MA423 Matrix Computations

Lecture 3: Accuracy and Stability Analysis

Rafikul Alam
Department of Mathematics
IIT Guwahati

Outline

- Backward stability of algorithms
- ill-conditioning
- Accuracy of computed solutions

Finite Precision Arithmetic

Arithmetic operations on computers are inexact. For example, computing

$$a + b$$

on a computer, produces

Finite Precision Arithmetic

Arithmetic operations on computers are inexact. For example, computing

$$a+b$$

on a computer, produces

$$(a+b)(1+e)$$

where e is called the rounding error.

Finite Precision Arithmetic

Arithmetic operations on computers are inexact. For example, computing

$$a+b$$

on a computer, produces

$$(a+b)(1+e)$$

where e is called the rounding error.

The rounding error is bounded by $|\mathbf{e}| \leq \mathbf{u}$, where $\mathbf{u}$ is the unit roundoff.

Single precision: $\mathbf{u} \simeq \mathbf{5.96} \times \mathbf{10}^{-8}$

Finite Precision Arithmetic

Arithmetic operations on computers are inexact. For example, computing

$$\mathsf{a+b}$$

on a computer, produces

$$\mathsf{(a+b)(1+e)}$$

where e is called the rounding error.

The rounding error is bounded by $|\mathbf{e}| \leq \mathbf{u}$, where $\mathbf{u}$ is the unit roundoff.

Single precision: $\mathbf{u} \simeq \mathbf{5.96} \times \mathbf{10^{-8}}$

Double precision: $\mathbf{u} \simeq \mathbf{1.11} \times \mathbf{10^{-16}}$

Finite Precision Arithmetic

Arithmetic operations on computers are inexact. For example, computing

$$a+b$$

on a computer, produces

$$(a+b)(1+e)$$

where e is called the rounding error.

The rounding error is bounded by $|\mathbf{e}| \leq \mathbf{u}$, where $\mathbf{u}$ is the unit roundoff.

Single precision: $\mathbf{u} \simeq \mathbf{5.96} \times \mathbf{10}^{-8}$

Double precision: $\mathbf{u} \simeq \mathbf{1.11} \times \mathbf{10}^{-16}$

IEEE standard enables one to keep track of small errors that are made when two numbers are added, subtracted, multiplied or divided on a computer.

Propagation of rounding errors

Errors are unavoidable in numerical computation.

Propagation of rounding errors

Errors are unavoidable in numerical computation.

- Finite precision arithmetic leads to rounding errors.

Propagation of rounding errors

Errors are unavoidable in numerical computation.

- Finite precision arithmetic leads to rounding errors.
- Errors introduced by an algorithm during computations (rounding and truncations) is called computational errors.

Propagation of rounding errors

Errors are unavoidable in numerical computation.

- Finite precision arithmetic leads to rounding errors.
- Errors introduced by an algorithm during computations (rounding and truncations) is called computational errors.
- During computation, an algorithm either magnifies these errors (unstable algorithm) or keep them under check (stable algorithm).

Errors

Two common measures of error are

$$\text{Absolute Error} := \|\text{True Value} - \text{Approx. Value}\|$$

Errors

Two common measures of error are

$$\text{Absolute Error} := \|\text{True Value} - \text{Approx. Value}\|$$

$$\text{Relative Error} := \frac{\text{Absolute Error}}{\|\text{True Value}\|}$$

Errors

Two common measures of error are

$$\text{Absolute Error} := \|\text{True Value} - \text{Approx. Value}\|$$

$$\text{Relative Error} := \frac{\text{Absolute Error}}{\|\text{True Value}\|}$$

But

Errors

Two common measures of error are

$$\text{Absolute Error} := \|\text{True Value} - \text{Approx. Value}\|$$

$$\text{Relative Error} := \frac{\text{Absolute Error}}{\|\text{True Value}\|}$$

But

- true value is usually unknown,
- so we estimate or bound errors rather than compute them exactly.

Errors

Two common measures of error are

$$\text{Absolute Error} := \|\text{True Value} - \text{Approx. Value}\|$$

$$\text{Relative Error} := \frac{\text{Absolute Error}}{\|\text{True Value}\|}$$

But

- true value is usually unknown,
- so we estimate or bound errors rather than compute them exactly.

Relative error is commonly used for analysis of rounding errors and stability of algorithms. On the other hand, absolute error is used for analysis of truncation errors.

Algorithm

An algorithm is a function $\mathrm{ALG} : (X, \|\cdot\|) \longrightarrow (Y, \|\cdot\|)$ such that

Algorithm

An algorithm is a function $\mathrm{ALG} : (X, \|\cdot\|) \longrightarrow (Y, \|\cdot\|)$ such that

- computation of $\mathrm{ALG}(\text{input})$ involves only a finite number of steps

Algorithm

An algorithm is a function $\mathrm{ALG} : (X, \|\cdot\|) \longrightarrow (Y, \|\cdot\|)$ such that

- computation of $\mathrm{ALG}(\text{input})$ involves only a finite number of steps
- and each step performs a finite number of elementary arithmetic operations.

