ERRORS AND SENSITIVITY IN SCIENTIFIC COMPUTING

Dr. Jitendra Kumar

Professor
Department of Mathematics
Indian Institute of Technology ROPAR

Let $\tilde{x}$ be an approximation of value x. Then the absolute and relatives errors are defined as

$$E_{abs} = |x - \tilde{x}| \qquad E_{rel} = \left|\frac{x-\tilde{x}}{x}\right|$$

Example 1: $x = 10^{16}$ and $\tilde{x} = 1.001 \times 10^{16}$

$$E_{abs} = 0.001 \times 10^{16} = 10^{13}$$

$$E_{rel} = \frac{10^{13}}{10^{16}} = 10^{-3}$$

Example 2: $x = 10^{2}$ and $\tilde{x} = 1.001 \times 10^{2}$

$$E_{abs} = 0.001 \times 10^{2} = 0.1$$

$$E_{rel} = \frac{0.1}{10^{2}} = 10^{-3}$$

Example 3: $x = 1000$ and $\tilde{x} = 1000.1$

$$E_{abs} = 0.1$$

$$E_{rel} = \frac{0.1}{1000} = 10^{-4} = 0.01\%$$

Example 4: $x = 1$ and $\tilde{x} = 1.1$

$$E_{abs} = 0.1$$

$$E_{rel} = \frac{0.1}{1} = 0.1 = 10\%$$

Physical Problem → Mathematical Model → Numerical Approximation

SOURCE OF ERRORS

- Error in initial data/parameters
- Errors in Mathematical Model
- **Round-off Error**
  - Computer's inability to represent numbers exactly (chopping and rounding)
  - unavoidable error
- **Truncation Error**
  - Approximations due to discretization or truncation of Series
  - Controllable

Illustrative Example: Consider $x = 1.23456$ and $y = 1.22222$

If x and y and approximated by their first three digit, i.e., $\hat{x} = 1.23$ and $\hat{y} = 1.22$

$$\hat{x} - \hat{y} = 1.23 - 1.22 = 0.01$$

Note that error in x and y was less than 1%.

Compute error in $x - y$:

$$\left|\frac{0.01234 - 0.01}{0.01234}\right| \approx 20\%$$

Real-World Disasters Due to Rounding Errors

1. **Failure of U.S. Anti-Missile Defense System (Gulf War 1991)**
   - US missile defense system tried to stop incoming Scud missiles.
   - Small **rounding error** in representing 0.1 seconds in computer memory.
   - Over **several hours**, the error added up (to ~0.34 seconds) → system miscalculated missile position (~500 meters).
   - Impact: **Missed missile hit a barracks → 28 soldiers killed**.

2. **Vancouver Stock Exchange Bug (1982)**
   - Software used to calculate stock prices had **rounding errors**.
   - Magnitude of errors: Index was **truncated** to **three decimal places** for each trade.
   - Over hundreds of thousands of trades, the small errors added up to **several thousand dollars** lost per trader
   - Impact: **Financial discrepancies** across the exchange → **losses for investors and trading firms**.

Takeaway: Even tiny numerical errors in computers can accumulate and cause **serious real-world** problems, from safety disasters to financial losses.

Round-off Error

Let $x = 2, y = 2, z = \sqrt{2} * 10^{-15}$

Evaluate $(y + z) - x$

```
>> x=2;
>> y=2;
>> z=sqrt(2)*1e-15;
>> (y+z)-x

ans =

   1.3323e-15

>> z

z =

   1.4142e-15

>>
```

$\approx 6\%$ error

Let $x = 10^8$

Evaluate $\dfrac{1}{\sqrt{x^2+1} - x}$

```
>> x=1e+8;
>> 1/(sqrt(x^2+1)-x);
>> 1/(sqrt(x^2+1)-x)

ans =

   Inf
```

- Accumulation of roundoff error while subtracting numbers of similar magnitude
- Accumulation of roundoff error while adding large and small (in magnitude) number
- If $|y| \ll 1$ then x/y may accumulate large roundoff error
- If $|y| \gg 1$ then xy may accumulate large roundoff error
- Overflow/underflow: number is too large or too small to fit into the floating point system

Round-off Error (Some Fixes)

Let $x = 2, y = 2, z = \sqrt{2} * 10^{-15}$

Instead of Evaluating $(y + z) - x$

Evaluate $(y - x)$ +z

```
>> x=2;
>> y=2;
>> z=sqrt(2)*1e-15;
>> (y-x)+z

ans =

   1.4142e-15

>> z

z =

   1.4142e-15

>>
```

Let $x = 10^8$

Instead of Evaluating $\frac{1}{\sqrt{x^2+1} - x}$

Evaluate $\sqrt{x^2+1} + x \approx 2 \times 10^8$

```
>> x=1e8;
>> sqrt(x^2+1)+x

ans =

   200000000

>>
```

Truncation Error

Results from approximation of an exact mathematical expression, e.g., truncation of infinite series, Discretization of ODE/PDE, Finite Differences, etc.

