Métodos numéricos para ingeniería

Juan Félix Ávila Herrera
Enrique Vílchez Quesada

MATERIAL WEB

```
const PLACING = 0;
const DRAWING = 1;
var center = new cfloat(0, 0);
var radius = 1.5;
var root_colors = [];
var mode = PLACING;
var tolerance = 0.1;
var r_color;
var color_text;
var directions;
var coloring = 0;
var a = 1;
```

alphaæditorial

ENRIQUE VÍLCHEZ QUESADA

Máster en Tecnología e Informática Educativa de la Escuela de Informática de la Universidad Nacional de Costa Rica (UNA). Máster en Entornos Virtuales de Aprendizaje de la Universidad Técnica Nacional de Costa Rica. Diplomado en Diseño, Desarrollo y Evaluación de *Software* Educativo de la Universidad del Norte de Colombia. Licenciado en la Enseñanza de la Matemática de la Universidad Nacional de Costa Rica.

Profesor de la Escuela de Informática de la Universidad Nacional de Costa Rica, imparte los cursos de Estructuras Discretas, Fundamentos de Informática e Investigación de Operaciones, fungiendo como docente y coordinador. Investigador asociado a distintas actividades y proyectos vinculados con el desarrollo de *software* y de materiales educativos computarizados.

Ganador del premio Wolfram Innovator Award 2021 otorgado por la empresa Wolfram Research por aportes en el área de la innovación educativa empleando tecnologías Wolfram. Miembro asociado del Comité Latinoamericano de Matemática Educativa (CLAME). Mención Honorífica al Galardón *Alumni* 2019 con Sello Universidad Técnica Nacional de Costa Rica, por el desempeño y trayectoria profesional en la docencia e investigación.

Autor de diversos artículos y libros y conferencista a nivel nacional e internacional en múltiples congresos relacionados con los campos de matemática educativa, informática educativa y matemática aplicada.

JUAN FÉLIX ÁVILA HERRERA

Doctor en Investigación de Operaciones del *Florida Institute of Technology*, (EE. UU.). Máster en Computación del Instituto Tecnológico de Costa Rica. Licenciado en Matemáticas de la Universidad de Costa Rica. Profesor catedrático de la Escuela de Informática de la Universidad Nacional de Costa Rica (UNA) y profesor de la Escuela de Matemáticas de la Universidad de Costa Rica.

Autor de publicaciones sobre matemáticas y computación. Ha participado en numerosos congresos y simposios, tanto a nivel nacional como internacional, en las áreas ya mencionadas.

Juan Félix Ávila Herrera
Enrique Vílchez Quesada

Métodos numéricos para ingeniería

alphaeditorial

Catalogación en la publicación – Biblioteca Nacional de Colombia

Ávila Herrera, Juan Félix, autor
Métodos numéricos para ingeniería / Juan Félix Ávila Herrera y Enrique Vílchez Quesada. -- Primera edición. -- Bogotá : Alpha Editorial, 2023.
370 páginas.

Incluye índice analítico.

ISBN 978-958-778-891-4 -- 978-958-778-892-1 (digital)

1. Métodos numéricos - Procesamiento de datos 2. Análisis numérico - Procesamiento de datos 3. Wolfram (Lenguaje de programación de computadores) 4. Ingeniería I. Vílchez Quesada, Enrique, autor

CDD: 518.0285 ed. 23 CO-BoBN– a1115607

Alpha Editorial S.A.
Calle 62 20–46 /esquina, Bogotá
Teléfono (601) 746 0102
cliente@alpha-editorial.com
www.alpha-editorial.com

Libros digitales
www.alphaeditorialcloud.com

Primera edición: Bogotá, 2023

Edición: Samantha Córdoba Hernández
Revisión de estilo: Mercedes Rangel
Portada: Ana Paula Santander Durán

ISBN 978-958-778-891-4
ISBN 978-958-778-892-1 DIGITAL

Impreso en Colombia
Printed and made in Colombia

A mi familia por renovar
continuamente mi fe en la
humanidad. Que Dios los
bendiga hoy y siempre.

JUAN FÉLIX ÁVILA HERRERA

A mi familia que constituye el
reflejo más puro de la felicidad
y el amor profundo.

ENRIQUE VÍLCHEZ QUESADA

Índice general

Índice de figuras

Índice de tablas

Introducción

Los métodos numéricos constituyen un área de la matemática encargada de encontrar soluciones numéricas aproximadas para distintos tipos de problemas tales como: la solución de ecuaciones con una o varias variables, la interpolación polinomial, el cálculo de derivadas e integrales, la solución de ecuaciones diferenciales, la solución de sistemas de ecuaciones lineales, entre otros.

En ingeniería, resalta la importancia de este campo científico como un recurso para la resolución de diversos tipos de problemas, tratando de minimizar la cantidad de cálculos necesarios. Para ello, el empleo de un ambiente de programación se torna indispensable con el objetivo de implementar los distintos algoritmos que se desarrollarán en la teoría clásica. En este libro se ha optado por utilizar pseudocódigo y el lenguaje de programación *Wolfram Language*, además de cierto tipo de documentos de uso libre llamados CDF (documentos con un formato computable), diseñados con fines didácticos por los autores.