Algorithm

An algorithm is a function $\mathrm{ALG} : (X, \|\cdot\|) \longrightarrow (Y, \|\cdot\|)$ such that

- computation of $\mathrm{ALG}(\text{input})$ involves only a finite number of steps
- and each step performs a finite number of elementary arithmetic operations.

Let $F(d)$ be a solution of a problem with given data d and $\mathrm{ALG}(d)$ be the computed solution.

Algorithm

An algorithm is a function $\mathrm{ALG} : (X, \|\cdot\|) \longrightarrow (Y, \|\cdot\|)$ such that

- computation of $\mathrm{ALG}(\text{input})$ involves only a finite number of steps
- and each step performs a finite number of elementary arithmetic operations.

Let $F(d)$ be a solution of a problem with given data d and $\mathrm{ALG}(d)$ be the computed solution. Then the accuracy of the computed solution $\mathrm{ALG}(d)$ is measured by the (relative) error

Algorithm

An algorithm is a function $\mathrm{ALG} : (X, \|\cdot\|) \longrightarrow (Y, \|\cdot\|)$ such that

- computation of $\mathrm{ALG}(\text{input})$ involves only a finite number of steps
- and each step performs a finite number of elementary arithmetic operations.

Let $F(d)$ be a solution of a problem with given data d and $\mathrm{ALG}(d)$ be the computed solution. Then the accuracy of the computed solution $\mathrm{ALG}(d)$ is measured by the (relative) error

$$\text{Error} = \frac{\|\mathrm{ALG}(d) - F(d)\|}{\|F(d)\|}$$

Algorithm

An algorithm is a function $\mathrm{ALG} : (X, \|\cdot\|) \longrightarrow (Y, \|\cdot\|)$ such that

- computation of $\mathrm{ALG}(\text{input})$ involves only a finite number of steps
- and each step performs a finite number of elementary arithmetic operations.

Let $F(d)$ be a solution of a problem with given data d and $\mathrm{ALG}(d)$ be the computed solution. Then the accuracy of the computed solution $\mathrm{ALG}(d)$ is measured by the (relative) error

$$\text{Error} = \frac{\|\mathrm{ALG}(d) - F(d)\|}{\|F(d)\|}$$

Definition: An algorithm ALG is said to be backward stable (stable) if

Algorithm

An algorithm is a function $\mathrm{ALG} : (X, \|\cdot\|) \longrightarrow (Y, \|\cdot\|)$ such that

- computation of $\mathrm{ALG}(\text{input})$ involves only a finite number of steps
- and each step performs a finite number of elementary arithmetic operations.

Let $F(d)$ be a solution of a problem with given data d and $\mathrm{ALG}(d)$ be the computed solution. Then the accuracy of the computed solution $\mathrm{ALG}(d)$ is measured by the (relative) error

$$\text{Error} \;=\; \frac{\|\mathrm{ALG}(d) - F(d)\|}{\|F(d)\|}$$

Definition: An algorithm ALG is said to be backward stable (stable) if

- $\mathrm{ALG}(d) = F(d + \Delta d)$ for some $\Delta d \in X$ such that $\dfrac{\|\Delta d\|}{\|d\|} = \mathcal{O}(\mathbf{u})$.

Algorithm

An algorithm is a function $\mathrm{ALG} : (X, \|\cdot\|) \longrightarrow (Y, \|\cdot\|)$ such that

- computation of $\mathrm{ALG}(\text{input})$ involves only a finite number of steps
- and each step performs a finite number of elementary arithmetic operations.

Let $F(d)$ be a solution of a problem with given data d and $\mathrm{ALG}(d)$ be the computed solution. Then the accuracy of the computed solution $\mathrm{ALG}(d)$ is measured by the (relative) error

$$\text{Error} = \frac{\|\mathrm{ALG}(d) - F(d)\|}{\|F(d)\|}$$

Definition: An algorithm ALG is said to be backward stable (stable) if

- $\mathrm{ALG}(d) = F(d + \Delta d)$ for some $\Delta d \in X$ such that $\dfrac{\|\Delta d\|}{\|d\|} = \mathcal{O}(\mathbf{u})$.

The quantity $\dfrac{\|\Delta d\|}{\|d\|}$ is called the backward error.

Examples

Example 1: Consider $Ax = b$. Then $x = F(A, b) = A^{-1}b$. Let $\hat{x} = \mathrm{ALG}(A, b)$.