Example: Taylor's polynomials are approximations of some functions

Taylor's Formula

Taylor's Polynomial

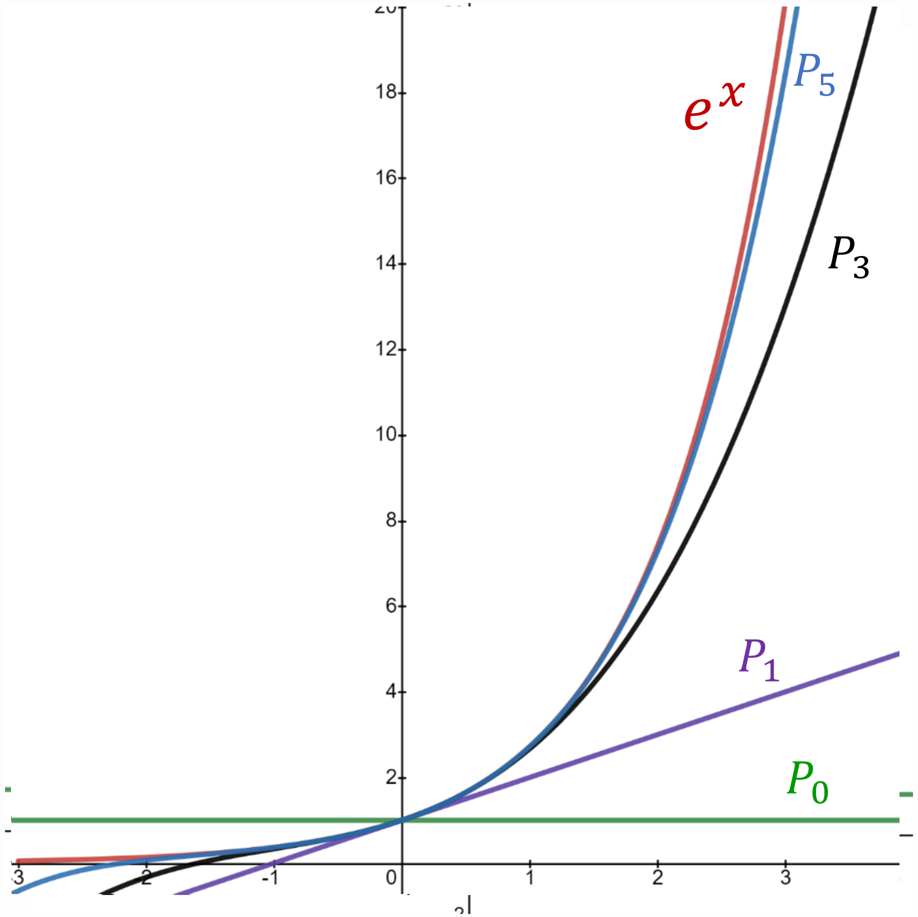
$$f(x) = \overbrace{f(x_0) + f'(x_0)(x - x_0) + \cdots + \frac{f^{(n)}(x_0)}{n!}(x - x_0)^n}^{\text{Taylor's Polynomial}} + R_n$$

Remainder: $R_n = \dfrac{f^{(n+1)}(\xi)}{(n+1)!}(x - x_0)^{n+1},\ x_0 < \xi < x$ Truncation Error

$$P_n(x) = f(x_0) + f'(x_0)(x - x_0) + \frac{f''(x_0)}{2!}(x - x_0)^2 + \cdots + \frac{f^{(n)}(x_0)}{n!}(x - x_0)^n$$

Taylor's Polynomial of order n

Example: Taylor's Polynomial of e^x around $x = 0$.



$$P_5(x) = 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24} + \frac{x^5}{120}$$

$$P_4(x) = 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + \frac{x^4}{24};$$

$$P_3(x) = 1 + x + \frac{x^2}{2} + \frac{x^3}{6}$$

$$P_2(x) = 1 + x + \frac{x^2}{2};$$

$P_0(x) = 1;$ $\quad P_1(x) = 1 + x;$

Combined effect of Truncation error and Round-off error

Consider $y = x^4 - x^3 - x^2 - x + 1;$

Its derivative $\frac{dy}{dx} = 4x^3 - 3x^2 - 2x - 1;$

Approximation of the derivative:

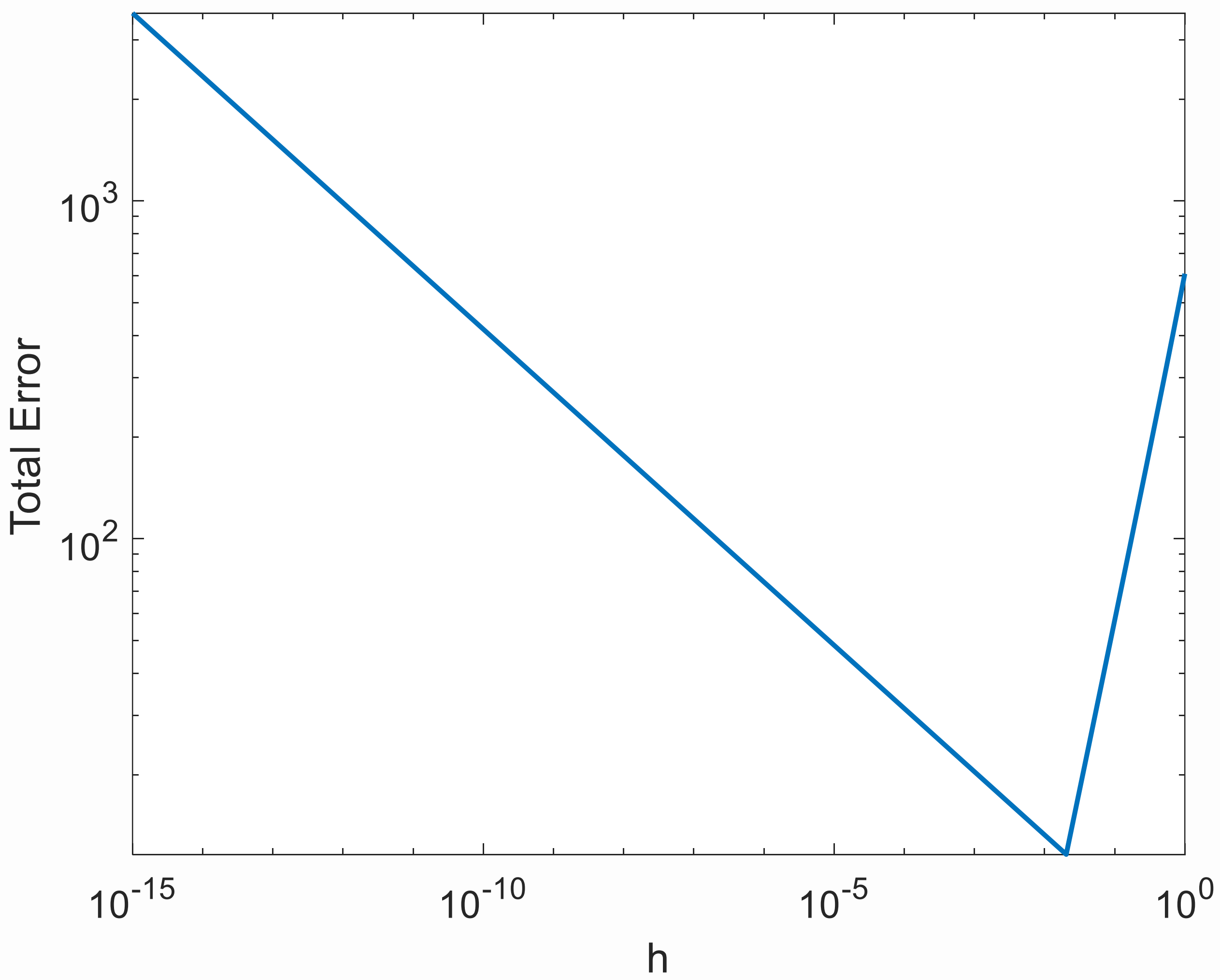
Forward difference

$$\frac{dy}{dx} \approx \frac{y(x+h) - y(x)}{h} \approx \frac{\tilde{y}(x+h) - \tilde{y}(x)}{h}$$

Truncation error   round off error

$$\text{Total Error} = \left| \frac{dy}{dx} - \left(\frac{\tilde{y}(x+h) - \tilde{y}(x)}{h} \right) \right|$$

Total error $x = 10$



Combined effect of Truncation error and Round-off error

Using Taylor's theorem, we have

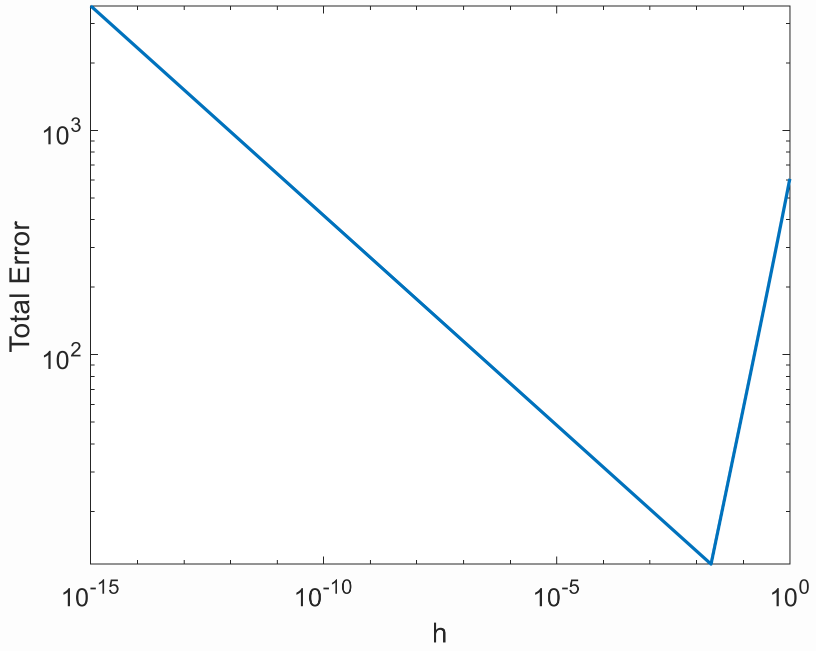
$$y(x_{i+1}) = y(x_i) + (x_{i+1} - x_i)y'(x_i) + \frac{(x - x_i)^2}{2} f''(\xi) \qquad \Rightarrow y'(x_i) = \frac{y(x_{i+1}) - y(x_i)}{h} - \frac{h}{2} f''(\xi)$$