El texto recurre a una imagen de mucha relevancia, tal y como se aprecia en la figura 1. Este ícono señala la existencia de un CDF en código fuente de *Wolfram Language* que puede ser descargado para uso local en la computadora del usuario. Los CDF son importantes para el lector con la intención de profundizar los temas abarcados en el libro, observar la respuesta de un ejercicio o retroalimentarse mediante una exposición paso a paso de las distintas técnicas y métodos numéricos utilizados.

Figura 1. Imagen iconográfica del libro.

También, en el texto el lector encontrará este tipo de recuadro que se emplea para resaltar algún aspecto importante dentro de las explicaciones realizadas por los autores. Asimismo, se usan otros recuadros titulados como **Ficha N.º** que resumen los elementos teóricos más relevantes del tema abordado. Además, cabe destacar que en este texto se supondrá como separador decimal el uso del punto. Adicionalmente, se emplean otros separadores similares a este recuadro para hacer referencia a definiciones y teoremas, todo ello con el objetivo de mejorar la lectura comprensiva de esta obra.

Todas las direcciones a los archivos descargables se encuentran disponibles en:

`https://www.alpha-editorial.com/Papel/9789587788914/Metodos+Numericos+Para+Ingenieria`

También puede seguir los siguientes pasos:

1) Acceder a https://www.alpha-editorial.com
2) Escribir el nombre del libro en el buscador.
3) En la página del libro encontrará el vínculo al material web.

El PDF publicado en esta dirección electrónica presenta una serie de códigos junto a los cuales se asocia la fuente de las implementaciones programadas en *Wolfram Language*. Cada código socializado se menciona de forma explícita en algún lugar del texto. A este respecto, el lector, al encontrarse con un código dentro del libro, debe buscar la dirección *URL* vinculada con él, si desea recibir la información complementaria.

La obra aquí desarrollada ofrece un interesante enfoque que combina de manera versátil la teoría, la práctica y la simulación de procesos mediante el uso del software *Wolfram Mathematica*.

Heredia, Costa Rica, 2022. J. Ávila y E. Vílchez

Capítulo 1

Aritmética de computadora y análisis de algoritmos

"Dadme un punto de apoyo y moveré el mundo".
Arquímedes de Siracusa

1.1. Números en punto flotante

El primer problema que abordaremos es la representación de los números reales en una calculadora o computadora. ¿Podemos representar todos los números reales en una computadora? La respuesta es definitivamente no. En efecto, como sabemos, hay una cantidad infinita de números reales y (por lo menos hasta el año 2023) solo una cantidad finita de espacio de memoria disponible en las calculadoras y computadoras. Por lo tanto, no podemos representar o (intuitivamente) "vertir" todos los números reales en una computadora (indistintamente de la capacidad que tenga). Esto nos lleva a concluir que solo podremos representar un subconjunto de los números reales.

> Recordemos que la forma más común para la representación de los números es la forma en base decimal (en base diez) que, como bien sabemos, se enseña desde la escuela primaria. Para describir los números usamos 10 dígitos a saber: 0, 1, 2, $\cdots$, 9.

Ejemplo 1.1. Notemos que un número real, como por ejemplo 234.561, puede expresarse en base decimal como:

$$234.561 = \boxed{2} \times 10^2 + \boxed{3} \times 10^1 + \boxed{4} \times 10^0 + \boxed{5} \times 10^{-1} + \boxed{6} \times 10^{-2} + \boxed{1} \times 10^{-3}$$

Descargue un archivo: código 1.

> Este CDF permite expresar un número dado mediante una sumatoria de potencias de 10.

Necesitamos para el ejemplo anterior y para casos similares, solo los dígitos 0–9 y potencias de 10. Recordemos también que los números reales se dividen en racionales (con una expansión decimal infinita periódica) e irracionales (con una expansión decimal infinita no periódica).

Ficha N.º 1.

Se sabe que si $x \in \mathbb{R}$ entonces admite una representación decimal infinita. De esta manera:

$$x = \pm \sum_{k=-\infty}^{+\infty} d_k \times 10^k$$

Ejemplo 1.2. Para el ejemplo 1 tenemos que $d_{-3} = 1$, $d_{-2} = 6$, $d_{-1} = 5$ $d_0 = 4$, $d_1 = 3$, $d_2 = 2$. Los demás d_k's son cero.

1.1.1. Números binarios

Hay otras formas alternativas de representación numérica como por ejemplo, la binaria (que se usa mucho en ciencias de la computación), en la que solo se emplean el 0 y el 1 como dígitos, y nos referimos a ellos con el nombre de bits. En esta modalidad, los bits se multiplican por potencias de 2 en lugar de usar las de 10.

> Usualmente, para indicar que un número n está en base b se usa la notación $(n)_b$.

Cuando $b = 10$, y no hay peligro de confusión, se suele omitir la indicación de la base, es decir:

$$(n)_{10} = n$$

Ejemplo 1.3. Hallar el valor decimal del número binario 101, que representamos como $(101)_2$. Veamos:

$$(101)_2 = \boxed{1} \times 2^2 + \boxed{0} \times 2^1 + \boxed{1} \times 2^0 = 4 + 0 + 1 = 5$$

Tenemos entonces que el número binario $(101)_2$ corresponde a 5 en representación decimal. Podríamos escribir $(101)_2 = (5)_{10}$, pero se sobreentiende que $(5)_{10}$ es simplemente 5

Descargue un archivo: código 2.

Este CDF permite hacer la conversión de un número dado en base binaria a su equivalente dado en base decimal.

