Tema 1 - Escalas, condición y estabilidad

Curso de Física Computacional

M. en C. Gustavo Contreras Mayén

22 de agosto de 2012

Contenido

Contenido

Contenido

Contenido

Contenido

Contenido

¿Qué es la física computacional?

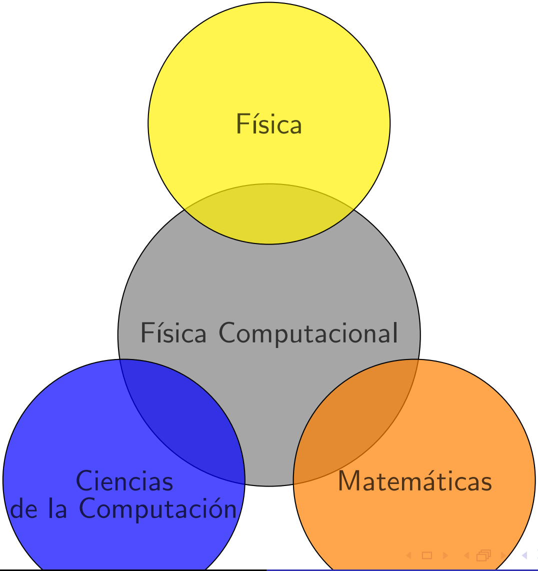
La física computacional es una nueva manera de hacer investigación en física, próxima al experimento y a la teoría.

En el laboratorio se realizan mediciones en sistemas físicos reales (restringida a la factibilidad de recursos técnicos), y que luego los físicos teóricos explican esas mediciones mediante las teorías.

Areas de investigación en la física

- Problemas que no tienen solución analítica.
- Validar aproximaciones y hacer efectivas las teorías propuestas.
- Comparar cuantitativamente teorías y mediciones experimentales.
- Visualizar conjuntos de datos complejos.
- Control y medición de experimentos.

- Predicción del clima
- Superconductividad
- Genoma Humano
- Visión y lenguaje
- Fusión nuclear
- Oceanografía
- Ciencia de los materiales
- Diseño de semiconductores
- Astrofísica relativista
- Sistemas de combustión
- Estructura biológica
- Diseño de fármacos
- Turbulencia
- Recuperación de petróleo y gas
- Cromodinámica cuántica


Física
Física Computacional
Ciencias
de la Computación
Matemáticas

Método numérico

Se puede representar como una cadena de algoritmos A_i con $(i = 1, 2, 3, \ldots, N)$ en la entrada y salida.

Entrada		Salida
	$A_1 \rightarrow A_2 \rightarrow A_3 \rightarrow \ldots \rightarrow A_N$	
Datos iniciales		Solución numérica

La solución obtenida por un método numérico es aproximada, es decir, hay cierta diferencia entre la solución exacta y la solución numérica.

Principales causas de la diferencia

- Falta de correspondencia entre el problema (modelo) matemático y el fenómeno físico real.
- Errores en los datos iniciales (parámetros de entrada).
- Errores en el método numérico usado para resolver el problema.
- Errores de redondeo en las operaciones aritméticas.

Conceptos principales relacionados con modelos y métodos numéricos

- Aproximación.
- Estabilidad.
- Convergencia.

Aproximación

Es la proximidad de un modelo numérico al modelo original (diferencial, integral, etc.) o el grado de aproximación, caracteriza el error que se introduce al hacer discreto el modelo continuo.

El grado de aproximación n se estima mediante un factor que tiene el error entre dos modelos.

Estabilidad

- Caracteriza la manera de propagación de los errores iniciales dentro del algoritmo en el proceso del cálculo.
- Si el incremento de errores iniciales es considerable y sin ningún control, entonces el método numérico se llama inestable.
- Si los errores de cálculos dependen continuamente de los errores iniciales, entonces el método se llama estable.

Convergencia

- Significa que la solución numérica converge hacia la solución exacta cuando el tamaño de la malla h tiene a cero, o el número de truncación N tiende al infinito.

Errores en los métodos numéricos

- Truncamiento: se debe a las aproximaciones utilizadas en la fórmula matemática del modelo.
- Redondeo: se asocia al hecho de que la representación de un número en la computadora se hace con un conjunto limitado de dígitos.

Series de Taylor

- Las soluciones numéricas son en su mayoría, aproximaciones de las soluciones exactas.
- Gran parte de los métodos numéricos se basan en la aproximación de funciones por medio de polinomios.

El desarrollo de Taylor es una serie infinita de potencias, representa de manera exacta a una función dentro de un cierto radio alrededor de un punto dado.

Al comparar el desarrollo polinomial de la solución numérica con la serie de Taylor de la solución exacta, es posible evaluar el error, conocido como error de truncamiento.

Si se ignoran todos los términos de la serie de Taylor, excepto algunos cuantos, se puede obtener un polinomio que se aproxime a la función verdadera.

A éste polinomio se le llama *serie de Taylor truncada* y se usa como punto de partida para obtener métodos numéricos.