In computer, let $y(x)$ be approximated (round off error) by $\tilde{y}(x)$, i.e.,

$$y(x_{i+1}) = \tilde{y}(x_{i+1}) + e_{i+1} \qquad \& \qquad y(x_i) = \tilde{y}(x_i) + e_i$$

$$\Rightarrow y'(x_i) = \frac{\tilde{y}(x_{i+1}) - \tilde{y}(x_i)}{h} + \frac{e_{i+1} - e_i}{h} - \frac{h}{2} f''(\xi)$$

True Value — Finite Difference Approximation — Round off Error — Truncation Error



Assuming $e_i \leq \epsilon$, for all i and $f''(\xi) \leq M$, then the Total Error is given by

$$\Rightarrow \left| y'(x_i) - \frac{\tilde{y}(x_{i+1}) - \tilde{y}(x_i)}{h} \right| \leq \frac{2\epsilon}{h} + \frac{M}{2} h$$

Condition Number of a Function

Consider $f(x) = \dfrac{x}{1 - x}$

x	$f(x)$
0.97	32.3333
0.98	49.0000
-0.97	-0.4924
-0.98	-0.4949

About 1% error in x leads to more than 51% error in $f(x)$ (rows 0.97, 0.98)

About 1% error in x leads to more than 0.50% error in $f(x)$ (rows -0.97, -0.98)

The **condition number** of a function measures how much the output value of the function can change for a small change in the input argument.

Condition Number of a Function

Let $\tilde{x}$ be an approximation of x. (OR $\tilde{x}$ is a perturbed value of x)

We want to see the effect of the discrepancy between $\tilde{x}$ and x on the value of the function.

Using Taylor's series (Mean value theorem): $f(x) = f(\tilde{x}) + (x - \tilde{x})f'(\xi)$

$$\frac{f(x) - f(\tilde{x})}{f(x)} = \frac{f'(\xi)}{f(x)}(x - \tilde{x}) \Rightarrow \frac{f(x) - f(\tilde{x})}{f(x)} \approx \frac{xf'(x)}{f(x)} \frac{(x - \tilde{x})}{x}$$

Change in output — condition number — change in input

The condition number of evaluation of f at the point x is a measure of the ratio of the relative change in a function $f(x)$ to the relative change in x.

Sensitivity of Mathematical Models

$$\text{Condition Number} = \frac{xf'(x)}{f(x)} = \frac{1}{1-x}$$

$$\frac{f(x) - f(\tilde{x})}{f(x)} \approx \frac{xf'(x)}{f(x)} \frac{(x - \tilde{x})}{x}$$

Change in output — condition number — change in input

$$f(x) = \frac{x}{1-x}$$

x	$f(x)$
0.97	32.3333
0.98	49.0000
-0.97	-0.4924
-0.98	-0.4949

$$\text{Condition Number}\Big|_{x=0.98} = \frac{1}{1-x} \approx 50$$

About 1% error in x leads to more than 51% error in $f(x)$

About 1% error in x leads to more than 0.50% error in $f(x)$

$$\text{Condition Number}\Big|_{x=-0.98} = \frac{1}{1-x} \approx 0.50$$

Example: Evaluate $f(x) = \sqrt{x^2+1} - x$ for $x = 10^8$

Condition number of $f: \approx -1$ Problem is well conditioned

Computing steps:

1. Compute $x^2 + 1 =: t_1$ — Well conditioned $k \approx 2$
2. Compute $\sqrt{t_1} =: t_2$ — Well conditioned $k \approx \frac{1}{2}$
3. Compute $t_2 - x$ — Ill conditioned $k \gg 1$

Problem is well-conditioned but the obvious algorithm used to evaluate it is unstable.

Different algorithm must be used for evaluating the original expression.

$$f(x) = \sqrt{x^2+1} - x = \frac{1}{\sqrt{x^2+1} + x}$$

3. Compute $t_2 + x \coloneqq t_3$ — Well conditioned $k \approx \frac{1}{2}$
4. Compute $\frac{1}{t_3}$ — Well conditioned $k \approx 1$

Takeaways

Rounding errors are unavoidable → but understanding them prevents catastrophic mistakes.

Conditioning tells us about the **problem**:

- Well-conditioned → input errors don't grow much.
- Ill-conditioned → small input errors get magnified.

Stability tells us about the **algorithm**:

- Stable → controls rounding/perturbation errors.
- Unstable → errors grow during computation.

Key relationship:

- *Well-conditioned problem + stable algorithm → reliable results.*
- *Ill-conditioned problem → unreliable results, regardless of algorithm.*

Preconditioning: transform a hard (ill-conditioned) system into an easier (well-conditioned one).