Ficha N.º 2.

Para pasar de un número entero positivo dado en forma decimal a uno dado en forma binaria, podemos dividir por 2 el número (o su respectivo) cociente hasta obtener un cociente que sea igual a 0. El binario equivalente se determina concatenando los bits de los residuos se empieza por el último y luego se usan los precedentes.

Ejemplo 1.4. Para convertir el número 13 a su representación binaria, procedemos de la siguiente forma:

$$\begin{array}{r|l} 13 & 2 \\ \hline \boxed{1} & 6 \end{array} \nearrow \qquad \begin{array}{r|l} 6 & 2 \\ \hline \boxed{0} & 3 \end{array} \nearrow \qquad \begin{array}{r|l} 3 & 2 \\ \hline \boxed{1} & 1 \end{array} \nearrow \qquad \begin{array}{r|l} 1 & 2 \\ \hline \boxed{1} & 0 \end{array}$$

Para construir el binario basta con leer de izquierda a derecha los números encerrados en cajas. Obtenemos así que $13 = (1101)_2$.

Descargue un archivo: código 3.

Este CDF permite hacer la conversión de un número dado en base 10 a su equivalente dado en base 2.

1.1.2. Notación científica normalizada

Sin importar el tipo de representación que se use para los números reales, debemos contestar la pregunta ¿cómo efectuar una representación "adecuada" de los

números reales en una computadora? Para responder esto necesitamos algunas definiciones.

Definición 1.1

Si $x \in \mathbb{R} - \{0\}$ es cualquier número, entonces existe $n \in \mathbb{Z}$ tal que este x puede expresarse de la forma:

$$x = \pm \underbrace{0.d_1 d_2 \cdots d_k d_{k+1} \cdots}_{r} \times 10^n = \pm r \times 10^n$$

con $d_1 \neq 0$, y $0 \leq d_i \leq 9$, para $i > 1$. Llamaremos a esta expresión la forma normalizada de x en notación científica.

Notemos en la anterior definición que $\frac{1}{10} \leq r < 1$. Además, se acostumbra ampliar esta definición haciendo $r = 0$ para el caso $x = 0$.

Ejemplo 1.5. Veamos algunos casos en los que obtenemos la forma normalizada para un número no nulo dado:

1) Si $x = 3.416$, entonces, dividiendo por 10 y multiplicando por 10, obtenemos $x = 0.3416 \times 10^1$. En este caso, observamos que $d_1 = 3$, $d_2 = 4$, $d_3 = 1$, $d_4 = 6$ y $d_i = 0$ para $i > 4$.

2) Si $x = 0.015$, entonces (multiplicando por 10 y dividiendo por 10), obtenemos $x = 0.15 \times 10^{-1}$, de donde concluimos que $d_1 = 1$, $d_2 = 5$ y $d_i = 0$ para cualquier $i > 2$.

3) Si $x = e = 2.71828 \cdots$, entonces $x = 0.271828 \cdots \times 10^1$, de donde observamos, por ejemplo, que $d_1 = 2$, $d_2 = 7$. En este caso no podemos indicar todos los d_i's pues e posee una expansión decimal infinita no periódica.

Descargue un archivo: código 4.

Este CDF permite convertir un número real a su forma en notación científica normalizada.

La representación decimal de un número real $x = \pm 0.d_1 d_2 \cdots d_k d_{k+1} \cdots \times 10^n$ en un dispositivo (computadora o calculadora) tiene que ver con la cantidad de dígitos d_k de x que se deciden almacenar en la memoria conjuntamente con el rango del exponente n que se aceptará.

Es importante señalar que en muchos casos, como cuando se usa una calculadora, no tenemos oportunidad de modificar esta representación y simplemente debemos aceptarla.

En el caso de computadoras y más específicamente, en el caso de algunos lenguajes de programación, es posible ajustar algunos parámetros respecto a la representación. Para entender mejor de qué se trata esto, presentamos la siguiente definición.

Definición 1.2

Sea $k \in \mathbb{N}$, $n_1, n_2 \in \mathbb{Z}$. Designemos $D = \{0, 1, \cdots, 9\}$, Definamos el conjunto de números reales de precisión finita:

$$\mathcal{F} = F_{(k,n_1,n_2)} = \left\{\pm 0.d_1 d_2 \cdots d_k \times 10^n : d_i \in D, d_1 \neq 0, n_1 \leq n \leq n_2\right\} \cup \{0\}$$

A los elementos de este conjunto $\mathcal{F}$ les llamaremos números en punto flotante.

Notamos que $\mathcal{F} \subseteq \mathbb{R}$, pero $\mathcal{F} \neq \mathbb{R}$, es decir, $\mathcal{F}$ es un subconjunto propio de $\mathbb{R}$.

Ficha N.º 3.

El conjunto $\mathcal{F} = F_{(k,n_1,n_2)}$ posee $18 \times 10^{k-1}(n_2 - n_1 + 1) + 1$ números.

