Cuando tu ordenador te engaña

Fernando Chamizo

26 de abril de 2012

Resumen y objetivos

La sorprendente precisión y rapidez con la que trabajan los ordenadores personales de hoy en día nos hacen olvidar que el análisis trata fundamentalmente con cantidades arbitrariamente grandes o pequeñas que escapan de sus capacidades. Ilustraremos con varios ejemplos que en algunas ocasiones los resultados computacionales deben ser considerados críticamente. Se intenta que el contenido de esta sesión sea de utilidad directa en la práctica docente. Los programas y ejemplos están realizados con el software matemático Sage que es gratuito y de código abierto y se puede emplear en el aula, incluso sin ninguna instalación a través de internet o por medio de versiones *live*.

En `http://www.uam.es/fernando.chamizo/auditorio.html` habrá una copia revisada de estas notas y de la presentación asociada.

Índice

1. Introducción **2**

2. Algunos ejemplos extraños **3**
2.1. Gráficas caprichosas 3
2.2. Racionalización irracional 4
2.3. El "chiste" de reduzca 6
2.4. Circunferencias no circulares 7

3. Explicaciones y más ejemplos **10**
3.1. El ϵ-máquina y sumas no reordenables 10
3.2. Más cuentas, menos precisión 12
3.3. ¿Cómo se inventaron los logaritmos neperianos? 13
3.4. Equilibrio inestable 15

4. Los ordenadores y los humanos se equivocan **16**
4.1. Ordenadores deficientes 16
4.2. Errores o engaños para educar 17
4.3. La cuadriculación del círculo 17

5. Apéndice **18**

1. Introducción

Los ordenadores modernos son digitales, funcionan con ceros y unos y sólo permiten representar un número limitado de cantidades discretas. Por otro lado el análisis trabaja intrínsecamente con pasos al límite, cantidades que se aproximan indefinidamente a otras, o que se hacen arbitrariamente grandes o arbitrariamente pequeñas.

Tras estas descripciones radicalmente opuestas, cabe preguntarse si es posible ayudarse del ordenador dentro del análisis. La respuesta es un sí rotundo en un mundo tecnológico en el que la ingeniería, basada en gran medida en modelos continuos regidos por ecuaciones diferenciales, depende sin paliativos de los ordenadores.

En términos generales, el ordenador nos dará lo que queremos y será un instrumento fiable para explicar análisis en el aula, o para comprobar o proponer ejercicios, siempre que tomemos ciertas precauciones consistentes esencialmente en no llevar al ordenador más allá de sus límites. Por poner una analogía, en cierta manera inversa a la situación descrita, aunque sepamos que existen los átomos, a gran escala podemos considerar la materia como algo continuo y regido por las leyes de la mecánica clásica. Cuando reducimos la escala de medida hay cosas que no concuerdan, por ejemplo las partículas de polen en un líquido parecen tener vida propia y desafían a la mecánica de fluidos moviéndose espasmódicamente (movimiento browniano). Bajando la escala varios órdenes de magnitud hasta niveles subatómicos, la física se vuelve muy extraña y todavía hoy no se entiende por completo. Lo más curioso es que en ocasiones los átomos nos explican fenómenos macroscópicos, como procesos químicos, por ello se introdujeron como hipótesis antes de que nadie tuviera evidencia mínimamente directa de su existencia. De la misma forma, a veces la estructura directa del ordenador se manifiesta en cálculos a gran escala con efectos inesperados.

En lo que respecta a la educación, el uso del ordenador en el aula no requiere que el profesor adquiera grandes conocimientos de informática. Basta por ejemplo en la enseñanza del cálculo, una pequeña "actualización" porque cuando se inventaron las derivadas e integrales no había ordenadores.

Las nuevas tecnologías son la puerta a un nuevo mundo de posibilidades en la docencia de las matemáticas. Sin embargo, hay que reconocer que ha creado también un problema colateral, bien distinto de que el ordenador no sea fiable en situaciones límite, y es que en

diversos ámbitos educativos se ha identificado lo tecnológico con lo bueno. ¿Merece la pena que redactemos nuestras clases en PowerPoint usando el editor de ecuaciones para escribir las fórmulas? No deja de ser curioso que las mismas personas que acusan en los medios a los docentes de desconocer las nuevas tecnologías, se distingan por una formación y ocupación bien lejanas de los saberes técnicos. Por otro lado, parece que las personas con un contacto continuo por ejemplo con la informática o las telecomunicaciones son más proclives, fuera del ámbito comercial, a manifestar sus limitaciones.

2. Algunos ejemplos extraños

2.1. Gráficas caprichosas

El concepto más representativo del cálculo infinitesimal es la derivada. La definición es bien conocida y representa la velocidad instantánea: se calcula el incremento del espacio (de la función) en relación con el incremento del tiempo (la variable) cuando éste es indefinidamente pequeño.

$$f'(x) = \lim_{h\to 0} \frac{f(x+h) - f(x)}{h}.$$

La idea de una cantidad que tiende a cero requirió muchos años de maduración en la historia de las matemáticas y todavía causa cierta perplejidad hasta que nos acostumbramos a él. Es batalla perdida (fuera de manipulaciones simbólicas) implementarlo en nuestro ordenador, debido a su arquitectura interna. La derivada como velocidad instantánea es una idealización matemática, todo lo que miden los instrumentos más precisos son velocidades medias en intervalos muy pequeños. Así pues para funciones "normales"

$$f'(x) \approx \frac{f(x+h) - f(x)}{h} \qquad \text{cuando } h \neq 0 \text{ es pequeño.}$$

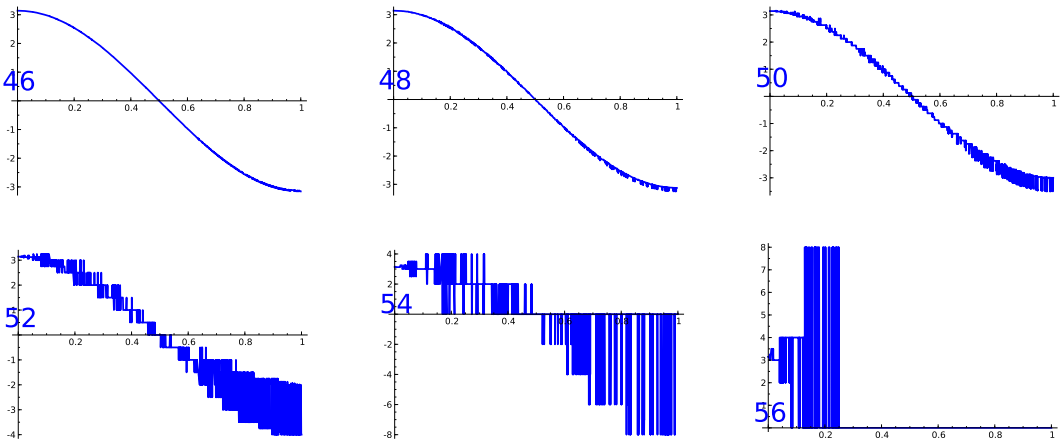
Además, cuanto más pequeño sea h, mejor será la aproximación.

