Prólogo

La asignatura se denomina Cálculo Numérico. El Cálculo Numérico; o Análisis Numérico; es la rama de las matemáticas encargada de diseñar algoritmos para obtener aproximaciones a una solución cuyo problema es abordado por el Algebra y el Análisis Matemático. Se denomina "Numérico" porque a diferencia del Análisis Matemático, cuyo objetivo es el estudio de un ente matemático en particular que modela el fenómeno estudiado, aquí el centro es el desarrollo del "método o algoritmo" que, mediante un conjunto de operaciones finitas y numéricas, aproxima la solución.

Una definición mas formal es aquella brindada por Henrici, quien la declara como "La teoría de los métodos constructivos en Análisis Matemático". La palabra constructivo hace referencia a que, propuesto un problema matemático, además de estudiar la existencia de la solución, hay que brindar un procedimiento para calcularla de manera explícita. Los métodos se construyen aplicando algoritmos, entendido aquí como una especificación no ambigua de operaciones aritméticas en un orden prefijado. Es decir, no se aplica un cómputo simbólico (manipulación de expresiones algebraicas) sino que operan números. Por este motivo los métodos aplicados en el análisis numérico se denominan comúnmente Métodos Numéricos.

Una definición formal de Método Numérico indica que "es un conjunto de reglas que permite la obtención, mediante un número finito de operaciones elementales, de un resultado que se aproxima de alguna manera a la solución del problema en cuestión". Podemos integrar ambos conceptos vertidos hasta ahora al definir el Cálculo Numérico como "el estudio de los métodos numéricos y los procedimientos usados para obtener soluciones aproximadas a problemas matemáticos".

En definitiva, los métodos numéricos permiten realizar el cálculo aproximado de soluciones en muchos problemas matemáticos. Estas técnicas se basan en procedimientos que consideran aspectos del cálculo que surgen en aplicaciones de las Matemáticas sobre el mundo real y que se ajustan a los medios actuales de cálculo automático digital; es decir los métodos numéricos cobran especial interés cuando son implementados en un ordenador, de tal manera que el grueso de los cálculos es realizado por este.

Esta asignatura pretende que el estudiante complete los conocimientos de Álgebra y Cálculo Diferencial e Integral con conceptos y procedimientos que le permitan de un modo efectivo alcanzar la solución de problemas que en estos ámbitos se plantean. Así al finalizar la cursada se espera que el mismo pueda:

- ✓ Comprender los principios necesarios para generalizar las soluciones específicas de los problemas científicos y de ingeniería a modelos de simulación mediante herramientas informáticas basadas en los algoritmos matemáticos.
- ✓ Ser capaz de analizar, representar y resolver los problemas científicos y de ingeniería empleando herramientas informáticas que automaticen los cálculos.
- ✓ Adquirir las herramientas y los criterios mínimos necesarios para resolver problemas numéricamente y/o evaluar la validez y la precisión de los resultados obtenidos mediante algoritmos.
- ✓ Ser capaz de analizar un problema y tener el criterio para aplicar el método que mejor se adapte a la naturaleza del problema.

- ✓ Poder describir un problema y su solución numérica a otros actores utilizando la terminología adecuada y técnicas de comunicación interpersonal.

La amplitud de los problemas que puede abordar esta disciplina es realmente muy amplia, y se puede dividir en dos grupos fundamentales:

- ✓ Problemas de dimensión finita: son aquellos cuya respuesta se corresponde con un conjunto finito de números, tales como las ecuaciones algebraicas, determinantes, etc.
- ✓ Problemas de dimensión infinita: son aquellos en cuya solución o planteamiento intervienen elementos descriptos por una cantidad infinita de números, como integración y derivación numérica, cálculo de ecuaciones diferenciales, interpolación, etc.

Aunque también se suelen dividir atendiendo a su naturaleza o motivación para el empleo del cálculo numérico:

- ✓ Problemas de tal complejidad que no poseen solución analítica
- ✓ Problemas en los cuales existe una solución analítica, pero que, por su complejidad u otros motivos, no puede explotarse de forma sencilla en la práctica.
- ✓ Problemas para los cuales existen métodos sencillos pero que, para elementos que se emplean en la práctica, requieren una cantidad de cálculos excesiva; mayor que la necesaria para un método numérico.

Para finalizar se compartirán dos aspectos importantes. Por un lado, resumir los objetivos de esta disciplina:

1. Dado un problema matemático, encontrar los métodos numéricos, que bajo ciertas condiciones permiten obtener una solución aproximada del problema propuesto, y
2. Analizar las condiciones bajo las cuales la solución del algoritmo es próxima a la verdadera, estimando los errores en la solución aproximada.

Se trata pues, de encontrar métodos aproximados para resolver todo tipo de problemas matemáticos y analizar los errores producidos en estas aproximaciones.

El segundo aspecto importante para destacar consiste en retomar algo que ya se había comentado antes: El análisis numérico cobra especial importancia con la llegada de los ordenadores. Estos son útiles para cálculos matemáticos extremadamente complejos, y su creciente potencia de cálculo ha incrementado el uso y desarrollo de métodos numéricos para resolver multitud de problemas, lo cual hace posible abordar problemas tan complejos que antes serían impracticables. Muchas de las disciplinas de inteligencia artificial utilizan en el fondo pequeños algoritmos numéricos fusionados u absorbidos por la propia metodología aplicada, gran parte de las aplicaciones científicas, así como los editores de video, música e imágenes los aplican en diferentes niveles de complejidad, y muchas técnicas usadas en otras materias aplican un método numérico. Esto ha provocado que el estudio de los métodos numéricos haya evolucionado enormemente en los últimos años y esté convirtiéndose en una de las ramas más importantes de las matemáticas y, desde luego una de las de más actualidad.
Paradójicamente el uso de ordenadores plantea a los científicos y desarrolladores de soluciones científicas basadas en métodos numéricos tener que realizar un estudio adicional de los errores que provocan la limitada capacidad de un ordenador para representar un número. Además, como lo indica la definición de cálculo numérico, la solución es una aproximación, por lo tanto, también posee un error inherente. Por este motivo en esta unidad se inicia con los conceptos de errores y los errores en el uso de ordenadores.

La naturaleza de los errores

Puesto que vamos a trabajar con soluciones numéricas, lo primero que debemos saber y aceptar es que las mismas generan errores.
Este aspecto puede generar contradicciones, después de todo en el ideario general es casi natural la percepción de que la Ingeniería debe generar soluciones exactas donde no hay espacio para cometer errores. La percepción del error es comúnmente fatalista, por ejemplo:

El Titanic, el gigantesco trasatlántico construido en los astilleros de Belfast, en Irlanda del Norte, se hundió la noche del 14 de abril de 1912 después de chocar con un iceberg en la mitad del océano Atlántico, durante su viaje inaugural desde Southampton a Nueva York. La tragedia, que dejó un saldo de 1.532 personas muertas de los 2.222 pasajeros que iban a bordo, se produjo por una suma de errores humanos, partiendo por el hecho de que, al ser considerado "insumergible", el buque no disponía de medios ni botes salvavidas suficientes como para hacer frente a un naufragio, cosa que disparó el número de víctimas.
También se responsabilizó del luctuoso hecho al capitán Edward Smith por conducir la nave a gran velocidad y con cierta imprudencia. Como si fuera poco, el marino Frederick Fleet, el vigía encargado de avistar posibles peligros para el barco y quien realizara las campanadas de alerta luego de divisar un inmenso bulto negro en medio de la noche (que resultó ser un gigantesco iceberg), testificaría posteriormente que de haber tenido binoculares, los cuales inexplicablemente estaban almacenados en ese momento dentro de unos casilleros por órdenes superiores, habría visto mucho antes el iceberg y, por ende, habría podido dar la alarma a tiempo.

El 23 de julio de 1983, el Vuelo 143 de Air Canada, un Boeing 767-200, se quedó sin combustible a una altitud de 12.500 metros (41.000 pies), casi a media distancia de su vuelo desde Montreal a Edmonton, Canadá. La tripulación fue capaz de planear el avión a un aterrizaje de emergencia en el Gimli Industrial Park Airport, una antigua base de la Fuerza Aérea de Canadá en Gimli, Manitoba.

La investigación reveló que la carga de combustible se calculó mal, dado que no existía una formación adecuada en los sistemas métricos decimales recientemente adoptados, tras sustituir al sistema imperial. En vez de usar como unidad de peso el kilogramo se utilizó la libra (que equivale a 0,45 Kg). Como consecuencia el avión solo contaba con la mitad de combustible necesario. Además, el sistema informático que mide y calcula el nivel de combustible estaba descompuesto y no se podía reparar debido a la escasez de repuestos.

Sin embargo, hay casos famosos donde un error ha cambiado el mundo para bien, por ejemplo:

La penicilina, el antibiótico más conocido del mundo que desde su descubrimiento ha salvado la vida a millones de persona, quizás no se hubiese creado si el médico británico Alexander Fleming no hubiera olvidado limpiar su laboratorio antes de salir de vacaciones.

En la mañana del 28 de septiembre de 1928 Flemming se encontraba estudiando cultivos de bacterias de estafilococo en el sótano del laboratorio del Hospital St. Mary, en Londres, aunque luego debió ausentarse casi por un mes de la capital inglesa, olvidando cerca de una ventana abierta una placa de Petri (un instrumento de laboratorio) que contenía bacterias.

Al regresar de sus vacaciones, Flemming advirtió la placa de Petri olvidada. Al observarla se percató de que las muestras se habían contaminado con una especie de moho que había entrado por la ventana a través del viento. Movido por la curiosidad, el científico en vez de tirar su experimento arruinado a la basura colocó su placa de Petri al microscopio y observó asombrado que no sólo el moho había contaminado todo el contenido de la placa, sino que alrededor de éste, había un claro, una zona limpia en la que el moho había matado a todas las bacterias. Flemming acababa de descubrir los hongos de Penicillium, la base del famoso antibiótico que posee la capacidad de aniquilar una gran cantidad de bacterias que causan infecciones en el cuerpo humano.

Aunque no se tenga una percepción fatalista del error, es muy probable que se lo asocie a un sistema de sanción o penalización como un estímulo para alcanzar la perfección. En Chapra (2007, p. 53) se ejemplifica esta situación de la siguiente manera “Los estudiantes y los practicantes de la ingeniería trabajan constantemente para limitar este tipo de errores en sus actividades. Cuando hacen un examen o realizan sus tareas, son sancionados, mas no premiados por sus errores”.