Ejemplo 1.6. El conjunto $\mathcal{F} = F_{(3,-2,2)}$ está formado por todos aquellos números x que se puedan escribir de la forma:

$$\pm 0.d_1 d_2 d_3 \times 10^n, \text{ con } -2 \leq n \leq 2$$

Tenemos entonces $0.123 \times 10^2 = 12.3 \in \mathcal{F}$. Sin embargo, $0.123 \times 10^3 = 123 \notin \mathcal{F}$. De forma similar notamos que $-0.123 \times 10^{-3} = -0.00123 \notin \mathcal{F}$. Observamos que el número más grande de $\mathcal{F}$ es $0.999 \times 10^2 = 99.9$, y que el más pequeño es $0.999 \times 10^{-2} = -99.9$. Esto no significa que $\mathcal{F} = [-99.9, 99.9]$. De hecho, el conjunto $\mathcal{F}$ es finito y posee 9001 elementos.
Otros aspecto interesante de notar sobre $\mathcal{F}$ es que alrededor del 60 % de sus valores están en $]-1, 1[$ y solo 199 de sus valores son enteros. Estos datos se obtuvieron con la ayuda de un pequeño programa en el software comercial denominado *Wolfram Mathematica*.

Descargue un archivo: código 5.

Este CDF permite hallar el número de elementos en el conjunto $F_{(k,n_1,n_2)}$.

Si usamos el conjunto $\mathcal{F}$ del ejemplo anterior para representar los números reales en un dispositivo, notaremos que las cosas son bastante diferentes de lo que usualmente tenemos en calculadoras y computadoras. Por ejemplo, no podemos representar el número 150, pues es mayor que el valor máximo de $\mathcal{F}$. Lo mismo ocurre con -150, pues es menor que el mínimo de $\mathcal{F}$. A esta deficiencia se le conoce en inglés como *overflow*.
Con los números "cercanos" a 0, tenemos una situación parecida. En este ejemplo, el menor número positivo en $\mathcal{F}$ es 0.001. Si nos dan para representar un número cuyo valor absoluto es menor que 0.001, lo que usualmente se hace es asignarle el valor de 0. A esta deficiencia se le conoce en inglés como *underflow*.
Con el fin de representar un número $x \in \mathbb{R}$ mediante un elemento del conjunto $\mathcal{F} = F_{(k,n_1,n_2)}$, es necesario, primero, determinar si $min \leq x \leq max$, en donde min es el valor mínimo de $\mathcal{F}$ y max su valor máximo. El siguiente paso es desestimar los dígitos que se hallen después de la posición k. Existen dos maneras (clásicas) de conseguir esto, y lo consignamos en la siguiente definición:

Definición 1.3

Supongamos que $\mathcal{F} = F_{(k,n_1,n_2)}$ y que $x \in \mathbb{R} - \{0\}$ tiene como forma normalizada a $x = \pm 0.d_1d_2 \cdots d_kd_{k+1} \cdots \times 10^n$, $d_1 \neq 0$, $0 \leq d_i \leq 9$ para $i > 1$ con $n_1 \leq n \leq n_2$. Una aproximación $x^* \in \mathcal{F}$ de x, se puede conseguir de las siguientes dos formas:

1) Por truncación a k dígitos: en este caso, simplemente se desestiman los dígitos a partir de la posición $k+1$ (inclusive).

$$x = \pm 0\underbrace{d_1d_2d_3 \cdots d_k}_{\text{Se conservan}}\underbrace{d_{k+1}d_{k+2}d_{k+3} \cdots}_{\text{Se desestiman}} \times 10^n$$

Esto nos da $x^* = \pm 0.d_1d_2d_3 \cdots d_k \times 10^n$.
Abreviaremos esta modalidad mediante $AT(k)$ (aritmética de truncamiento a k dígitos para el conjunto $\mathcal{F}$). Además, llamaremos a $0.d_1d_2d_3 \cdots d_k$ la mantisa y a n el exponente. Los dígitos utilizados en la mantisa para expresar un número se denominan dígitos significativos.

2) Por redondeo a k dígitos: en este caso se debe analizar el dígito d_{k+1}. Veamos.

   a) Si $d_{k+1} < 5$, se procede análogamente a lo que se hace cuando se usa $AT(k)$, es decir, se desestiman los dígitos de la posición

$k+1$ (inclusive).

b) Si $d_{k+1} \geq 5$ sumamos $5 \times 10^{n-(k+1)}$ al x (¡original!) y luego aplicamos $AT(k)$. En la práctica, esta operación se reduce simplemente a sumar un uno al dígito d_k.

Abreviamos esta modalidad mediante $AR(k)$ (aritmética de redondeo a k dígitos para el conjunto $\mathcal{F}$).

Llamaremos a x^* el punto flotante de x ya sea que se obtenga por alguna de las modalidades $AT(k)$ o $AR(k)$. En ambos casos llamaremos a $\epsilon = x^* - x$, el error de representación.

Notamos también, ya sea que usemos $AT(k)$, o bien, $AR(k)$, que para un $x \in \mathbb{R} - \{0\}$ representable en $\mathcal{F}$, su punto flotante x^* representará (simultáneamente) una infinidad de valores que coinciden con x en los primeros k (o bien $k-1$) dígitos.

Ejemplo 1.7. Consideremos los números en punto flotante del ejemplo 1.6, a saber, $\mathcal{F} = F_{(3,-2,2)}$. En este caso usamos 3 dígitos de representación. Si empleamos $AT(3)$, tenemos, por ejemplo, que el número 0.354×10^1 servirá para representar a cualquiera de los números:

$$0.35467,\ 0.35411,\ 0.35495787,\ 0.35488745$$

solo por mencionar algunos de la infinidad que podríamos nombrar.