Consideremos por ejemplo $f(x) = \operatorname{sen}(\pi x)$. Es una función sencilla, media oscilación en $[0, 1]$, con derivada acotada y la buena aproximación de ésta por el cociente incremental funciona sin sorpresas. Por ejemplo

$$f'(1/6) = \pi\frac{\sqrt{3}}{2} = 2{,}720699\ldots \qquad \frac{f(1/6 + 10^{-4}) - f(1/6)}{10^{-4}} = 2{,}720452\ldots$$

En consonancia con esto, si dibujamos la gráfica de $g(x) = \big(f(x+h) - f(x)\big)/h$ tiene forma de coseno, incluso tomando valores de h no demasiado pequeños. Sin embargo utilizando Sage cerca de $2^{-50} \approx 10^{-15}$ empiezan a ocurrir cosas muy raras (lo mismo ocurriría con otro software posiblemente a escalas comparables). Ciertamente son valores muy pequeños pero si dejamos a la clase que experimente, es plausible que algún alumno llegue a ellos y pregunte por este fenómeno.

En las siguientes gráficas $h = 2^{-N}$ con N el número que aparece destacado.



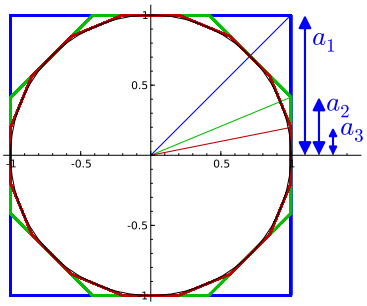
Algo malo ocurre en este rango y además parece ser un hecho universal. Probando con otras funciones "normales", incluso polinomios como $f(x) = x^2$, vemos figuras similares.

2.2. Racionalización irracional

Un hipotético alumno aventajado nos podría preguntar de dónde sale π o, en términos más prácticos, cómo calcular π de alguna forma algorítmica. Por cierto, recurrir a circunferencias muy grandes hechas de material flexible que se pueda desenrollar para comparar radio y longitud no es buena idea si lo que nos preocupa es la precisión (y practicar con el ordenador).

Esta pregunta es una buena ocasión para contar el método de aproximación de Arquímedes, o más bien la mitad, porque nos vamos a olvidar de los polígonos inscritos.

La mitad del lado de un cuadrado circunscrito en una circunferencia unidad mide claramente $a_1 = 1$ y el perímetro del cuadrado es $8a_1 = 8$. En un octógono regular circunscrito la longitud de la mitad del lado es $a_2 = \sqrt{2} - 1$ y por tanto su perímetro será $16a_2 =$. En general, si llamamos a_n a la longitud de la mitad del lado del polígono regular circunscrito de 2^{n+1} lados (dividir por dos cada vez) entonces el perímetro es $2 \cdot 2^{n+1}a_n$. En el límite se debe obtener la longitud de la circunferencia, que es 2π. Por consiguiente, dividiendo entre dos, tenemos una sucesión que tiende a π y un dibujo nos sugiere que lo hace con mucha rapidez.



a_n = mitad del lado del polígono de 2^{n+1} lados

$$\boxed{2^{n+1}a_n \to \pi}$$

Para que este método sea útil, deberíamos buscar algún algoritmo para calcular a_n. Simple trigonometría nos dice que $a_n = \tan\left(\pi/2^{n+1}\right)$ y ahora, escribiendo $\alpha = \pi/2^n$, tenemos mediante trigonometría un poco más difícil pero no demasiado:

$$a_n = \tan\frac{\alpha}{2} = \sqrt{\frac{1-\cos\alpha}{1+\cos\alpha}} = \frac{1-\cos\alpha}{\operatorname{sen}\alpha} = \frac{\frac{1}{\cos\alpha}-1}{\frac{\operatorname{sen}\alpha}{\cos\alpha}} = \frac{\sqrt{1+\tan^2\alpha}-1}{\tan\alpha} = \frac{\sqrt{1+a_{n-1}^2}-1}{a_{n-1}}.$$

Esta fórmula permite calcular a_n de manera recurrente. A partir de $a_1 = 1$ obtenemos $a_2 = \sqrt{2}-1 = 0{,}414213$ y a partir de a_2 se obtiene $a_3 = 0{,}198912$ y así sucesivamente.

Escribamos dos variantes del algoritmo, siempre partiendo de $a_1 = 1$:

$$a_n = \frac{\sqrt{1+a_{n-1}^2}-1}{a_{n-1}} \qquad \text{y} \qquad a_n = \frac{a_{n-1}}{\sqrt{1+a_{n-1}^2}+1}.$$

Ambas fórmulas son idénticas racionalizando. Sin embargo nos empeñamos en distinguirlas y llamamos b_n a los resultados de la segunda fórmula (a sabiendas de que son iguales). Haciendo un pequeño programa vemos que en el quinto término aparece una pequeña divergencia prácticamente inapreciable.

n	$2^{n+1}a_n$	$2^{n+1}b_n$
1	4.00000000000000	4.00000000000000
2	3.31370849898476	3.31370849898476
3	3.18259787807453	3.18259787807453
4	3.15172490742926	3.15172490742926
5	3.14411838524587	3.14411838524590
6	3.14222362994234	3.14222362994246
7	3.14175036916970	3.14175036916897

El problema es cómo va evolucionando.

