Errores en el cálculo numérico

Manuel Carlevaro

Departamento de Ingeniería Mecánica

Grupo de Materiales Granulares - UTN FRLP

manuel.carlevaro@gmail.com

Cálculo Avanzado • 2023

XƎLATEX ·

¿Para qué?

- Análisis numérico
- Manipulación simbólica
- Colección y análisis de datos
- Visualización
- Simulación

¿Para qué?

- Análisis numérico
- Manipulación simbólica
- Colección y análisis de datos
- Visualización
- Simulación

¿Cómo?

- Modelado: sistema de ecuaciones, ecuaciones diferenciales, integral
- Método numérico: elección, parametrización, estimación de errores
- Programación: Python, C/C++, Fortran, Julia
- Ejecución del código
- Interpretación de resultados: visualización, análisis estadístico, rediseño y ejecución

Errores

1. **Error inherente**. Provienen desde el principio en los datos originales y están fuera del alcance del control de cálculo. Ejemplo: incertezas en las mediciones.
2. **Error de truncamiento**. Se producen como resultado de reemplazar un proceso infinito por uno finito. Ejemplo: usar solo los primeros términos de una serie de Taylor.
3. **Error de redondeo**. Se originan en la representación con precisión finita de los números en una computadora.
4. **Error por equivocación**. Causado por realizar una operación aritmética incorrectamente.

Errores

1. **Error inherente**. Provienen desde el principio en los datos originales y están fuera del alcance del control de cálculo. Ejemplo: incertezas en las mediciones.
2. **Error de truncamiento**. Se producen como resultado de reemplazar un proceso infinito por uno finito. Ejemplo: usar solo los primeros términos de una serie de Taylor.
3. **Error de redondeo**. Se originan en la representación con precisión finita de los números en una computadora.
4. **Error por equivocación**. Causado por realizar una operación aritmética incorrectamente.

---

1 – 3 son errores **inevitables** $\mapsto$ control del error.

4 es **evitable**.

Números enteros y de punto flotante

Una cuenta simple: $a + a + a - 3a = 0$

```
>>> 0.1 + 0.1 + 0.1 - 0.3
5.551115123125783e-17
```

En computadoras hay **dos tipos** de números:

- **Punto fijo:** cantidad fija de números decimales
  Ejemplo: $35.6247, 0.4573, -1.0000$
  Enteros: 0 números decimales $\mapsto$ **exacta**
- **Punto flotante:** cantidad fija de cifras significativas
  Ejemplo:
  $0.1900 \cdot 10^4, 0.8691 \cdot 10^{-6}, -0.2000 \cdot 10^{-13}$ (4 cifras significativas) $\mapsto$ representación **aproximada**

```
>>> 7 + 0.000000000000001
7.000000000000001
>>> 7 + 0.0000000000000001
7.0
>>> 0.1 + 0.2
0.30000000000000004
```

Estándar IEEE 754: tres **enteros**

$$r = (-1)^s \, m \, b^e$$

s: signo, m: mantisa, b: base (10, 2, 16), e: exponente.
Ejemplo decimal:

$$s = 0$$
$$m = 31415926$$
$$b = 10$$
$$e = -7$$
$$r = (-1)^0 31415926 \cdot 10^{-7} = 3.1415926$$

Ejemplo binario:

$$r = \pm m \cdot 2^n, \quad m = 0.d_1 d_2 \cdots d_k, \quad d_1 > 0$$
$$r = \pm(d_1 2^{-1} + d_2 2^{-2} + \cdots + d_n 2^{-n}) \cdot 2^n$$

número de máquina de k-dígitos.