En la práctica, el conjunto $\mathcal{F}$ queda determinado por el dispositivo o programa de computadora que usemos. Los números que se ofrecen en los ejercicios y ejemplos de este texto son en su mayoría suceptibles de ser representados en tales dispositivos o programas.

Prestaremos así, más atención al proceso de redondear o truncar una cantidad numérica. En los ejemplos numéricos siguientes se da un conjunto $\mathcal{F}$ específico, sin embargo, no es difícil modificar dicha situación para el caso en que se usa, por ejemplo, una calculadora de bolsillo.

Ejemplo 1.8. Veamos algunos casos usando $AT(5)$ en $\mathcal{F} = F_{(5,-10,10)}$.

1) Sea $x = 0.123456789$: en este caso $x^* = 0.12345$ y el error de representación está dado por $\epsilon = -6.789 \times 10^{-6}$.

2) Sea $x = 0.023456789$: normalizando x obtenemos:

$$x = 0.23456789 \times 10^{-1}$$

De esta forma, tenemos que $x^* = 0.23456 \times 10^{-1} = 0.023456$.

Notamos que emplear aritmética de trucamiento es tan simple como "conservar cierta cantidad de dígitos después del punto decimal".

Utilicemos ahora $AR(5)$ en $\mathcal{F} = F_{(5,-10,10)}$.

1) Sea $x = 0.123456789$: notamos que $d_6 = 6 \geq 5$, por lo tanto, debemos agregar 1 a $d_5 = 5$, de donde obtenemos entonces que $x^* = 0.12346$.

2) Sea $x = 0.023456789 = 0.23456789 \times 10^{-1}$: en este caso, $d_6 = 7 \geq 5$, por lo tanto, debemos agregar 1 a $d_5 = 6$. De esta forma tenemos que $x^* = 0.23457 \times 10^{-1} = 0.023457$.

3) Sea $x = 0.99999\cdots$: notamos que $d_6 = 9$, por lo que debemos agregar un 1 a $d_5 = 9$. Esto deja $d_5 = 0$ y agregamos un 1 a $d_4 = 9$, etc. Finalmente, concluimos que $x^* = 1$. Otra forma de abordar este problema es sumando primero. Veamos:

$$5 \times 10^{n-(k+1)} = 5 \times 10^{0-(5+1)} = 5 \times 10^{-6}$$

Esto produce $1.00000499\cdots$ y al aplicar $AT(5)$ sobre esta cantidad obtenemos $x^* = 1$.

Descargue un archivo: código 6.

Redondeo y truncamiento de números en un conjunto $F_{(k,n_1,n_2)}$.

Ya sea que usemos $AT(k)$ o $AR(k)$ notamos que x^* es una aproximación de x. Debemos, entonces, buscar mecanismos o indicadores que nos permitan determinar si x^* se trata de una "buena" o "mala" aproximación.

Definición 1.4

Sea x^* una aproximación de x, definimos el error absoluto de aproximar x usando x^* como:

$$EA(x, x^*) = |x - x^*|$$

Además, definimos el error relativo de aproximar x usando x^* como:

$$ER(x, x^*) = \frac{|x - x^*|}{|x|}$$

toda vez que $x \neq 0$.

Por otro lado, debemos mencionar que el error absoluto mide la diferencia entre el valor real y la aproximación, mientras que el error relativo toma en cuenta el orden del valor que se está aproximando. No es lo mismo cometer un error de medida en cantidades "pequeñas" que en cantidades "grandes": no es lo mismo un error de 0.1 cm en la medida de la altura de una persona que en la altura de una montaña.

Ejemplo 1.9. Sean $x = 0.1$, $x^* = 0.2$, $y = 100.1$ y $y^* = 100.2$. Calculemos para este par de casos su respectivos errores absolutos y relativos. Empecemos con el valor x:

$$\begin{cases} EA(0.1, 0.2) = |0.1 - 0.2| = 0.1 \\ \\ ER(0.1, 0.2) = \dfrac{|0.1 - 0.2|}{0.1} = 1 \end{cases}$$

Veamos ahora el caso de y:

$$\begin{cases} EA(100.1, 100.2) = |100.1 - 100.2| = 0.1 \\ \\ ER(100.1, 100.2) = \dfrac{|100.1 - 100.2|}{100.1} = 0.000999001 \end{cases}$$

Notamos en este caso que aunque ambos errores relativos son iguales, los errores absolutos son muy diferentes. Vemos así que el error relativo es más informativo que el error absoluto.

> Es importante señalar que aunque se use aritmética de rendondeo o de truncamiento para cierta cantidad de dígitos, podemos efectuar el cálculo de los errores absolutos y relativos usando para ello una mayor precisión.

El siguiente ejemplo es una muestra de lo anterior.

Ejemplo 1.10. Veamos algunos casos.

1) Sea $x = 0.3$ y $x^* = 0.32$, entonces $EA(x, x^*) = 0.02$ y $ER(x, x^*) = 0.66\cdots$.

2) Al usar $AR(5)$, para $\pi = 3.1415926535897932385\cdots$ obtenemos $\pi^* = 3.1416$. Por lo tanto, $EA(\pi, \pi^*) = 7.34641 \times 10^{-6}$ y $ER(\pi, \pi^*) = 2.33843 \times 10^{-6}$.

Aunque hoy en día las calculadoras y computadoras nos permiten trabajar con muchos dígitos de precisión, los cálculos que realizamos en análisis numérico generan errores significativos, muchas veces nada desdeñables.

