Errors and Uncertainties in Computations

Rubin H Landau
With
Sally Haerer and Scott Clark

Computational Physics for Undergraduates
BS Degree Program: Oregon State University

"Engaging People in Cyber Infrastructure"
Support by EPICS/NSF & OSU

Problem: Life + Errors (Uncertainties)

- **Always part of computation**
- **Finite precision $\Rightarrow$ uncertainties = "errors"**
- **Don't be afraid; Don't play with garbage**
- **Errors accumulate with steps U_i**

$$\text{start} \rightarrow U_1 \rightarrow U_2 \rightarrow \ldots \rightarrow U_n \rightarrow \text{end} \quad (1)$$

- p = probability U_i correct
- $P = p^n$ = probability n steps correct
- $n = 1000,\ p = 0.9993 \Rightarrow P = 1/2$
- (whoops!)

Theory: Types of Errors (4 plagues)

1. **Blunders: typos, wrong program, wrong data, …**

2. **Random errors: electronic fluctuations, cosmic rays, …**
   - **rare, but if 10^8 steps?**
   - **can't control**
   - **repeat calculation**

3. **Approximation errors, algorithm, approx math:**

$$e^{-x} = \sum_{n=0}^{\infty} \frac{(-x)^n}{n!} \simeq \sum_{n=0}^{N} \frac{(-x)^n}{n!} = e^{-x} + \mathcal{E}(x, N) \qquad (2)$$

   - **decreases as *N* increases**
   - **vanishes as *N* $\rightarrow \infty$**

4. **Round-Off Errors** (cont)

Errors (cont)-- Roundoff errors:

- **Because: numbers via finite # bits**
- **≈ uncertainties in measurement**
- **Some numbers represent exact (2^n)**
- **Accumulates with steps ⇒ unstable**
- **⇒ *garbage*: RO ≈ result:**

$$2\left(\frac{1}{3}\right) - \frac{2}{3} = 0.66666 - 0.66667 = -0.00001 \neq 0 \qquad (3)$$

- ***Significant figures***
  - *a* = 11223344556677889900 *(4)*
    = 1.1223344556677889 * 10^{19}
  - exponent: stored separately
    small ⇒ full precision
- **Most significant part: 1.12233**
- **Least significant part: 44556677**
  - error prone

Disaster Model: Subtractive Cancellation

- **If you subtract two large numbers and end up with a small result, there will be less significance in the small result**

$$a = b - c \quad \Rightarrow \quad a_c = b_c - c_c \tag{5}$$

$$a_c = b(1+\epsilon_b) - c(1+\epsilon_c) \tag{6}$$

$$\Rightarrow \quad \frac{a_c}{a} = 1 + \epsilon_b \frac{b}{a} - \frac{c}{a}\epsilon_c \tag{7}$$

$$\simeq \quad 1 - \frac{b}{a}(\epsilon_b - \epsilon_c) \rightarrow \infty \tag{8}$$

HW: Cancellation in power series, x = 100:

$$e^{-x} = 1 - x + \frac{x^2}{2} - \frac{x^3}{6} + \cdots = 1 - 100 + 5000 + \cdots \tag{9}$$

Model for Multiplicative Errors

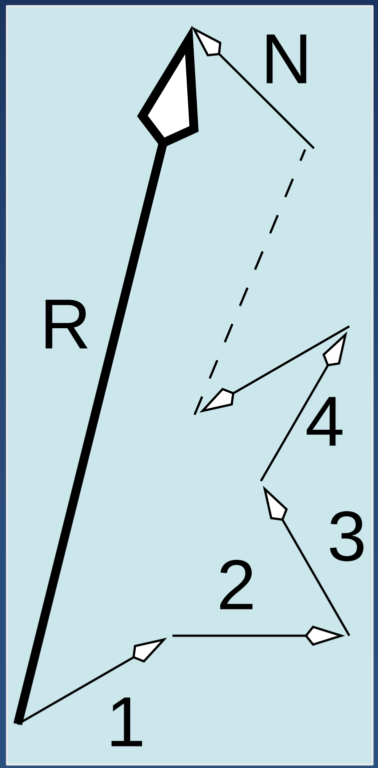
- Propagation of error in multiplication

$$a = b \times c \quad \Rightarrow \quad a_c = b_c \times c_c, \qquad (10)$$

$$\Rightarrow \quad \frac{a_c}{a} = (1+\epsilon_b)(1+\epsilon_c) \simeq 1 + \epsilon_b + \epsilon_c \qquad (11)$$

- How add errors? $|\epsilon_b| + |\epsilon_c|, \quad |\epsilon_b| - |\epsilon_c|$?

$\epsilon_{\text{ro}} \approx \sqrt{N}\epsilon_m$



  - Algorithm Model: N steps random walk (Chap.5)
  - Each step $\simeq$ machine precision

If non random: $\epsilon \simeq N\epsilon_m, \quad N!\epsilon_m$

E.G.: several hour calculation, 10^{10} Flops

$\Rightarrow \quad \epsilon \simeq 10^7 \epsilon_m$

Singles: $\epsilon_m \simeq 10^{-7} \quad \Rightarrow \quad \epsilon \simeq 1$ (whoops!)