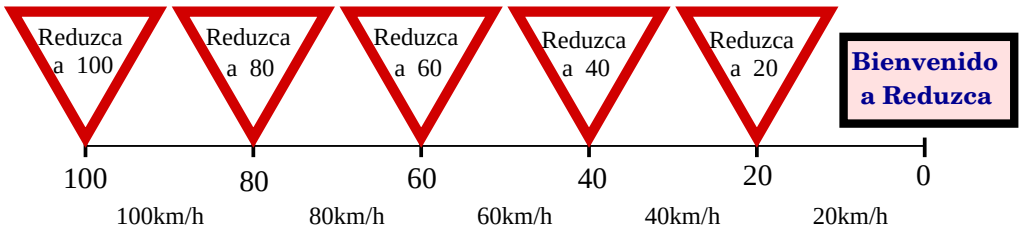
n	$2^{n+1}a_n$	$2^{n+1}b_n$
18	3.14159252378847	3.14159265362739
19	3.14160058362634	3.14159265359919
20	3.14159252378847	3.14159265359214
21	3.14144515887080	3.14159265359038
22	3.14097079560249	3.14159265358994
23	3.13398329388536	3.14159265358983
24	3.11105678802532	3.14159265358980
25	3.05362474788830	3.14159265358980
26	2.61983729517922	3.14159265358979

Ni siquiera podemos continuar mucho más allá. Con Sage (con otro software pasaría algo similar) el ordenador se niega a calcular a_{29} porque dice que no puede dividir por cero. Sin embargo b_n funciona bien y a partir de a_{26} sigue dando π con las 14 cifras decimales correctas que muestra, sin que se vea ningún fallo según avanza n.

Si dos expresiones son algebraicamente iguales, ¿por qué el ordenador las trata de forma tan diferente? ¿Por qué la versión racionalizada es peor?

2.3. El "chiste" de reduzca

En breve, este "chiste" consiste en un conductor que va por la carretera y ve un cartel que avisa "Reduzca a 100", después otro que dice "Reduzca a 80" y así sucesivamente, hasta que llega a un gran cartel que indica "Bienvenido a Reduzca".



¿Cuánto tiempo tardaría este coche en llegar a Reduzca? Es un buen (y sencillo) ejercicio para nuestros alumnos. Hay 5 tramos de 20km. El primero se recorre a 100km/h, el segundo a 80km/h, hasta el último que se recorre a 20km/h. Utilizando que $\Delta t = \Delta e/v$, el tiempo total es

$$\frac{20}{100} + \frac{20}{80} + \frac{20}{60} + \frac{20}{40} + \frac{20}{20} = \frac{137}{60} = 2\text{h } 17\text{m}.$$

Si ponemos más señales, claramente se tardará más tiempo en llegar, pero ¿está el tiempo de llegada acotado? La pregunta no es trivial. Una cantidad que crece puede estar acotada, como sucede con $(n-1)/n$, y por otra parte, aunque la velocidad final sea menor cuantas más señales haya, es efectiva en una distancia menor. Pensemos por ejemplo que una partícula con ecuación de movimiento $x(t) = (t-1)^3 + 1$ se mueve en una unidad de tiempo de 0 a 1 aunque su velocidad decrece y termina siendo $x'(1) = 0$.

Para ser más concretos, digamos que preguntamos a los alumnos si pueden hallar un número N de señales igualmente espaciadas de forma que en vez de tardar más de dos horas, tarde más de dos días. Siendo realista, el problema es muy difícil incluso para los mejores alumnos pero podríamos guiarlos para que al menos experimenten con un ordenador.

Ahora hay N tramos de $100/N$km. El primero se recorre a 100km/h, el segundo a $(100 - 100/N)$km/h, el tercero a $(100 - 2\cdot 100/N)$km/h; hasta el último que se recorre a $\big(100 - (N-1)100/N\big)$km/h, esto es, $100/N$km/h. Dos días son 48h, por tanto buscamos con el ordenador

valores de N tales que

$$\frac{100/N}{100} + \frac{100/N}{100 - 100/N} + \frac{100/N}{100 - 200/N} + \dots \frac{100/N}{100/N} > 48.$$

Una primera conjetura es que si con $N = 5$ señales se tardan más de 2h, para que tarde 24 veces esta cantidad la desigualdad quizá requiera $N = 5 \cdot 24 = 120$. Sin embargo un pequeño programa prueba que la suma del primer miembro es sólo 5,368868..., ni siquiera se ha triplicado.

Una tabla correspondiente a N del orden de decenas de miles muestra que el incremento se hace cada vez menor y sugiere que quizá el tiempo converja a algún valor que no alcanza a 48.

N	Suma	Δ
100000	12.090146	.....
200000	12.783290	0.693144
300000	13.188755	0.405464
400000	13.476436	0.287681
500000	13.699580	0.223143

N	Suma	Δ
600000	13.881901	0.182321
700000	14.036051	0.154150
800000	14.169583	0.133531
900000	14.287366	0.117782
1000000	14.392726	0.105360

2.4. Circunferencias no circulares

Les contamos a nuestros alumnos que las derivadas y las integrales sirven por ejemplo para determinar los movimientos de los planetas. Con $F = ma$, se escriben las ecuaciones que los regulan en términos de posiciones y sus derivadas segundas (como funciones del tiempo) pero introduciendo el momento lineal todo se traduce en relaciones entre funciones y sus derivadas primeras.

¿Cómo resolver numéricamente estas ecuaciones? Olvidémonos de las ecuaciones específicas de la gravitación, que son complicadas, y tomemos como base para experimentar el movimiento de una partícula $\big(x(t), y(t)\big)$ regulado por unas relaciones muy sencillas:

$$\begin{cases} x' = -y \\ y' = x \end{cases}$$

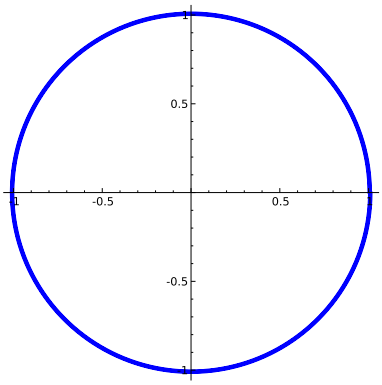
Si partimos por ejemplo del punto $\big(x(0), y(0)\big) = (1, 0)$ es fácil comprobar que $x(t) = \cos t$, $y(t) = \operatorname{sen} t$ es solución (de hecho es la única).