Definición 1.5

Sea $x \in \mathbb{R}$, decimos que x^* aproxima a x con t dígitos de precisión si $ER(x, x^*) < 5 \times 10^{-t}$, o bien,

$$\left| \frac{x - x^*}{x} \right| < 5 \times 10^{-t}$$

Ejemplo 1.11. Si se sabe que x^* aproxima a x con 8 dígitos de precisión, entonces se cumple que:

$$\left| \frac{x - x^*}{x} \right| < 5 \times 10^{-8} = 0.00000005$$

Vemos así que hay ocho ceros antes del dígito 5.

Descargue un archivo: código 7.

Este CDF calcula el error absoluto y el error relativo entre dos cantidades. Se indica además el número de dígitos de exactitud.

Ejemplo 1.12. Sea $x = 3.5$, hallemos un intervalo para los x^* que aproximan a x con 5 dígitos de precisión. Para lograr esto planteamos la desigualdad $ER(x, x^*) < 5 \times 10^{-5}$, o bien:

$$\left| \frac{3.5 - x^*}{3.5} \right| < 5 \cdot 10^{-5}$$

o equivalentemente:

$$-17.5 \cdot 10^{-5} < x^* - 3.5 < 17.5 \cdot 10^{-5}$$

por lo tanto, $x^* \in]3.49983, 3.50018[$.

En este libro se verá una gran cantidad de métodos numéricos que permiten aproximar ciertos valores, tales como los ceros de una ecuación, derivadas, integrales, entre otros. En muchos de los ejemplos y ejercicios se solicitará hacer el cálculo con cierta cantidad de dígitos de exactitud y es por eso que la definición anterior es necesaria.

Antes de terminar esta sección es importante indicar que, aun cuando se cuente con un dispositivo, que permita manipular números con "muchos"

> dígitos de precisión, algunas veces preferimos utilizar solo unos "cuantos" de ellos. Para lograr este propósito utilizamos la misma idea referente a los criterios para representar un número real en un conjunto de precisión finita, esto es, usaremos redondeo o truncamiento.

Ilustramos esto con un ejemplo.

Ejemplo 1.13. Si una calculadora de bolsillo nos da:

$$x^* = 2.8284271 = 0.28284271 \times 10$$

como una aproximación para $x = \sqrt{8}$, podemos obtener una aproximación por truncamiento a tres dígitos, a saber, $x_1^* = 2.82$. También, podemos obtener una aproximación por rendondeo a tres dígitos, a saber, $x_2^* = 2.83$.
Se debe notar que el error relativo entre x y x^* es de 0.0000000087491, lo que permite 8 dígitos de exactitud. Además, el error relativo entre x y x_1^* es de 0.00842712 lo que facilita 3 dígitos de exactitud. Finalmente, el error relativo entre x y x_2^* es de 0.000556095 lo que permite también 3 dígitos de exactitud.
La razones para optar por aproximaciones de inferior precisión tales como x_1^* o x_2^*, son muy variadas. A veces es simplemente porque para el propósito de nuestras actividades o cálculos posteriores, con esas aproximaciones obtenemos resultados que consideraremos "satisfactorios".

1.2. Aritmética de computadora

Hablemos ahora de las operaciones aritméticas de computadora. Como es de esperarse, estas operaciones están supeditadas a la precisión del dispositivo. Es por esta razón que utilizaremos cuatro nuevos símbolos para las operaciónes aritméticas de computadora, a saber: $\oplus$, $\ominus$, $\otimes$ y $\oslash$, correspondienes (respectivamente) a las operaciones aritméticas usuales: $+$, $-$, $\times$ y $/$. Definiremos esta operaciones de computadora de la siguiente forma:

$x \oplus y = [x^* + y^*]^*$	$x \ominus y = [x^* - y^*]^*$
$x \otimes y = [x^* \times y^*]^*$	$x \oslash y = [x^* / y^*]^*$

> Dicho en términos más sencillos, si deseamos efectuar una de estas operaciones aritméticas de computadora con x y y, tomamos el punto flotante de x y de y, efectuamos la operación aritmética correspondiente y finalmente tomamos el punto flotante de ese resultado.

La justificación de esto es que solo podemos emplear números con punto flotante

para representar los resultados. La idea se puede extender para incluir otras operaciones tales como potenciación, raíz cuadrada, trigonométricas, entre otras.

Ejemplo 1.14. Sea $x = 0.8888888$ y $y = 0.987654321$. Supongamos que estamos usando $AT(3)$ en $\mathcal{F} = F_{(3,-10,10)}$. Tenemos así que $x^* = 0.888$ y $y^* = 0.987$. Hagamos algunos cálculos aritméticos. Es importante señalar que lo referente a los cálculos de los errores relativos y absolutos, se hace con mayor precisión.

1) Observemos que $x \oplus y = [0.888 + 0.987]^* = [1.875]^* = 1.87$. Por otro lado, $x + y = 1.87654$. De esta forma, el error absoluto en esta operación está dado por:
$$EA(x + y, x \oplus y) = 0.00654321$$
y el error relativo está dado por:
$$ER(x + y, x \oplus y) = 0.00348684$$
Como $0.00348684 < 0.005$, se nota que $x \oplus y$ aproxima a $x + y$ con 3 dígitos de precisión.

