UNIVERSIDAD DEL NORTE
Editorial

JORGE VELÁSQUEZ ZAPATEIRO
VIRGILIO OBESO FERNÁNDEZ

ANÁLISIS NUMÉRICO

notas de clase

Análisis Numérico

Notas de clase

Análisis Numérico

Notas de clase

Jorge Velásquez Zapateiro
Magister en Matemáticas
Virgilio Obeso Fernández
Magister en Matemáticas

Editorial Universidad del Norte
Barranquilla (Colombia)
2008

Velásquez Zapateiro, Jorge.
Análisis numérico : notas de clase / Jorge Velásquez Zapateiro, Virgilio Obeso Fernández. -- Barranquilla : Editorial Universidad del Norte, 2008.

286 p. : il. ; 28 cm.
Incluye referencias bibliográficas
ISBN 978-958-8252-58-2 (impreso)
ISBN 978-958-741-900-9 (PDF)

1. Análisis numérico : notas de clase. I. Velásquez Zapateiro, Jorge. II. Obeso Fernández, Virgilio. III. Tít.

UNIVERSIDAD DEL NORTE
Vigilada Mineducación
www.uninorte.edu.co
Km 5, vía a Puerto Colombia, A.A. 1569
Área metropolitana de Barranquilla (Colombia)

Coordinadora editoral
Zoila Sotomayor O.

Editor
Jorge Velásquez Zapateiro

Diseño de portada
Joaquín Camargo Valle

Corrección de textos
Henry Stein

Impreso y hecho en Colombia
Javegraf (Bogotá)
Printed and made in Colombia

Índice general

Capítulo 1

Números en la computadora

La aparición del computador ha hecho posible la solución de problemas que por su tamaño antes eran excluidos. Desafortunadamente, los resultados son afectados por el uso de la **aritmética de precisión finita**, en la cual para cada número se puede almacenar tantos dígitos como lo permita el diseño del computador.

Así, por ejemplo, de nuestra experiencia esperamos tener siempre expresiones verdaderas, como $2 + 2 = 4$, $3^2 = 9$, $(\sqrt{5})^2 = 5$, pero en la aritmética de precisión finita $\sqrt{5}$ no tiene un solo número fijo y finito que lo representa.

Como $\sqrt{5}$ no tiene una representación de dígitos finitos, en el interior del computador se le da un valor aproximado, cuyo cuadrado no es exactamente 5, aunque con toda probabilidad estará lo bastante cerca a él para que sea aceptable.

1.1. Sistemas decimal y binario

El sistema numérico de uso frecuente es el **sistema decimal**. La base del sistema decimal es 10. Ahora bien, la mayoría de las computadoras no usan el sistema decimal en los cálculos ni en la memoria, sino el sistema binario, que tiene base 2, y su memoria consiste de registros magnéticos, en los que cada elemento sólo tiene los estados **encendido** o **apagado**.

La base de un sistema numérico recibe el nombre de **raíz**. Para el sistema decimal, como se dijo, es 10 y para el binario es 2.
La base de un número se denota con un **subíndice**, así que $(3.224)_{10}$ es 3.224 en base 10, $(1001.11)_2$ es 1001.11 en base 2.

El valor de un número base r es $(abcdefg.hijk)_r$ y se calcula como

$a\times r^6+b\times r^5+c\times r^4+d\times r^3+e\times r^2+f\times r^1+g\times r^0+h\times r^{-1}+i\times r^{-2}+j\times r^{-3}+k\times r^{-4}$

1.2. Del sistema decimal al sistema binario

Consideremos el número 17 en base 10 (de aquí en adelante se omite la base si ésta es 10), éste se puede escribir en base 2 de la siguiente forma:

$$(17)_{10} = (10001)_2$$

en efecto:

$$1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 16 + 1 = 17$$

o también:

$$427.325 \approx (110101011.\overline{0101001})_2$$

Ahora $(1001.11101)_2 = 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 1 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3} + 0 \times 2^{-4} + 1 \times 2^{-5} = 9.90625$.

En general, si N es un número natural, entonces existen cifras $a_0, a_1, a_2, a_3, \ldots,$
$a_K \in \{0, 1\}$ tales que

$$N = a_K \times 2^K + a_{K-1} \times 2^{K-1} + a_{K-2} \times 2^{K-2} + \cdots + a_1 \times 2^1 + a_0 \times 2^0$$

Para encontrar la representación binaria de un número natural N se puede establecer un algoritmo, el cual se concreta si dividimos la expresión anterior entre dos, teniendo entonces que

$$\frac{N}{2} = a_K \times 2^{K-1} + a_{K-1} \times 2^{K-2} + a_{K-2} \times 2^{K-3} + \cdots + a_1 \times 2^0 + \frac{a_0}{2}$$

Si llamamos

$$P_0 = a_K \times 2^{K-1} + a_{K-1} \times 2^{K-2} + a_{K-2} \times 2^{K-3} + \cdots + a_1 \times 2^0$$

entonces

$$\frac{N}{2} = P_0 + \frac{a_0}{2}$$

Luego a_0 es el resto que resulta de dividir a N entre dos. Dividiendo ahora P_0 entre dos se tiene que

$$\frac{P_0}{2} = a_K \times 2^{K-2} + a_{K-1} \times 2^{K-3} + a_{K-2} \times 2^{K-4} + \cdots + a_3 \times 2^1 + a_2 \times 2^0 + \frac{a_1}{2}$$

con lo que

$$\frac{P_0}{2} = P_1 + \frac{a_1}{2}$$

donde

$$P_1 = a_K \times 2^{K-2} + a_{K-1} \times 2^{K-3} + a_{K-2} \times 2^{K-4} + \cdots + a_3 \times 2^1 + a_2 \times 2^0$$

o sea que a_1 es el resto de dividir P_0 entre dos.Y se continua este procedimiento hasta que se encuentre un número K tal que $P_K = 0$. De lo anterior se tiene el siguiente algoritmo:

$$N = 2P_0 + a_0$$

$$P_0 = 2P_1 + a_1$$

$$\vdots$$

$$P_{K-2} = 2P_{K-1} + a_{K-1}$$

$$P_{K-1} = 2P_K + a_K \qquad P_K = 0$$

Ejemplo 1.2.1. *Utilice el algoritmo anterior para escribir 1357 en notación binaria.*