Computacionalmente usamos $\big(x(t+h) - x(t)\big)/h \approx x'(t)$, $\big(y(t+h) - y(t)\big)/h \approx y'(t)$, como en un ejemplo anterior, para concluir que si h es pequeño tenemos con gran aproximación

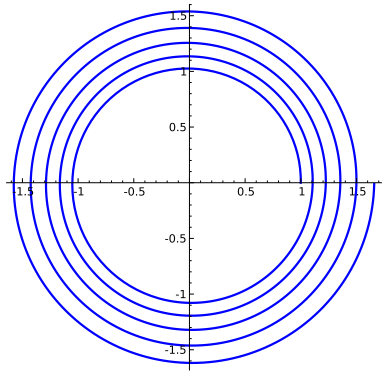
$$\begin{cases} x(t+h) = x(t) - hy(t) \\ y(t+h) = y(t) + hx(t) \end{cases}$$

Entonces a partir de $\big(x(0), y(0)\big)$ podemos aproximar $\big(x(h), y(h)\big)$ y después $\big(x(2h), y(2h)\big)$ y así sucesivamente. Uniendo los puntos, tendremos una aproximación de la circunferencia $t \mapsto (\cos t, \operatorname{sen} t)$. Este método, llamado *método de Euler* funciona razonablemente bien para t pequeño.

Es de prever (por Taylor) que en cada punto se cometa un error de orden h^2. Llegar de 0 hasta T requiere T/h iteraciones y por tanto el error total es como hT. Si este número es pequeño, todo debería ir bien, mientras que si no lo es, digamos $hT \approx 1$, entonces la circunferencia se deforma notablemente. Los experimentos con el ordenador avalan esta idea:



$h = 0{,}001,\ t \in [0, 10\pi]$



$h = 1/31,\ t \in [0, 10\pi]$

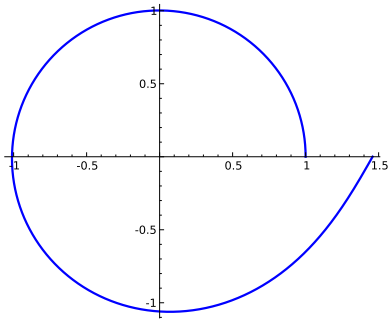
La regla aproximada es entonces que para ver una órbita completa (la circunferencia unidad), $T = 2\pi$ lleva a que un incremento h que sea bastante menor que una décima, es suficiente.

La sorpresa aparece si complicamos un poco la ecuación de cierta manera. Por ejemplo

$$\begin{cases} x' = -y + \frac{1}{2}x(x^2 + y^2 - 1) \\ y' = x + \frac{1}{2}y(x^2 + y^2 - 1) \end{cases}$$

Todavía la solución bajo $\big(x(0), y(0)\big) = (1, 0)$ es $x(t) = \cos t$, $y(t) = \operatorname{sen} t$, porque los términos que hemos añadido se anulan para esta solución.

En principio no hay nada malo y el análisis anterior acerca del error debería funcionar. Así pues, para $T = 2\pi = 6{,}28\ldots$ un incremento $h = 10^{-3}$ debería ser más que suficiente. Sin embargo unos experimentos nos hacen ver cuán equivocados estamos. No podemos ver la trayectoria circular de nuestro planeta o partícula usando el ordenador. Además los cálculos muestran una sensibilidad increíble. La segunda figura es lo que se obtendría reemplazando 1/2 por 0,5599, mientras que con 0,57 ya Sage se quejaría diciendo que salen números demasiado grandes.



$h = 0{,}001,\ t \in [0, 2\pi]$

$h = 0{,}001,\ t \in [0, 2\pi]$

3. Explicaciones y más ejemplos

3.1. El ϵ-máquina y sumas no reordenables

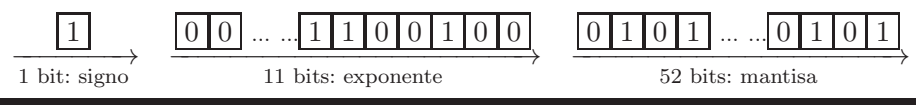
Un primer intento de explicar las gráficas caprichosas es que con el software empleado, no "caben" en el ordenador números algo menores que 2^{-50}. Esto no es cierto. De hecho en cualquier calculadora de bolsillo podemos representar sin problemas números tan pequeños como 10^{-99} y en el software científico básico típicamente hasta 10^{-308}.

La explicación pasa por el concepto de "ϵ-máquina". Esencialmente es el máximo error relativo que permanece inadvertido para un ordenador. En particular, es la mayor cantidad $\epsilon > 0$ para la cual $1 + \epsilon = 1$ donde $+$ aquí significa la suma que realmente hace el ordenador. De la misma forma se cumplirá $0{,}002 + 0{,}002\epsilon = 0{,}002$ pero $0{,}002 + \epsilon \neq 0{,}002$. Por ejemplo $\epsilon = 10^{-3}$ daría $0{,}00001 + 0{,}000002 = 0{,}000012$ porque la posición de la coma en ambos números sólo difiere en un lugar, pero $0{,}1 + 0{,}000002 = 0{,}1$ porque la coma está desplazada más de tres lugares. El ϵ-máquina no tiene nada que ver con el número más pequeño representable.

En una calculadora de bolsillo se puede tener $\epsilon \approx 10^{-12}$ pero a veces es difícil detectarlo con la definición anterior porque muchas calculadoras piensan (con razón) que los números enteros son más habituales en nuestras cuentas que los decimales y redondean lo que nos muestran en pantalla aunque internamente tenga más precisión. Con un ordenador la situación suele ser más transparente porque en los lenguajes de programación suelen separarse los enteros de los no enteros mediante tipos. Una gran parte del software científico está basado en C/C++, de hecho el propio sistema operativo (las tripas de nuestro ordenador en términos de software) dependen de este lenguaje (en cualquiera de los sistemas operativos habituales). Según las especificaciones normativas, en doble precisión el ϵ-máquina en C/C++ es $2^{-53} \approx 10^{-16}$. Hay una razón técnica para ello y es que al nivel del procesador (las tripas de nuestro ordenador en términos de hardware) es natural trabajar con 64 bits y lo que se hace es reservar 52 de ellos para la mantisa (número sin exponente) 11 de ellos para el exponente con signo y 1 para el signo del número.

¿Cómo se almacena internamente $-1{,}3521606 \cdot 10^{30} = -\frac{2^{104}}{15} = -\frac{16}{15}2^{100}$**?**

Signo: $- \to 1$
Exponente: $100 \to 1100100$ en binario
Mantisa: $16/15 \to 1{,}010101\ldots$ en binario



Los 52 bits ocupados por la mantisa hacen que un número sea invisible al sumar si desplazamos la "coma binaria" 53 bits. Por tanto $\epsilon = 2^{-53}$.