Y más adelante completa

> Aunque la perfección es una meta digna de alabarse, es difícil, si no imposible, alcanzarla. Por ejemplo, a pesar de que el modelo obtenido mediante la segunda ley de Newton es una aproximación excelente, en la práctica jamás predecirá con exactitud la caída del paracaidista. Fenómenos tales como la velocidad del viento y alguna ligera variación de la resistencia del aire desviarían la predicción. Si tales desviaciones son sistemáticamente grandes o pequeñas, habría entonces que formular un nuevo modelo. No obstante, si su distribución es aleatoria y se agrupan muy cerca de la predicción, entonces las desviaciones se considerarían insignificantes y el modelo parecerá adecuado. (Chapra, p. 53)

El objetivo principal del Análisis Numérico es el estudio de las técnicas que permiten aproximar soluciones a los problemas que aparecen en fenómenos físicos y vivientes. Estos problemas son en general del tipo continuo, por lo que se requiere el diseño y análisis matemático de algoritmos aproximantes para simular los fenómenos de la naturaleza mediante ordenadores. Esto se debe a que la resolución de tales fenómenos mediante métodos analíticos muchas veces

es muy costosa de plantear, de tal manera que es impracticable, o simplemente es imposible de planear. Sin embargo, modelar problemas reales mediante un ordenador no es una tarea fácil. Requiere entender muy bien las posibilidades y las limitaciones tanto del cálculo aproximado como de los algoritmos utilizados. Y esto vale desde aproximar un simple número o calcular una raíz, hasta resolver sistemas lineales de gran tamaño y complejas ecuaciones diferenciales.

Uno de los objetivos principales de la computación científica es desarrollar métodos precisos y eficaces y para ello, es necesario poder controlar las diferentes fuentes de error a fin de no modificar los resultados calculados.
Los resultados numéricos se ven afectados por muchos tipos de error. Algunas fuentes de error son muy difíciles de eliminar, otras pueden ser reducidas o suprimidas al reescribir las fórmulas mediante expresiones que eficienticen el cálculo minimizando recursos y/o descartando los términos que provocan la aparición del error o su propagación. Los cambios en la secuencia computacional utilizada también contribuyen en la minimización de errores.
La importancia de la detección y manejo de los errores es fundamental debido a que los mismos se propagan desde el inicio del cómputo hasta el resultado final, a veces con una considerable amplificación, otras veces observándose oscilaciones. Esto puede provocar que el método aplicado sea correcto pero impracticable por la forma en que está implementado o por sus consecuencias en términos de la representación de los números involucrados que realice el procesador en la memoria.
Así, resulta importante poder distinguir entre el nuevo error producido durante cada calculo en particular (error de procesamiento) y el error heredado (propagado) desde los datos durante todos los cálculos. La siguiente constituye una fusión de las fuentes de error propuestas por Hernández (2018) y Hernández (2011):

- ✓ Error en los datos de entrada: los datos iniciales o de entrada pueden ser el resultado de mediciones influenciadas por errores sistemáticos o por perturbaciones temporales. Esto incluye errores en la medición de estos datos (por utilizar el instrumento inadecuado que lleva al redondeo o truncamiento), o la representación de estos en la máquina (cuando el dato es ingresado sin formato adecuado, por ejemplo, un número decimal en lugar de una fracción). También se presentan cuando se utilizan resultados de otros trabajos que contienen sus correspondientes errores finales.
- ✓ Error de redondeo durante el cálculo o de representación: la limitación de los números de punto flotante en un ordenador lleva asociada una perdida frecuente de información que, de acuerdo con el contexto, puede o no ser importante. Sobre este tipo de errores profundizaremos en una sección posterior, si embargo se comparte dos casos típicos: si el procesador utilizado no tiene la capacidad de operar con números de más de s dígitos, entonces el producto exacto de dos números con s dígitos de longitud no debería ser utilizado en cálculos posteriores porque el resultado debe ser recortado, provocando un error. Otro ejemplo común es aquel que se presenta cuando un procesador que opera con punto flotante se halla ante la situación de sumar un número relativamente pequeño b a otro número a. Aquí, sucederá que algunos dígitos de b se pierden y no tendrán efecto en futuros cálculos que dependan del valor de $a + b$. El efecto de estos redondeos puede ser trágicamente notable para ciertos tipos de algoritmos, a los que se denominarán “inestables”.
- ✓ Error de truncamiento o de discretización: este tipo de error ocurre cuando un proceso limite es truncado antes de llegar al valor limite. Por ejemplo, cuando una serie infinita es truncada después de un número finito de términos, o cuando una derivada es aproximada a través de un cociente en diferencias. Otro ejemplo es cuando una función no lineal es aproximada con una función lineal en un intervalo definido.

- ✓ Simplificaciones en el modelo matemático. En la mayoría de las aplicaciones matemáticas se producen idealizaciones. En un problema de mecánica, por ejemplo, es habitual asumir que la cuerda que sostiene un péndulo carece de masa. En otros tipos de problemas, puede ser una complicación considerar un cuerpo como un objeto homogéneo y lleno completamente de materia, en vez de estar compuesto por átomos. Este tipo de error es bastante más complicado de estimar que los descritos anteriormente.
- ✓ Error humano y/o máquina: en cualquier trabajo que se necesite operar con números, son comunes los errores de administración de información (por ejemplo, al copiar datos), errores en cálculos manuales y discrepancias en los datos procesados. Se debe estar alerta a descubrir la presencia de estas situaciones (por ejemplo, verificar que la versión impresa de una tabla no contenga errores de impresión. A veces las rutinas computaciones también contienen errores, rastreables a aspectos no considerados por el programador. Aquí se pueden incluir errores técnicos como no considerar el tipo de procesador a utilizar. Un buen plan de trabajo debe prever controles para evitar o disminuir los errores humanos. En cuanto a los errores de máquina, suelen ser excepcionales, y su solución está fuera de nuestro alcance, salvo la verificación del nivel de obsolescencia del producto.

También se puede pueden agrupar estas fuentes de error en:

- ✓ Errores Inherentes o de Modelo: González (2011) los define como los errores que afectan a los datos del problema numérico y pueden tener distintos orígenes. Pueden ser el resultado de la incertidumbre en cualquier medición. También cuando existe una limitación en la cantidad de dígitos que podamos en el instrumento de medida, por ejemplo: si ingresamos el valor π en una calculadora, jamás obtendremos ni podremos usar su valor exacto, lo que nos llevará a asumir un error de redondeo o truncamiento. En definitiva, los errores inherentes no pueden ser evitados. Aquí también tendríamos los errores humanos. Se denominan de Modelo porque aparecen como consecuencia de las diferencias entre el fenómeno real estudiado y las abstracciones que se materializan en al crear el modelo matemático. Son imposibles de remediar porque son producto de factores intrínsecos a la naturaleza y las personas. Otra característica de esta fuente de errores es que no se pueden cuantificar.
- ✓ Errores de Método: son errores que surgen de la representación y manipulación de los datos numéricos utilizados en los cálculos necesarios para el desarrollo del modelo. En este tipo de fuente de errores se incluyen los errores redondeo y de truncamiento. A diferencia de los errores del modelo, estos errores si se pueden cuantificar y serán objeto de análisis en nuestra cursada.

El error por el uso del ordenador

La cantidad de información numérica que puede mantener un ordenador es finita, por lo tanto, el procesador solo podrá procesar un conjunto finito de números. Como consecuencia de esto cada vez que las operaciones generan información que no pueda ser representada en ese conjunto finito, el procesador se ve obligado a reemplazarlo con una aproximación (el número en sistema binario más cercano). Este reemplazo es lo que genera el error de redondeo; y cada vez que se vuelve a realizar operaciones con esa información (que ya posee un error de redondeo) se produce lo que se denomina “Propagación de error de redondeo”. Esta propagación puede generar resultados totalmente incorrectos.

Para tener una idea de lo que estamos tratando de exponer, empezaremos con un ejemplo trivial. Intentemos realizar en Scilab (la herramienta informática que vamos a utilizar en la asignatura) una división simple. Tratemos de dividir 2 por 3, sabemos que el resultado es una fracción simple. Sabemos además que el resultado es $0,\widehat{66}$. Sin embargo, Scilab nos muestra

```
--> 2./3
 ans  =

    0.6666667
```

Este resultado no es exactamente $\frac{2}{3}$, está claramente y "razonablemente" cerca, pero no es el valor exacto. El número obtenido es resultado de la configuración de Scilab y la forma en que la ha calculado el ordenador esta operación. En definitiva, nosotros no hemos definido cuan cerca está el número de su representación fraccionaria, sino que lo hizo el ordenador.

Veamos otro ejemplo simple: ¿Cuánto debería darnos como resultado la siguiente operación?

$$1 \times 10^{20} + 1 - 1 \times 10^{20} = ?$$

Esperaríamos que Scilab nos devuelva 1. Veamos el resultado de aplicar esta simple operación

```
--> 1e20+1-1e20
 ans  =

    0.
```

Obtenemos un 0, pero deberíamos haber obtenido 1, ¿Algo anda mal con Scilab? ¿Ingresamos mal los valores? ¿Ese el cómputo correcto? ¿Algo hicimos mal? En realidad, no. Hay una razón por la cual estas simples operaciones aritméticas pueden producir resultados inesperados. De esto se trata esta sección, sin embargo, antes probemos un tercer ejemplo, probablemente todos estemos al tanto de la siguiente igualdad:

$$a^2 - b^2 \equiv (a+b)(a-b)$$

Así que la vamos a aplicar considerando que:

$a = \sqrt{1+x}$

$b = \sqrt{x}$

Entonces la igualdad quedaría

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

Que es lo mismo que

$$1 = \left(\sqrt{1+x} + \sqrt{x}\right)\left(\sqrt{1+x} - \sqrt{x}\right)$$

Entonces tenemos una suma de raíces cuadradas que debería ser igual a uno sobre la diferencia de las raíces cuadradas:

$$\frac{1}{\sqrt{1+x} - \sqrt{x}} = \sqrt{1+x} + \sqrt{x}$$

Y podemos expresarla así:

$$\frac{1}{\sqrt{1+x}-\sqrt{x}} - (\sqrt{1+x}+\sqrt{x}) = 0$$

Ahora la evaluaremos numéricamente para los siguientes valores

1) 1×10^{-20}
2) 1
3) 1×10^{3}
4) 1×10^{10}
5) 1×10^{20}

Y esperaríamos que siempre nos dé como resultado 0.