Examples

Example 1: Consider $Ax = b$. Then $x = F(A, b) = A^{-1}b$. Let $\hat{x} = \mathrm{ALG}(A, b)$.Then

ALG stable $\Longrightarrow \hat{x} = \mathrm{ALG}(A, b) = F(A + \Delta A, b + \Delta b)$, that is,

Examples

Example 1: Consider $Ax = b$. Then $x = F(A, b) = A^{-1}b$. Let $\hat{x} = \mathrm{ALG}(A, b)$.Then

ALG stable $\Longrightarrow$ $\hat{x} = \mathrm{ALG}(A, b) = F(A + \Delta A, b + \Delta b)$, that is, $(A + \Delta A)\hat{x} = b + \Delta b$ such that $\dfrac{\|\Delta A\|}{\|A\|} = \mathcal{O}(\mathbf{u})$ and $\dfrac{\|\Delta b\|}{\|b\|} = \mathcal{O}(\mathbf{u})$.

Examples

Example 1: Consider $Ax = b$. Then $x = F(A, b) = A^{-1}b$. Let $\hat{x} = \mathrm{ALG}(A, b)$.Then

ALG stable $\Longrightarrow \hat{x} = \mathrm{ALG}(A, b) = F(A + \Delta A, b + \Delta b)$, that is, $(A + \Delta A)\hat{x} = b + \Delta b$ such that $\dfrac{\|\Delta A\|}{\|A\|} = \mathcal{O}(\mathbf{u})$ and $\dfrac{\|\Delta b\|}{\|b\|} = \mathcal{O}(\mathbf{u})$.

Example 2: Consider the LU decomposition $A = LU$.

Examples

Example 1: Consider $Ax = b$. Then $x = F(A, b) = A^{-1}b$. Let $\hat{x} = \mathrm{ALG}(A, b)$.Then

ALG stable $\Longrightarrow \hat{x} = \mathrm{ALG}(A, b) = F(A + \Delta A, b + \Delta b)$, that is, $(A + \Delta A)\hat{x} = b + \Delta b$ such that $\dfrac{\|\Delta A\|}{\|A\|} = \mathcal{O}(\mathbf{u})$ and $\dfrac{\|\Delta b\|}{\|b\|} = \mathcal{O}(\mathbf{u})$.

Example 2: Consider the LU decomposition $A = LU$. Let $[L, U] = \mathrm{ALG}(A)$.

Examples

Example 1: Consider $Ax = b$. Then $x = F(A, b) = A^{-1}b$. Let $\hat{x} = \mathrm{ALG}(A, b)$.Then

ALG stable $\Longrightarrow \hat{x} = \mathrm{ALG}(A, b) = F(A + \Delta A, b + \Delta b)$, that is, $(A + \Delta A)\hat{x} = b + \Delta b$ such that $\dfrac{\|\Delta A\|}{\|A\|} = \mathcal{O}(\mathbf{u})$ and $\dfrac{\|\Delta b\|}{\|b\|} = \mathcal{O}(\mathbf{u})$.

Example 2: Consider the LU decomposition $A = LU$. Let $[L, U] = \mathrm{ALG}(A)$. Then ALG stable $\Longrightarrow A + \Delta A = LU$ for some ΔA such that $\|\Delta A\|/\|A\| = \mathcal{O}(\mathbf{u})$.

Examples

Example 1: Consider $Ax = b$. Then $x = F(A, b) = A^{-1}b$. Let $\hat{x} = \mathrm{ALG}(A, b)$. Then

ALG stable $\Longrightarrow \hat{x} = \mathrm{ALG}(A, b) = F(A + \Delta A, b + \Delta b)$, that is, $(A + \Delta A)\hat{x} = b + \Delta b$ such that $\dfrac{\|\Delta A\|}{\|A\|} = \mathcal{O}(\mathbf{u})$ and $\dfrac{\|\Delta b\|}{\|b\|} = \mathcal{O}(\mathbf{u})$.

Example 2: Consider the LU decomposition $A = LU$. Let $[L, U] = \mathrm{ALG}(A)$. Then ALG stable $\Longrightarrow A + \Delta A = LU$ for some ΔA such that $\|\Delta A\|/\|A\| = \mathcal{O}(\mathbf{u})$.

Example 3: Suppose $\mathrm{ALG}(d)$ computes $f(d) = e^d$ for $d \in \mathbb{R}$. Then ALG is stable if $\mathrm{ALG}(d) = f(d + \Delta d) = e^{d+\Delta d}$ and $|\Delta d|/|d| = \mathcal{O}(\mathbf{u})$.

Examples

Example 1: Consider $Ax = b$. Then $x = F(A, b) = A^{-1}b$. Let $\hat{x} = \mathrm{ALG}(A, b)$.Then

ALG stable $\Longrightarrow \hat{x} = \mathrm{ALG}(A, b) = F(A + \Delta A, b + \Delta b)$, that is, $(A + \Delta A)\hat{x} = b + \Delta b$ such that $\dfrac{\|\Delta A\|}{\|A\|} = \mathcal{O}(\mathbf{u})$ and $\dfrac{\|\Delta b\|}{\|b\|} = \mathcal{O}(\mathbf{u})$.