2) Notemos que $x \ominus y = [0.888 - 0.987]^* = [-0.099]^* = -0.099$ con un error absoluto de 0.000234568 y un error relativo de 0.002375. De nuevo, vemos que el resultado posee 3 dígitos de precisión.

3) Por otro lado, $x \otimes y = [0.888 \times 0.987]^* = [0.876456]^* = 0.876$ con un error absoluto de 0.00191495 y un error relativo de 0.00218125. También, vemos que el resultado posee 3 dígitos de precisión.

4) Finalmente, $x \oslash y = [0.888/0.987]^* = [0.899696]^* = 0.899$ obteniendo un error absoluto de 0.000999991 y un error relativo de 0.0011111. Una vez más, vemos que el resultado posee 3 dígitos de precisión.

Repitamos el ejemplo anterior pero usando aritmética de redondeo.

Ejemplo 1.15. Sea $x = 0.8888888$ y $y = 0.987654321$. Supongamos que estamos usando $AR(3)$ sobre $\mathcal{F} = F_{(3,-10,10)}$. Tenemos en este caso que $x^* = 0.889$ y $y^* = 0.988$. De nuevo, es importante señalar que lo referente a los errores relativos y absolutos se hace con mayor precisión.

1) Observemos que $x \oplus y = [0.889 + 0.988]^* = 1.88$. Por otro lado, tenemos que $x + y = 1.87654$. De esta forma, el error absoluto en esta operación está dado por:
$$EA(x + y, x \oplus y) = 0.00345679$$
y el error relativo está dado por:
$$ER(x + y, x \oplus y) = 0.00184211$$

Como $0.00184211 < 0.005$, se nota que $x \oplus y$ aproxima a $x + y$ con 3 dígitos de precisión.

2) Hacemos ahora $x \ominus y = [0.889 - 0.988]^* = -0.099$ con un error absoluto de 0.000234567 y un error relativo de 0.00237499. De nuevo, vemos que el resultado posee 3 dígitos de precisión.

3) Hacemos ahora $x \otimes y = [0.889 \times 0.889]^* = 0.878$ con un error absoluto de 0.0000850489 y un error relativo de 0.000096876. El resultado posee 4 dígitos de precisión.

4) Finalmente, hacemos $x \oslash y = [0.889/0.988]^* = 0.9$ con un error absoluto de 9.112499×10^{-10} y un error relativo de 1.0125×10^{-9}. El resultado tiene 9 dígitos de precisión.

Descargue un archivo: código 8.

Este CDF permite efecturar operaciones aritméticas con precisión finita.

El uso reiterado de las operaciones aritméticas (y similares), induce un nuevo error conocido como error por propagación. Dado que cada vez que efectuamos una operación aritmética debemos hacer redondeo o truncamiento, según sea la opción adoptada, es de esperar que conforme se incremente la cantidad de operaciones en un cálculo, se aumente el error por propagación, produciendo una diferencia significativa (que podemos medir mediante el error absoluto o relativo) entre el valor real (exacto) y el arrojado por la computadora.
Los errores por redondeo o por propagación pueden resultar fatales cuando se necesita ser muy preciso en un cálculo. Se recomienda al lector leer sobre un accidente ocurrido utilizando un misil Patriot durante la Guerra del Golfo. Este accidente se debió a un error de redondeo, produciendo que se atacara un cuartel del ejército estadounidense, matando a 28 soldados e hiriendo a muchos otros. Consultar el siguiente sitio web: https://www-users.cse.umn.edu/~arnold/disasters/patriot.

> Una situación en la cual el error de propagación adquiere atención especial es cuando se realiza la resta de cantidades "muy similares" entre ellas.

Veamos un cálculo en el que se da este caso.

Ejemplo 1.16. Sea $x = 0.123456$ y $y = 0.123457$. Si usamos $AR(3)$, tenemos entonces que:

$$x \ominus y = [0.123 - 0.123]^* = 0$$

Observamos, sin embargo, que $x - y = -1 \times 10^{-6}$, por lo tanto, el error absoluto está dado por $EA = 1 \times 10^{-6}$ y el error relativo por $ER = 1$. Por supuesto,

este error relativo nos alerta (en esta y en situaciones similares) que debemos cuestionarnos si el resultado obtenido resulta aceptable. La respuesta a esto depende en buena medida de la naturaleza del problema que estamos tratando.

Un caso clásico que evidencia los problemas de efectuar restas o diferencias de cantidades "similares" empleando una aritmética "pobre", es la resolución de ecuaciones cuadráticas $ax^2 + bx + c = 0$, mediante la fórmula general para cuadráticas, a saber:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Como el lector lo podrá advertir, el problema se podrá dar cuando, al calcular una de las raíces, resulte que $-b \approx \sqrt{b^2 - 4ac}$, en donde el símbolo $\approx$ se lee "aproximadamente igual".