Una consecuencia poco agradable de la existencia del ϵ-máquina es que las sumas aparentemente ¡dejan de ser conmutativas! Estrictamente, lo que sucede es que no son asociativas pero el efecto es más llamativo cuando se combinan las propiedades conmutativa y asociativa.

Pensemos por ejemplo en una calculadora con $\epsilon = 5 \cdot 10^{-4}$, entonces, según muestran las siguientes tablas, 2.01+0.00003+0.00004+0.00004 $\neq$ 0.00003+0.00004+0.00004+2.01

	2.01
+	0.00003
=	2.01
+	0.00004
=	2.01
+	0.00004
=	$\boxed{2.01}$

	0.00003
+	0.00004
=	0.00007
+	0.00004
=	0.00011
+	2.01
=	$\boxed{2.0101}$

El fenómeno se acentúa cuando el número de sumandos crece, lo cual ocurre muchas veces cuando programamos. Por ejemplo, se conoce la bella fórmula

$$\pi^4 = \sum_{m=1}^{\infty} \frac{90}{m^4}.$$

Supongamos que deseamos utilizar la suma de muchos términos de la serie, digamos 100000, para conseguir una aproximación de π^4. Lo natural es hacer la suma "hacia adelante", es decir, empezando en 1 y acabando en 100000. Consideramos también la suma "hacia atrás" comenzando a sumar a partir del término 100000.

El resultado es que con el orden natural, se comete un error de $1.93 \cdot 10^{-11}$ mientras que el orden inverso, es $4.26 \cdot 10^{-14}$, esto es, 400 veces menor. La explicación es que, como han ilustrado las tablas anteriores, los errores son mayores cuando los sumandos son de órdenes de magnitud bien distintos. Al sumar en sentido inverso, tratamos de evitar en parte esta situación.

En la suma en el orden habitual partimos de 90 y hacemos sumas cometiendo un error relativo típico de $\epsilon = 2^{-53}$, que corresponde a un error absoluto de $90 \cdot 2^{-53} \approx 10^{-14}$. Si todos estos errores se acumulasen, se tendría un error total de 10^{-9}. ¿Por qué, incluso en este caso malo, el error es demasiado bueno? Porque los errores aleatorios e independientes no se suman realmente. Imaginenos que tiramos una moneda y sumamos 1 si sale cara y -1 si sale cruz. Algunos resultados profundos de probabilidad (teorema central del límite) implican que lo que cabe esperar es que al cabo de N tiradas tengamos una cantidad comparable a $\sqrt{N}$. El ordenador, redondea en las operaciones unas veces hacia arriba y otras hacia abajo y si los sumandos son siempre comparables, el error relativo es como el ϵ-máquina por la raíz del número de sumas. En el primer caso debería ser por tanto comparable a $10^{-14}\sqrt{100000}$ (y lo es con un factor 6).

3.2. Más cuentas, menos precisión

La existencia del ϵ-máquina hace que restar números muy parecidos sea desastroso.

Recordemos que típicamente ϵ es alrededor de 10^{-16} en un ordenador y que esta cantidad es muchos órdenes de magnitud mayor que los números más pequeños que puede representar. Así el ordenador no tendrá ningún problema en calcular correctamente ϵ/ϵ como 1. Sin embargo, el resultado de $\big((1+\epsilon)-1\big)/\epsilon$ será 0 porque la máquina no puede distinguir $1+\epsilon$ de 1. De la misma forma, $(1+\epsilon)/\epsilon - 1/\epsilon$ será cero para el ordenador.

La situación es peor de lo que parece, de hecho la cantidad $(1+\delta)/\delta - 1/\delta$ estará bastante indeterminada si δ es comparable a ϵ. Por ejemplo, si $\delta \approx 2\epsilon$, entonces $1+\delta$ y $1+\delta \pm \delta/2$ serán esencialmente lo mismo para el ordenador y la resta bien podría ser $3/2$ o $1/2$ para el ordenador.

¿Cómo se aplica todo esto a nuestro ejemplo que no se dejaba racionalizar? Recordemos que en nuestra construcción, $a_n = \tan(\pi/2^{n+1})$ entonces según crece n, la sucesión converge rápidamente (exponencialmente) a cero. Con el primer algoritmo tenemos que

$$2^{n+1}a_n = \frac{\sqrt{1+a_{n-1}^2}-1}{2^{-n-1}a_{n-1}} \approx \frac{\big(1+\frac{1}{2}a_{n-1}^2\big)-1}{2^{-n-1}a_{n-1}}$$

porque para x pequeño $\sqrt{1+x} \approx 1 + x/2$ (Taylor o teorema del valor medio) y estamos esencialmente en la misma situación descrita anteriormente con $\delta = a_{n-1}^2/2$. En cuanto a_{n-1} empiece a acercarse a $\sqrt{\epsilon}$ comenzarán los problemas. Necesariamente, cuando $a_{n-1} \approx 10^{-8}$, lo cual es un número bastante grande en comparación con ϵ, los resultados serán bastante indeterminados.

Una tabla con los últimos cinco valores muestra este fenómeno:

n	$2^{n+1}a_n$	a_{n-1}	$a_{n-1}^2/2$
22	3.14097079560249	7.48978891103459e-7	2.80484689659283e-13
23	3.13398329388536	3.74432897043525e-7	7.00999971942033e-14
24	3.11105678802532	1.86799960964045e-7	1.74471127080845e-14
25	3.05362474788830	9.27167173631585e-8	4.29819483929991e-15
26	2.61983729517922	4.55025545938059e-8	1.03524123728115e-15

Con el segundo algoritmo, si b_{n-1} es comparable a $\sqrt{\epsilon}$ lo único que ocurre es que el error en $\sqrt{1+b_{n-1}^2}+1$ será 2 con un error comparable a ϵ y con ello b_n tendrá un error relativo comparable a ϵ. La única influencia de ello es que no debemos confiar en cifras que estén más allá de 16 lugares desde la primera cifra significativa de b_n.