Example 2: Consider the LU decomposition $A = LU$. Let $[L, U] = \mathrm{ALG}(A)$. Then ALG stable $\Longrightarrow A + \Delta A = LU$ for some ΔA such that $\|\Delta A\|/\|A\| = \mathcal{O}(\mathbf{u})$.

Example 3: Suppose $\mathrm{ALG}(d)$ computes $f(d) = e^d$ for $d \in \mathbb{R}$. Then ALG is stable if $\mathrm{ALG}(d) = f(d + \Delta d) = e^{d+\Delta d}$ and $|\Delta d|/|d| = \mathcal{O}(\mathbf{u})$.

Example 4: Consider $F(x) = \sqrt{x+1} - \sqrt{x}$ and $x := 10^5$.

Examples

Example 1: Consider $Ax = b$. Then $x = F(A, b) = A^{-1}b$. Let $\hat{x} = \mathrm{ALG}(A, b)$.Then

ALG stable $\Longrightarrow \hat{x} = \mathrm{ALG}(A, b) = F(A + \Delta A, b + \Delta b)$, that is, $(A + \Delta A)\hat{x} = b + \Delta b$ such that $\dfrac{\|\Delta A\|}{\|A\|} = \mathcal{O}(\mathbf{u})$ and $\dfrac{\|\Delta b\|}{\|b\|} = \mathcal{O}(\mathbf{u})$.

Example 2: Consider the LU decomposition $A = LU$. Let $[L, U] = \mathrm{ALG}(A)$. Then ALG stable $\Longrightarrow A + \Delta A = LU$ for some ΔA such that $\|\Delta A\|/\|A\| = \mathcal{O}(\mathbf{u})$.

Example 3: Suppose $\mathrm{ALG}(d)$ computes $f(d) = e^d$ for $d \in \mathbb{R}$. Then ALG is stable if $\mathrm{ALG}(d) = f(d + \Delta d) = e^{d+\Delta d}$ and $|\Delta d|/|d| = \mathcal{O}(\mathbf{u})$.

Example 4: Consider $F(x) = \sqrt{x+1} - \sqrt{x}$ and $x := 10^5$.

Unstable algorithm: $\mathrm{ALG}(x) = \sqrt{x+1} - \sqrt{x}$.

Examples

Example 1: Consider $Ax = b$. Then $x = F(A, b) = A^{-1}b$. Let $\hat{x} = \mathrm{ALG}(A, b)$.Then

ALG stable $\Longrightarrow \hat{x} = \mathrm{ALG}(A, b) = F(A + \Delta A, b + \Delta b)$, that is, $(A + \Delta A)\hat{x} = b + \Delta b$ such that $\dfrac{\|\Delta A\|}{\|A\|} = \mathcal{O}(\mathbf{u})$ and $\dfrac{\|\Delta b\|}{\|b\|} = \mathcal{O}(\mathbf{u})$.

Example 2: Consider the LU decomposition $A = LU$. Let $[L, U] = \mathrm{ALG}(A)$. Then ALG stable $\Longrightarrow A + \Delta A = LU$ for some ΔA such that $\|\Delta A\|/\|A\| = \mathcal{O}(\mathbf{u})$.

Example 3: Suppose $\mathrm{ALG}(d)$ computes $f(d) = e^d$ for $d \in \mathbb{R}$. Then ALG is stable if $\mathrm{ALG}(d) = f(d + \Delta d) = e^{d+\Delta d}$ and $|\Delta d|/|d| = \mathcal{O}(\mathbf{u})$.

Example 4: Consider $F(x) = \sqrt{x+1} - \sqrt{x}$ and $x := 10^5$.

Unstable algorithm: $\mathrm{ALG}(x) = \sqrt{x+1} - \sqrt{x}$.
In 5-digit decimal arithmetic, MATLAB gives $\mathrm{ALG}(10^5) = 0$.

Examples

Example 1: Consider $Ax = b$. Then $x = F(A, b) = A^{-1}b$. Let $\hat{x} = \mathrm{ALG}(A, b)$.Then

ALG stable $\Longrightarrow \hat{x} = \mathrm{ALG}(A, b) = F(A + \Delta A, b + \Delta b)$, that is, $(A + \Delta A)\hat{x} = b + \Delta b$ such that $\dfrac{\|\Delta A\|}{\|A\|} = \mathcal{O}(\mathbf{u})$ and $\dfrac{\|\Delta b\|}{\|b\|} = \mathcal{O}(\mathbf{u})$.