Entonces en Scilab se define la función y se la aplica a todos los valores

```
--> deff('[y]=f(x)','y=1./(sqrt(1+x)-sqrt(x))- (sqrt(1+x)+sqrt(x))')

--> valores=[1e-20, 1, 1e3, 1e10, 1e20];

--> f(valores)
 ans  =

   0.  -4.441D-16  -7.219D-12   0.2233264   Inf
```

Veamos qué resultados obtenemos:

1) Para 1×10^{-20} se obtiene 0 como se esperaba.
2) Para 1 se obtiene $-4{,}441 \times 10^{-16}$ que está "razonablemente" cerca de 0, pero ¿bajo qué criterio podemos aseverar que es **razonablemente** cerca o no?
3) Para 1×10^{3} se obtiene $-7{,}219 \times 10^{-12}$ que también es un número pequeño. Estamos ante una situación similar a la anterior, ¿es lo suficientemente pequeño? Al menos no es tan pequeño como el anterior. ¿Por qué se alejó del cero?
4) Para 1×10^{10} se obtuvo 0,2233264, un número definitivamente que no es cero.
5) Finalmente, para 1×10^{20} el resultado `inf` evidencia que en algún momento el denominador se hace 0, pero para que esto sea así, se debería cumplir que $\sqrt{1+x} = \sqrt{x}$. Esto es un poco extraño.

¿Qué estará sucediendo? Por lo pronto lo único que les puedo asegurar de este lado, es que no hay errores en los cálculos. Para entender los resultados obtenidos es necesario estudiar la aritmética del ordenador.

Representación de Números en Sistemas de Numeración

La forma en que los ordenadores realizan operaciones aritméticas presenta ciertas particularidades. Antes de embarcarse en las mismas es necesario comprender como los números son representados en el ordenador. En primer lugar, definiremos que es un **Sistema de Numeración**. Es una convención para representar cantidades. Todo sistema de numeración posee un conjunto de dígitos básicos mediante el cual puede representar cualquier número. La cantidad de dígitos que utiliza un sistema numérico es definida por la **base,** que es el número

que se usa como referencia para construir un sistema. El sistema de base 10 utiliza 10 dígitos (0, 1, 2, 3, 4, 5, 6, 7, 8, 9) para representar números. Tales dígitos son satisfactorios por sí mismos para contar de 0 a 9. Para grandes cantidades se usa la combinación de estos dígitos básicos; donde la posición o valor de posición especifica su magnitud. El dígito en el extremo derecho de un número entero representa un número del 0 al 9. El segundo dígito a partir de la derecha representa un múltiplo de 10. El tercer dígito a partir de la derecha representa un múltiplo de 100 y así sucesivamente. Por ejemplo, si se tiene el número 86.409 se tienen 8 grupos de 10.000, seis grupos de 1.000, cuatro grupos de 100, cero grupos de 10, y nueve unidades, es decir

$$(8 \times 10^4) + (6 \times 10^3) + (4 \times 10^2) + (0 \times 10^1) + (9 \times 10^0) = 86.409$$

Este tipo de representación se denomina **Notación Posicional**.

Teorema fundamental de la numeración y el Sistema Binario

Este teorema es el fundamento para la notación posicional. Establece la forma general de construir números en un sistema de numeración posicional.

Sean:

- N: número válido en el sistema de numeración
- r: base del sistema de numeración. Número de símbolos permitidos en el sistema.
- x: un símbolo cualquiera de los permitidos en el sistema de numeración.
- n: número de dígitos de la parte entera
- ,: coma fraccionaria. Símbolo utilizado para separar la parte entera de la parte decimal.
- m: número de dígitos de la parte decimal

La fórmula general para construir un número (cualquier número) N en un sistema de numeración posicional con base r es la siguiente

$$N = x_{(n-1)} \dots x_1 x_0, x_{-1} \dots x_{-m} = x_n r^n + \cdots + x_1 r^1 + x_0 r^0 + x_{-1} r^{-1} + \cdots + x_{-m} r^{-m}$$

$$N = \sum_{i=-m}^{n} x_i r^i$$

El valor del número será la suma de cada dígito multiplicado por la potencia de la base correspondiente a la posición que ocupa en el número. Observe además que la formula determinada es un polinomio. Por lo tanto, se puede afirmar que el número N tiene asociado un polinomio $P(r)$ donde r es la base del número N, y los coeficientes x_i de este polinomio constituyen su representación numérica en la base r.

¿Cuál es el sistema de numeración que gestiona internamente un ordenador?

Debido a que las unidades lógicas fundamentales de los ordenadores digitales eran componentes electrónicos, durante mucho tiempo su única forma de representar la información de manera física fue mediante impulsos eléctricos (apagado/encendido) o magnetización (magnetizado/desmagnetizado). Por lo tanto, por una cuestión de practicidad, los números en el ordenador se representan generalmente con un sistema binario o de base 2. En este sistema la unidad se conoce con el nombre de bit, el cual permite almacenar dos estados posibles: 0 o 1; que son utilizados para mapear los impulsos electros o la magnetización. Así, los números en

este sistema se forman dando un peso comprendido entre 0 y 1 a cada potencia de 2. Por ejemplo, el número $(1011{,}01)_2$ se puede escribir como:

$$(1011{,}01)_2 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2}$$

¿Cuál es la metodología que sigue para representar números binarios en un ordenador?

Por razones prácticas, en un ordenador solo puede manejarse una cantidad finita de bits para cada número; esta cantidad o longitud varía de una máquina a otra según el fin que se persiga. Por ejemplo, para trabajos de ingeniería, es mejor trabajar con números grandes; por otro lado, una longitud pequeña es más económica y útil para cálculos y procesamientos administrativos.

Los procesadores de los ordenadores están diseñados para representar números en una cantidad fija de bits. Estos bloques de bits se denominan **palabras**. Además, por una cuestión de estandarización internacional, las palabras se agrupan en 8, 16, 32, 48 o 64 bits. Actualmente se está masificando el uso de ordenadores cuya arquitectura es de 64 bits. Lamentablemente utilizar palabras es una decisión de diseño que trae como consecuencia que las mismas solo podrán representar un número finito de números, y por lo tanto para aquellos números que estén fuera del rango aceptado se deberán aplicar estrategias para no perder la información.

Representación de Números Enteros

Existen diversos métodos para representar números enteros en las palabras de un ordenador. Si se considera el Método de Magnitud con Signo aplicado a una palabra de 16 bits (el tamaño usual para representar enteros en los lenguajes de programación y en los ordenadores), el primer bit representa el signo (0 significa positivo, mientras que 1 significa negativo). Los 15 bits restantes permiten almacenar números desde 000000000000000 a 111111111111111 en sistema binario. Este método expone claramente la limitada capacidad que poseen los ordenadores para almacenar números, en el sentido que el tamaño de la palabra determina el rango finito de valores que puede almacenar. Si se convierte la cota superior al decimal se obtendrá:

$$(1 \times 2^{14} + (1 \times 2^{13}) + (1 \times 2^{12}) + \cdots + (1 \times 2^1) + (1 \times 2^0) = 2^{15} - 1 = 32767$$

Por lo tanto, cada palabra de 16 bits puede contener cualquier número comprendido en el rango [-32768, 32767].

Ejemplo 1: si desea almacenar el valor 525 en base 10 en una palabra de 16 bits, primero deberíamos convertirlo en binario. Scilab posee una función denominada *dec2bin()* que realizará por nosotros el cambio de base

```
--> dec2bin(525)
 ans  =

 1000001101
```

Por tanto, su representación en la palabra de 16 bits del ordenador será la siguiente:

0	0	0	0	0	0	1	0	0	0	0	0	1	1	0	1

Donde se recuerda que la primera posición es el signo y el resto de los dígitos permite representar el número en sistema binario. Observe que el hecho de utilizar una palabra de 16 bits permite contar con un rango de valores que contiene sin problemas el número expresado. El siguiente ejemplo visualiza aún más esta situación.

Ejemplo 2: Represente el entero decimal -173 en un ordenador de 16 bits usando el método de magnitud con signo. Similar al ejercicio anterior, pero recordando que el signo lo podemos representar en la palabra por lo que podemos hacer lo siguiente

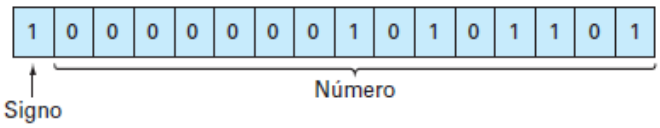
```
--> dec2bin(-173)
at line    12 of function dec2bin ( C:\Program Files\scilab-6.0.2\module

dec2base: Wrong value for input argument #1: Must be between 0 and 2^52.

--> dec2bin(173)
 ans  =

 10101101
```

Que en la palabra tendrá la siguiente forma



En la práctica se utilizan métodos más eficientes que incorporan en forma directa el signo dentro de la magnitud del número para no desperdiciar un bit. Un Ejemplo de este tipo de métodos es la técnica denominada Complemento de 2.

Representación de los números reales

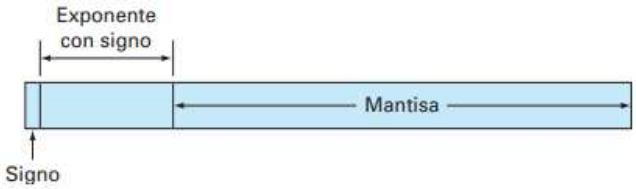
Las cantidades reales se representan en el ordenador usando la técnica del Punto Flotante. Esta representación es similar a la expresión de números mediante notación científica. Por ejemplo, si quisiéramos expresar el número 4000000000 en base 10, sin tener que escribir los 9 ceros, mediante la notación científica quedaría expresada de la siguiente manera: 4×10^9. La representación mediante notación científica se vale de dos números auxiliares: la mantisa y el exponente. De la misma forma, la técnica del Punto Flotante utiliza la mantisa m y el exponente p para representar números en el sistema binario mediante la siguiente expresión

$$m \times 2^p$$

Con $\frac{1}{2} \leq |m| < 1$

donde $m = mantisa, b = 2 = base\ del\ sistema\ numérico\ utilizado, p = exponente$

Tanto la **mantisa** (también denominada **significando)**, como el **exponente** o **característica** constituyen valores enteros. La representación del punto flotante en una palabra del sistema binario se realiza mediante el siguiente esquema



El primer bit se reserva para el signo, luego se utiliza una serie de bits para representar el exponente con signo y finalmente la última serie de bits se reserva para representar la mantisa.

¿Qué significado tiene la expresión Con $\frac{1}{2} \leq |m| < 1$?

Normalmente el valor en coma flotante es previamente **normalizado** con el objetivo de almacenar la mayor cantidad de cifras significativas (concepto que se verá en una sección posterior) y de esa manera optimizar la capacidad de almacenamiento de la palabra.

