TEORIA DE ERRORES

□

Fuentes De error

Errores inherentes: (EI)
Son los errores que afectan a los datos del problema numérico y pueden tener distintos orígenes. Por ejemplo pueden ser el resultado de la incertidumbre en cualquier medición, o por ejemplo cuando queremos ingresar en una calculadora los valores de $\sqrt{2}, \pi$, ya que usaremos solo una cantidad finita de dígitos para representarlos.

Errores de redondeo: (ER)
Son los posibles errores de representación que se produzcan al realizar cada cálculo de nuestro algoritmo.

Errores de discretización o truncamiento: (ED)
Son los que se producen al pasar del problema matemático al numérico, por ejemplo cuando se desprecia el término complementario, suplantando una suma infinita por una finita.

Error Final

Error inherente propagado: La forma en que se propaguen los errores inherentes quedara definida por el problema numérico.

Error de redondeo propagado: El error de redondeo final será el producto de la propagación de los errores de redondeo en los cálculos y dependerá del algoritmo que elijamos.

Errores de discretización o truncamiento: Es cualitativa y cuantitativamente el que definimos para la fuente de error

Definiciones

- Problema Matemático: Es una descripción clara de la conexión funcional entre los datos de entrada y de salida.
- Problema Numérico: Es una descripción clara de la conexión funcional entre los datos numéricos de entrada y de salida. Debe implicar una cantidad finita de operaciones elementales realizables por computadora
- Método Numérico: Es un procedimiento para aproximar un problema matemático con un problema numérico.

- **Algoritmo:** Es una descripción completa y bien definida de una cantidad finita de operaciones elementales a través de las cuales es posible transformar los datos de entrada en los datos de salida.

- **Error absoluto, Error relativo y cota del error**

  Sea b el valor de una cierta magnitud y sea $\bar{b}$ el valor medido o calculado:

  $$e_b = b - \bar{b}$$

  $$er_b = \frac{e_b}{b}$$

  $$er_b \cong \frac{e_b}{\bar{b}} \quad que\ es\ buena\ si\ |e_b| \ll |b|$$

- **Decimales significativos**

  $\bar{a}\ tiene\ t\ decimales\ significativos \Leftrightarrow |a - \bar{a}| \leq k10^{-t}$ (k puede tomar el valor 1 o 0.5). Tomaremos k=0.5.
  Cuando el resultado de una operación tenga mas de t decimales, escribiremos este valor con t decimales solamente siguiendo la regla: si el decimal que esta en el lugar t+1 es menor que 5 dejamos los t decimales del numero como están; si el decimal de la posición t+1 es igual o mayor que 5 entonces le sumamos 10^{-t} a nuestro valor y tomamos los primeros t decimales que quedan. Cuando realizamos este proceso decimos que nuestro resultado esta **escrito exactamente con sus decimales significativos.**

Propagación de los errores inherentes

Supongamos que tenemos un problema numérico donde $X \in R^n \quad Y \in R^m$, es decir las componentes del vector X son los datos de entrada y el vector Y representa los resultados, entonces:

$$Y(X) = \begin{bmatrix} y_1(X) \\ y_2(X) \\ \vdots \\ y_m(X) \end{bmatrix}$$

Supongamos que conocemos $\bar{X} = \begin{bmatrix} \bar{x}_1 \\ \bar{x}_2 \\ \vdots \\ \bar{x}_n \end{bmatrix}$ $y\ e_{xi} = x_i - \bar{x}_i$

Las operaciones que involucran al vector Y son derivables entonces desarrollando por Taylor en un entorno de $\bar{X}$ tenemos:

$$Y(X) = Y(\bar{X}) + \frac{\partial(y_1,\dots y_m)}{\partial(x_1,\dots x_n)}(\bar{X})(X-\bar{X}) + T(X-\bar{X})$$

Donde el segundo sumando indica el producto de la matriz Jacobiana por el vector que indica la variación de los datos respecto de los valores que usamos para realizar los cálculos y el tercer término es el resto del desarrollo de Taylor. Por lo tanto si

$$x_i - \bar{x_i} \ll 1 \quad \forall\, i = 1, \dots, n$$

Podemos despreciar el término complementario resultando:

$$y_i(X) - y_i(\bar{X}) \cong \sum_{j=1}^{n} \frac{\partial y_i}{\partial x_j}(\bar{X})(x_j - \bar{x}_j) \quad i = 1, \cdots, m$$

en ter min *os del error*

$$e_{yi} \cong \sum_{j=1}^{n} \frac{\partial y_i}{\partial x_j}(\bar{X})\, e_{xj}$$

$$Si \left|e_{x_j}\right| \le r_j, \quad j = 1, \dots n \quad resulta$$

$$\left|e_{y_i}\right| \le \sum_{j=1}^{n} \left|\frac{\partial y_i}{\partial x_j}(X)\right| r_j$$