Incluso si suponemos que no hay compensación entre diferentes errores relativos, nótese que cometer k veces un error relativo ϵ es como multiplicar esa cantidad por $(1+\epsilon)^k$ y para que esto llegue a duplicar a la cantidad, habría que tomar $k \approx \epsilon^{-1}\log 2$, porque $(1+\epsilon)^{\epsilon^{-1}} \approx e$, lo cual sería un número descomunal de iteraciones.

3.3. ¿Cómo se inventaron los logaritmos neperianos?

¿Alguna vez nos ha preguntado un alumno para qué sirven los logaritmos? Una respuesta parcial pero con base histórica es que sirven para transformar multiplicaciones en sumas, que son más fáciles. La siguiente pregunta es para qué sirven los logaritmos neperianos. Quizá ya nos hemos acostumbrado a ello, pero eso de tomar como base 2,71828... es bien raro. La respuesta está necesariamente relacionada con el análisis: son los logaritmos que tienen una derivada sencilla, $1/x$. Sin embargo aquí la historia nos juega una mala pasada porque el libro de J. Neper (también transcrito Napier) sobre las *maravillas de los logaritmos* es de 1614 mientras que Newton nació en 1642.


MIRIFICI

Logarithmorum

Canonis descriptio,

Ejusque usus, in utraque

Trigonometria; ut etiam in

omni Logistica Mathematica,

Amplissimi, Facillimi, & expeditissimi explicatus.

Authore ac Inventore,

IOANNE NEPERO,

Barone Merchistonii,

&c. Scoti.

EDINBURGI,

Ex officinâ ANDREÆ HART

Bibliopolæ, cIↃ. DC. XIV.

6. Def. Logarithmus ergo cujusque sinus, est numerus quam proxime definiens lineam, quae aequaliter crevit, interea, dum sinus totius, linea proportionaliter in sinum illum decrevit, existente utroque motu synchrono, atque initio æquiveloce.

Por tanto, el Logaritmo de cualquier seno es el número que con aproximación define la línea que se *incrementa igualmente* al mismo tiempo que la línea para el *seno completo decrece proporcionalmente* en dicho seno, siendo ambos movimientos síncronos y con la misma velocidad inicial.

La definición de logaritmo dada por Neper es literalmente bastante oscura porque depende de los conceptos *incrementar igualmente* y *decrecer proporcionalmente*, que tienen un sentido no tan fácil de sospechar (que Neper aclara en definiciones anteriores). Además por necesidades prácticas (tablas astronómicas para la navegación) quería aplicar el logaritmo a las razones trigonométricas, que aparecen en la propia definición, y como iba a trabajar con 7 decimales normalizó las cosas para que el 1 sea un 10^7, éste es el *seno completo* del que habla Neper.

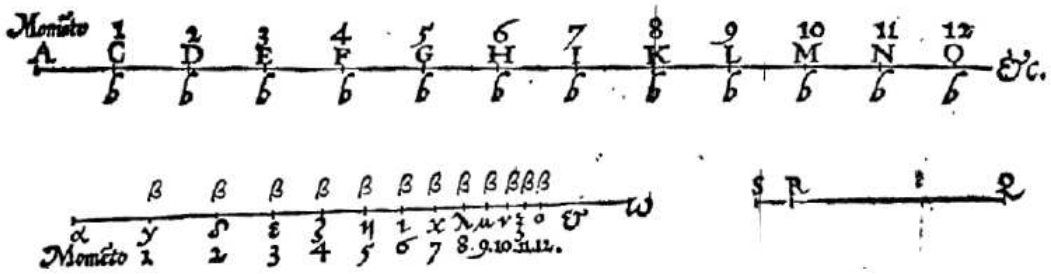
¿Qué significa en lenguaje sencillo la definición de Neper? Pues esencialmente definió los logaritmos neperianos mediante una versión del "chiste" de reduzca con infinitas señales, una en cada punto. De este modo la velocidad es igual a la distancia al destino. Si la distancia inicial a Reduzca es 1 en ciertas unidades (¿*hectokilómetros*?) entonces se prueba que el logaritmo neperiano de la distancia del coche a Reduzca es en valor absoluto igual al tiempo.

Lo que hizo realmente Neper fue suponer que la distancia a Reduzca era 10^7 y evitar el

valor absoluto cambiando el signo, por ello el logaritmo neperiano que definió, digamos $\ln_{\text{orig}}$, es ligeramente diferente al actual, $\ln_{\text{act}}$. Concretamente

$$\ln_{\text{orig}} x = -10^7 \ln_{\text{act}} \frac{x}{10^7}.$$

Lo que Neper llama *incrementarse igualmente* es el paso uniforme del tiempo que representaba en una recta con divisiones equidistribuidas. Por otro lado, llamaba *decrecer proporcionalmente* a seguir la trayectoria del coche, que representaba con divisiones que se van juntando porque las señales van disminuyendo la velocidad. En su tratado hace unos dibujos para explicar estos conceptos que darían las posiciones cuando cronometramos un coche normal y el coche de Reduzca.



Volviendo a nuestro problema, digamos que T es el tiempo que se tarda en llegar a Reduzca cuando hay N señales. Tomemos como unidad los cientos de kilómetros, entonces el último tramo mide $1/N$ y si se tarda T_u en llegar a él, se tiene $T = T_u + 1$ porque el último tramo se recorre en una hora (longitud $1/N$, velocidad $1/N$). Evidentemente cuantas más señales haya más se tardará y, por lo dicho anteriormente con infinitas señales se tardaría $-\ln(1/N) = \ln N$ en llegar al último tramo. Así pues:

$$T = 1 + T_u < 1 + \ln N \qquad \Longrightarrow \qquad N > e^{T-1}.$$

Por tanto si el tiempo debe ser mayor que 48 horas, se tendrá $N > e^{47}$. Para hacerse una idea de este número, si en nuestra suma pudiéramos operar cada par de sumandos en 10^9 segundos (demasiado optimista sin métodos especiales de programación) tendríamos que esperar ¡unos 8180 años! El ordenador no puede ayudarnos.