Solución

$$\begin{aligned}
1357 &= 678 \times 2 + 1, & a_0 &= 1\\
678 &= 339 \times 2 + 0, & a_1 &= 0\\
339 &= 169 \times 2 + 1, & a_2 &= 1\\
169 &= 84 \times 2 + 1, & a_3 &= 1\\
84 &= 42 \times 2 + 0, & a_4 &= 0\\
42 &= 21 \times 2 + 0, & a_5 &= 0\\
21 &= 10 \times 2 + 1, & a_6 &= 1\\
10 &= 5 \times 2 + 0, & a_7 &= 0\\
5 &= 2 \times 2 + 1, & a_8 &= 1\\
2 &= 1 \times 2 + 0, & a_9 &= 0\\
1 &= 0 \times 2 + 1, & a_{10} &= 1
\end{aligned}$$

luego

$$1357 = a_{10}a_9a_8a_7a_6a_5a_4a_3a_2a_1a_0 = (10101001101)_2 \qquad \square$$

Supongamos que ahora se tiene $Q \in R$ con $0 < Q < 1$, entonces existen $b_1, b_2, b_3, b_4 \cdots \in \{0, 1\}$ tal que

$$Q = 0.b_1b_2b_3b_4b_5 \ldots$$

y por tanto

$$Q = b_1 \times 2^{-1} + b_2 \times 2^{-2} + b_3 \times 2^{-3} + \cdots + b_k \times 2^{-k} + \ldots$$

Si multiplicamos Q por dos se tiene que

$$2Q = b_1 + b_2 \times 2^{-1} + b_3 \times 2^{-2} + \cdots + b_k \times 2^{-k+1} + \ldots$$

si $F_1 = frac(2Q)$, donde $frac(x)$ es la parte fraccionaria de x, y $b_1 = [|2Q|]$, donde $[|x|]$ es la parte entera de x, entonces

$$F_1 = b_2 \times 2^{-1} + b_3 \times 2^{-2} + \cdots + b_k \times 2^{-k+1} + \ldots$$

Multiplicando ahora F_1 por dos se tiene que

$$2F_1 = b_2 + b_3 \times 2^{-1} + \cdots + b_k \times 2^{-k+2} + \cdots = b_2 + F_2$$

donde $F_2 = frac(2F_1)$, y $b_2 = [|2F_1|]$. Continuando este proceso formamos dos sucesiones $\{b_k\}$ y $\{F_k\}$, dadas por $b_k = [|2F_{k-1}|]$ y $F_k = frac(2F_{k-1})$, con $b_1 = [|2Q|]$ y $F_1 = frac(2Q)$, se tiene entonces que la representación binaria de Q es

$$Q = \sum_{i=1}^{\infty} b_i 2^{-i}$$

Ejemplo 1.2.2. *Utilice el algoritmo anterior para escribir 0.234 en notación binaria.*

Solución

Sea $Q = 0.234$, entonces

$$2Q = 0.468, \qquad b_1 = [|0.468|] = 0 \qquad F_1 = frac(0.468) = 0.468$$

$$2F_1 = 0.936, \qquad b_2 = [|0.936|] = 0 \qquad F_2 = frac(0.936) = 0.936$$

$$
\begin{aligned}
2F_2 &= 1.872, \qquad b_3 = [|1.872|] = 1 \qquad F_3 = frac(1.872) = 0.872 \\
2F_3 &= 1.744, \qquad b_3 = [|1.744|] = 1 \qquad F_4 = frac(1.744) = 0.744 \\
2F_4 &= 1.488, \qquad b_5 = [|1.488|] = 1 \qquad F_5 = frac(1.488) = 0.488 \\
2F_5 &= 0.976, \qquad b_6 = [|0.976|] = 0 \qquad F_6 = frac(0.976) = 0.976 \\
2F_6 &= 1.952, \qquad b_7 = [|1.952|] = 1 \qquad F_7 = frac(1.952) = 0.952 \\
&\vdots \\
&\vdots
\end{aligned}
$$

De lo anterior se tiene que

$$Q = 0.234 = (0.0011101\ldots)_2 \qquad \square$$

1.3. Números en punto flotante

Definición 1.3.1. *Los números en punto flotante son números reales de la forma*

$$\pm\alpha \times \beta^e$$

donde α tiene un número de dígitos limitados, β es la base y e es el exponente que hace cambiar de posición al punto decimal.

Definición 1.3.2. *Un número real x tiene una representación punto flotante normalizada si*

$$x = \pm\alpha \times \beta^e$$

con $\frac{1}{\beta} < |\alpha| < 1$

En el caso que x tenga representación punto flotante normalizada, entonces

$$x = \pm 0, d_1 d_2 ... d_k \times \beta^e$$

donde si $x \neq 0$, $d_1 \neq 0$, $0 \leq d_i < \beta$, $i = 1, 2, 3, ...k$ y $L \leq e \leq U$.

Definición 1.3.3. *El conjunto de los números en punto flotante se le llama* **conjunto de números de máquina**.

El conjunto de número de máquina es finito, ya que si $x = \pm 0.d_1 d_2 d_3 d_4 \cdots d_t \times \beta^e$, con $d_1 \neq 0$, $0 \leq d_i \leq \beta - 1$, $L \leq e \leq U$, entonces para asignarle valor a d_1 hay $\beta - 1$ posibles valores y para d_i, $i = 2, 3, 4, \ldots t$ hay β posibles asignaciones, luego, entonces existirán

$(\beta-1)\underbrace{\beta\beta\cdots\beta}_{t-1 \text{ factores}} = (\beta-1)\beta^{t-1}$, fracciones positivas.