Ejemplo 1.17. Resuelva $x^2 + 100x + 1 = 0$, empleando $AT(4)$.
Notemos primero que los valores exactos de las raíces son:

$$x_1 = -50 - 7\sqrt{51} \approx -99.99, \quad \text{y} \quad x_2 = -50 + 7\sqrt{51} \approx -0.010001$$

Hagamos ahora el cálculo usando $AT(4)$. Es conveniente advertir que aquí emplearemos el signo de igualdad para expresar que las cantidades a ambos lados del $=$ son equivalentes en $AT(4)$.
En este caso, $\Delta = 9996$, y por lo tanto, $\sqrt{\Delta} = 99.97$. Calculemos la primera raíz como sigue:

$$x_1^* = \frac{-b - \sqrt{\Delta}}{2a} = \frac{-100 - 99.97}{2} = -\frac{199.7}{2} = -99.85$$

Observamos que:

$$\begin{cases} EA(x_1, x_1^*) = 0.139999\cdots \\ \\ ER(x_1, x_1^*) = 0.0014001\cdots \end{cases}$$

La segunda raíz involucra la resta de cantidades muy "semejantes":

$$x_2^* = \frac{-b + \sqrt{\Delta}}{2a} = \frac{-100 + 99.97}{2} = \frac{-0.03}{2} = -0.015$$

Tenemos entonces que:

$$\begin{cases} EA(x_2, x_2^*) = 0.004999 \\ \\ ER(x_2, x_2^*) = 0.49985 \end{cases}$$

Para mejorar la aproximación de x_2 sin necesidad de aumentar la precisión de

aritmética empleada, podemos echar a mano a una equivalencia algebraica de la fórmula cuadrática general. Veamos:

$$x_2 = \frac{-b + \sqrt{b^2 - 4ac}}{2a} = \frac{-b + \sqrt{b^2 - 4ac}}{2a} \times \frac{-b - \sqrt{b^2 - 4ac}}{-b - \sqrt{b^2 - 4ac}}$$

Utilizando la tercera fórmula notable obtenemos:

$$x_2 = \frac{b^2 - (b^2 - 4ac)}{2a\left(-b - \sqrt{b^2 - 4ac}\right)} = \frac{-2c}{b + \sqrt{b^2 - 4ac}}$$

Si empleamos ahora está última fórmula, se infiere que:

$$x_2^{**} = \frac{-2 \cdot 1}{100 + 99.97} = \frac{-2}{199.9} = -0.01$$

Notamos entonces que:

$$\begin{cases} EA(x_2, x_2^{**}) = 1 \times 10^{-6} \\ \\ ER(x_2, x_2^{**}) = 0.00009999 \end{cases}$$

Estos errores para x_2^{**} presentan una mejora considerable respecto al cálculo x_2^* anterior hecho para aproximar x_2.

Ejemplo 1.18. Usando $AT(3)$ realizaremos el cálculo de $x = \left(1 + 2^{-8} - 1\right) \cdot 2^8$. Notamos primero que el valor exacto de x es:

$$x = \left(1 + 2^{-8} - 1\right) \cdot 2^8 = x = \left(2^{-8}\right) \cdot 2^8 = x = 2^{8-8} = 2^0 = 1$$

Si usamos $AT(3)$, tenemos que:

$$x^* = \left(\underbrace{0.1 \times 10 + 0.312 \times 10^{-2}}_{} - 0.1 \times 10\right) \cdot 0.256 \times 10^3 =$$
$$(0.1 \times 10 - 0.1 \times 10) \cdot 0.256 \times 10^3 = 0 \cdot 0.256 \times 10^3 = 0$$

En este caso, tanto el error absoluto como el relativo es de una unidad y claramente detectamos que el valor real y el aproximado son "muy" diferentes.

Otro caso típico que genera errores importantes (usando una aritmética de computadora "pobre"), tiene que ver con la evaluación de polinomios.

Ejemplo 1.19. Sea

$$f(x) = x^3 + 3x^2 + 3x + 1$$

Determinemos el valor numérico de $z = f(1.99)$ usando $AT(3)$.
Veamos algunas partes del cálculo:

- $1.99^3 = 7.88$
- $1.99^2 = 3.96$
- $7.88 + 11.8 = 19.6$
- $19.6 + 5.97 = 25.5$
- $25.5 + 1 = 26.5$

Por lo tanto, usando $AT(3)$, obtenemos $z^* = 26.5$. Dado que $z = 26.7309$, tenemos que $EA(z, z^*) = 0.230899$ y por otro lado, $ER(z, z^*) = 0.00863791$.
Modifiquemos ahora el polinomio $f(x)$ utilizando la forma anidada que se muestra a continuación:

$$f(x) = x^3 + 3x^2 + 3x + 1 = x[x(x + 3) + 3] + 1$$

Veamos algunos detalles del cálculo:

- $1.99 + 3 = 4.99$
- $9.93 + 3 = 12.9$
- $25.6 + 1 = 26.6$

Por lo tanto, usando $AT(3)$, obtenemos $z^* = 26.6$. Dado que $z = 26.7309$, tenemos que $EA(z, z^*) = 0.130899$ y por otro lado $ER(z, z^*) = 0.00489692$.
Notamos, entonces, que el error relativo se reduce casi a la mitad del obtenido en el cálculo anterior, y se da una mejoría.

1.3. Introducción al análisis de algoritmos

Muchos de los métodos numéricos que estudiaremos en este libro se consignan en forma de algoritmos. En varios de estos algoritmos es normal encontrar estructuras repetitivas que dependen de un parámetro $n \in \mathbb{N}$ que determina la "calidad" de la solución para el problema abordado. Ciertamente, n también determina la rapidez con la que un programa de computadora ejecuta el algoritmo. Resulta interesante señalar que dos algoritmos que supuestamente realizan la misma tarea, podrían diferir considerablemente en el tiempo que utilizan.