Ejemplo 1: $0.1011 \cdot 2^3$:

$$r = 4 + 1 + \frac{1}{2} = 5.5$$

Ejemplo 1: $0.1011 \cdot 2^3$:

$$r = 4 + 1 + \frac{1}{2} = 5.5$$

Ejemplo 2: Números enteros con exponente fijo $e = n$ $b = 2$, $n = 3$ ($r \geq 0$)

Representación:	000	001	010	011	100	101	110	111
Valor decimal:	0	1	2	3	4	5	6	7

Overflow (desbordamiento): $3 + 5$ = `011` + `101` = `1000`

Ejemplo 1: $0.1011 \cdot 2^3$:

$$r = 4 + 1 + \frac{1}{2} = 5.5$$

Ejemplo 2: Números enteros con exponente fijo $e = n$ $b = 2$, $n = 3$ ($r \geq 0$)

Representación:	000	001	010	011	100	101	110	111
Valor decimal:	0	1	2	3	4	5	6	7

Overflow (desbordamiento): $3 + 5 = \texttt{011} + \texttt{101} = \texttt{1000}$

Ejemplo 3: e entero variable (punto flotante) $b = 2$, $n = 2$, $-2 \leq e \leq 2$

Representación:	$0.00 \cdot 2^e$	$0.01 \cdot 2^e$	$0.10 \cdot 2^e$	$0.11 \cdot 2^e$
Valor decimal:	0	$\frac{1}{4} \cdot 2^e$	$\frac{1}{2} \cdot 2^e$	$\frac{3}{4} \cdot 2^e$

$$r \in \left\{0, \frac{1}{16}, \frac{1}{8}, \frac{3}{16}, \frac{1}{4}, \frac{3}{8}, \frac{1}{2}, \frac{3}{4}, 1, \frac{3}{2}, 2, 3\right\}, \quad 2 + \frac{1}{8} = 0.10001 \notin \{\cdot\}$$

Características del formato:

- Permite representar números de órdenes de magnitud enormemente dispares (limitado por la longitud del exponente)
- Proporciona la misma precisión relativa para todos los órdenes (limitado por la longitud de la mantisa)
- Permite cálculos entre magnitudes (número grande $\times$ número pequeño) conservando la precisión de ambos en el resultado.
- Representación en notación científica

$$6.022 \cdot 10^{23} \longleftrightarrow \texttt{6.022E23}$$

- Existe solo un número finito de números de máquina, y son menos "densos" a medida que el número es más grande. Hay tantos números entre 2 y 4 como entre 1024 y 2048.
- El menor número de máquina positivo ε_m para el cual $1 + \varepsilon_m > 1$ se denomina **precisión de la máquina**. No se pueden representar números en los intervalos $[1, 1 + \varepsilon_m], [2, 2 + 2\varepsilon_m], \cdots$
- Exponentes de la norma IEEE 754:
  - Precisión simple: 2^{-126} a 2^{128} ($1.175 \cdot 10^{-38}$ a $3.403 \cdot 10^{38}$)
  - Precisión doble: 2^{-1022} a 2^{1024} ($2.225 \cdot 10^{-308}$ a $1.798 \cdot 10^{308}$)

- Normalización: $d_1 > 0 \mapsto$ representación única
- $d_1 = 1$ **no** se almacena (dígito principal implícito)
- Precisión simple: 4 bytes = 32 bits:
  - s 1 bit
  - e 8 bits
  - m 23 bits
- Sesgo de exponente: $e = E - 127$
  ($e = E - 1023$ en doble precisión, 11 bits)

Ejemplo:

$$\underbrace{1}_{-}\underbrace{10000001}_{e}\underbrace{10100100000000000000000}_{m}$$

- Signo: primer bit 1: negativo
- Exponente: $\mathtt{10000001}_2 = 129$. $129 - 127 = 2$
- Mantisa:

$$1.101001_2 = 1 + \frac{1}{2} + \frac{1}{2^3} + \frac{1}{2^6} = \frac{105}{64} = 1.640625$$

- Resultado:

$$r = (-1)1.640625 \cdot 2^2 = -6.5625$$

- Precisión doble: 8 bytes = 64 bits:
  - s 1 bit
  - e 11 bits
  - m 52 bits

Precisión infinita: $\pm .d_1 d_2, \cdots b^e$

Precisión n-dígitos $\pm 0.d_1 d_2, \cdots d_n b^e$

Parte entera $[\cdot]$: $[123.456] = 123$

$x = 0.m \cdot b^e$

Truncamiento

$$\mathcal{T}(x) = [b^n \cdot 0.m] \cdot b^{e-n}$$

Ejemplo ($n = 3$, base 10):

$$\mathcal{T}(1/3) = 10^{-3}[10^3 \cdot 0.333\ldots] = 0.333$$
$$\mathcal{T}(2/3) = 10^{-3}[10^3 \cdot 0.666\ldots] = 0.666$$