Pero como el número de exponentes es $U - L + 1$, en total habrán $(\beta - 1)\beta^{t-1}(U - L + 1)$ números de máquina positivos y tomando los números máquina negativos, el total de números de máquina es $2(\beta - 1)\beta^{t-1}(U - L + 1) + 1$, teniendo en cuenta que el cero es también un número de máquina. Esto significa que cualquier número real debe ser representado por uno de los $2(\beta-1)\beta^{t-1}(U-L+1)+1$ número de máquina.

Ejemplo 1.3.1. *Como ejemplo tomemos $\beta = 2$, $t = 3$, $L = -2$ y $U = 2$. En este caso, las mantisas serían $(0.100)_2$, $(0.101)_2$, $(0.110)_2$ y $(0.111)_2$, los cuales son la representación en base dos de los números reales $\frac{1}{2}$, $\frac{5}{8}$, $\frac{3}{4}$ y $\frac{7}{8}$ respectivamente. El total de números de máquina aparece en la siguiente tabla:*

TABLA 1

-2	-1	0	1	2
$(0.100)_2 \times 2^{-2}$	$(0.100)_2 \times 2^{-1}$	$(0.100)_2 \times 2^{0}$	$(0.100)_2 \times 2^{1}$	$(0.100)_2 \times 2^{2}$
$(0.101)_2 \times 2^{-2}$	$(0.101)_2 \times 2^{-1}$	$(0.101)_2 \times 2^{0}$	$(0.101)_2 \times 2^{1}$	$(0.101)_2 \times 2^{2}$
$(0.110)_2 \times 2^{-2}$	$(0.110)_2 \times 2^{-1}$	$(0.110)_2 \times 2^{0}$	$(0.110)_2 \times 2^{1}$	$(0.110)_2 \times 2^{2}$
$(0.111)_2 \times 2^{-2}$	$(0.111)_2 \times 2^{-1}$	$(0.111)_2 \times 2^{0}$	$(0.111)_2 \times 2^{1}$	$(0.111)_2 \times 2^{2}$

que corresponden respectivamente a los números reales de la siguiente tabla

TABLA 2

$\frac{4}{32}$	$\frac{8}{32}$	$\frac{16}{32}$	$\frac{32}{32}$	$\frac{64}{32}$
$\frac{5}{32}$	$\frac{10}{32}$	$\frac{20}{32}$	$\frac{40}{32}$	$\frac{80}{32}$
$\frac{6}{32}$	$\frac{12}{32}$	$\frac{24}{32}$	$\frac{48}{32}$	$\frac{96}{32}$
$\frac{7}{32}$	$\frac{14}{32}$	$\frac{28}{32}$	$\frac{56}{32}$	$\frac{112}{32}$

El total de números de máquina es: $2(2-1)\times 2^2(2+2+1)+1=41$, los cuales son:

$$0,\ \pm\frac{4}{32},\ \pm\frac{5}{32},\ \pm\frac{6}{32},\ \pm\frac{7}{32},\ \pm\frac{8}{32},\ \pm\frac{10}{32},\ \pm\frac{12}{32},\ \pm\frac{14}{32},\ \pm\frac{16}{32},\ \pm\frac{20}{32},\ \pm\frac{24}{32}$$
$$\pm\frac{28}{32},\pm\frac{32}{32},\pm\frac{40}{32},\pm\frac{48}{32},\pm\frac{56}{32},\pm\frac{64}{32},\pm\frac{80}{32},\pm\frac{96}{32},\pm\frac{112}{32}$$

1.4. Notación científica normalizada

En la sección anterior hablamos de representación punto flotante y punto flotante normalizado. De acuerdo con eso, si $x\in R$ está en base 10, éste se puede normalizar tomando

$$x=\pm r\times 10^n$$

con $0.1\le r<10$ y n un entero. Obviamente, si $x=0$, entonces $r=0$

Ejemplo 1.4.1. *Represente* 732.5051 *y* -0.005612 *en la notación punto flotante normalizado*

Solución

El número 732.5051 se puede representar como punto flotante normalizado escribiendo $732.5051=0.7325051\times 10^3$.
De la misma manera, $-0.005612=-0.5612\times 10^{-2}$ □

Por otro lado, si x está en el sistema binario, se puede representar en punto flotante normalizado si se escribe de la forma

$$x=\pm q\times 2^m$$

donde $0.5\le q<1$ y m es un entero.

Ejemplo 1.4.2. *Represente* $(101.01)_2$ *y* $(0.0010111)_2$ *en la notación punto flotante normalizado*

Solución

El número $(101.01)_2$ se puede representar como punto flotante normalizado escribiendo $(101.01)_2=0.10101\times 2^3$

De igual forma $(0.0010111)_2 = 0.10111 \times 2^{-2}$

Nota: $(0.1)_2 = 1 \times 2^{-1} = 0.5.$ $\square$

En una computadora los números se representan de la manera como hemos explicado arriba, pero con ciertas restricciones sobre q y m impuestas por la longitud de la palabra. Si suponemos que se tiene una computadora hipotética, la cual llamaremos Norm-32, y si además suponemos que ésta tiene una longitud de palabra de 32 bits (1bit = 1**Bi**nary Digi**t**), éstos se distribuyen de la manera siguiente:

donde los dos primeros espacios son reservados para los signos, y se les asigna cero si el signo es positivo y uno si es negativo, los siguientes siete espacios para el exponente y los restantes para la mantisa.