Example 2: Consider the LU decomposition $A = LU$. Let $[L, U] = \mathrm{ALG}(A)$. Then ALG stable $\Longrightarrow A + \Delta A = LU$ for some ΔA such that $\|\Delta A\|/\|A\| = \mathcal{O}(\mathbf{u})$.

Example 3: Suppose $\mathrm{ALG}(d)$ computes $f(d) = e^d$ for $d \in \mathbb{R}$. Then ALG is stable if $\mathrm{ALG}(d) = f(d + \Delta d) = e^{d+\Delta d}$ and $|\Delta d|/|d| = \mathcal{O}(\mathbf{u})$.

Example 4: Consider $F(x) = \sqrt{x+1} - \sqrt{x}$ and $x := 10^5$.

Unstable algorithm: $\mathrm{ALG}(x) = \sqrt{x+1} - \sqrt{x}$.
In 5-digit decimal arithmetic, MATLAB gives $\mathrm{ALG}(10^5) = 0$.

Stable algorithm: $\mathrm{ALG}(x) = 1/(\sqrt{x+1} + \sqrt{x})$.

Examples

Example 1: Consider $Ax = b$. Then $x = F(A, b) = A^{-1}b$. Let $\hat{x} = \mathrm{ALG}(A, b)$.Then

ALG stable $\Longrightarrow \hat{x} = \mathrm{ALG}(A, b) = F(A + \Delta A, b + \Delta b)$, that is, $(A + \Delta A)\hat{x} = b + \Delta b$ such that $\dfrac{\|\Delta A\|}{\|A\|} = \mathcal{O}(\mathbf{u})$ and $\dfrac{\|\Delta b\|}{\|b\|} = \mathcal{O}(\mathbf{u})$.

Example 2: Consider the LU decomposition $A = LU$. Let $[L, U] = \mathrm{ALG}(A)$. Then ALG stable $\Longrightarrow A + \Delta A = LU$ for some ΔA such that $\|\Delta A\|/\|A\| = \mathcal{O}(\mathbf{u})$.

Example 3: Suppose $\mathrm{ALG}(d)$ computes $f(d) = e^d$ for $d \in \mathbb{R}$. Then ALG is stable if $\mathrm{ALG}(d) = f(d + \Delta d) = e^{d+\Delta d}$ and $|\Delta d|/|d| = \mathcal{O}(\mathbf{u})$.

Example 4: Consider $F(x) = \sqrt{x+1} - \sqrt{x}$ and $x := 10^5$.

Unstable algorithm: $\mathrm{ALG}(x) = \sqrt{x+1} - \sqrt{x}$.
In 5-digit decimal arithmetic, MATLAB gives $\mathrm{ALG}(10^5) = 0$.

Stable algorithm: $\mathrm{ALG}(x) = 1/(\sqrt{x+1} + \sqrt{x})$.

In 5-digit decimal arithmetic, MATLAB gives $\mathrm{ALG}(10^5) = 1.5811 \times 10^{-3}$, the correct value in 5-digit arithmetic. $\blacksquare$.

Accuracy

Backward stability of ALG guarantees
$\mathrm{ALG}(d) = F(d + \Delta d)$ and $\|\Delta d\|/\|d\| = \mathcal{O}(\mathbf{u})$.

Accuracy

Backward stability of ALG guarantees
$\mathrm{ALG}(d) = F(d + \Delta d)$ and $\|\Delta d\|/\|d\| = \mathcal{O}(\mathbf{u})$.

What can be said about

Accuracy

Backward stability of ALG guarantees
$\mathrm{ALG}(d) = F(d + \Delta d)$ and $\|\Delta d\|/\|d\| = \mathcal{O}(\mathbf{u})$.

What can be said about

$$\begin{aligned} \text{Error} \quad &= \quad \frac{\|\mathrm{ALG}(d) - F(d)\|}{\|F(d)\|} = \frac{\|F(d+\Delta d) - F(d)\|}{\|F(d)\|} \\ &= \quad \text{Propagated data error?} \end{aligned}$$

Accuracy

Backward stability of ALG guarantees
$\mathrm{ALG}(d) = F(d + \Delta d)$ and $\|\Delta d\|/\|d\| = \mathcal{O}(\mathbf{u})$.

What can be said about

$$\begin{aligned} \text{Error} &= \frac{\|\mathrm{ALG}(d) - F(d)\|}{\|F(d)\|} = \frac{\|F(d + \Delta d) - F(d)\|}{\|F(d)\|} \\ &= \text{Propagated data error?} \end{aligned}$$

- An algorithm has no control over propagated data error.

Accuracy

Backward stability of ALG guarantees
$\mathrm{ALG}(d) = F(d + \Delta d)$ and $\|\Delta d\|/\|d\| = \mathcal{O}(\mathbf{u})$.