Como comentario final, nótese que depués de simplificar la desigualdad de nuestro problema es

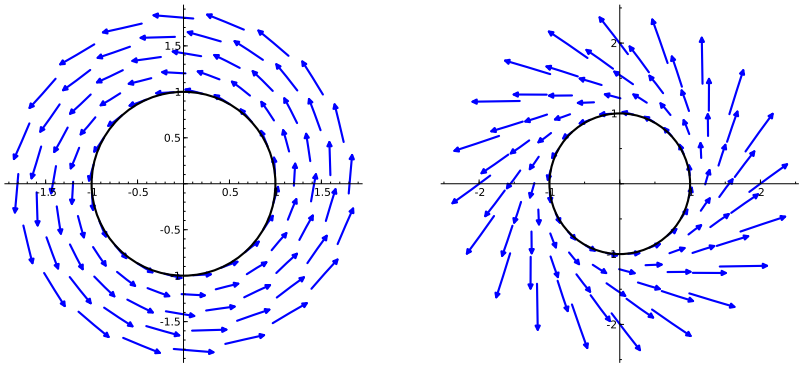
$$\frac{1}{N} + \frac{1}{N-1} + \frac{1}{N-2} + \cdots + \frac{1}{3} + \frac{1}{2} + \frac{1}{1} > 48$$

Una comparación de sumas (superiores) con integrales prueba que el primer miembro es mayor que $\int_1^{N+1} x^{-1} dx = \ln(N+1)$, por tanto cualquier N con $N > e^{48} - 1$ es válido para resolver el problema. Además, de lo visto anteriormente se puede deducir que e^{47} no es suficiente.

3.4. Equilibrio inestable

¿Es posible que un cono invertido se sostenga apoyado únicamente en su vértice? Si dejamos que rote, todos hemos visto una peonza pero en otro caso sabemos positivamente que no. Algo así ocurre con el problema planteado. Es "inestable" en el sentido de que su segunda versión presenta gran sensibilidad con respecto a las condiciones iniciales.

Nuestro planeta o partícula, en cada instante tiene una velocidad dada por el vector $(-y, x)$ con las primeras ecuaciones y por $\left(-y+\frac{1}{2}x(x^2+y^2-1), x+\frac{1}{2}y(x^2+y^2-1)\right)$ con las segundas ecuaciones. Si dibujamos ambos campos de vectores cerca de la circunferencia unidad, veremos que hay una diferencia significativa.



En el primer caso, si un pequeño error hace que saltemos a un punto ligeramente fuera de la circunferencia unidad, entraremos en otro movimiento circular de radio ligeramente diferente. Es por ello que cuando los errores son apreciables vemos espirales. Por otro lado, el segundo dibujo nos indica que en cuanto nos alejamos lo suficiente de la circunferencia unidad las velocidades están en dirigidas de forma que nos alejamos más todavía de ella. Las partículas tienden a escapar una vez que salen de la circunferencia unidad.

Por si este razonamiento cualitativo sabe a poco, hay una forma de tratar todo esto de manera más analítica. Digamos que escribimos nuestras ecuaciones en polares (radio r y ángulo α), es decir con $x(t) = r(t)\cos\alpha(t)$, $y(t) = r(t)\operatorname{sen}\alpha(t)$, entonces

$$\begin{cases} r'\cos\alpha - r\alpha'\operatorname{sen}\alpha = -r\operatorname{sen}\alpha + Kr(r^2-1)\cos\alpha \\ r'\operatorname{sen}\alpha + r\alpha'\cos\alpha = r\cos\alpha + Kr(r^2-1)\operatorname{sen}\alpha \end{cases}$$

donde $K = 0$ para el primer caso y $K = 1/2$ para el segundo. multiplicando la primera ecuación por $\cos\alpha$ y la segunda por $\operatorname{sen}\alpha$ y sumando se llega a una ecuación sencilla para la variación del radio:

$$r' = Kr(r^2-1).$$

Si $K = 0$ entonces tenemos que el radio es constante, es decir, el único movimiento posible es en circunferencias concéntricas y, como hemos dicho antes, los pequeños errores sólo harán que saltemos de una a otra cercana. Si $K = 1/2$, tras unos cálculos rutinarios, se comprueba que

$$r(t) = \frac{1}{\sqrt{1 - \lambda_0 e^t}}$$

verifica la ecuación. Aquí hay que ajustar λ_0 para que en el instante inicial r tome su valor inicial. Si imponemos $r(0) = 1$, como habíamos hecho, entonces $\lambda_0 = 0$ y obtenemos de nuevo la circunferencia unidad $r = 1$. Pero si un pequeño error mueve la trayectoria, y llegamos en $t = t_0$ a un punto con $r(t_0) > 1$ entonces entraremos en una solución con λ_0 pequeño pero positivo, y esa solución es muy mala porque en un tiempo finito, exactamente cuando $t = \ln(1/\lambda_0)$ tiene una singularidad. En otras palabras, esperando un tiempo finito llega al infinito.

4. Los ordenadores y los humanos se equivocan

4.1. Ordenadores deficientes

Los ejemplos anteriores dejan a salvo el honor de los ordenadores y nos induce a pensar que el material informático nunca se equivoca, sino que se usa mal, fuera de sus especificaciones. Todos hemos tenido experiencias bien diferentes en cuanto a los sistemas operativos (ralentizaciones, cuelgues, problemas en el inicio) que muchas veces se excusan apelando a posible software dañino. Sin embargo, los cálculos puros, en el imaginario popular permanecen a salvo de todo error. Esta idea no es correcta del todo.

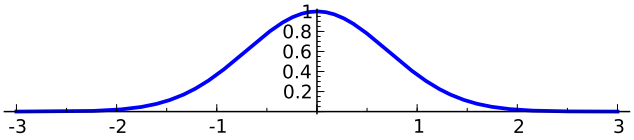
Uno de los ejemplos más llamativos, dentro del software habitual, estaba en Excel 2007, que al calcular el producto de 850 y 77,1 daba como resultado 100000. Había otros ejemplos de multiplicaciones incorrectas en Excel con resultados absurdos, hasta que un parche lo subsanó.

Posiblemente no tan notorio cuantitativamente pero mucho más desastroso para la compañía (no hay parches de hardware), fue el error encontrado alrededor de 1995 en algunos modelos de los microprocesadores Intel Pentium. El problema era que en algunos casos concretos dividía ligeramente mal. Los primeros ejemplos hechos públicos (por un investigador que estaba calculando con mucha precisión los inversos de los primos gemelos), daban errores para ciertas divisiones a partir de la décima cifra decimal. Era muy difícil que un usuario medio notase algo raro pero Intel se vio forzada a reemplazar los microprocesadores de muchos clientes.