Numerical reliability comes from **understanding** both the **problem** (conditioning) and the **method** (stability).

Good algorithms can control errors, but they cannot change the nature of the problem itself.

ITERATIVE METHOD

A method for solving the linear system $Ax = b$ is called ***iterative*** if it is a ***numerical method*** computing a ***sequence of approximate solutions*** $x^{(k)}$ that converges to the exact solution x as the number of iterations k goes to ∞.

DEFINITION (Convergence of an Iterative Method):

An iterative method is said to **converge** if for any choice of initial vector $x^{(0)} \in \mathbb{R}^n$, the sequence of approximate solutions $x^{(k)}$ converges to the exact solution x, i.e., $\lim_{k\to\infty} x^{(k)} = x$.

DEFINITION (Residual/Error):

We call the vector $r_k = b - Ax^{(k)}$ **residual** (respectively **error** $e_k = x^{(k)} - x$) at the kth iteration.

REMARK:

In general, we have no knowledge of e_k because the exact solution x is unknown. However, it is easy to compute the residual r_k, so the convergence is usually dedicated on the residual in practice.

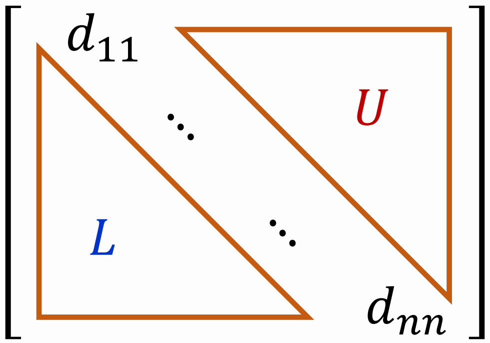
Consider a system of linear equations $A_{n\times n}x_{n\times 1} = b_{n\times 1}$

Jacobi Iteration Method

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1, j\neq i}^{n} a_{ij}x_j^{(k)}\right)$$

$$x^{(k+1)} = -D^{-1}(L+U)\,x^{(k)} + D^{-1}b$$

$A =$



L: Lower triangular part of A

D : Diagonal entries of A

U : Upper triangular part of A

Gauss Seidel Method

$$x_i^{(k+1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j<i} a_{ij}x_j^{(k+1)} - \sum_{j>i} a_{ij}x_j^{(k)}\right)$$

$$x^{(k+1)} = -(D+L)^{-1}Ux^{(k)} + (D+L)^{-1}b$$

Iterative Methods: $x^{(k+1)} = Gx^{(k)} + Hb$

Necessary and Sufficient Conditions:

The iterative methods converge for any initial guess if and only if **all the eigenvalues** of the iteration matrix G have absolute value **less than 1**.

OR

The iterative methods converge if and only if the **spectral radius** (largest absolute eigenvalue) of G is less than 1, i.e., $\rho(G) < 1$.

Sufficient Conditions :

1. If any **norm of iteration matrix G** is less than 1, i.e. $\|G\| < 1$, then the iterative methods converge for any initial guess.

2. If A is **strictly diagonally dominant** by rows (or by columns) then the Jacobi and Gauss-Seidel methods converge for any initial guess.

Problem 1: Solve the system of linear equations: $0.835x_1 + 0.667x_2 = 0.168$

$0.333x_1 + 0.266x_2 = 0.067$

Jacobi

Iterations	x_1	x_2
1	1	1
2	-0.59760479	-1
3	1	1.000009
4	-0.597611983	-1
5	1	1.00001801
6	-0.597619176	-1
⋮	⋮	⋮
9996	-0.633954765	-1
9997	1	1.0455148
9998	-0.633962121	-1
9999	1	1.04552401
10000	-0.633969478	-1

Gauss-Seidel

Iterations	x_1	x_2
1	1	1
2	-0.5976	1.000009
3	-0.59761	1.000018
4	-0.59762	1.000027
5	-0.59763	1.000036
6	-0.59763	1.000045
⋮	⋮	⋮
9996	-0.67113	1.092056
9997	-0.67114	1.092065
9998	-0.67115	1.092075
9999	-0.67115	1.092084
10000	-0.67116	1.092094

Problem 2: Solve the system of linear equations: $0.835x_1 + 0.667x_2 = 0.168$

$$0.333x_1 + 0.266x_2 = 0.066$$

Jacobi

Iterations	x_1	x_2
1	1	1
2	-0.5976	-1.00376
3	1.003003	0.99625
4	-0.59461	-1.00752
5	1.006006	0.992499
6	-0.59161	-1.01128
⋮	⋮	⋮
9996	14.54216	-20.0024
9997	16.17919	-17.9569
9998	14.54522	-20.0063
9999	16.18226	-17.9607
10000	14.54829	-20.0101

Gauss-Seidel

Iterations	x_1	x_2
1	1	1
2	-0.5976	0.99625
3	-0.59461	0.992499
4	-0.59161	0.988749
5	-0.58862	0.984998
6	-0.58562	0.981248
⋮	⋮	⋮
9996	30.0264	-37.3413
9997	30.02953	-37.3452
9998	30.03266	-37.3492
9999	30.0358	-37.3531
10000	30.03893	-37.357