What can be said about

$$\begin{aligned} \text{Error} &= \frac{\|\mathrm{ALG}(d) - F(d)\|}{\|F(d)\|} = \frac{\|F(d+\Delta d) - F(d)\|}{\|F(d)\|} \\ &= \text{Propagated data error?} \end{aligned}$$

- An algorithm has no control over propagated data error.
- Propagated data error is entirely determined by sensitivity of F at d to small perturbation.

Accuracy

Backward stability of ALG guarantees
$\mathrm{ALG}(d) = F(d + \Delta d)$ and $\|\Delta d\|/\|d\| = \mathcal{O}(\mathbf{u})$.

What can be said about

$$\begin{aligned} \text{Error} &= \frac{\|\mathrm{ALG}(d) - F(d)\|}{\|F(d)\|} = \frac{\|F(d+\Delta d) - F(d)\|}{\|F(d)\|} \\ &= \text{Propagated data error?} \end{aligned}$$

- An algorithm has no control over propagated data error.
- Propagated data error is entirely determined by sensitivity of F at d to small perturbation.
- Analysis of propagated data error is a part of Perturbation Theory.

Condition number

- A problem is said to be sensitive or ill-conditioned at d if a small relative change in d can cause a large relative change in the solution $F(d)$.

Condition number

- A problem is said to be sensitive or ill-conditioned at d if a small relative change in d can cause a large relative change in the solution $F(d)$. That is,

$$\frac{\|F(d+\Delta d)-F(d)\|}{\|F(d)\|}$$

can be very large compared to $\dfrac{\|\Delta d\|}{\|d\|}$.

Condition number

- A problem is said to be sensitive or ill-conditioned at d if a small relative change in d can cause a large relative change in the solution $F(d)$. That is,

$$\frac{\|F(d+\Delta d) - F(d)\|}{\|F(d)\|}$$

can be very large compared to $\dfrac{\|\Delta d\|}{\|d\|}$.

- For small changes in d, the condition number

$$\mathrm{cond}_F(d) := \max\left(\frac{\text{relative change in solution}}{\text{relative change in input data}}\right)$$

Condition number

- A problem is said to be sensitive or ill-conditioned at d if a small relative change in d can cause a large relative change in the solution $F(d)$. That is,

$$\frac{\|F(d+\Delta d) - F(d)\|}{\|F(d)\|}$$

can be very large compared to $\dfrac{\|\Delta d\|}{\|d\|}$.

- For small changes in d, the condition number

$$\mathrm{cond}_F(d) := \max\left(\frac{\text{relative change in solution}}{\text{relative change in input data}}\right)$$

$$= \mathrm{limsup}_{\epsilon\to 0}\left(\frac{\|F(d+\Delta d) - F(d)\|}{\epsilon\|F(d)\|} : \frac{\|\Delta d\|}{\|d\|} \leq \epsilon\right)$$

provides a measure of sensitivity of $F(d)$ at d.

Ill-conditioning

- For small relative changes in d we have

$$\frac{\|F(d+\Delta d) - F(d)\|}{\|F(d)\|} \lesssim \mathrm{cond}_F(d) \frac{\|\Delta d\|}{\|d\|}$$

Ill-conditioning

- For small relative changes in d we have

$$\frac{\|F(d+\Delta d) - F(d)\|}{\|F(d)\|} \lesssim \mathrm{cond}_F(d)\frac{\|\Delta d\|}{\|d\|}$$

$$\begin{pmatrix} \text{Propagated data} \\ \text{error} \end{pmatrix} \lesssim \text{cond.} \times \begin{pmatrix} \text{Error in} \\ \text{data} \end{pmatrix}$$

Ill-conditioning

- For small relative changes in d we have

$$\frac{\|F(d+\Delta d) - F(d)\|}{\|F(d)\|} \lesssim \mathrm{cond}_F(d) \frac{\|\Delta d\|}{\|d\|}$$

$$\begin{pmatrix} \text{Propagated data} \\ \text{error} \end{pmatrix} \lesssim \text{cond.} \times \begin{pmatrix} \text{Error in} \\ \text{data} \end{pmatrix}$$

- Thus $F(d)$ is ill-conditioned if $\mathrm{cond}_F(d) \gg 1$. Otherwise, the problem is well-conditioned.

Ill-conditioning

- For small relative changes in d we have

$$\frac{\|F(d+\Delta d)-F(d)\|}{\|F(d)\|} \lesssim \mathrm{cond}_F(d)\frac{\|\Delta d\|}{\|d\|}$$

$$\begin{pmatrix}\text{Propagated data}\\ \text{error}\end{pmatrix} \lesssim \text{cond.} \times \begin{pmatrix}\text{Error in}\\ \text{data}\end{pmatrix}$$

- Thus $F(d)$ is ill-conditioned if $\mathrm{cond}_F(d) \gg 1$. Otherwise, the problem is well-conditioned.