Dado que un número real distinto de cero $x = \pm q \times 2^m$, siempre puede normalizarse de tal manera que $\frac{1}{2} \leq q < 1$, podemos suponer que el primer bit en q es 1, y por lo tanto no requiere almacenamiento; esto es, si se quiere almacenar en Norm-32 el número 0.828125, el cual es equivalente a $(0.110101)_2 \times 2^0$, esto se hace de la forma indicada:

0	0	0	0	0	0	0	0	0	1	0	1	0	1	.	.	.	.	.	.	0

Ejemplo 1.4.3. *Represente y almacene en punto flotante normalizado -0.125*

Solución
Se tiene que

$$0.125 \times 2 = 0.25 \quad | \quad 0$$
$$0.25 \times 2 = 0.5 \quad | \quad 0$$
$$0.5 \times 2 = 1 \quad | \quad 1$$

entonces

$$-0.125 = (-0.001)_2 = (-0.1)_2 \times 2^{-2}$$

Además

$$2 = (10)_2$$

de manera que su representación es

1	1	0	0	0	0	0	1	0	0	0	0	0	0	0	.	.	.	.	0

Ejemplo 1.4.4. *Represente y almacene en punto flotante normalizado 117.125*

Solución

Sabemos que

$$117 = (1110101)_2$$

y que

$$0.125 = (0.001)_2$$

luego

$$117.125 = (1110101.001)_2 = (0.1110101001)_2 \times 2^7$$

y $7 = (111)_2$, luego para almacenarlo se hace de la siguiente forma:

0	0	0	0	0	0	1	1	1	1	1	0	1	0	1	.	.	.	.	0

Ahora, hemos dicho que $|m|$ no requiere más de 7 bits, lo cual significa que $|m| \leq (1111111)_2 = 2^7 - 1 = 127$, de modo que el exponente de 7 dígitos binarios proporciona un intervalo de 0 a 127, pero el uso exclusivo de enteros positivos para el exponente no permite una representación adecuada para números pequeños; para que esto pueda ser posible se toma el exponente en el intervalo $[-63, 64]$

También hemos dicho que q requiere no más de 24 bits, por lo tanto los números de nuestra máquina hipotética tienen una precisión limitada que corresponde a entre 7 y 8 dígitos decimales, ya que el bit menos significativo en la mantisa representa unidades del orden $2^{-24} \approx 10^{-7}$. Esto quiere decir que números expresados mediante más de siete dígitos decimales serán objeto de una aproximación cuando se dan como datos de entrada o como resultados de operaciones.

1.5. Errores y notación fl(x)

1.5.1. Norma vector

Definición 1.5.1. *Sea V un espacio vectorial. Una función $g : V \longrightarrow \mathbb{R}$ es una norma vector si $\forall x, y \in V$ y α un escalar, se cumple que*

1. $g(x) \geq 0$ y $g(x) = 0$ si y sólo si $x = 0$
2. $g(\alpha x) = |\alpha| g(x)$

3. $g(x+y) \leq g(x) + g(y)$

Entre las clases de norma están las denominadas **p-normas**, las cuales se definen como

Definición 1.5.2. *Para* $1 \leq p < \infty$ *se definen así:*

$$||x||_p = \Big[\sum_{i=1}^{n} |x_i|^p\Big]^{\frac{1}{p}}$$

Otra norma muy usada en análisis numérico es la norma del máximo cuya definición presentamos ahora:

Definición 1.5.3. *Sea* $x \in R^n$*, definimos la* **norma del máximo** *como*

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

Nota: Si $p = 1$, se tiene que $||x||_1 = \Big[\sum_{i=1}^{n} |x_i|\Big]$, y si $p = 2$, se tiene que $||x||_2 = \Big[\sum_{i=1}^{n} |x_i|^2\Big]^{\frac{1}{2}}$

1.5.2. Error absoluto y relativo

Definición 1.5.4. *Si* $x \in \mathbb{R}^n$ *y* $x^\star \in \mathbb{R}^n$ *es una aproximación a* x*, definimos el error absoluto como*

$$E = \|x - x^\star\|$$

Definición 1.5.5. *Si* $x \in \mathbb{R}^n$ *y* $x^\star \in \mathbb{R}^n$ *es una aproximación a* x*, definimos el error relativo como*

$$E_r = \frac{\|x - x^\star\|}{\|x\|} \qquad x \neq 0$$

Nota: Si n = 1, entonces $E = |x - x^\star|$ y $E_r = \dfrac{|x - x^\star|}{|x|} \qquad x \neq 0.$

Definición 1.5.6. *Si* $x \in \mathbb{R}$ *y* $x^\star \in \mathbb{R}$ *es su aproximación, se dice que* $x^\star$ *tiene por lo menos p - β cifras significativas exactas si* $E \leq \frac{1}{2}\beta^{-p}$*.*

Definición 1.5.7. *Si $x \in \mathbb{R}$ y $x^\star \in \mathbb{R}$ es su aproximación, se dice que $x^\star$ tiene por lo menos p - β dígitos significativos exactos si $E_r \leq \frac{1}{2}\beta^{-p+1}$.*

Como hemos dicho, los números pueden sufrir aproximaciones cuando se dan como datos de entrada o como resultados de operaciones; estas aproximaciones se pueden hacer de dos formas:

- **Truncamiento:** En este proceso el número se representa por medio del mayor número de la máquina menor que el número dado.
- **Redondeo:** En este proceso el número se representa por el número de máquina más cercano al número dado.

Los errores de redondeo pueden ser sutiles cuando se realizan cálculos individuales, pero éstos pueden perjudicar la precisión computacional si existen dos situaciones:

1. Cuando se suman una sucesión de números, especialmente si éstos decrecen en valor absoluto.
2. Cuando se hace la diferencia entre dos números casi idénticos, ya que se cancelan los dígitos principales.

