|_2 = \frac{\|x\|_2^2 + \|y\|_2^2}{|x^T y|}$$

I write $(x; y) = (x^T \;\; y^T)^T$ here to avoid excessive transposing.

Note that $\|(x; y)\|_2 = \sqrt{\|x\|_2^2 + \|y\|_2^2}$, and the matrix 2-norm satisfies $\|x^T\|_2 = \|x\|_2$ according to the Cauchy-Schwarz inequality.

Hence there are two ways how $\kappa_2(\,\cdot\,, (x; y))$ can become large:

- $\|x\|_2$ is much larger than $\|y\|_2$ or vice versa.

  Note that the numerator $= O(\|x\|^2)$ but the denominator $= O(\|x\|)$.

  This is the ill-conditioning of multiplication as discussed on slide 38 and usually not an issue.

- $\|x\|_2 \approx \|y\|_2$ but $|x^T y|$ is much smaller.

  This is cancellation as discussed on slide 36 and the main source of inaccuracy in inner products.

Conditioning and Stability

Stability of inner product

Let $x, y \in \mathbb{R}^n$ and $x \odot y = x_1 \otimes y_1 \oplus \ldots \oplus x_n \otimes y_n$ with the sum evaluated from left to right. If $n\,\texttt{eps()} < 1$, then

$$x \odot y = \tilde{x} \cdot y \quad \text{where} \quad \frac{|\tilde{x}[i] - x[i]|}{|x[i]|} \le \frac{n\,\texttt{eps()}}{1 - n\,\texttt{eps()}}.$$

Hence $\odot$ is backward stable.

Proof. See Assignment 1.

Stability of matrix-vector product

Let $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^n$ and $A \odot b = \big(A[1,:] \odot b \;\; \ldots ; A[m,:] \;\, b\big)^T$.
If $n\,\texttt{eps()} < 1$, then

$$A \odot b = Ab \quad \text{where} \quad \frac{|\tilde{A}[i,j] - A[i,j]|}{|A[i,j]|} \le \frac{n\,\texttt{eps()}}{1 - n\,\texttt{eps()}}.$$

Proof. The claim follows immediately from the stability of the inner product.

Conditioning and Stability

Conditioning of matrix-vector product

Analogous to $\kappa_2\big(\,\cdot\,,(x;y)\big) = \frac{\|x\|_2^2+\|y\|_2^2}{|x^Ty|}$, one can show

$$\kappa_2\big((A,b)\mapsto Ab\big) = \frac{\|A\|_2^2+\|b\|_2^2}{\|Ab\|_2}.$$

For notational convenience, I write $\kappa\big(f(x)\big)$ rather than $\kappa(f,x)$ here and below.

However, the proof and interpretation of this claim do require some extra work, namely we would have to clarify the meaning of $\|(A,b)\|_2$. I skip this technical difficulty because this result is actually rarely used in applications.

Instead, let us consider the conditioning of Ab if we only allow perturbations in either A or b but not both at the same time.

According to the stability result for $A \odot b$, it is enough to consider perturbations in A to assess the impact of rounding errors.

In this case, we have

$$\kappa_2(A\mapsto Ab) = \kappa_2(b\mapsto Ab) = \frac{\|A\|_2\,\|b\|_2}{\|Ab\|_2}.$$

Proof. See next slide.

Conditioning and Stability

Conditioning of matrix-vector product (continued)

Copied from previous slide:

$$\kappa_2(A \mapsto Ab) = \kappa_2(b \mapsto Ab) = \frac{\|A\|_2 \, \|b\|_2}{\|Ab\|_2}.$$

Proof (not examinable).

- $\kappa_2(b \mapsto Ab)$ follows immediately from the definition.
- $\kappa(A \mapsto Ab)$: The derivative of Ab with respect to A is the linear map $\Delta A \mapsto \Delta A b$, and we have

$$\|\Delta A \mapsto \Delta A b\|_2 = \sup_{\Delta A \in \mathbb{R}^{m \times n} \setminus \{0\}} \frac{\|Ab\|_2}{\|A\|_2} = \|b\|_2.$$

Interpretation. Matrix-vector product is well conditioned unless b is almost eliminated by A.