Problem 3: Solve the system of linear equations: $0.835x_1 + 0.667x_2 = 0.168$

$$0.333x_1 + 0.265x_2 = 0.068$$

Jacobi

Iterations	x_1	x_2
1	1	1
2	-0.5976	-1
3	1	1.007556
4	-0.60364	-1
5	1	1.015141
6	-0.6097	-1
⋮	⋮	⋮
2996	1	$3.05E+08$
2997	$-2.4E+08$	-1
2998	1	$3.06E+08$
2999	$-2.4E+08$	-1
3000	1	$3.08E+08$

Gauss-Seidel

Iterations	x_1	x_2
1	1	1
2	-0.5976	0.992518
3	-0.59163	0.985065
4	-0.58567	0.977639
5	-0.57974	0.970241
6	-0.57383	0.962871
⋮	⋮	⋮
2996	$-3.72E+16$	$4.68E+16$
2997	$-3.74E+16$	$4.70E+16$
2998	$-3.75E+16$	$4.71E+16$
2999	$-3.76E+16$	$4.73E+16$
3000	$-3.78E+16$	$4.75E+16$

Key Observations:

Underlined System of Linear Equations

$$0.835x_1 + 0.667x_2 = 0.168$$
$$0.333x_1 + 0.266x_2 = 0.067$$

Exact Solution: $(1, -1)$

$$0.835x_1 + 0.667x_2 = 0.168$$
$$0.333x_1 + 0.266x_2 = 0.066$$

Exact Solution: $(-666, 834)$

1. System is very sensitive
   - How to measure?

2. Convergence of iterative schemes
   - How to check?

$$0.835x_1 + 0.667x_2 = 0.168$$
$$0.333x_1 + 0.265x_2 = 0.068$$

Exact Solution: $(1, -1)$

J: $(1, 3.08E + 08)$ — $\rho(A) = 1.0019$

GS: $(-3.78E + 16, 4.75E + 16)$ — $\rho(A) = 1.0038$

Underlined System of Linear Equations

$0.835x_1 + 0.667x_2 = 0.168$	$0.835x_1 + 0.667x_2 = 0.168$
$0.333x_1 + 0.266x_2 = 0.067$	$0.333x_1 + 0.266x_2 = 0.066$
Exact Solution: $(1, -1)$	Exact Solution: $(-666, 834)$
GS Approximation: $(-0.6711, 1.0920)$	GS Approximation: $(-30.0389, -37.357)$

Residials: $b - Ax^{(k)}$

$$\begin{bmatrix} -5.81 \times 10^{-5} \\ -2.0704 \times 10^{-5} \end{bmatrix} \qquad \begin{bmatrix} -2.4750 \times 10^{-6} \\ -1.0000 \times 10^{-4} \end{bmatrix}$$

3. Residuals are small but the approximation is weird
   - Where is the problem?

Vector Norm: Let $x, y \in \mathbb{R}^n$. The norm of a vector is number that measures "size" or "length" of a vector. It satisfies

(i) $\|x\| > 0$ for $x \neq 0$ and $\|x\| = 0$ for $x = 0$

(ii) $\|\lambda x\| = |\lambda|\|x\|, \quad \forall\, \lambda \in \mathbb{R}$

(iii) $\|x + y\| \leq \|x\| + \|y\|$

Examples:

- The p-norm of the vector $x = (x_1, x_2, \dots, x_n)^T \in \mathbb{R}^n$ is defined as

$$\|x\|_p = \left(\sum_{i=1}^{n} |x_i|^p\right)^{\frac{1}{p}} \qquad (p = 2:\ \text{Euclidean Norm})$$

- The ∞-norm of the vector $x = (x_1, x_2, \dots, x_n)^T \in \mathbb{R}^n$ is defined as

$$\|x\|_\infty = \max_{1 \leq i \leq n} |x_i|$$

Matrix Norm: A number associated with a matrix that is often requires in analysis of Matrix based algorithm.

Matrix norms give some notion of "size" of a matrix or "distance" between the two matrices.

Let $A, B \in \mathbb{R}^{n\times n}$. Similar to vector norm, matrix norm also satisfies the following properties

(i) $\|A\| > 0$ for $A \neq 0$ and $\|A\| = 0$ for $A = 0$

(ii) $\|\lambda A\| = |\lambda|\|A\|, \quad \forall\, \lambda \in \mathbb{R}$

(iii) $\|A + B\| \leq \|A\| + \|B\|$

Examples:

- The *Frobenius* norm

$$\|A\|_F = \left(\sum_{i,j} |a_{ij}|^2\right)^{1/2}$$

- The *max* norm

$$\|A\|_{\max} = \max_{i,j} |a_{ij}|$$

Def. For any vector norm, we can also define a corresponding matrix norm (called **induced matrix norm**) as

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

Def. We say that a matrix norm $\|\cdot\|$ is consistent (**compatible**) with a vector norm if

$$\|Ax\| \le \|A\|\|x\|, \qquad \forall x \in \mathbb{R}^n$$

Def. We say that a matrix norm $\|\cdot\|$ is **sub-multiplicative** if $\forall\ A, B \in \mathbb{R}^{n\times n}$

$$\|AB\| \le \|A\|\|B\|$$

Note: All norms do not satisfy compatibility and sub-multiplicative properties. However, all **induced matrix norms** and **Frobenius norm** satisfy these properties.