Por lo anterior, si deseamos estimar el error cometido al aproximar un número positivo $x = \pm 0.d_1d_2...d_td_{t+1}... \times \beta^m$, $d_i \neq 0$ mediante un número de máquina, notado $fl(x)$, esto se hace de la siguiente forma:

- Con redondeo
  1. $fl(x) = \pm 0.d_1d_2...d_t \times \beta^m$ si $0 \leq d_{t+1} < \frac{\beta}{2}$.
  2. $fl(x) = \pm(0.d_1d_2...d_t + \beta^{-t}) \times \beta^m$ si $\frac{\beta}{2} \leq d_{t+1} < \beta$
- Con truncamiento
  $fl(x) = \pm 0.d_1d_2...d_t \times \beta^m$

Se puede probar que si hay redondeo, los errores absoluto y relativo son $E \leq \frac{1}{2}\beta^{m-t}$ y $E_r \leq \frac{1}{2}\beta^{1-t}$; si hay truncamiento, son $E \leq \beta^{m-t}$ y $E_r \leq \beta^{1-t}$.

En nuestra computadora hipotética Norm-32, si $x = (0.d_1d_2d_3...d_{24}d_{25}d_{26}...) \times 2^m$, el número $x^{'} = (0.d_1d_2d_3...d_{24}) \times 2^m$ obtenido por truncamiento se encuentra a la izquierda de x en la recta real y el número $x^{''} = (0.d_1d_2d_3...d_{24} + 2^{-24}) \times 2^m$ obtenido por redondeo se localiza a la derecha de x (ver figura 1.1). El más cercano a x entre $x^{'}$ y $x^{''}$ se selecciona para representar a x en

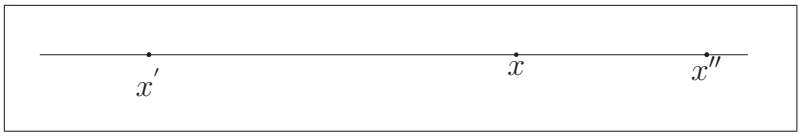

Figura 1.1

la computadora. Obsérvese que si $x^{'}$ representa mejor a x, entonces

$$|x - x^{'}| \leq \frac{1}{2}|x^{'} - x^{''}| \leq \frac{1}{2} \times 2^{-24} \times 2^{m} = 2^{m-25}$$

Luego el error relativo es

$$E_r = \frac{|x - x^{'}|}{|x|} \leq \frac{2^{m-25}}{q \times 2^{m}} = \frac{2^{-25}}{q} \leq \frac{2^{-25}}{\frac{1}{2}} = 2^{-24}$$

Y si x está más cercano a $x^{''}$, entonces

$$|x - x^{''}| \leq \frac{1}{2}|x^{''} - x^{'}| = 2^{m-25}$$

Luego entonces el error relativo es

$$E_r = \frac{|x - x^{''}|}{|x|} \leq 2^{-24}$$

Es posible que durante el transcurso del cálculo se genere un número $\pm q \times 2^{m}$, donde m quede por fuera del rango permitido por la computadora. Si m es demasiado grande, se dice que se produce un sobreflujo o desbordamiento por exceso (**overflow**) y se interrumpen los cálculos; si m es, por el contrario, muy pequeño, se dice que ocurre un subflujo o desbordamiento por defecto (**underflow**) y suele dársele el valor cero; en Norm-32 esto ocurre para $m > 127$ o $m < -127$ respectivamente.

Definición 1.5.8. *Sean x e y puntos flotantes, definimos $\oplus$, $\ominus$, $\otimes$ y $\oslash$, llamadas operaciones de punto flotante, de la siguiente forma:*

$$x \oplus y = fl(fl(x) + fl(y))$$
$$x \ominus y = fl(fl(x) - fl(y))$$
$$x \otimes y = fl(fl(x) \times fl(y))$$
$$x \oslash y = fl(fl(x)/fl(y))$$

donde $+, -, \times, /$ son las operaciones usuales

Para ilustrar estas operaciones sean $x, y \in \mathbb{R}$ tales que $fl(x) = \dfrac{24}{32}$ y $fl(y) = \dfrac{7}{32}$, son números punto flotantes, dados en el ejemplo 1.3.1, entonces

$$x \oplus y = fl\Big(\frac{24}{32} + \frac{7}{32}\Big) = fl\Big(\frac{31}{32}\Big) = \frac{32}{32} = 1$$

$$x \ominus y = fl\Big(\frac{24}{32} - \frac{7}{32}\Big) = fl\Big(\frac{17}{32}\Big) = \frac{16}{32}$$

$$x \otimes y = fl\Big(\frac{24}{32} \times \frac{7}{32}\Big) = fl\Big(\frac{21}{128}\Big) = \frac{20}{128} = \frac{5}{32}$$

$$x \oslash y = fl\Big(\frac{24}{32}\Big/\frac{7}{32}\Big) = fl\Big(\frac{24}{32} \times \frac{32}{7}\Big) = fl\Big(\frac{24}{7}\Big) = fl\Big(\frac{768}{224}\Big) = \frac{784}{224} = \frac{112}{7}$$

Observemos que si $fl(x) = \frac{96}{32}$ y $fl(y) = \frac{4}{32}$, entonces

$$x \oslash y = fl\Big(\frac{96}{32}\Big/\frac{4}{32}\Big) = fl\Big(\frac{96}{4}\Big) = \frac{112}{32}$$

(fenómeno **overflow**), ya que $\frac{96}{4} > \frac{112}{32}$

En resumen, si $fl(x)$ es el número de máquina más cercano a x y tomamos $\delta = \dfrac{fl(x) - x}{x}$, entonces $fl(x) = x(1 + \delta)$ y $|\delta| \leq \frac{1}{2}\beta^{1-t} = \epsilon$ o $|\delta| \leq \beta^{1-t} = \epsilon$ usando aritmética de redondeo o truncamiento respectivamente. El número ϵ se conoce como **error de redondeo unitario o unidad de redondeo**. En Norm-32 la unidad de redondeo es 2^{-24}.