Propagación del error en las operaciones elementales

Suma y resta

$$Y(x_1, x_2) = x_1 \pm x_2$$

$$e_y = e_{x_1 \pm x_2} = \frac{\partial Y}{\partial x_1}(x_1, x_2)\, e_{x_1} + \frac{\partial Y}{\partial x_2}(x_1, x_2)\, e_{x_2} \Rightarrow$$

$$e_{x_1 \pm x_2} = e_{x_1} + e_{x_2}$$

y el error relativo es

$$er_y = er_{x_1 \pm x_2} = \frac{\bar{x}_1}{\bar{x}_1 \pm \bar{x}_2} er_{x_1} \pm \frac{\bar{x}_2}{\bar{x}_1 \pm \bar{x}_2} er_{x_2}$$

Producto

$$Y\left(x_1, x_2\right) = x_1.x_2$$

$$e_y = e_{x_1.x_2} = \overline{x}_2 e_{x_1} + \overline{x}_1 e_{x_2}$$

$$er_y = er_{x_1} + er_{x_2}$$

División

$$Y\left(x_1, x_2\right) = \frac{x_1}{x_2} \quad x_2 \neq 0$$

$$e_y = e_{x_1/x_2} = \frac{1}{\overline{x}_2} e_{x_1} - \frac{\overline{x}_1}{\left(\overline{x}_2\right)^2} e_{x_2}$$

$$er_y = er_{x_1/x_2} = er_{x_1} - er_{x_2}$$

Raíz Cuadrada

$$Y\left(x\right) = \sqrt{x} \quad x > 0$$

$$e_y = e_{\sqrt{x}} = \frac{1}{2\sqrt{\overline{x}}} e_x$$

$$er_y = er_{\sqrt{x}} = 0.5 er_x$$

Diremos que una operación elemental es estable, si dada una cota para los errores inherentes relativos, los errores propagados relativos se mantienen acotados por un valor independiente de los datos de entrada.

Representación de números

Sistemas de numeración

Nuestro sistema de numeración es posicional. Un sistema de numeración posicional queda caracterizado por la base (B) que debe ser un numero natural mayor o igual a 2 y por un conjunto de B símbolos que determinan el" alfabeto" del sistema de numeración, debiendo representar los mismos los enteros de 0 a B-1.
En nuestro sistema decimal: B=10, y los dígitos son: 0, 1,2,…,9 la representación de un numero racional es como sigue:

$$x = \pm a_M a_{M-1} ... a_0 , a_{-1} a_{-1} ... a_{-N} \quad 0 \le a_i \le 9 \quad a_i \in N$$

$$\Leftrightarrow$$

$$x = \pm \sum_{n=-N}^{M} a_n 10^n$$

Las computadoras representan internamente los números en sistema binario. Aquí los 'bits' juegan el papel de los factores de las sucesivas potencias de 2 en la descomposición de un numero:

$$x = \pm a_M a_{M-1} ... a_0 , a_{-1} a_{-1} ... a_{-N} = \pm \sum_{n=-N}^{M} a_n 2^n \quad a_n = 0 \vee a_n = 1$$

Ejemplos

$$101_2 = 1.2^0 + 0.2^1 + 1.2^2 = 5$$

$$11{,}1 = 1.2^{-1} + 1.2^0 + 1.2^1 = 3.5$$

$$1011 = 1.2^0 + 1.2^1 + 0.2^2 + 1.2^3 = 11$$

Representación en sistemas de punto fijo

Se toman dos números fijos n_1 y n_2 tales que $n = n_1 + n_2$ asignándose n_1 lugares a los dígitos enteros y n_2 lugares a los dígitos decimales.
Ejemplos
Si n=10, n_1 =4 y n_2 =6
25.543 se representara 0025 543000
0.0673 se representara 0000 067300
En este tipo de representación el numero 16537 no se representa a pesar de tener solo 5 dígitos.