Experiment: Determine Errors

1. Basic algorithm questions:
   a) does it converge?
   b) if not, quit
   c) how precise are converged results?

2. Converged $\neq$ correct

3. How $$ (time consuming) ?

Experiment: Expected Behavior

- ϵ_{apprx} = approximation error = Ans_exact - Ans_algorithm
- Algorithmic error decreases rapidly

$$\epsilon_{\text{aprx}} \simeq \frac{\alpha}{N^{\beta}} \to 0, \qquad (N \to \infty), \tag{12}$$

- Round oFF error increases slowly

$$\epsilon_{RO} \simeq \sqrt{N}\epsilon_m \tag{13}$$

- Want minimum of sum

$$\begin{aligned} \epsilon_{\text{tot}} &= \epsilon_{\text{approx}} + \epsilon_{\text{RO}}, \\ &\simeq \frac{\alpha}{N^{\beta}} + \sqrt{N}\epsilon_m \end{aligned} \tag{14}$$

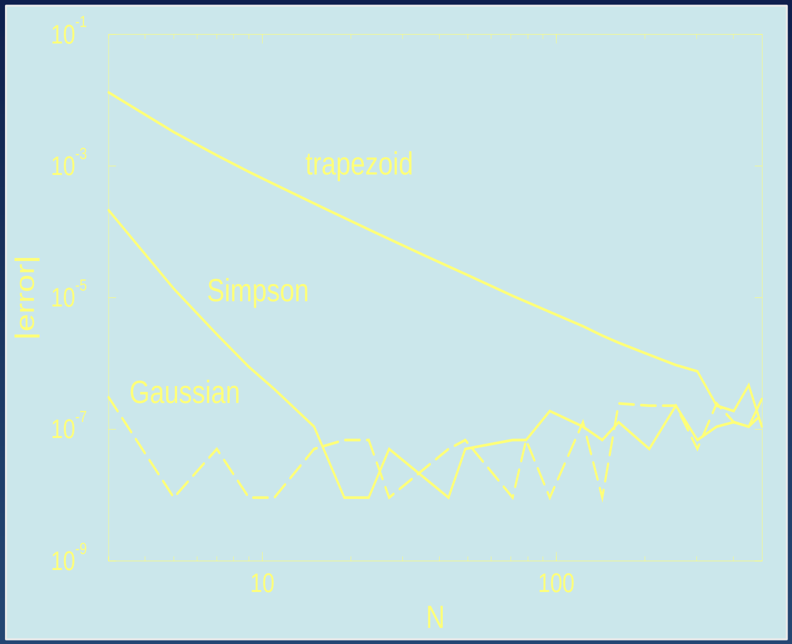
- Want N small $\Rightarrow$ faster, accurate

Experimental Approach

- Determine α & β via applied math (see text)
- Min ε via comparison to known answer
  - test (similar) problem
- If *N* looks good, assume *2N* exact:

$$A(N) \approx \mathcal{A} + \frac{\alpha}{N^\beta} + \textbf{RO}$$

$$\epsilon_{tot} = A(N) - A(2N) \approx \frac{\alpha}{N^\beta}$$



Look & understand!
converge → diverge

- Plot $\log_{10}(\varepsilon)$ *vs* $\log_{10}(N)$
  - ordinate = # decimal places precision
  - slope = β

Time for Exercises in Lab

Exercise: Subtractive Cancellation

1. Equivalent solutions to $ax^2 + bx + c = 0$

$$x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}, \qquad x'_{1,2} = \frac{-2c}{b \pm \sqrt{b^2 - 4ac}}.$$

Subtractive cancellation if *4ac* << b^2 *($\sqrt{\ldots}$ ≈ b)*

a. Compute all 4 solutions

b. Demo how error ↑ (a = b = 1, c = 10^{-n})

c. Relate to machine precision

d. Have program output good solutions

Subtractive Exercise (cont)

2. Three equivalent (math) sums:

$$S_N^{(1)} = \sum_{n=1}^{2N} (-1)^n \frac{n}{n+1}. \qquad (1)$$

$$S_N^{(2)} = \sum_{n=1}^{N} \frac{2n}{2n+1} - \sum_{n=1}^{N} \frac{2n-1}{2n} \qquad \text{(even \& odd)}, \qquad (2)$$

$$S_N^{(3)} = \sum_{n=1}^{N} \frac{1}{2n(2n+1)} \qquad \text{(partial sums)}. \qquad (3)$$

Write program to calculate S(1), S(2), S(3) $1 \le N \le 100$

Assume S(3) is exact

Log-log plot relative error: $\log_{10} \left| \frac{S_N^{(1)} - S_N^{(3)}}{S_N^{(3)}} \right|$ *vs* $\log_{10}(N)$

Get expected straight line?

Exercise: error in e^{-x} (cont)

3. Mathematical definition versus algorithm

$$e^{-x} = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \cdots \simeq \sum_{n=0}^{N} \frac{(-x)^n}{n!} \qquad (1)$$

a. Examine cancellation of terms for $x \approx 10$

b. Convergence $\Rightarrow$ terms decrease

`new = (n+1)/x old` $\Rightarrow n+1 \approx x$

c. Demo: near-perfect cancellation here?

d. exp$(-x)$ = $1/$exp(x) better, yet still RO

e. Plot error *vs* N for $1 \leq x \leq 100$ (see easyPtPlot.java.)