Redondeo

$$\mathcal{R}(x) = [b^n \cdot 0.m + 0.5] \cdot b^{e-n}$$

Ejemplo ($n = 3$, base 10):

$$\mathcal{R}(1/3) = 10^{-3}[10^3 \cdot 0.333\ldots + 0.5] = 0.333$$
$$\mathcal{R}(2/3) = 10^{-3}[10^3 \cdot 0.666\ldots + 0.5] = 0.667$$

Si $x < 0$ $\mathcal{R}(x) = -\mathcal{R}(-x)$

$$|x - \mathcal{R}(x)| \leq \frac{1}{2} b^{e-n} \text{ (unidad de redondeo precisión de la máquina)}$$

a: Valor exacto, $\tilde{a}$: valor aproximado

$$\epsilon = a - \tilde{a}$$

es el **error** de $\tilde{a}$. Entonces:

$$a = \tilde{a} + \epsilon$$

Error absoluto:

$$|a - \tilde{a}|$$

Error relativo

$$\epsilon_r = \frac{\epsilon}{a} = \frac{a - \tilde{a}}{a}$$

a es desconocido. Si $|\epsilon| \ll |\tilde{a}|$

$$\epsilon_r \approx \frac{\epsilon}{\tilde{a}}$$

Tampoco conocemos ϵ. En la práctica obtenemos una cota de error de $\tilde{a}$:

$$|\epsilon| \leq \beta, \quad |a - \tilde{a}| \leq \beta$$

Del mismo modo:

$$|\epsilon_r| \leq \beta_r, \quad \left|\frac{a - \tilde{a}}{a}\right| \leq \beta_r$$

$$f(x_1, x_2, \cdots, x_n), \quad x_i = x_i^0 + \delta x_i$$

Expansión en serie de Taylor de primer orden:

$$f \approx f_0 + \sum_{i=1}^{n} \frac{\partial f}{\partial x_i}\delta x_i \text{ o } \delta f = |f - f_0| \leq \left|\frac{\partial f}{\partial x_1}\right| \delta x_1 + \cdots + \left|\frac{\partial f}{\partial x_n}\right| \delta x_n$$

Casos especiales: $x = \tilde{x} + \epsilon_x$, $y = \tilde{y} + \epsilon_y$, $|\epsilon_x| \leq \beta_x$, $|\epsilon_y| \leq \beta_y$

- Sumas y restas:

$$\begin{aligned} |\epsilon| &= |x - y - (\tilde{x} - \tilde{y})| \\ &= |x - \tilde{x} - (y - \tilde{y})| \\ &= |\epsilon_x - \epsilon_y| \leq |\epsilon_x| + |\epsilon_y| \leq \beta_x + \beta_y \end{aligned}$$

- Productos y divisiones:

$$\begin{aligned} |\epsilon_r| &= \left|\frac{xy - \tilde{x}\tilde{y}}{xy}\right| = \left|\frac{xy - (x - \epsilon_x)(y - \epsilon_y)}{xy}\right| = \left|\frac{\epsilon_x y + \epsilon_y x - \epsilon_x \epsilon_y}{xy}\right| \\ &= \left|\frac{\epsilon_x y + \epsilon_y x}{xy}\right| \leq \left|\frac{\epsilon_x}{x}\right| + \left|\frac{\epsilon_y}{y}\right| = |\epsilon_{rx}| + |\epsilon_{ry}| \leq \beta_{rx} + \beta_{ry} \end{aligned}$$

- E. Kreyszig, H. Kreyszig y E.J. Norminton. *Advanced Engineering Mathematics*. Hoboken, USA: John Wiley & Sons, Inc, 2011. Capítulo 19.
- Carlos Moreno González. *Introducción al cálculo numérico*. Madrid, España: Universidad Nacional de Educación a Distancia, 2014 Capítulo 1.
- S.C. Chapra y R.P. Canale. *Numerical Methods for Engineers*. 8.ª ed. New York, United States: McGraw-Hill Education, 2021. Capítulo 3.
- J.H. Mathews y K.D. Fink. *Numerical methods using MATLAB*. New Jersey, United States: Pearson Education Inc., 2004. Capítulo 1.