- How large $\mathrm{cond}_F(d)$ is large enough? The answer depends on how choosy you are!

- If $\mathrm{cond}_F(d) = 10^s$ then s digits may be lost in the solution computed by a stable algorithm.

Estimating the condition number

If F is differentiable at d then

Estimating the condition number

If F is differentiable at d then

$$\mathrm{cond}_F(d) \simeq \frac{\|J_F(d)\| \, \|d\|}{\|F(d)\|},$$

Estimating the condition number

If F is differentiable at d then

$$\mathrm{cond}_F(d) \simeq \frac{\|J_F(d)\|\,\|d\|}{\|F(d)\|},$$

where $J_F(d) = \left[\dfrac{\partial F_i}{\partial x_j}(d)\right]$ is the Jacobian of F at d.

Estimating the condition number

If F is differentiable at d then

$$\text{cond}_F(d) \simeq \frac{\|J_F(d)\| \, \|d\|}{\|F(d)\|},$$

where $J_F(d) = \left[\dfrac{\partial F_i}{\partial x_j}(d)\right]$ is the Jacobian of F at d.

Example: Consider $F(d) = \sqrt{d}$. Then $J_F(d) = F'(d) = 1/2\sqrt{d}$, for $d \neq 0$ and $\mathrm{cond}_F(d) = 1/2$. ■

Estimating the condition number

If F is differentiable at d then

$$\mathrm{cond}_F(d) \simeq \frac{\|J_F(d)\| \, \|d\|}{\|F(d)\|},$$

where $J_F(d) = \left[\dfrac{\partial F_i}{\partial x_j}(d)\right]$ is the Jacobian of F at d.

Example: Consider $F(d) = \sqrt{d}$. Then $J_F(d) = F'(d) = 1/2\sqrt{d}$, for $d \neq 0$ and $\mathrm{cond}_F(d) = 1/2$. ■

Example: Consider $F(d_1, d_2) = d_1 - d_2$. Then $J_F(d) = [1, -1]$ and

$$\mathrm{cond}_F(d) = \frac{2\|d\|_\infty}{|d_1 - d_2|}.$$

Estimating the condition number

If F is differentiable at d then

$$\text{cond}_F(d) \simeq \frac{\|J_F(d)\| \, \|d\|}{\|F(d)\|},$$

where $J_F(d) = \left[\frac{\partial F_i}{\partial x_j}(d)\right]$ is the Jacobian of F at d.

Example: Consider $F(d) = \sqrt{d}$. Then $J_F(d) = F'(d) = 1/2\sqrt{d}$, for $d \neq 0$ and $\mathrm{cond}_F(d) = 1/2$. ■

Example: Consider $F(d_1, d_2) = d_1 - d_2$. Then $J_F(d) = [1, -1]$ and

$$\mathrm{cond}_F(d) = \frac{2\|d\|_\infty}{|d_1 - d_2|}.$$

For $d_1 := 1$, and $d_2 := 1 - 10^{-5}$, $\mathrm{cond}_F(d) = 2 \times 10^5$. ■

Example

Propagated data error for $\tan(x)$ near $\pi/2$.

Example

Propagated data error for $\tan(x)$ near $\pi/2$.

Consider $x := .15707 \times 10^1$ and $x + \delta x := .15709 \times 10^1$.

Example

Propagated data error for $\tan(x)$ near $\pi/2$.

Consider $x := .15707 \times 10^1$ and $x + \delta x := .15709 \times 10^1$.

Then error in the data

$$|\delta x|/|x| = 1.2733 \times 10^{-4}.$$

Example

Propagated data error for $\tan(x)$ near $\pi/2$.

Consider $x := .15707 \times 10^1$ and $x + \delta x := .15709 \times 10^1$.

Then error in the data
$$|\delta x|/|x| = 1.2733 \times 10^{-4}.$$

We have $\tan(x + \delta x) = -9.6457 \times 10^3$, $\tan(x) = 1.0381 \times 10^4$ and

Example

Propagated data error for $\tan(x)$ near $\pi/2$.

Consider $x := .15707 \times 10^1$ and $x + \delta x := .15709 \times 10^1$.

Then error in the data

$$|\delta x|/|x| = 1.2733 \times 10^{-4}.$$

We have $\tan(x + \delta x) = -9.6457 \times 10^3$, $\tan(x) = 1.0381 \times 10^4$ and

$$|\tan(x + \delta x) - \tan(x)|/\tan(x) = 1.9291$$

Example

Propagated data error for $\tan(x)$ near $\pi/2$.

Consider $x := .15707 \times 10^1$ and $x + \delta x := .15709 \times 10^1$.