Por ejemplo, suponga que posee el siguiente número

$$\frac{1}{34} = 0{,}029411765 \ldots$$

Del cual solo se pueden almacenar 4 dígitos decimales. Si se expresa en forma de punto flotante quedaría así

$$\frac{1}{34} = 0{,}0294 \times 10^{0}$$

Más adelante aprenderá que el cero a la derecha de la coma no es una cifra significativa, por tanto, podría expresarse de la siguiente manera

$$\frac{1}{34} = 0{,}2941 \times 10^{-1}$$

Donde todos los decimales son cifras significativas. La normalización genera como consecuencia que el valor absoluto de m quede limitado a

$$\frac{1}{b} \leq |m| < 1$$

Donde b es la base. Por ejemplo, para un sistema de base 10, m estaría entre 0,1 y 1; mientras que para un sistema de base 2, m estaría entre 0,5 y 1.

Entonces, cuando el valor que se desea representar en la mantisa m no se halla en el rango que admite se debe "normalizar" para que sea válido. La normalización provocará que cambie también el valor del exponente p, el cual es el encargado de absorber el factor extra del cambio realizado. Por lo tanto, este rango sirve en realidad para generar una representación única del número.

Una nomenclatura para expresar la mantisa en términos de una cadena de dígitos binarios es la siguiente:

$$m = \pm u_1 u_2 u_3 \ldots u_t$$

Donde el primer dígito es un bit que representa el signo, y cuyo valor es 0 si el signo es positivo y 1 si el signo es negativo; mientras que los u_i indican los dígitos del número binario y su posición o peso dentro del número representado. Esta nomenclatura es equivalente a la siguiente expresión en términos de punto flotante

$$m = \pm u_1 u_2 u_3 \ldots u_t = \pm(u_1 \times 2^{-1} + u_2 \times 2^{-2} + u_3 \times 2^{-3} + \cdots + u_t \times 2^{-t})$$

Con la representación anterior es fácil observar porque la mantisa se representa en el rango $0{,}5 \leq m < 1$.

De manera similar la representación del exponente es:

$$p = \gamma_0\gamma_1\gamma_2 \dots \gamma_L$$

Donde γ_0 se reserva también para el signo, mientras que el resto de los dígitos se utilizan para almacenar la representación binaria de un entero de tamaño fijo.

Por lo tanto, la representación binaria de un número expresado en punto flotante mediante esta nomenclatura de dígitos de bits es

$$\pm\gamma_0\gamma_1\gamma_2 \dots \gamma_L u_1 u_2 u_3 \dots u_t$$

Este conjunto de bits que expresan la representación de números expresadas de esta forma es la "palabra".

La representación de punto flotante permite que tanto fracciones como números muy grandes se expresen en el ordenador. Si este número flotante es aplicado en base 2 también se denomina **número máquina**. Los números máquina forman un conjunto finito de números racionales.

Entonces dado un número máquina, solo existen dos posibilidades: que esté en el rango de los que permite la palabra o caso contrario que no esté en ese rango. De la misma manera luego de realizar una operación el número obtenido puede caer dentro o fuera del rango. Si el número resultante no está en el rango (por ser muy grande o chico) el ordenador lo señalará con un comentario.

El uso de punto flotante tiene diversas desventajas. Por ejemplo, los números de punto flotante requieren más espacio y más tiempo de procesado que los números enteros. Más importante aún es que su uso introduce una fuente de error debido a que la mantisa conserva sólo un número finito de cifras significativas. Este es el denominado error de redondeo del que hemos estado comentando anteriormente.

Estándar IEEE 574

¿No se ha preguntado hasta ahora cual es la cantidad de dígitos reservada a la mantisa y por consecuencia al exponente?

La aritmética con palabras se halla normalizada por el estándar IEEE 574. Establece la definición de la precisión de los tipos de datos permitidos. La precisión en esta norma hace referencia al tamaño de la palabra y la forma en la que esta es dividida para almacenar la mantisa, el signo y el exponente.

La norma propone los siguientes tipos de precisión permitidos:

- Precisión Única: palabras de 32 bits (23 bits para la mantisa, 8 bits para el exponente)
- Precisión Doble: palabras de 64 bits (52 bits para la mantisa, 11 bits para el exponente)
- Precisión Extendida.

La precisión única es usada para la definición de los tipos de datos que manejan muchos lenguajes de programación para representar el número en punto flotante, por ejemplo, en C es

usado por el tipo de dato `float`. Este lenguaje también utiliza la precisión doble propuesta por el estándar para definir el tipo de datos `double`. Es un esfuerzo por aumentar la exactitud de los cálculos adicionando más bits a la mantisa. Se logra utilizando dos palabras, la primera se estructura para signo, exponente y mantisa; mientras que los bits de la segunda se utilizan para ampliar la mantisa de la primera. La desventaja es que consume más memoria.

Respecto de la precisión extendida se puede mencionar que, si bien algunos lenguajes de programación y herramientas científicas lo incorporan, en la práctica aún no resultan útiles. Desafortunadamente, los tipos de datos con precisión extendida no son realmente portátiles, provocando que no sean uniformes en diversas implementaciones. Por lo tanto, en general, no se recomienda el uso de estos tipos de datos y, de hecho, en la gran mayoría de las aplicaciones solo se usaría un tipo de datos de doble precisión. Como regla general, debería tener una muy buena razón para no usarlos por defecto.

Este estándar también define varios tipos de números flotantes especiales que son combinaciones especiales de bits. Por ejemplo:

- ✓ Infinito Positivo: conocido como `inf` y definido como $x < \inf \quad \forall\, x\ finito$, esto es: un número especial que es estrictamente más grande que cualquier número finito. Se supone que representa el resultado de cálculos que se desbordan cuando los resultados son demasiado grandes para caber en la memoria del tipo de datos dado.
- ✓ Infinito Negativo: conocido como `-inf` y definido como $x > \inf \quad \forall\, x\ finito$, esto es: un número especial que es estrictamente más pequeño que cualquier número finito. Se supone que representa el resultado de cálculos que se desbordan cuando los resultados son demasiado pequeños para caber en la memoria del tipo de datos dado.
- ✓ Not-a-number: conocido como `nan` y definido como $x = nan \leftrightarrow x \neq x$. El estándar propone la existencia de un objeto curioso llamado NAN cuyo significado literal es "no es número". Está destinado a representar los resultados de cálculos inválidos. Por ejemplo, si desea realizar una división por cero, ¿cuál es el resultado? Esta operación inválida genera `nan`. Dicho de otra manera, el resultado no es un número válido, y es este aspecto lo que desea expresar la desigualdad $x \neq x$.

Existen muchos otros números de punto flotante especiales y otras particularidades interesantes a considerar cuando se realizan operaciones con números de coma flotante en ordenadores. Todo ingeniero que trabaje con herramientas científicas para trabajos físicos o matemáticos debería conocerlas, especialmente los de las ciencias de la computación, ya que de no hacerlo podría tener consecuencias realmente catastróficas, como el siguiente:

El 4 de junio de 1996 el cohete no tripulado *Ariane 5*, lanzado por la Agencia Espacial Europea, explotó 40 segundos después de despegar. Después de una década de desarrollo con una inversión de 7 mil millones de dólares, el cohete hacía su primer viaje al espacio. El cohete y su carga estaban valuados en 500 millones de dólares. Un comité investigó las causas de la explosión y en dos semanas emitió un reporte. La causa de la falla fue un error de software en el sistema de referencia inercial. Específicamente un número de punto flotante de 64 bits, relativo a la velocidad horizontal del cohete con respecto a la plataforma, fue convertido a un entero con signo de 16 bits. El número entero resultó mayor que 32,768, el entero con signo más grande que puede almacenarse en una palabra de 16 bits, fallando por ello la conversión.*

Por tal motivo para mayores detalles puede consultar el artículo: What every computer scientist should know about floating-point arithmetic, publicado en el año 1991 recuperado de: https://doi.org/10.1145/103162.103163

Sin embargo, con los detalles ofrecidos hasta ahora se puede entender los resultados que arrojó Scilab, que aparentan ser errores de cálculo pero que no lo son y tienen su explicación en la forma en que los ordenadores representan los números en punto flotante.

Como resumen de todo lo explicado hasta aquí podemos decir que la representación de números reales mediante la técnica del Punto Flotante es una forma de codificar un número real mediante dos enteros binarios: la mantisa y el exponente. Esto trae aparejado algunas consecuencias. La primera observación inmediata es que no todos los números reales pueden representarse exactamente. Esto se debe al número finito de bits disponibles dentro de la palabra para la mantisa y el exponente, esto es, solo se puede representar un conjunto finito de valores.

Manejo del Error en el Cálculo Numérico

En esta disciplina se estudian básicamente los errores de redondeo y los errores de truncamiento, que son los que pueden cometer los ordenadores cuando se aplican métodos numéricos. De hecho, los métodos numéricos se tornan valiosos si se los implementa en un ordenador. Obtenida una aproximación $\bar{x}$ al valor verdadero x_t que representa la magnitud de la solución, entonces estamos ante la siguiente relación

$$x_t = \bar{x} + e_t$$

Donde e es el error cometido, t es una referencia a `true` que indica que el valor de la variable es exacto, y por lo tanto x_t es el valor exacto y e_t se denomina **error verdadero**.

Por lo tanto

$$e_t = x_t - \bar{x}$$

Cuando $\bar{x} > x_t$, se dice que la aproximación es **por exceso**, mientras que si $\bar{x} < x_t$, se dice que la aproximación es **por defecto**.

Generalmente, en los métodos numéricos que se estudiarán interesa la magnitud del error sin considerar su signo, entonces se define

$$|e_t| = |x_t - \bar{x}|$$

Como el **error absoluto verdadero** cometido al realizar la aproximación. Este error presenta una desventaja debido a que no toma en consideración el orden de la magnitud del valor que se estima.

Por ejemplo, un error de un centímetro es mucho más significativo si se está midiendo un remache en lugar de la longitud un puente.

Una manera de tomar en cuenta las magnitudes de las cantidades que se evalúan consiste en normalizar el error respecto al valor verdadero, es decir:

$$|\varepsilon_t| = \frac{|e_t|}{|x_t|} = \frac{|x_t - \bar{x}|}{|x_t|}$$

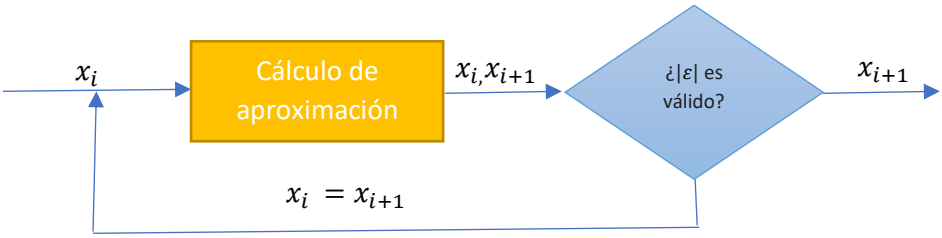
Lo cual se conoce como **error relativo verdadero**, y donde $x_t \neq 0$.