Representación en sistemas de punto flotante

En este sistema cada número real puede ser representado en la forma:

$$x = a.10^b, con\ |a| < 1, b \in Z$$

Donde el exponente: b indica la posición del punto decimal con respecto al primer digito de la mantisa: a.
Se dice que un sistema es de punto flotante normalizado si imponemos a la mantisa la condición que su primer digito después del punto decimal sea distinto de cero, o sea:

$$0.1 \leq |a| < 1$$

Una computadora asigna una cantidad finita de t cifras para la mantisa y otra de e cifras para el exponente de modo que:
N=t+e
Ejemplos
Si n=6, t=4 y e=2
6385 se representara 6385 04
25.5 se representara 2550 02

Nosotros consideraremos solamente sistemas de representación de punto flotante normalizado y la correspondiente aritmética de punto flotante.

Corte o truncamiento

Dado un número real dentro del rango de la maquina, procedemos a escribirlo en punto flotante normalizado:

$$x = a\,10^b \quad con\ |a| = 0.a_{-1}a_{-2}...a_{-t}a_{-t-1}\ldots$$

Definimos

$$\bar{a} = signo(a)\,0.a_{-1}a_{-2}...a_{-t}$$

$$\bar{x} = \bar{a}\,10^b$$

Será almacenado exactamente en la maquina como **aproximación de x por truncamiento.**

Redondeo o redondeo simétrico

$$x = a\,10^b \quad con\ |a| = 0.a_{-1}a_{-2}...a_{-t}a_{-t-1}\ldots$$

Definimos

$$\bar{a} = signo(a)\begin{cases} 0.a_{-1}a_{-2}...a_{-t} & si\ 0 \leq a_{-t-1} \leq 4 \\ 0.a_{-1}a_{-2}...a_{-t} + 10^{-t} & si\ 5 \leq a_{-t-1} \end{cases}$$

$$\bar{x} = \bar{a}\,10^b$$

Será almacenado exactamente en la maquina como **aproximación de x por redondeo.**

Error relativo máximo de representación

Si x pertenece al rango de la maquina, de las dos formas de almacenamiento vistas anteriormente deducimos que:

$$x = a\,10^{b}\ \ y$$

$$\left|a-\bar{a}\right| \le \begin{cases} 10^{-t}\ \ truncamiento \\ 0.510^{-t}\ \ redondeo \end{cases}$$

de donde

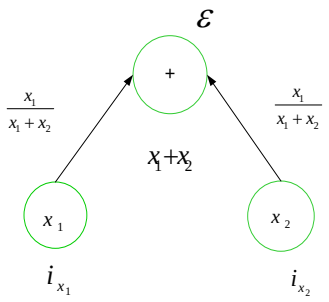
$$\left|er_x\right| = \frac{\left|x-\bar{x}\right|}{\left|x\right|} = \frac{\left|a-\bar{a}\right|10^{b}}{\left|a\right|10^{b}} \le \frac{\left|a-\bar{a}\right|}{0.1} \le \begin{cases} 10^{1-t} & corte \\ 0.510^{1-t} & redondeo \end{cases}$$

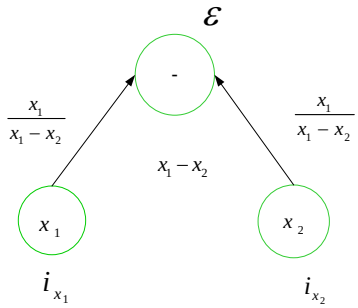
Calculo de la propagación de errores inherentes y de redondeo utilizando la grafica de un proceso

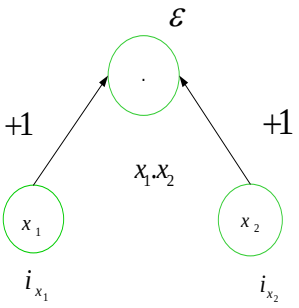
Grafica de un proceso

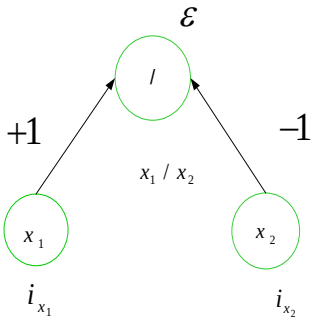
Una grafica de un proceso es la representación de un algoritmo, con una convención para identificar las flechas que aparecen en la grafica, de forma que sea fácil determinar el error relativo total (propagación del inherente mas propagación del de redondeo) en el resultado final