1.6. Análisis de error

Sea $\otimes$ un operador con el cual representamos una cualquiera de las operaciones básicas +, - $\times$, $\div$, y sean x e y dos números cualesquiera, y si $x \otimes y$ debe calcularse y almacenarse, entonces la variación computada de $x \otimes y$ es $fl(x \otimes y)$, por consiguiente cabe preguntarse ¿qué tan preciso es $fl(x \otimes y)$? Por lo anterior $fl(x \otimes y) = (x \otimes y)(1 + \delta)$ con $|\delta| \leq \epsilon$, si x, y son números de la máquina.

Si x e y no son números de la máquina, entonces $fl[fl(x) \otimes fl(y)] = (x + (1 + \delta_1) \otimes (y + (1 + \delta_2))(1 + \delta_3)$ con $\delta_i \leq \epsilon$.

1.7. Épsilon de la máquina

Ya se ha comentado que si una máquina funciona con una base β y utiliza t posiciones en la mantisa de sus números de punto flotante, entonces

$$fl(x) = x(1+\delta), \qquad |\delta| \leq \epsilon$$

donde $\epsilon = \frac{1}{2}\beta^{1-t}$ en caso de redondeo y $\epsilon = \beta^{1-t}$ en caso de truncamiento. El número ϵ (error de redondeo unitario) es una característica de la máquina, de su sistema operativo y de la manera en que efectúa los cálculos. El épsilon de la máquina es importante porque caracteriza la precisión de la máquina en tal forma que los programas computacionales sean razonablemente independientes de la máquina en la que se ejecutan; sirve además como criterio de parada de los algoritmos.

Definición 1.7.1. *Se define ϵ de la máquina (abreviadamente"macheps") como el número positivo más pequeño τ tal que sumado con 1 da como resultado un número mayor que 1, esto es, $\epsilon = \{\tau : \tau + 1\}$.*

Este número es posible hallarlo con el siguiente algoritmo:

ALGORITMO

```
Inicio
   eps ← 1.0
   Mq 1.0 + eps > 1.0
      epsilon ← eps
      eps ← 0.5 × eps
   FMq
   Escriba épsilon
Fin
```

En el caso de nuestra máquina hipotética "macheps"$= 2^{-24}$

1.8. Notación O de Landau

Con el propósito de determinar qué tan rápido crece o decrece una función, Edmund Landau introdujo la notación de órdenes de magnitud que lleva su nombre. Por ejemplo, el desarrollo de Taylor de la función exponencial se puede escribir como

$$e^x = 1 + x + \frac{x^2}{2!} + O(x^3), \qquad x \to 0$$

donde el último término significa que el término de error del teorema de Taylor es menor, en valor absoluto, que una constante que multiplica a x^3, cuando x está cerca de 0.

De manera formal se tiene la siguiente definición:

Definición 1.8.1. *Dos funciones $f(x)$ y $g(x)$ de variable real son del mismo orden de magnitud, escrito $f(x) = O(g(x))$, más propiamente,*

$$f(x) = O(g(x)), \qquad x \to \infty$$

si y sólo si existen constantes N y C tales que

$$|f(x)| \leq C|g(x)|, \qquad \forall x > N$$

Lo que intuitivamente significa que $f(x)$ no crece más rápido que $g(x)$

En general, si $a \in \mathbb{R}$, escribiremos

$$f(x) = O(g(x)), \qquad x \to a$$

si y sólo si existen constantes α, β tales que

$$|f(x)| \leq \beta|g(x)|, \qquad |x - a| < \alpha$$

Normalmente, el contexto determina el valor de a o si ésta es ∞.

Se denomina de orden constante a una función $O(1)$, logarítmico, si es $O(\log(n))$, lineal, si $O(n)$, cuadrático para $O(n^2)$, polinómico para $O(n^k)$ con $k \in \mathbb{N}$, y exponencial para $O(c^n)$ con $0 < c \in \mathbb{R}$. Es fácil comprobar que $O(\log(n)) = O(\log(n^c))$.

Además de la notación O grande, Landau también introdujo la notación o pequeña. Informalmente, $f(x) = o(g(x))$ significa que f crece mucho más lentamente que g y se hace cada vez más insignificante respecto a ella conforme crece x.

Formalmente, se tiene:

Definición 1.8.2. *$f(x) = o(g(x))$ para $x \to \infty$ si y sólo si $\forall \gamma > 0$ existe una constante N tal que*

$$|f(x)| \leq \gamma|g(x)|, \qquad \forall x > N$$

En general, se tiene que

Definición 1.8.3. *$f(x) = o(g(x)$, $x \to a$ si y sólo si $\forall\gamma > 0$ existe una constante η tal que*

$$|f(x)| \leq \gamma|g(x)|, \qquad \forall|x-a| < \eta$$

Cuando a es cero o infinito, y queda claro su valor por el contexto, se omite.

Es fácil observar que los símbolos O y o son equivalentes a $\leq$ y $<$.

El símbolo O tiene propiedades, las cuales mostramos en el siguiente teorema:

Teorema 1.8.1. *1. Si $f(x) = O(g(x))$ y $h(x) = O(g(x))$, entonces $\lambda f(x) + \nu h(x) = O(g(x))$*

2. Si $f(x) = O(g(x))$, $h(x) = O(k(x))$, entonces $f(x)h(x) = O(g(x)k(x))$

3. Si $f(x) = O(g(x))$, $g(x) = O(h(x))$, entonces $f(x) = O(h(x))$

Demostración. 1.- Como $f(x) = O(g(x))$ y $h(x) = O(g(x))$, entonces existen constantes N_1 y N_2 tales que

$$|f(x)| \leq N_1|g(x)| \qquad \text{y} \quad |h(x)| \leq N_2|g(x)|$$

luego

$$|\lambda f(x) + \nu g(x)| \leq |\lambda||f(x)| + |\nu||g(x)|$$

de modo que

$$|\lambda f(x) + \nu g(x)| \leq |\lambda|N_1|g(x)| + |\nu|N_2|g(x)| = (|\lambda|N_1 + |\nu|N_2)|g(x)|$$