En el caso de Sage (y de cualquier otro software) con el uso uno llega a detectar alguna cosa rara. Por ejemplo, al menos en la versión 4.5.1 de este software, si uno trata de dibujar una circunferencia de una manera original dando muchas vueltas a la curva paramétrica $(\cos t, \operatorname{sen} t)$, utilizando el comando para curvas paramétricas `parametric_plot( (cos(x), sin(x)), (0,200))`, el resultado muestra muchas rectas extrañas.

4.2. Errores o engaños para educar

Los humanos, y en particular los profesores, engañamos también a veces en el terreno matemático. Incluso utilizamos a veces inconscientemente el engaño con fines educativos. Por ejemplo, ¿cómo vemos representada en los libros la campana de Gauss $f(x) = e^{-x^2}$ y repetimos a nuestros alumnos? Dibujamos unas asíntotas generosas y un máximo bien pronunciado. Hoy en día, en que el ordenador y algunas calculadoras están al alcance del que desee dibujar una gráfica, cualquier alumno nos puede refutar mostrando con la misma escala en ambos ejes el siguiente dibujo:

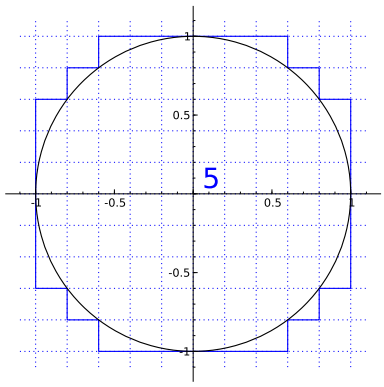


$f(x) = e^{-x^2}$

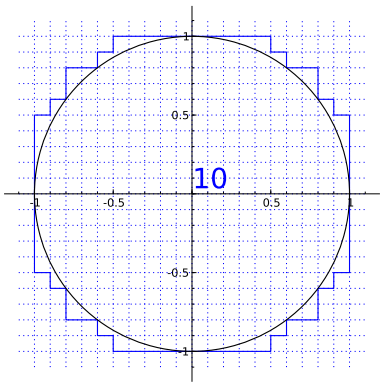
La realidad es que es imposible distinguir a simple vista la gráfica anterior de la de una función que se anula fuera de $[-3, 3]$. Tal hazaña requeriría distinguir una parte entre diez mil. Si cada unidad en los ejes fuera de 1cm, sobradamente tendríamos que ser capaces de ver una bacteria de tamaño medio. Incluso en $x = 2$ ya la separación del eje es menor que un 2 % de la unidad.

4.3. La cuadriculación del círculo

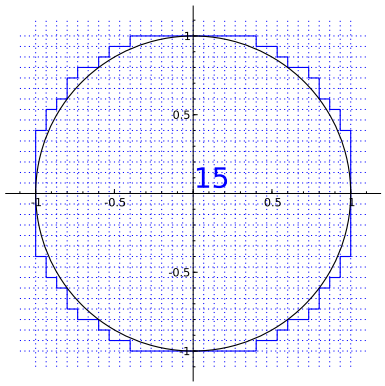
Pintar la aproximación de Arquímedes de la circunferencia unidad por polígonos inscritos y circunscritos, es un poco complicado. Supongamos que nos decidimos por un método más primitivo y aproximamos la longitud de la circunferencia unidad, es decir, 2π, por la longitud de una línea poligonal con segmentos contenidos en $x = nh$, $y = nh$, $n \in \mathbb{Z}$, lo más próxima posible al exterior de la circunferencia.



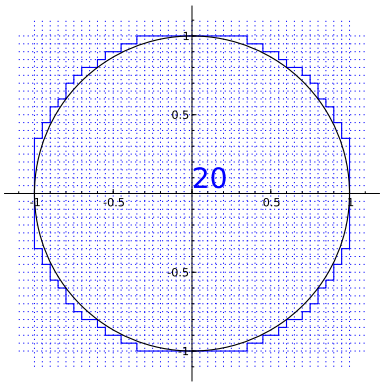
$h = 1/5$



$h = 1/10$



$h = 1/15$



$h = 1/20$

Dicha línea se acerca, por supuesto, indefinidamente a la circunferencia según $h \to 0^+$. Si hiciéramos un pequeño programa, el ordenador nos diría que para $h = 1/100$ la longitud de la línea poligonal es 8, también lo es para $h = 1/1000$ y para una millonésima. ¿Nos está engañando el ordenador? ¿Acaso $2\pi = 8$?

5. Apéndice

La elección de software es, como todo, cuestión de gustos. Si alguien está interesado en utilizar Sage la referencia universal es el sitio oficial:

`http://www.sagemath.org/`

y hay varios manuales aquí:

`http://www.sagemath.org/library.html`

Una vez que uno sabe de qué va la cosa. Hay algunas chuletas en

`http://wiki.sagemath.org/quickref`

Realmente la escrita en español por J.L. Varona, casi no necesita ningún conocimiento previo.

El programa es gratuito y de código abierto. Ocupa bastante en memoria y además está más bien dirigido hacia Linux; por ello antes de decidirse a instalarlo o a cambiar de sistema operativo, es preferible probarlo a través de la red o creando un disco *live*.

Esto último se consigue desplegando el menú `Download` de `http://www.sagemath.org/` y seleccionando `LiveCD`. Elegimos un servidor y descargamos en nuestro ordenador el fichero `.iso` correspondiente. El actual es `SageLIVE-511-47.iso`. Una vez que tenemos el fichero lo grabamos en un CD (no hace falta un DVD). Es muy importante que seleccionemos en

el programa que usemos para grabar la opción “imagen iso” o “imagen”. De otro modo se grabará como un fichero normal y no se ejecutará.

En el CD ahora tenemos el programa y un pequeño Linux. Introducimos el CD en nuestro ordenador y obligamos a que se reinicie desde ahí. Para ello, dependiendo de la configuración de nuestra BIOS, quizá sólo haya que reiniciar o quizá antes de que arranque nuestro sistema operativo habitual debemos pulsar F12 u otra tecla de función y seleccionar que arranque desde el disco.

Después de pasar un menú en el que basta esperar (no tocar opciones que no se entienden) y de esperar un rato más, tendremos Linux funcionando con Sage y, lo más importante, cuando terminemos y reiniciemos el ordenador, seguirá impoluto como antes (si no hemos hecho cosas que difícilmente se harían por descuido). Evidentemente, un sistema operativo que está dentro de un CD funcionará necesariamente ralentizado. Para tener la velocidad de crucero, hay que decidirse a instalar.