Por otro lado, este error se puede multiplicar por 100% para expresarlo en términos porcentuales quedando:

$$|\varepsilon_t|100\% = \frac{|e_t|}{|x_t|}100\% = \frac{|x_t - \bar{x}|}{|x_t|}100\%$$

Que representa el **error relativo verdadero porcentual**.

Un problema práctico de las definiciones anteriores es que normalmente en los problemas de ingeniería abordados por el análisis numérico no se contará con el valor verdadero.
La alternativa que brinda la disciplina consiste en normalizar el error mediante la aplicación de métodos iterativos que obtengan el error considerando las aproximaciones previamente realizadas, con la esperanza de que cada iteración realizada del método produzca mejores aproximaciones, basándose en una convergencia determinada por la aceptación de un error válido calculado en cada iteración del método. El término **iteración** hace referencia a la repetición de los pasos definidos dentro del método, o re-ejecución del método.
Esto es representado esquemáticamente en la siguiente imagen



Donde x_i es el valor de entrada para el método. El método calcula una aproximación x_{i+1} y en base ambos valores (x_i y x_{i+1}) se estima una aproximación al error absoluto o relativo. Si ese error es aceptable, entonces se dirá que x_{i+1} es una aproximación válida, sino se convertirá en el nuevo valor de entrada para el método numérico y se volverá a estimar la aproximación (este es el proceso iterativo).
Entonces

$$|e| = |x_{i+1} - x_i|$$

Es el **error absoluto de una aproximación**, donde x_{i+1} es la aproximación actual generada por el método numérico, mientras que x_i es el valor de una aproximación anterior.
De la misma manera

$$|\varepsilon| = \frac{|e|}{|x_{i+1}|} = \frac{|x_{i+1} - x_i|}{|x_{i+1}|}$$

Es el error relativo de una aproximación y

$$|\varepsilon|100\% = \frac{|e|}{|x_{i+1}|}100\% = \frac{|x_{i+1} - x_i|}{|x_{i+1}|}100\%$$

Es el error relativo porcentual.

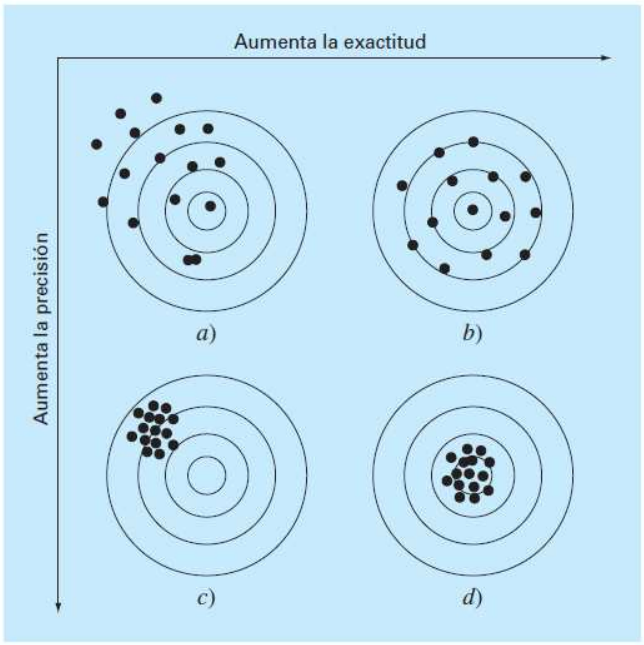
Análisis de Errores en el Cálculo Numérico

Debido a que la mayoría de los métodos numéricos son iterativos, y considerando la interesante forma que tienen para definir el error de las sucesivas aproximaciones, analizar el error de un resultado numérico plantea la necesidad de establecer criterios sistemáticos que permitan asegurar que las aproximaciones generadas sean lo suficientemente *exactas y precisas*, y por consecuente, que el error calculado sea válido.

La **exactitud** se refiere a que tan cercano está el valor calculado o medido (es decir $\bar{x}$) del valor verdadero (es decir x). Lo contrario a este concepto es la **inexactitud** (conocida también como **sesgo**), que se define como una desviación sistemática del valor verdadero.

La **precisión** se refiere a que tan cercanos se encuentran los valores calculados o medidos unos de otros. Lo contrario a este concepto se denomina **imprecisión** (también llamada **incertidumbre**), que hace referencia a la magnitud en la dispersión de los valores calculados o medidos.

La forma clásica de ilustrar estos conceptos se realiza a través de una analogía con los resultados sobre una diana en la práctica de tiro. Se plantean 4 resultados diferentes, uno en cada diana, para diferentes tiradores. Cada uno de los resultados se consideran, por analogía, como las aproximaciones que genera la técnica numérica empleada (los $\bar{x}$); mientras que el centro del blanco representa el valor verdadero (es decir x) y el tirador refleja el método numérico aplicado.



Si analizamos la diana ***c)***, puede observarse que los disparados están muy juntos, se dice que son bastantes precisos. Sin embargo, están claramente alejados del centro de la diana, es decir son inexactos. Los errores de redondeo que puede producir el ordenador tienden a generar inexactitud de los resultados, quitando eficacia al método. En cambio, los errores de truncamiento efectivamente producen inexactitud.

Observe ahora la diana ***a)***, en este caso el tirador acertó una vez, pero sus demás tiros parecen estar alrededor de un punto de la zona superior izquierda de la diana, aunque en un grado menor al de la diana c). En este caso los resultados son en su conjunto inexactos e imprecisos. En este caso probablemente hay una propagación de errores de truncamiento.

En las dianas ***b)*** y ***d)***, los tiros tienden a estar centrados respecto al centro, es decir resultan ser exactos, sin embargo, el tirador de la diana d) es más preciso, por ese motivo sus disparos fueron más efectivos o eficientes.

De esta analogía se puede aseverar que los métodos numéricos deben ser lo suficientemente exactos (eficacia) o sin sesgo para satisfacer los requisitos de un problema particular de ingeniería; y deben ser suficientemente precisos para ser adecuados en el diseño de la ingeniería (eficiencia).

En el análisis numérico tanto la inexactitud como la imprecisión son considerados errores en las predicciones, por los que se los referencia como errores. Al respecto, hay que considerar que:

- ✓ Los datos de entrada para un método numérico rara vez son exactos, puesto que se basan en ensayos experimentales o bien son estimados.
- ✓ Los métodos numéricos también introducen errores de varios tipos (esencialmente truncamiento y redondeo).
- ✓ Las operaciones con valores aproximados pueden provocar la propagación de errores, es decir una acumulación de errores que provoque un resultado inaceptable.

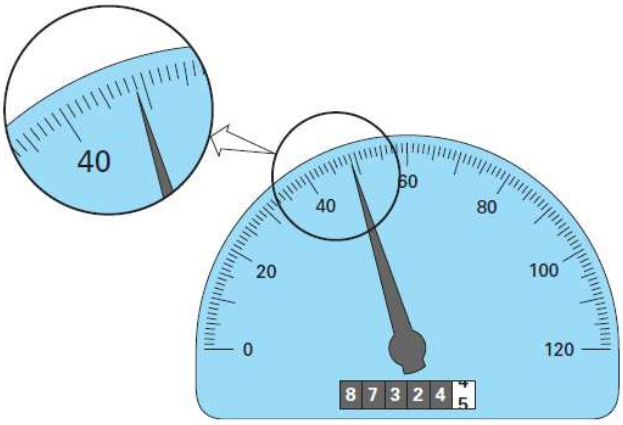
Este marco conceptual tiene por objetivo conducir a que uno de los criterios que se suele utilizar para garantizar la precisión y exactitud de las aproximaciones generadas por los métodos numéricos empleados radica en el convenido de las cifras significativas.

Cifras significativas

Como se mencionó, las cifras significativas constituyen un criterio clave para trabajar o informar con confianza un valor numérico aproximado en términos de precisión y exactitud. Se definen

como el número de dígitos, más un dígito estimado, que se pueden usar con confianza. Para ilustrar los componentes de esta definición, observe la siguiente figura extraída de un ejemplo de Chapra (2007), donde se visualizan un velocímetro y un odómetro (contador de kilometraje) de un automóvil.

Suponga que se desea establecer la velocidad marcada con dos enteros y un decimal. Alguien podría decir que marca 48, 8 $^{km}/_{h}$, mientras que otra persona podría indicar 48,9 $^{km}/_{h}$. Por lo tanto, debido a los límites del instrumento, únicamente se emplean con confianza los dos primeros dígitos. Para estimaciones del tercer dígito (o más allá) sólo se considerarían aproximaciones. Por lo que aquí se están usando 3 cifras significativas: 2 con confianza + 1 estimada. Observe que sería ridículo afirmar, considerando el velocímetro de la figura, que el automóvil viaja a 48,8642138 $^{km}/_{h}$.

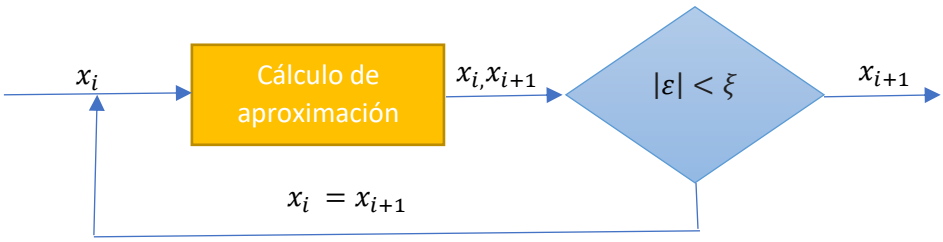


Si observamos nuevamente la figura, pero ahora centrándonos en el odómetro se puede constatar que muestra su valor con hasta seis dígitos confiables, es decir, el automóvil ha recorrido un poco menos de 87.324,5 km durante su uso. Aquí el séptimo dígito (y los siguientes) resultan inciertos. Por lo tanto, los valores que expresa el odómetro poseen 7 cifras significativas (6 con confianza + 1 estimado)

De esta manera, un valor numérico expresado en términos de cifras significativas aporta información no ambigua ni superflua acerca de una determinada medida experimental o aproximación generada, debido a que ocupan una posición igual o superior al orden o posición del error cometido.