por tanto existe una constante $N = |\lambda|N_1 + |\nu|N_2$ tal que

$$|\lambda f(x) + \nu g(x)| \leq N|g(x)|$$

de modo que

$$\lambda f(x) + \nu h(x) = O(g(x))$$

2.- Si $f(x) = O(g(x))$, $h(x) = O(k(x))$, entonces existen constantes N_1 y N_2 tales que

$$|f(x)| \leq N_1|g(x)| \qquad \text{y} \qquad |h(x)| \leq N_2|k(x)|$$

luego

$$|f(x)h(x)| = |f(x)||h(x)| \leq N_1|g(x)|N_2|k(x)| = (N_1N_2)|g(x)k(x)|$$

de modo que existe una constante $N = N_1N_2$ tal que

$$|f(x)h(x)| \leq N|g(x)k(x)|$$

y por lo tanto

$$f(x)h(x) = O(g(x)k(x))$$

3.- Si $f(x) = O(g(x))$, $g(x) = O(h(x))$, existen constantes N_1 y N_2 tales que

$$|f(x)| \leq N_1|g(x)| \qquad \text{y} \qquad |g(x)| \leq N_2|h(x)|$$

por lo tanto

$$|f(x)| \leq N_1|g(x)| \leq N_1(N_2|h(x)|) = N|h(x)|$$

donde $N = N_1N_2$, de modo que existe $N = N_1N_2$ tal que

$$|f(x)| \leq N|h(x)|$$

y por lo tanto $f(x) = Oh(x)$ □

Por supuesto, la notación O también permite comparar sucesiones, $\{a_n\}$, $\{b_n\}$ de forma $a_n \leq b_n$ de acuerdo con la siguiente definición:

Definición 1.8.4. *Sean $\{a_n\}$ y $\{b_n\}$ dos sucesiones, con $b_n > 0 \quad \forall n$, si existe una constante C tal que*

$$|a_n| \leq Cb_n$$

con $n \geq N$, para algún número natural N, entonces se dice que

$$a_n = O(b_n)$$

La definición anterior es equivalente a decir que $\lim\limits_{n\to\infty} \dfrac{|a_n|}{b_n} = L \neq \infty$

Ejemplo 1.8.1. *1. Como $\dfrac{1}{n^2} \leq \dfrac{2}{n(n+1)}$, entonces $\dfrac{1}{n^2} = O\Big(\dfrac{1}{n(n+1)}\Big)$*

2. Se sabe que $|\cos n| \leq 1$, luego $\cos n = O(1)$

3. $\operatorname{sen}\dfrac{x}{n} = O\Big(\dfrac{1}{n}\Big)$, ya que $\Big|\operatorname{sen}\dfrac{x}{n}\Big| \leq \dfrac{x}{n} = |x|\dfrac{1}{n}$

4. Como $\lim\limits_{n\to\infty} \dfrac{\sqrt{n+1}-\sqrt{n}}{1/\sqrt{n}} = \lim\limits_{n\to\infty} \dfrac{\sqrt{n}}{\sqrt{n+1}+\sqrt{n}} = \dfrac{1}{2}$, entonces

$$\sqrt{n+1}-\sqrt{n} = O\Big(\frac{1}{\sqrt{n}}\Big) \qquad \square$$

Hay 3 símbolos más, pero sólo presentaremos uno de ellos, el equivalente a =: se dice que $f(x) = \theta(g(x))$ si y sólo si $f(x) = O(g(x))$ y $g(x) = O(f(x))$. Note la diferencia entre escribir $f(x) = O(g(x))$ y $f(x) = \theta(g(x))$.

En este texto nos limitaremos al uso de la notación O grande, sobre todo para simplificar la escritura del término de error. No utilizaremos ninguno de los otros símbolos de Landau.

1.9. Pérdida de cifras significativas

Toda operación de punto flotante en un proceso computacional puede dar lugar a un error, que puede aumentar o disminuir. Una de las maneras más comunes de aumentar la importancia de un error se conoce como **pérdida de cifras significativas**. La pérdida de cifras significativas se puede generar por la longitud de la palabra que almacena los números, y en este caso es inevitable, pero también se puede tener por la programación, en este caso es evitable. Estos últimos aparecen, por ejemplo, al restar números muy cercanos. Supongamos que vamos a calcular $z = x - y$ y que tenemos aproximaciones $x^\star$ y $y^\star$ para x y y respectivamente, cada una de las cuales es buena hasta r cifras. Entonces $z^\star = x^\star - y^\star$ es una aproximación para z que también es buena hasta r cifras significativas, a menos que $x^\star$ y $y^\star$ coincidan en una o más cifras. En este último caso habrá cancelación durante la sustracción, y por lo tanto $z^\star$ será exacto hasta menos de r cifras.

Por ejemplo, si $x = 0.3721478693$ y $y = 0.3720230572$, entonces $x - y = 0.0001248121 = 0.1248121 \times 10^{-3}$. Si los cálculos se llevan en una computadora decimal con mantisa de cinco cifras, entonces $fl(x) = x^\star = 0.37215$ y $fl(y) = y^\star = 0.37202$, luego $z^\star = fl(x) - fl(y) = x^\star - y^\star = 0.00013$. El error relativo es

$$E_r = \left| \frac{(x - y) - (x^\star - y^\star)}{x - y} \right| \approx 4\,\%$$

que es un error relativo muy grande.