Ejemplo de diagrama para las operaciones elementales





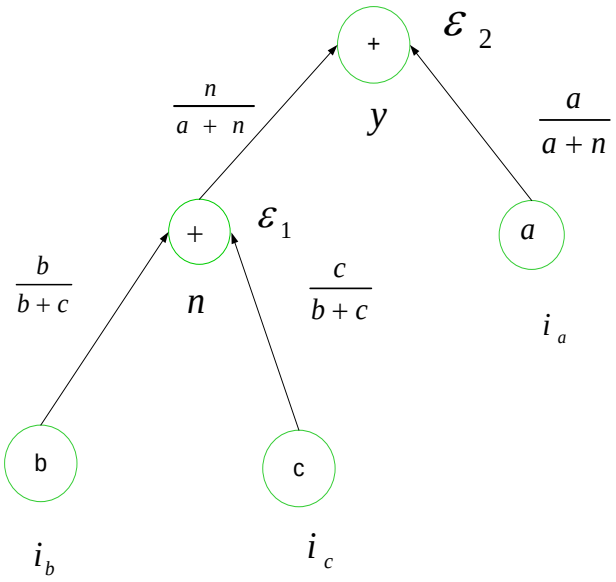




Ejemplo
Queremos efectuar la suma de tres números:

Y=a+b+c, usando el siguiente algoritmo
N=b+c
Y=a+n

Llamaremos a los errores relativos inherentes: i_a, i_b, i_c y a los errores relativos de redondeo en cada suma $\varepsilon_1 \; y \; \varepsilon_2$.
Sabemos que $|\varepsilon_1| \le \mu \; y \; |\varepsilon_2| \le \mu$
La grafica correspondiente es:



El error relativo total en n será:

$$er_n = \frac{i_b b + i_c c}{b + c} + \varepsilon_1$$

Y el error relativo final será:

$$er_y = \frac{er_n n + i_a a}{a + n} + \varepsilon_2 \Rightarrow er_y = \frac{a i_a + b i_b + c i_c}{a + b + c} + \frac{b + c}{a + b + c}\varepsilon_1 + \varepsilon_2$$

Si suponemos que r es una cota para los errores relativos inherentes, obtenemos una cota para el error relativo total:

$$\left|er_y\right| \le \frac{|a| + |b| + |c|}{|a + b + c|} r + \left(1 + \frac{|b + c|}{|a + b + c|}\right)\mu$$

El termino que multiplica a r se lo denomina **condición del problema (** C_p **)** y es el factor de amplificación de los errores relativos inherentes. La condición del problema depende exclusivamente del problema numérico.

El termino que multiplica a μ se denomina **termino de estabilidad (** T_e **)** y depende del problema numérico y del algoritmo.

Estabilidad

Un algoritmo es **numéricamente estable** si y solo si:

$$\frac{C_p + T_e}{C_p} /\gg 1$$

Números de condición

Podría ocurrir que los resultados de un problema tengan poca precisión esto puede deberse a dos cosas: el algoritmo puede no ser el mas conveniente en ese caso se dice que el **algoritmo esta mal condicionado o el algoritmo es numéricamente inestable;** o también puede ser consecuencia del problema numérico mismo, es decir los resultados pueden ser muy sensibles a las perturbaciones de los datos de entrada, independientemente del algoritmo elegido, en ese caso diremos que **el algoritmo es numéricamente inestable o que el problema numérico es inestable.**

Número de condición del problema

Supongamos que tenemos un problema numérico representado por:

$$Y = P(X) \quad P : R^n \to R^m$$

Definimos

$$C_p^i = \frac{\sum_{k=1}^{n} \left| \frac{\partial P_i}{\partial x_k}\left(\bar{X}\right) \right| \left|\bar{x}_k\right|}{\left|P_i\left(\bar{X}\right)\right|} \quad i = 1,\ldots,m$$

Y el número de condición del problema:

$$C_p = \max\left\{C_p^i, i = 1,\cdots m\right\} = \left\| \left[C_p^1, \cdots, C_p^m \right]^t \right\|$$