Example: Consider $\|A\|_{\max} = \max_{i,j} |a_{ij}|$, $\quad A = B = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}, x = \begin{bmatrix} 1 \\ 1 \end{bmatrix} \quad \Rightarrow AB = \begin{bmatrix} 2 & 2 \\ 2 & 2 \end{bmatrix}$ and $Ax = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$

$\|AB\|_{\max} = 2 > 1 = \|A\|_{\max}\|B\|_{\max}$ (Not sub-multiplicative)

$\|Ax\|_{\max} = 2 > 1 = \|A\|_{\max}\ \|x\|_{\max}$ (Not compatible)

Simplified forms of induced matrix norm: $\|A\| = \max_{x\neq 0} \frac{\|Ax\|}{\|x\|}$

The matrix norm induced by the vector 1-norm:

$$\|Ax\|_1 = \sum_{i=1}^{n} |(Ax)_i| = \sum_{i=1}^{n} \left| \sum_{j=1}^{n} a_{ij} x_j \right| \le \sum_{i=1}^{n} \sum_{j=1}^{n} |a_{ij}|\, |x_j| = \sum_{j=1}^{n} \sum_{i=1}^{n} |a_{ij}|\, |x_j| \le \sum_{j=1}^{n} \left(\max_k \sum_{i=1}^{n} |a_{ik}| \right) |x_j|$$

$$\|Ax\|_1 \le \left(\max_k \sum_{i=1}^{n} |a_{ik}| \right) \sum_{j=1}^{n} |x_j| \implies \|Ax\|_1 \le \left(\max_k \sum_{i=1}^{n} |a_{ik}| \right) \|x\|_1 \implies \frac{\|Ax\|_1}{\|x\|_1} \le \left(\max_k \sum_{i=1}^{n} |a_{ik}| \right)$$

We have $\frac{\|Ax\|_1}{\|x\|_1} \leq \left(\max_k \sum_{i=1}^{n} |a_{ik}| \right)$

$$\|A\|_1 = \max_{x \neq 0} \frac{\|Ax\|_1}{\|x\|_1}$$

To prove $\|A\|_1 = \max_k \sum_{i=1}^{n} |a_{ik}|$, we must find an $\hat{x} \in \mathbb{R}^n$ for which

$$\frac{\|A\hat{x}\|_1}{\|\hat{x}\|_1} = \max_k \sum_{i=1}^{n} |a_{ik}|$$

Suppose that the largest absolute column sum is attained in the mth column of A.

Set $\hat{x} = e_m$. **Then** $\|\hat{x}\|_1 = 1$ and $A\hat{x} = \begin{bmatrix} a_{1m} \\ a_{2m} \\ \vdots \\ a_{nm} \end{bmatrix}$

$$\frac{\|A\hat{x}\|_1}{\|\hat{x}\|_1} = \sum_{i=1}^{n} |a_{im}| = \max_k \sum_{i=1}^{n} |a_{ik}| \implies \|A\|_1 = \max_{x \neq 0} \frac{\|Ax\|_1}{\|x\|_1} = \max_k \sum_{i=1}^{n} |a_{ik}|$$

(Column Sum Norm)

Simplified forms of induced matrix norm

The matrix norm induced by the vector 1-norm:

$$\|A\|_1 = \max_j \sum_{i=1}^{n} |a_{ij}| \qquad \textbf{(Column Sum Norm)}$$

The matrix norm induced by the vector ∞-norm:

$$\|A\|_\infty = \max_i \sum_{j=1}^{n} |a_{ij}| \qquad \textbf{(Row Sum Norm)}$$

The matrix norm induced by the vector 2-norm:

$$\|A\|_2 = \sqrt{\rho(A^T A)} = \sqrt{\lambda_{\max} \text{ of } A^T A} \qquad \textbf{(Spectral Norm)}$$

Note: The number $\rho(A) = \max\{|\lambda|: \lambda \text{ is the eigenvalue of } A\}$ is called the spectral radius of A.

ILL-CONDITIONED LINEAR SYSTEMS $Ax = b$

A system of linear equations is said to ill-conditioned when some **small perturbation** in the system can produce **large changes** in the exact solution.