Then error in the data

$$|\delta x|/|x| = 1.2733 \times 10^{-4}.$$

We have $\tan(x + \delta x) = -9.6457 \times 10^3$, $\tan(x) = 1.0381 \times 10^4$ and

$$|\tan(x + \delta x) - \tan(x)|/\tan(x) = 1.9291$$

Thus error in the data is magnified 10^4 times in the solution.

Example

Propagated data error for $\tan(x)$ near $\pi/2$.

Consider $x := .15707 \times 10^1$ and $x + \delta x := .15709 \times 10^1$.

Then error in the data

$$|\delta x|/|x| = 1.2733 \times 10^{-4}.$$

We have $\tan(x + \delta x) = -9.6457 \times 10^3$, $\tan(x) = 1.0381 \times 10^4$ and

$$|\tan(x + \delta x) - \tan(x)|/\tan(x) = 1.9291$$

Thus error in the data is magnified 10^4 times in the solution.

We have

$$\mathrm{cond}_{\tan}(x) = 1.6306 \times 10^4.$$

Back to accuracy

$$\text{Backward stability of } \mathrm{ALG}$$
$$\Downarrow$$

Back to accuracy

$$\text{Backward stability of ALG}$$
$$\Downarrow$$
$$\mathrm{ALG}(d) = F(d + \Delta d) \text{ and } \|\Delta d\|/\|d\| = \mathcal{O}(\mathbf{u})$$
$$\Downarrow$$

Back to accuracy

$$\text{Backward stability of } \mathrm{ALG}$$
$$\Downarrow$$
$$\mathrm{ALG}(d) = F(d + \Delta d) \text{ and } \|\Delta d\|/\|d\| = \mathcal{O}(\mathbf{u})$$
$$\Downarrow$$

$$\begin{aligned} \text{Error} &= \frac{\|\mathrm{ALG}(d) - F(d)\|}{\|F(d)\|} = \frac{\|F(d+\Delta d) - F(d)\|}{\|F(d)\|} \\ &\lesssim \mathrm{cond}_F(d) \frac{\|\Delta d\|}{\|d\|}. \end{aligned}$$

Back to accuracy

$$\text{Backward stability of } \mathrm{ALG}$$
$$\Downarrow$$
$$\mathrm{ALG}(d) = F(d + \Delta d) \text{ and } \|\Delta d\| / \|d\| = \mathcal{O}(\mathbf{u})$$
$$\Downarrow$$

$$\begin{aligned} \text{Error} &= \frac{\|\mathrm{ALG}(d) - F(d)\|}{\|F(d)\|} = \frac{\|F(d+\Delta d) - F(d)\|}{\|F(d)\|} \\ &\lesssim \mathrm{cond}_F(d) \frac{\|\Delta d\|}{\|d\|}. \end{aligned}$$

Error $\lesssim$ cond.$\times$ Backward error.

Summary

- Accuracy refers to closeness of computed solution to the exact solution.

Summary

- Accuracy refers to closeness of computed solution to the exact solution.

- Stability of an algorithm alone cannot guarantee accuracy.

Summary

- Accuracy refers to closeness of computed solution to the exact solution.

- Stability of an algorithm alone cannot guarantee accuracy.

- Accuracy of solution produced by automated computation depends on three major factors:
  - properties of finite precision arithmetic
  - backward stability of the algorithm.
  - conditioning of the computational problem - sensitivity of solutions to small perturbations

Summary

- Accuracy refers to closeness of computed solution to the exact solution.

- Stability of an algorithm alone cannot guarantee accuracy.

- Accuracy of solution produced by automated computation depends on three major factors:

  - properties of finite precision arithmetic
  - backward stability of the algorithm.
  - conditioning of the computational problem - sensitivity of solutions to small perturbations

  Thus

$$\text{Accuracy} = \text{function(stability, conditioning, floating-point arithmetic)}$$

Summary

- Accuracy refers to closeness of computed solution to the exact solution.

- Stability of an algorithm alone cannot guarantee accuracy.

- Accuracy of solution produced by automated computation depends on three major factors:
  - properties of finite precision arithmetic
  - backward stability of the algorithm.
  - conditioning of the computational problem - sensitivity of solutions to small perturbations

  Thus

  $$\text{Accuracy} = \text{function(stability, conditioning, floating-point arithmetic)}$$

- A stable algorithm will compute an accurate solution to a well-conditioned problem.

Summary

- Accuracy refers to closeness of computed solution to the exact solution.
- Stability of an algorithm alone cannot guarantee accuracy.
- Accuracy of solution produced by automated computation depends on three major factors:
  - properties of finite precision arithmetic
  - backward stability of the algorithm.
  - conditioning of the computational problem - sensitivity of solutions to small perturbations

  Thus

  $$\text{Accuracy} = \text{function(stability, conditioning, floating-point arithmetic)}$$

- A stable algorithm will compute an accurate solution to a well-conditioned problem.
- Inaccuracy can result from numerical instability as well as from ill-conditioning.