Para dimensionar la importancia de esta afirmación se invita que tenga en cuenta lo siguiente:

- ✓ Si los datos numéricos que alimentan los métodos numéricos utilizados son resultado de mediciones experimentales, lo ideal es que sean previamente validados mediante procedimientos estadísticos (de manera resumida sería realizar más de una medida y proceder con el tratamiento estadístico de los datos para establecer así un resultado con un buen límite de confianza, idealmente superior al 90%), pero si esto no es posible lo más común es utilizar el denominado convenio de cifras significativas. ¿Objetivo perseguido? Poder establecer un nivel de confianza sobre los datos de entrada.
- ✓ La funcionalidad general de los métodos numéricos comentadas hasta ahora se refina: para desestimar un resultado generado por una iteración del método numérico el algoritmo calcula el error absoluto o relativo de la aproximación, que debe ser mayor a una tolerancia de error ξ previamente prefijada, esto es



Por lo tanto, es necesario:

1. Desarrollar criterios para especificar que tan precisos son los resultados obtenidos por el método numérico.
2. Atender la situación que se produce cuando un número utilizado en el método numérico no puede expresarse mediante un número finito de cifras.

Para lograr esto, se utiliza el convenido de cifras significativas. ¿Objetivo perseguido? Determinar con seguridad la confianza respecto de los números que devuelve el método numérico.

Dimensionada la importancia de expresar un número aproximado mediante sus cifras significativas, la siguiente pregunta es ¿Cómo se logra obtener esta expresión?

Reglas para determinar las cifras significativas de un número

Uno puede estar tentado en pensar que entre más dígitos se reporten y/o usen en las operaciones se van a obtener mejores resultados. Sin embargo, esta suposición conduce a errores debido al hecho de que los resultados pueden presentar muchas más cifras significativas que las que puede manejar el instrumento de medición o el ordenador que ejecuta el método numérico.

Recuerde que, del concepto de cifras significativas se puede intuir que las mismas tienen un significado real o aportan alguna información. Las cifras significativas de un número vienen determinadas generalmente por su incertidumbre.

Por ejemplo, consideremos una medida de longitud que arroja un valor de 4.325,3528 metros con un error de 0,8 metros. Puesto que el error es del orden de decenas de metro, es evidente que todas las cifras del número que ocupan una posición menor que las decenas no aportan ninguna información. Es decir, no tiene sentido brindar el número con una exactitud de diez decenas de mil, si afirmamos que el error es de casi un metro. Como puede notar, cuando se expresa un número debe evitarse siempre la utilización de cifras no significativas.

Por este motivo, el trabajo con cifras significativas plantea un conjunto de reglas que permiten determinar adecuadamente tanto las cifras significativas de los números involucrados en las operaciones, como del reporte adecuado de los resultados de esas operaciones en términos de cifras significativas.

Regla 1: cualquier digito distinto de cero es significativo. Por ejemplo, 438 tiene tres cifras significativas.

Regla 2: los ceros situados en medio de números diferentes de cero son significativos. Por ejemplo, 402 tiene tres cifras significativas, y 30002 tiene cinco cifras significativas.

Regla 3: los ceros a la izquierda del primer número distinto de cero no son significativos. Por ejemplo, 0,0023 tiene dos cifras significativas.

Regla 4: los ceros que se encuentran después de la coma y después de un digito distinto de cero, son significativos. Por ejemplo 10,00 tiene 4 cifras significativas, y 0,0030 tiene dos cifras significativas.

Regla 5: en los números enteros, los ceros situados después de un dígito distinto de cero pueden ser o no significativos. Por ejemplo, 600 puede tener una cifra significativa (6), dos (60), o tres (600). Para conocer el número correcto de cifras significativas necesitamos conocer más información acerca de cómo fue generado el número (por ejemplo, si el número es una medición, necesitamos conocer la precisión del instrumento de medición empleado). Otra manera de conocer el número correcto de cifras significativas dentro de la misma expresión

consiste en reportar el número en notación científica. Por ejemplo, 6×10^2 tiene una cifra significativa, $6{,}0 \times 10^2$ tiene dos cifras significativas, y $6{,}00 \times 10^2$ tiene tres cifras significativas.

Cifras significativas de un valor aproximado con respecto a un valor verdadero

Tanto los datos de entrada como los resultados obtenidos a través métodos numéricos también plantean una limitación adicional al hecho de determinar las cifras significativas de una aproximación por el uso del ordenador. Recuerde que esos valores pueden poseer más cifras significativas que las que puede contener las palabras). Además, las operaciones numéricas plantean diversos desafíos para que sus resultados no generen errores, y aunque parezca paradójico, una mayor amplitud de dígitos puede generar pérdida de cifras significativas.

Por tales motivos, es conveniente relacionar la tolerancia de error con el número de cifras significativas en la aproximación. En (Chapra, 2007) se indica que, si se considera el siguiente criterio, se podrá tener la seguridad de que el resultado es correcto en al menos n cifras significativas.

$$\xi = (0{,}5 \times 10^{2-n})\%$$

Ejemplo: Suponga que desea utilizar la Serie Maclaurin para aproximar $e^{0{,}5}$ con la seguridad de confiar al menos 3 cifras significativas, sabiendo que el resultado exacto es 1,648721

1) Se calcula la tolerancia de error que asegure la exactitud en al menos 3 cifras, significativas, esto es $n = 2$, entonces
$$\xi = (0{,}5 \times 10^{2-3})\% = 0{,}05\%$$
2) Ahora se procede a aplicar la serie de Maclaurin, incorporando en cada paso un término hasta lograr que el error relativo cometido sea menor que la tolerancia de error. Recuerde que a mayor cantidad de términos la serie es más exacta y su forma es la siguiente:
$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \cdots + \frac{x^n}{n!}$$
3) Observe el siguiente cuadro

Términos	Resultado	$\lvert\varepsilon_t\rvert 100\%$	$\lvert\varepsilon\rvert 100\%$
1	1	39,3	
2	1,5	9,02	33,3
3	1,625	1,44	7,69
4	1,645833333	0,175	1,27
5	1,648437500	0,0172	0,158
6	1,648697917	0,00142	0,0158

Observe que en la aproximación era exacta al agregar el cuarto término (luego de redondear) a la Serie de Maclaurin pero, al seguir la especificación de la tolerancia de error calculada, llegamos hasta la incorporación del sexto término de la serie. Además, en este punto la exactitud aumenta a cinco cifras.

Esto se debe a que tanto $\lvert\varepsilon\rvert 100\%$ como $\xi = (0{,}5 \times 10^{2-n})\%$ son del tipo conservadoras, es decir que el resultado es tan bueno, al menos como lo especifican.

En realidad, para el error relativo porcentual existen situaciones donde esta característica no se llega a cumplir, pero para la segunda siempre se cumple.

Por ese motivo, la experiencia muestra que una tolerancia de error $\xi = 0{,}1 \times 10^{-3}$ suele preservar la conservación de las cifras significativas.

Tratamiento del error de Redondeo

Como se mencionó anteriormente el uso de ordenadores para la resolución de métodos numéricos puede provocar la generación de errores de redondeo debido a que no se pueden expresar con exactitud ciertos números, tales como π o $\sqrt{7}$. Esto se aplica también a algunos números en base 10 debido a que los ordenadores utilizan la base 2. Y por si fuera poco también es probable que aparezca cuando se desean representar números muy grandes o pequeños. Además, se debe considerar que dichos números normalmente se utilizarán para operar con otros números generando posiblemente resultados que también serán redondeados, lo cual puede llevar a la pérdida de cifras significativas.

Si los valores que se operan están expresados en términos de cifras significativas y la operación necesita redondearse, **no es admisible que se gane o pierda incertidumbre mientras se realizan operaciones aritméticas**.

Por este motivo los ordenadores aplican reglas de redondeo según sean necesarias. La aplicación de estas reglas generalmente permite que los sucesivos errores de redondeo se compensen generando resultados precisos y exactos sin perder cifras significativas. Las reglas son:

Regla 1: para cada valor que se involucrará en la operación se conservan las cifras significativas y el resto se descarta. El último dígito menos significativo se trata de la siguiente manera (regla de redondeo):

a) Se aumenta en 1 si el primer dígito descartado es mayor que 5.
b) Se mantiene su valor si el primer dígito descartado es menor que 5.
c) Si el primer dígito descartado es 5 o es 5 seguido de ceros, entonces se debe evaluar el dígito menos significativo de la siguiente manera: si es par se mantiene su valor, pero si es impar se incrementa en 1.

Regla 2: en una suma o una resta el número de dígitos del resultado viene marcado por la posición del menor dígito común de todos los números que se suman o se restan. Por tanto, en una adición o una sustracción el número de cifras significativas de los números que se suman o restan no refieren a ningún criterio específico. Por ejemplo, si se evalúan las siguientes sumas de valores expresados en términos de cifras significativas previamente redondeadas:

a) $4{,}3 + 0{,}030 + 7{,}31 = 11{,}64 \cong 11{,}6$
b) $34{,}6 + 17{,}8 + 15 = 67{,}4 \cong 67$
c) $34{,}6 + 17{,}8 + 15{,}7 \cong 68{,}1$

Tanto para a) como c), el menor dígito significativo común entre los operandos es el dígito de una décima. Por lo tanto, el resultado debe expresarse considerando esta cifra. Po ese motivo el resultado de a) se redondea.

Para el caso del ejemplo b) el menor dígito significativo común es la unidad, por ese motivo se redondea y no se considera la parte decimal.

Por otro lado, como se comentará más adelante, el caso de la resta es muy particular. Para exponer de que se trata esto considere el siguiente ejemplo:

30,3475-30,3472=0,0003

Se puede observar que, si bien cada operando tiene 6 cifras significativas y coinciden en la posición del dígito menos significativo, el resultado solo posee 1 cifra significativa. Es decir, esta operación puede provocar **pérdida de cifras significativas**, por lo que se recomienda postergarlas lo más que sea posible, ya que la pérdida de cifras significativas puede afectar la precisión del cálculo.

Regla 3: en un producto o una división el resultado debe redondearse de manera que contenga el mismo número de dígitos significativos que el número de origen que posea menor número de dígitos significativos.