Definimos el vector de errores relativos inherentes:

$$er_x = \left[er_{x_1}, \cdots, er_{x_n} \right]^t \quad y \sup ongamos\ que \left\| er_x \right\| \le r$$

$$\Rightarrow$$

$$e_{y_i} = \sum_{k=1}^{n} \frac{\partial P_i}{\partial x_k}\left(\bar{X} \right) e_{x_k}$$

$$dividiendo, tomando \bmod ulo\ y\ a \cot ando\ resulta$$

$$\left| er_{y_i} \right| \le C_p^i \left\| er_x \right\|$$

$$Sea\ er_y = \left[er_{y_1}, \cdots, er_{y_m} \right]^t \quad resulta$$

$$\left\| er_y \right\| \le C_p \left\| er_x \right\| \le C_p r$$

Por lo tanto podemos decir que C_p depende de los datos de entrada y es una cota del cambio relativo que el resultado exacto del problema puede tener si se producen perturbaciones relativas en los datos de entrada acotadas por r.

Número de condición del algoritmo.

Antes de hablar del numero de condición del algoritmo, supongamos que la maquina opera con una unidad aritmético-lógica con 2t dígitos y luego almacena en la memoria el resultado redondeado a t dígitos. De acuerdo a lo visto anteriormente en el apartado"Error relativo máximo de representación" podemos escribir:

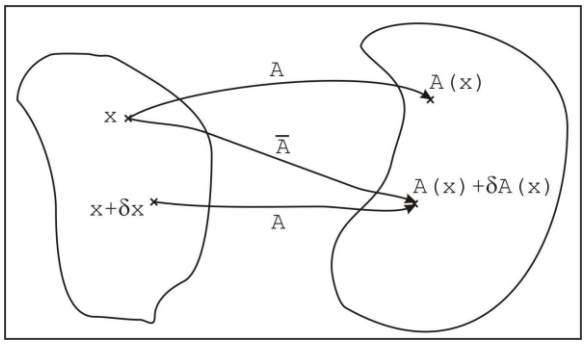
$$fl\left(x\ op\ y \right) = \left(x\ op\ y \right)\left(1 - \varepsilon \right) \quad \left| \varepsilon \right| \le \mu$$

Donde op representa una operación elemental y ε es el error relativo de redondeo o representación de la operación $\left(x\ op\ y \right)$

Simbolizamos por $y = A\left(X \right) : R^n \to R^m$ al algoritmo para resolver el problema: $Y = P\left(X \right) : R^n \to R^m$. Si no existieran errores de redondeo ocurriría: $A\left(X \right) = P\left(X \right)$.

Pero nos interesa cuantificar la influencia de los errores de redondeo, por lo tanto utilizaremos la notación $\bar{Y} = \bar{A}\left(X \right)$ para simbolizar el resultado considerando solo los errores de redondeo.

Ver grafico



A(x) es el valor exacto calculado con el algoritmo A.
$\bar{A}(X)$ es el valor calculado con una máquina.

Sea $y_i = A_i(X)$. Si el cálculo de $A_i(X)$ implica L operaciones, efectuando un análisis retrospectivo de errores tenemos:

$$\bar{y}_i = y_i\left(1+\sum_{k=1}^{L} F_{i,k}(X)\,\varepsilon_k\right) \quad con \quad |\varepsilon_k| \le \mu$$

Llamamos a los $F_{i,k}$ **factores de amplificación.**
Definimos

$$E_i = \sum_{k=1}^{L} |F_{i,k}(X)|$$

$$C_a^i = \frac{E_i}{C_p^i}$$

Con estos valores definimos el número de condición del algoritmo como:

$$C_a = \max\left\{C_a^i, i = 1,\cdots m\right\} = \left\|\left[C_p^1,\cdots,C_p^m\right]^t\right\|$$

Ejemplos

1) Sea el problema de resolver: $y = x^2$ utilizando el algoritmo:

   $y = A(x) = x.x$

   $\bar{y} = fl(x.x) = (x.x)(1-\varepsilon)$

   Como la única operación que hay que hacer es el producto resulta que : E=1

   Ahora debemos calcular $C_p^i = C_p$

   $$C_p = \frac{|\bar{x}|.|\bar{x}| + |\bar{x}|.|\bar{x}|}{|\bar{x}.\bar{x}|} = 2$$

   Luego

   $$C_a = \frac{1}{2}$$

2) Sea el problema de resolver: $y = a + b + c$ utilizando el algoritmo:
$y = A(a,b,c) = a + (b + c)$

Definimos:
Un algoritmo es **numéricamente estable** si y solo si $C_a / \gg 0$
Un algoritmo es **aceptable** si y solo si $C_a \cong 1$