La pérdida de cifras significativas se puede evitar (cuando sea posible) reescribiendo las ecuaciones bien sea utilizando artificios algebraicos, trigonométricos o series de Taylor.
Por ejemplo, calcular $y = \sqrt{x+1} - 1$ está implicando una pérdida de cifras significativas para valores cercanos a 0, ya que en este caso $\sqrt{x+1} \approx 1$, luego

se podría evitar esta pérdida reescribiendo la ecuación de la forma

$$y = (\sqrt{x+1} - 1)\frac{\sqrt{x+1}+1}{\sqrt{x+1}+1} = \frac{x}{\sqrt{x+1}+1}$$

Ejemplo 1.9.1. *Con el argumento anterior, si*

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

calcular $f(400)$ *con una aproximación por redondeo a cinco cifras significativas*

Solución
Como se pide una aproximación a cinco cifras significativas, entonces $\sqrt{402} \approx 20.05$ y $\sqrt{401} \approx 20.025$, de modo que

$$f(400) \approx 400^2(20.05 - 20.025) = 4000$$

Ahora sí reescribimos la $f(x)$ por una función equivalente dada por

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

y evaluamos la función $g(x)$ en $x = 400$, con una aproximación por redondeo a cinco cifras significativas se tiene que

$$g(400) \approx \frac{400^2}{\sqrt{402} - \sqrt{401}} = \frac{400^2}{20.05 - 20.025} \approx 3992.51$$

Pero el valor exacto de $f(x) = 3992.5218\cdots$; observamos entonces que $g(400)$ coincide con el valor real de $f(400)$ hasta la quinta cifra significativa.□

Otro caso en el que se puede presentar pérdida de cifras significativas es cuando se evalúa un polinomio. Una forma más eficiente para hacerlo y al mismo tiempo evitar dicha pérdida es mediante el uso del **método de Horner** de las multiplicaciones encajadas, el cual consiste en lo siguiente:
Dado el polinomio

$$P_n(x) = a_n x^n + a_{n-1}x^{n-1} + a_{n-2}x^{n-2}\cdots + a_3x^3 + a_2x^2 + a_1x + a_0$$

éste se podría escribir de la forma

$$Q_n(x) = (((((\cdots(a_nx + a_{n-1})x + a_{n-2})x + \cdots + a_3)x + a_2)x + a_1)x + a_0)$$

Ejemplo 1.9.2. *El polinomio*

$$P_4(x) = x^4 - 2x^3 + 3x^2 + 3x + 1$$

puede escribirse de la forma

$$Q_4(x) = ((((x-2)x+3)x+3)x+1)$$

Ejemplo 1.9.3. *Use aproximación a tres cifras con redondeo para evaluar* $P_4(3.21)$ *y* $Q_4(3.21)$*, siendo* $P_4(x)$ *y* $Q_4(x)$ *los polinomios dados en el ejemplo 1.9.2.*

Solución
Primero observemos que $P_4(3.21) = Q_4(3.21) = 81.564445481\cdots$

Además

$$P_4(3.21) \approx (3.21)^4 - 2(3.21)^3 + 3(3.21)^2 + 3(3.21) + 1$$

$$= 106 - 2 \times 33.1 + 3 \times 10.3 + 3 \times 3.21 + 1 = 81.33$$

y

$$Q_4(3.21) \approx ((((3.21-2)3.21+3)3.21+3)3.21+1)$$

o sea que

$$Q_4(3.21) \approx (((3.9+3)3.21+3)3.21+1) = ((22.1+3)3.21+1) = 80.6+1 = 81.6$$

□

Notemos que los errores absolutos al evaluar $P_4(3.21)$ y $Q_4(3.21)$ son respectivamente, $E_P = 0.2344$ y $E_Q = 0.03555$. Como se dijo, resulta más eficiente cuando se utiliza el método de Horner de multiplicaciones encajadas.

Otro ejemplo sería evaluar la función $f(x) = 1 - \cos x$. Al igual que antes, $1 \approx \cos x$ para valores cercanos a cero, y se presentará pérdida de dígitos significativos, entonces la función puede reescribirse como

$$f(x) = 1 - \cos x = \frac{(1-\cos x)(1+\cos x)}{1+\cos x} = \frac{\operatorname{sen}^2 x}{1+\cos x}$$

la cual puede calcularse con mucha más exactitud para valores cercanos a cero, o también a partir de la fórmula de Taylor alrededor de 0, esto es:

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + ...$$

luego

$$f(x) = 1 - \cos x = 1 - (1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + ...) = \frac{x^2}{2!} - \frac{x^4}{4!} + \frac{x^6}{6!} + ...$$

y si x está cercano de cero, podemos usar una serie truncada, tal como

$$f(x) = \frac{x^2}{2} - \frac{x^4}{24} + \frac{x^6}{720} + O(x^7)$$

luego si $x \to 0$, entonces

$$f(x) \approx \frac{x^2}{2} - \frac{x^4}{24} + \frac{x^6}{720} = \frac{x^2}{2}\Big(1 - \frac{x^2}{12} + \frac{x^4}{360}\Big) = \frac{x^2}{2}\Big(1 - \frac{x^2}{12}\Big(1 - \frac{x^2}{30}\Big)\Big)$$

Ejemplo 1.9.4. *Considere la función*

$$f(x) = \frac{e^x - e^{-x}}{x}$$

Use aritmética de redondeo a tres cifras decimales para evaluar $f(0.1)$, luego reemplace cada función exponencial por su tercer polinomio de Taylor y evalué entonces el polinomio resultante en $x = 0.1$

Solución
Al usar aritmética de redondeo a tres cifras se tiene entonces que $e^{0.1} = 1.11$ y $e^{-0.1} = 0.905$, luego

$$f(0.1) = \frac{1.11 - 0.905}{0.1} = 2.05$$

Pero el tercer polinomio de Taylor para e^x es

$$e^x = 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + O(x^4)$$

y para e^{-x} es

$$e^{-x} = 1 - x + \frac{x^2}{2} - \frac{x^3}{6} + O(x^4)$$

de modo que

$$e^x - e^x = 1 + x + \frac{x^2}{2} + \frac{x^3}{6} + O(x^4) - (1 - x + \frac{x^2}{2} - \frac{x^3}{6} + O(x^4)) = 2x + 2\frac{x^3}{6} + O(x^4)$$

luego el polinomio de Taylor para $f(x)$ es

$$P_3(x) = \frac{2x + \frac{2x^3}{6}}{x} = 2 + \frac{x^2}{3} + O(x^4)$$