Conditioning and Stability

Example

Assume $\varepsilon \ll 1$ and consider

$$A = \begin{pmatrix} 1 & \\ & \varepsilon \end{pmatrix}, \quad b_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad b_2 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \quad Ab_1 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad Ab_2 = \begin{pmatrix} 0 \\ \varepsilon \end{pmatrix}.$$

We observe:

- Ab_1 is well conditioned: $\kappa_2(b \mapsto Ab, b_1) = \frac{1 \times 1}{1} = 1$.
- Ab_2 is ill-conditioned: $\kappa_2(b \mapsto Ab, b_2) = \frac{1 \times 1}{\varepsilon} = \varepsilon^{-1}$

Thus, small relative perturbations in b_2 can lead to large relative changes in Ab_2. For example,

$$\tilde{b}_2 = \begin{pmatrix} \varepsilon \\ 1 \end{pmatrix}, \quad A\tilde{b}_2 = \begin{pmatrix} \varepsilon \\ \varepsilon \end{pmatrix}, \quad \frac{\|\tilde{b}_2 - b_2\|_2}{\|b_2\|_2} = \frac{\varepsilon}{1}, \quad \frac{\|A\tilde{b}_2 - Ab_2\|_2}{\|Ab_2\|_2} = \frac{\varepsilon}{\varepsilon}.$$

Illustration for $\varepsilon = 0$:

Conditioning and Stability

Remark

In many applications, we want to understand the numerical properties of Ab for a given fixed A but any arbitrary b. In such cases, it is common to study the following quantity instead of $\kappa(b \mapsto Ab)$.

Def: Condition number of a matrix

The condition number of a matrix $A \in \mathbb{R}^{m \times n}$ is given by

$$\kappa(A) = \sup_{b \in \mathbb{R} \setminus \{0\}} \kappa(b \mapsto Ab).$$

Warning!

The matrix condition number $\kappa(A)$ and function condition number $\kappa(f, x)$ have very similar names and notations, but these two quantities are different and it is important not to confuse the two.

Conditioning and Stability

Theorem

If $A \in \mathbb{R}^{n \times n}$ is invertible, then

$$\kappa(A) = \|A\|_2 \, \|A^{-1}\|_2.$$

Proof. We have for $Ab = c \iff b = A^{-1}c$ that

$$\kappa(A) = \sup_{b \in \mathbb{R}\setminus\{0\}} \frac{\|A\|_2 \, \|b\|_2}{\|Ab\|_2} = \sup_{c \in \mathbb{R}\setminus\{0\}} \frac{\|A\|_2 \, \|A^{-1}c\|_2}{\|c\|_2} = \|A\|_2 \, \|A^{-1}\|_2.$$

Conditioning and Stability

Summary

- Condition number: $\kappa_a(f, x) = \frac{\|\nabla f(x)\|_a}{\|f(x)\|_a} \, \|x\|_a$.
- Stability: $\exists \tilde{x} : \dfrac{|\tilde{x} - x|}{|x|} = O(\texttt{eps()}) \ \wedge \ \dfrac{|\tilde{f}(x) - f(\tilde{x})|}{|f(\tilde{x})|} = O(\texttt{eps()})$.
- Backward stability: $\exists \tilde{x} : \dfrac{|\tilde{x} - x|}{|x|} = O(\texttt{eps()}) \ \wedge \ \tilde{f}(x) = f(\tilde{x})$.
- $\tilde{f}(x)$ stable $\quad \Longrightarrow \quad = \big(1 + \kappa(f, x)\big) \, O(\texttt{eps()})$
- Application to scalar addition and multiplication, inner product and matrix-vector product.

Recommended exercise

Pick your favourite function (e.g. $\sqrt{\cdot}$, $\log(\cdot)$, $\div$, etc.), study its conditioning and check that it is stable assuming $\tilde{f}(x) = \texttt{T}\big(f(x)\big)$.

Further reading

- Trefethen, Bau (1997). *Numerical Linear Algebra*, Lectures 12-15.