Por tanto, a diferencia de la suma o la resta, en estas operaciones el número de dígitos significativos de las cantidades que intervienen en la operación sí determina el criterio para definir el número de dígitos significativos del resultado. Por ejemplo:

a) $\frac{24 \times 4{,}52}{100{,}0} = 1{,}0848 \cong 1{,}1$
b) $\frac{24 \times 4{,}02}{100{,}0} = 0{,}9648 \cong 0{,}96$
c) $3{,}14159 \times 0{,}25^2 \times 2{,}352 = 0{,}4618141 \ldots \cong 0{,}46$

En los tres ejemplos expuestos el menor número de cifras significativas de los diferentes factores que intervienen en las operaciones es dos: se trata concretamente del número 24 en los ejemplos a) y b), y del número 0,25 en el ejemplo c). Por tanto, los resultados se deben redondear a las dos cifras significativas que poseen esos valores.

Regla 4: En el logaritmo de un número se deben mantener tantos dígitos a la derecha de la coma decimal como cifras significativas tiene el número original.

Regla 5: En el antilogaritmo de un número se deben mantener tantas cifras significativas como dígitos decimales posee el número original.

Ejemplos para números en base 10

a) $\log 3{,}53 = 0{,}5477747 \cong 0{,}548$
b) $\log(1{,}200 \times 10^{-5}) = -4{,}9208188 \cong -4{,}9208$
c) $Anti\ \log(8{,}9) = 10^{8{,}9} = 7{,}94328 \times 10^8 \cong 8 \times 10^8$
d) $Anti\ \log(8{,}900) = 10^{8{,}9} = 7{,}94328 \times 10^8 \cong 7{,}94 \times 10^8$

En el ejemplo a) el número de cifras significativas del número 3,53 es tres y, por lo tanto, el número de decimales que tiene su solución es tres.

El número del ejemplo b) tiene cuatro cifras significativas y su logaritmo se expresa con 4 decimales, llegando a totalizar 5 cifras significativas.

En cuanto a los antilogaritmos de los ejemplos c) y d), el primero tiene una sola cifra decimal y su solución se expresa con una cifra significativa; el segundo tiene tres cifras decimales por lo que el resultado debe tener 3 cifras significativas.

<u>Una conclusión inicial sobre el análisis de error</u>

La convención de cifras significativas es una valiosa herramienta para obtener y asegurar aproximaciones con alta calidad de exactitud y precisión. Sin embargo, si profundizáramos en la propagación de errores, podríamos destacar que hay situaciones donde su aplicación no brinda soluciones del todo satisfactorios; debido a que la realización de operaciones aritméticas con cifras significativas en ocasiones aumenta la incertidumbre respecto a lo esperado, como consecuencia de considerar en una unidad la incertidumbre del último dígito de un número.

Es claro que este aumento de la incertidumbre será tanto mayor cuanto mayor sea el número de operaciones que se encadenen y, por tanto, sería conveniente determinar el valor de la incertidumbre si se quiere estar seguro de conocer la progresión del error cometido en las operaciones realizadas. Esto refuerza la idea de que el estudio de la incertidumbre no puede ser reemplazado por ninguna técnica, y de hecho no realizarla es lo que provoca la pérdida de cifras significativas. Sin embargo, los libros de cálculo, así como los de física o química normalmente

expresan la tendencia a realizar cálculos con datos cuya precisión viene indicada sólo por el convenio de las cifras significativas.

Una práctica común en la resolución de problemas **es mantener al menos un dígito de más durante los cálculos** para prevenir el error de redondeo (dígito de reserva). Como los ordenadores y calculadoras actuales normalmente pueden trabajar con más de un dígito de reserva lo importante será realizar el redondeo después de que se hayan acabado los cálculos (es decir, si usas el ordenador solo tienes que cerciorarte de que el resultado final sea expresado en términos de cifras significativas).

Si en las aplicaciones donde se utilizan métodos numéricos es prácticamente imposible determinar exactamente la magnitud de los errores de redondeo. ¿Qué más podemos hacer?

Lo único que podemos hacer; y ¡que es de fundamental importancia! es identificar las posibles causas que puedan provocar una propagación de errores mayor a la admisible. Esto se reflejará en la elaboración de mejores algoritmos o en la elección del método más apropiado para resolver un problema como consecuencia de la conceptualización de dos términos que surgen del estudio de la propagación de errores: algoritmo inestable y problema "mal condicionado". Un algoritmo inestable hace referencia a que es posible (y se debe) mejorar el método utilizado para la resolución del problema, mientras que el segundo término es mucho más esencial.

Propagación de errores

¿Han oído hablar del efecto mariposa? Hace referencia a cuan sensible es un sistema físico ante un pequeño error en las condiciones iniciales, todo esto enmascarado dentro de una historia de ciencia ficción. En efecto, en el año 1952 el escritor estadounidense Ray Bradbury escribió un interesante relato acerca de un grupo de exploradores que viajan al pasado prehistórico de la Tierra. Por accidente, uno de los exploradores se salió del sendero que les permitía ser observadores sin intervención en la historia y pisa una pequeña mariposa. Uno podría afirmar que este hecho no tiene relevancia en el devenir de la historia. Sin embargo, cuando volvieron al presente se les presentaron dos cambios de diferente magnitud. Por un lado, la composición del aire presenta ligeros cambios, mientras que por otro lado resulta que la última elección presidencial en EEUU había cambiado.

Este relato expresa una metáfora donde la mariposa pisada representa *un pequeño error*, que va afectando los sucesos de la historia a escalas cada vez mayores a medida que se suceden los años. Se puede discutir la validez de la probabilidad de que la ausencia del aleteo de una mariposa genere semejante cambio en la historia del mundo, pero su objetivo es narrar *lo importante que podría ser un pequeño error si se propaga y amplifica* por mucho tiempo.

Partamos del siguiente ejemplo: Encuentre el número entero más cercano a la siguiente expresión:

$$50\sqrt{5}\sqrt{7}$$

Estamos ante un predicamento. No se sabe cuál es el valor exacto de las raíces cuadradas. Ante esta situación una propuesta inicial es tomar valores aproximados para cada una de las raíces. Supongamos que optamos por tomar dos decimales para cada una de las raíces, después de todo, por alguna razón desconocida es una práctica común entre los estudiantes (y las personas ajenas a esta materia) trabajar solamente con esa cantidad de decimales. Entonces podremos indicar sin redondear que:

$$\sqrt{5} \approx 2{,}23 \; y \; \sqrt{7} \approx 2{,}64$$

Entonces

$$50 \,.\, 2{,}23 \,.\, 2{,}64 = 294{,}36 \Rightarrow Entero\ más\ cercano\ es\ 294$$

Pero si utilizamos las raíces redondeadas a dos decimales, los mismos cálculos nos devolverían

$$\sqrt{5} \approx 2{,}24 \; y \; \sqrt{7} \approx 2{,}64$$

$$50 \,.\, 2{,}24 \,.\, 2{,}64 = 295{,}68 \Rightarrow Entero\ más\ cercano\ es\ 296$$

Observe cómo el hecho de redondear ha afectado notoriamente el resultado obtenido. Por otro lado, observemos el resultado que nos brinda Scilab

```
--> 50*sqrt(5)*sqrt(7)
 ans  =

   295.80399
```

A fines del ejemplo, obtenemos el mismo resultado que el anterior (296 sería el entero más cercano). Sin embargo, si analizamos la parte decimal podemos ver que hay una diferencia de por lo menos 0,10. Se podría decir que Scilab ha usado mayor cantidad de decimales al realizar el cálculo. Sin embargo, a pesar de usar una herramienta científica se ha trabajado con números aproximados (por tanto, existe un error) y estos ejemplos permiten dimensionar el efecto de la propagación de errores cuando se trabaja con números redondeados; los cuales tienen un efecto especialmente notorio cuando se trabaja con operaciones del tipo multiplicación o división.

Causas de errores graves, y su propagación en los cálculos del ordenador

Además de brindar una representación inexacta de los números, la aritmética realizada en el ordenador también es inexacta. Usando números con representación en Punto flotante con m dígitos en la mantisa, las operaciones aritméticas elementales no se pueden ejecutar de manera exacta siempre. Por ello no se puede esperar reproducir de forma exacta las operaciones aritméticas en un ordenador digital. Deberemos contentarnos con sustituirlas por otras, denominadas **Operaciones de Punto Flotante** (cuyos símbolos se expresan de la siguiente manera: $\oplus,\ominus,\otimes,\oslash$), que las aproximen tanto como sea posible. Esto se puede conseguir, definiéndolas, usando el soporte que brinda las cifras significativas y el redondeo. Así se define:

Suma: $x \oplus y = fl(fl(x) + fl(y))$

Resta: $x \ominus y = fl(fl(x) - fl(y))$

Multiplicación $x \otimes y = fl(fl(x) \times fl(y))$

División $x \oslash y = fl(fl(x)/fl(y))$

Donde de manera idealizada se expresa que se normaliza cada valor a su representación en punto flotante, luego se aplica la operación correspondiente en aritmética exacta a esas representaciones, y finalmente se procede a la conversión del resultado exacto a su representación de punto flotante.

Se ha comprobado que las operaciones en punto flotante no verifican las siguientes reglas aritméticas normales:

a) $x \oplus y = x \nRightarrow y = 0$
b) $x \oplus (y \oplus z)$ puede ser distinto de $(x \oplus y) \oplus z$. Observe que esto significa que a veces puede cumplirse la asociatividad de la suma, mientras que otras veces no.
c) $x \otimes (y \otimes z)$ puede ser diferente de $(x \otimes y) \otimes z$. Al igual que con el caso anterior, puede o no cumplirse la propiedad distributiva de la multiplicación.

Por una cuestión técnica, ante la incertidumbre de estas propiedades, se considera que efectivamente no se cumplen, con el objetivo de evitarlas.

La propagación de errores se define como la generación de errores al realizar operaciones de coma flotante que generan resultados con perdidas de cifras significativas o que no se pueden representar en la palabra del ordenador, y que posteriormente se vuelven a utilizar en los próximos cálculos realizados.

El análisis de esta propagación de errores intenta definir como los errores de la aproximación y los errores de redondeo que se presentan en el método numérico se propagan y cambian el resultado final que debería generar el método.

No es el objetivo de la materia adentrarse en las diferentes técnicas que se utilizan para realizar estos estudios. Generalmente se aplican factores denominados comúnmente **índices de condicionamiento**, que determinan si y cómo los pequeños errores relativos que se cometan en los valores utilizados producen errores relativos muy grandes en los resultados finales. En ese caso se dice que el problema está **mal condicionado**.

Curiosamente los estudios científicos arrojan que los pequeños errores relativos en las operaciones elementales descriptas, más la raíz cuadrada, no influyen fuertemente en la propagación de los errores. Entonces ¿Qué provoca que la imprecisión en los resultados finales?