1. Inaccurate Right Hand Side (b)

Let $Ax = b$ and $A\tilde{x} = (b + \delta b)$

$$\Rightarrow A(\tilde{x} - x) = \delta b \Rightarrow \tilde{x} - x = A^{-1}\,\delta b \Rightarrow \|(\tilde{x} - x)\| \leq \|A^{-1}\|\|\delta b\| \Rightarrow \frac{\|(\tilde{x} - x)\|}{\|x\|} \leq \frac{\|A^{-1}\|\|\delta b\|}{\|x\|}$$

Again $Ax = b \Rightarrow \|b\| = \|Ax\| \Rightarrow \|b\| \leq \|A\|\|x\| \Rightarrow \frac{1}{\|x\|} \leq \frac{\|A\|}{\|b\|}$

$$\Rightarrow \frac{\|(\tilde{x} - x)\|}{\|x\|} \leq \|A^{-1}\|\|A\|\frac{\|\delta b\|}{\|b\|} \Rightarrow \frac{\|(\tilde{x} - x)\|}{\|x\|} \leq k(A)\frac{\|\delta b\|}{\|b\|}$$

Condition Number: $k(A) = \|A\|\|A^{-1}\|$

1. Inaccurate Right Hand Side (b) $$\frac{\|(\tilde{x}-x)\|}{\|x\|} \leq k(A)\frac{\|\delta b\|}{\|b\|}$$

If the system is well conditioned $(k(A) \approx 1)$, small change in b can have only a small effect on the solution. However, in the case of ill-conditioning $(k(A) \gg 1)$, small change in b may lead to large error in the solution.

2. Inaccurate Matrix Entries (A) $$\frac{\|(\tilde{x}-x)\|}{\|x\|} \leq \frac{[k(A)]^2}{(1-\|A^{-1}\delta A\|)}\frac{\|\delta A\|}{\|A\|}$$

3. Inaccurate (A) & (b) $$\frac{\|(\tilde{x}-x)\|}{\|x\|} \leq \frac{k(A)}{(1-\|A^{-1}\delta A\|)}\left(k(A)\frac{\|\delta A\|}{\|A\|}+\frac{\|\delta b\|}{\|b\|}\right)$$

Recall the problem:

$$0.835x_1 + 0.667x_2 = 0.168$$
$$0.333x_1 + 0.266x_2 = 0.067$$

Exact Solution: $(1, -1)$

$$0.835x_1 + 0.667x_2 = 0.168$$
$$0.333x_1 + 0.266x_2 = 0.066$$

Exact Solution: $(-666, 834)$

Condition Number of the coefficient matrix (k_1) : 1.7543×10^6

Questions Raised

- ✓ System is very sensitive
  - How to measure?

- ➢ Residual is small but the approximation is weird
  - Where is the problem?

Relation between Residual and Relative Error

System: $Ax = b$ Residual: $r = b - A\tilde{x}$ *Some* approximation of x: $\tilde{x}$

$$Ax = b \Longrightarrow \|b\| = \|Ax\| \Longrightarrow \|b\| \le \|A\|\|x\| \Longrightarrow \frac{1}{\|x\|} \le \frac{\|A\|}{\|b\|}$$

$$r = b - A\tilde{x} \Longrightarrow r = Ax - A\tilde{x} = A(x - \tilde{x}) \Longrightarrow (x - \tilde{x}) = A^{-1}r \Longrightarrow \|x - \tilde{x}\| = \|A^{-1}r\|$$

$$\Longrightarrow \|x - \tilde{x}\| \le \|A^{-1}\|\|r\| \Longrightarrow \frac{\|x - \tilde{x}\|}{\|x\|} \le \frac{\|A^{-1}\|\|r\|}{\|x\|} = \frac{\|A^{-1}\|\|A\|\|r\|}{\|b\|} \Longrightarrow \frac{\|x - \tilde{x}\|}{\|x\|} \le k(A)\frac{\|r\|}{\|b\|}$$

If the system is well conditioned $(k(A) \approx 1)$, small residual can be used to estimate error.

However, in the case of ill-conditioned system, residual may not estimate the actual error.

linear System of Linear Equations

$0.835x_1 + 0.667x_2 = 0.168$

$0.333x_1 + 0.266x_2 = 0.067$

Exact Solution: $(1, -1)$

GS Approximation: $(-0.6711, 1.0920)$

$0.835x_1 + 0.667x_2 = 0.168$

$0.333x_1 + 0.266x_2 = 0.066$

Exact Solution: $(-666, 834)$

GS Approximation: $(-30.0389, -37.357)$

Residuals:

$b - Ax^{(k)}$

$$\begin{bmatrix} -5.81 \times 10^{-5} \\ -2.0704 \times 10^{-5} \end{bmatrix}$$

$$\begin{bmatrix} -2.4750 \times 10^{-6} \\ -1.0000 \times 10^{-4} \end{bmatrix}$$

$k_1 : 1.7543 \times 10^6$

Key Takeaways

Importance of Rounding Off Errors

- Rounding errors are inevitable in numerical computations due to finite precision.
- Even small errors can **accumulate** in iterative calculations or long sequences of operations.
- Understanding rounding errors helps in **designing stable numerical algorithms**.

Sensitivity of Calculations → Function Evaluation

- **Sensitivity** measures how changes in input affect the output.
- Large condition number→ small input errors can produce **large output errors**.

Sensitivity of Linear Systems

- **High condition number** → system is **ill-conditioned** → small errors in b or A → large errors in x.
- For a system $Ax = b$: solution's sensitivity depends on **condition number of matrix $\boldsymbol{A}$**.

Understanding rounding errors and system sensitivity both are crucial for reliable numerical computations and for designing robust algorithms.

Thank You