La causa más grave reside en malos diseños del software, que realizan operaciones con datos que “no son compatibles”. Para explicar a lo que nos estamos refiriendo y a modo de practicidad, supongamos que el ordenador trabaja con un sistema decimal, cuya mantisa soporta 4 dígitos decimales y el exponente trabaja con 2 dígitos, de los cuales el primero es usado para el signo. Si consideramos que además la palabra reserva un bit para el signo del número, entonces la palabra que usaremos consta de 7 bits. Usaremos el término “normalizar” a la expresión de un valor numérico en notación científica para poder representarla en la palabra del ordenador. Por

ejemplo, si contamos con 3 números: 3,0; 7.956.000 y -0,0000025211, entonces normalizarlos significa que:

a) $3{,}0 = 0{,}3000 \times 10^{1}$
b) $7.956.000 = 0{,}7956 \times 10^{7}$
c) $-0{,}0000025211 = -0{,}2521 \times 10^{-5}$

Comprendido lo anterior, procederemos a ejemplificar los errores de diseño más graves que se cometen para darnos una idea clara de como afectan al resultado del método:

1) Suma o resta de números muy diferentes en términos de magnitud
   Se plantea sumar 0,002 a 600 en nuestro ordenador, entonces
   $0{,}002 = 0{,}2000 \times 10^{-2}$ con 4 cifras significativas
   $600 = 0{,}6000 \times 10^{3}$ con 4 cifras significativas
   Para poder sumar estos números deben estar normalizados al mismo exponente

$$\begin{array}{rr} & 0{,}000002 \times 10^{3} \\ + & 0{,}600000 \times 10^{3} \\ \hline & 0{,}600002 \times 10^{3} \end{array}$$

   Pero este resultado se debe expresar en una mantisa de 4 dígitos, por lo tanto, el redondeo generaría el siguiente resultado: $0{,}600000 \times 10^{3}$. Es decir, en términos de resultado, la suma nunca se realizó. Por este motivo se recomienda evitar sumar o restar dos números muy diferentes.
2) Resta de números muy parecidos. Ya habíamos compartido que la resta de números puede provocar pérdida de cifras significativas. Ahora observaremos otro aspecto de especial cuidado cunado se intenta restar números muy parecidos. Suponga que se desea restar 0,2144 de 0,2145. Entonces

$$\begin{array}{rr} & 0{,}2145 \times 10^{0} \\ - & 0{,}2144 \times 10^{0} \\ \hline & 0{,}0001 \times 10^{0} \end{array}$$

   Como la mantisa de la respuesta está desnormalizada, el ordenador automáticamente la normaliza y el resultado se almacena como $0{,}1000 \times 10^{-3}$.
   Como puede observar no se ha generado ningún error, pero si observa bien, el resultado antes de ser normalizado solo cuenta con 1 cifra significativa, por lo cual no se puede confiar plenamente en su exactitud, ya que un pequeño error en alguno de los números originales produciría un error relativo muy grande en la respuesta de un problema.
   Para demostrar esta afirmación suponga que la operación realizada forma parte de la siguiente expresión

$$X = (A - B).C$$

   Donde $A = 0{,}2145 \times 10^{0}$, $B = 0{,}2144 \times 10^{0}$ y $C = 0{,}1000 \times 10^{5}$.

   Entonces $X = (0{,}0001 \times 10^{0})(0{,}1000 \times 10^{5}) = 0{,}00001 \times 10^{-5}$

   Este terminó se normaliza a $X = 0{,}0001 \times 10^{-4} = 1$

   Que es el resultado correcto.
   Ahora suponga que se hubiera obtenido $A = 0{,}2146 \times 10^{0}$ con un error relativo porcentual de 0,046%, un valor muy bueno.

   Realizando la misma operatoria tenemos

$$\begin{array}{rr} - & 0{,}2146 \times 10^{0} \\ & 0{,}2144 \times 10^{0} \\ \hline & 0{,}0002 \times 10^{0} \end{array}$$

Como la mantisa de la respuesta está desnormalizada, el ordenador automáticamente la normaliza y el resultado se almacena como $0{,}2000 \times 10^{-3}$.

Entonces $X = (0{,}0002 \times 10^{0})(0{,}1000 \times 10^{5}) = 0{,}00002 \times 10^{-5}$

Este terminó se normaliza a $X = 0{,}0002 \times 10^{-4} = 2$

Que es un resultado 100% incorrecto, pero provocado por un error relativo muy pequeño. Observe como el error que provoca la resta de números muy parecidos puede llegar a pasar desapercibido.

3) Overflow y Underflow. Con frecuencia una operación entre dos números válidos puede generar un resultado tan grande o pequeño que el ordenador no puede representarlo.
Ejemplo overflow en multiplicación: sea que tiene

$$\begin{array}{rr} x & 0{,}5000 \times 10^{8} \\ & 0{,}2000 \times 10^{9} \\ \hline & 0{,}1000 \times 10^{17} \end{array}$$

En este caso no puede almacenarse el valor de la característica porque se requiere 3 dígitos.
Ejemplo de overflow en división: sea que se tiene

$$\frac{2000000}{0{,}000005} = \frac{0{,}2000 \times 10^{7}}{0{,}000005 \times 10^{-5}} = 0{,}4000 \times 10^{12}$$

Como se mencionó cuando se compartía el Estándar IEEE 574, los ordenadores reportan este tipo de error mediante un mensaje (inf).

El underflow puede aparecer en multiplicación o división, pero no es tan grave como el overflow.
Ejemplo de underflow en multiplicación: sea que tiene

$$(0{,}3000 \times 10^{-5})(0{,}0200 \times 10^{-3}) = 0{,}006 \times 10^{-8} = 0{,}6000 \times 10^{-10}$$

Como el exponente se ha excedido en un dígito no puede representarse en el ordenador, como consecuencia este valor se expresa como 0 (¿recuerda que Scilab arrojó este resultado en uno de los ejemplos iniciales?). Dicho valor posee un error relativo muy pequeño, por lo que generalmente no tiene real relevancia. No obstante, en algunas situaciones puede provocar errores indeseables, suponga que el cálculo anterior forma parte de $X = A.B.C$,donde el resultado obtenido anteriormente representa la operación entre A y B. Entonces independientemente del valor que tenga C, el resultado será 0. Si ud reordena la expresión como $X = A.C.B$ entonces puede ser que evite el underflow. Por ejemplo, si $C = 0{,}4000 \times 10^{7}$, entonces

$$(0{,}3000 \times 10^{-5})(0{,}4000 \times 10^{7}) = 0{,}1200 \times 10^{2}$$

Puede comprobar que el resultado final procediendo de esta manera es

$$X = 0{,}2400 \times 10^{-3}$$

Que es el resultado correcto, por lo que hay que tener cuidado en el diseño de las operaciones.

4) División por un número muy pequeño. Dados

$$A = 0{,}1120 \times 10^9 = 112000000$$
$$B = 0{,}1000 \times 10^6 = 100000$$
$$C = 0{,}900 \times 10^{-3} = 0{,}0009$$

Realice la siguiente operación: $X = A - B/C$

El resultado de $\frac{B}{C} = 0{,}1111 \times 10^9$ y el resultado de $X = 0{,}00009 \times 10^9$. Observe que este resultado de debe ser normalizado, por lo tanto, quedará $X = 0{,}9000 \times 10^6$. Si se analiza este resultado, que es correcto, podemos observar que solo posee una cifra significativa.

Esto hace suponer que es un resultado que es muy sensible a errores previos.

Por tal motivo, estudiaremos que resultado hubiéramos obtenido si en C la aproximación fuera afectada por un error relativo bajo, digamos del 0,01% lo que provoca que $C = 0{,}9001 \times 10^{-3}$.

En esta situación se obtiene $\frac{B}{C} = 0{,}1110 \times 10^9$ y $X = 0{,}1000 \times 10^7$

El error relativo porcentual cometido por la segunda aproximación es

$$|\varepsilon|100\% = \frac{|900000 - 1000000|}{|900000|}100\% = 11\%$$

El error relativo en C se multiplicado (propagado) cerca 1100 veces al resultado final. Este tipo de situaciones pueden provocar resultados carentes de significado o que no tengan relación con la respuesta verdadera, por lo cual deben evitarse.

5) Error de discretización. Cuando un número no se puede representar de manera exacta en base 2 aplicando la técnica de punto flotante, se comete el denominado error de discretización o cuantificación. El nombre se debe a que los números máquina conforman conjuntos discretos en lugar de continuos.

Para ejemplificar esto, usaremos Scilab para realizar diversas operaciones con la sumatoria $\sum_{i=1}^{10000} 0{,}0001$, el cual debería devolver 1.

A continuación, se muestra la función desarrollada en Scilab que representa esta sumatoria

```
function [x]=sumatoria()
    x=0
    for i=1:10000
        x=x+0.0001
    end
endfunction
```

Si ejecutamos la función, obtendremos lo siguiente

```
--> sumatoria()
 ans  =

    1.
```

Lo cual es correcto. Sin embargo, se debe comentar que Scilab está configurado por defecto para que muestre 10 dígitos. Suponga que deseamos visualizar más dígitos. El resultado no debería afectarse de ninguna manera, porque el resultado exacto es 1.

Con la función format() establecemos el valor en 11 dígitos y observamos el resultado obtenido:

```
--> format(11)
--> sumatoria()
 ans  =

    1.
```

Seguimos obteniendo el mismo resultado, podríamos parar aquí, pero por curiosidad veremos que pasa si aumentamos en un dígito más la visualización de números, varias veces. Observe lo que sucede para 16 dígitos

```
--> format(16)

--> sumatoria
 ans  =

   0.9999999999999
```

¿Qué ha sucedido? Se puede deducir que, para una cantidad de dígitos menor, se estuvo redondeando el resultado. La palabra presenta una limitación para representar una cadena de bits.

Fuentes consultadas

- ✓ https://www.ecured.cu/Matem%C3%A1tica_num%C3%A9rica
- ✓ Chapra, S. C.(2007). Métodos Numéricos para ingenieros, 5ta Edición. McGRAW-HILL. México.
- ✓ Hernandez, S. (2018). Apuntes de Cálculo Numérico con aplicaciones sobre Euler Math Toolbox. Ediciones UNPaedita Universidad Nacional de la Patagonia Austral. Argentina
- ✓ Vigueras Campuzano, A. (2016). Cálculo Numérico. Teoría, problemas y algunos programas con Máxima. Crai UPCT Ediciones Universidad Politécnica de Cartagena.
- ✓ What every computer scientist should know about floating-point arithmetic, publicado en el año 1991 recuperado de: https://doi.org/10.1145/103162.103163