Definición de Serie de Taylor

Una función $f(x)$ es analítica en $x = a$ si $f(x)$ se puede representar por medio de una serie de potencias en términos de $h = x - a$ dentro de un radio de convergencia $D > |x - a| > 0$.

Una condición necesaria para que una función sea analítica es que todas sus derivadas sean continuas tanto en $x = a$ como en alguna vecindad alrededor de ese punto.

Un punto en donde una función $f(x)$ no es analítica recibe el nombre de punto singular.

Si $f(x)$ es diferenciable en todas las partes de la vecindad de x_0, excepto en x_0 , entonces x_0 es un punto singular.

Los polinomios son analíticos en todas partes.

Si f es analítica alrededor de $x = a$, se puede representar $f(x)$ de manera exacta en la vecindad de $x = a$ por medio de una serie de Taylor, dada por:

$$
\begin{aligned}
f(x) = f(a) + hf'(a) + \frac{h^2}{2} f''(a) + \frac{h^3}{6} f'''(a) + \ldots \\
+ \ldots + \frac{h^m}{m!} f^m(a) + \ldots
\end{aligned}
$$

La serie de Taylor es única, esto quiere decir que no existe otra serie de potencias en $h = x - a$ para representar $f(x)$

El desarrollo de Taylor de una función alrededor de $x = 0$ recibe el nombre de serie de Maclaurin.

En aplicaciones prácticas, se debe de truncar la serie de Taylor después cierto orden, ya que es imposible incluir un número infinito de términos. Si la serie se trunca después del término N, se expresa por:

$$\begin{aligned} f(x) &= f(a) + hf'(a) + \frac{h^2}{2} f''(a) + \frac{h^3}{6} f'''(a) + \ldots \\ &= + \ldots + \frac{h^N}{N!} f^N(a) + O(h^{N+1}) \end{aligned}$$

Donde $h = x - a$ y $O(h^{N+1})$ representa el error por el truncamiento de los términos de orden $N + 1$

El error global se puede representar como:

$$O(h^{N+1}) = f^{N+1}(a + \xi h) \frac{h^{N+1}}{(N+1)!} \qquad 0 \leq \xi \leq 1$$

Dado que ξ no puede calcularse con exactitud, se aproxima el término del error, haciendo $\xi = 0$

$$O(h^{N+1}) \simeq f^{N+1}(a) \frac{h^{N+1}}{(N+1)!}$$

Si $N = 1$ la serie de Taylor truncada es:

$$f(x) \simeq f(a) + f'(a)h \qquad h = x - a$$

incluyendo el efecto del error, tenemos que

$$f(x) \simeq f(a) + f'(a)h + O(h^2)$$

donde

$$O(h^2 \simeq f''(a + \xi h)\frac{h^2}{2} \qquad 0 \leq \xi \leq 1$$

Representación de los números en las computadoras

Base decimal

El valor decimal de un número de base r es

$$(abcdefg.hijk)_r$$

que se calcula como:

$$\begin{aligned} ar^6 + br^5 + cr^4 + dr^3 + er^2 + fr + g + \\ + hr^{-1} + ir^{-2} + jr^{-3} + kr^{-4} \end{aligned}$$

La menor y mayor magnitud de un número real que se pueden representar en una computadora, varían de acuerdo con el diseño tanto de hardware como de software.

Los números reales en una computadora no son continuos. Si nos fijamos en número cercano a cero, el número positivo más pequeño en una IBM es 2.9×10^{-39}

Por tanto, no se pueden representar números entre 0 y 2.9×10^{-39}

A éste intervalo se le conoce como épsilon de la máquina.

Epsilon de la máquina

Hay dos maneras de definir el épsilon de la máquina: un épsilon absoluto y un épsilon relativo.

Este último es el más usado. Como el conjunto de números en la computadora es finito, la siguiente definición de épsilon relativo tiene sentido:

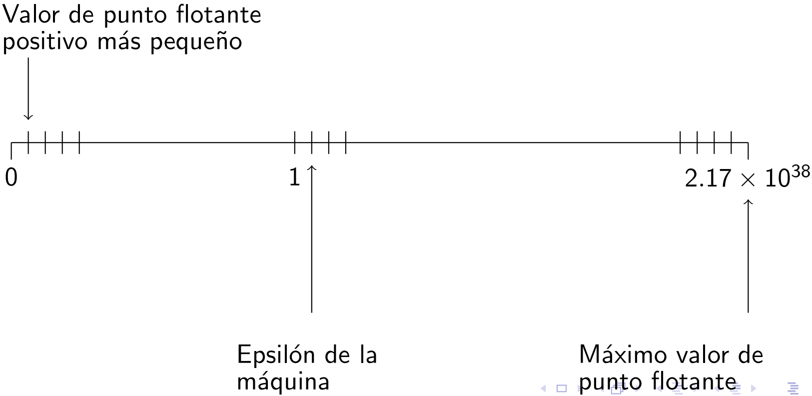
$$\epsilon_{maq} = \epsilon = \text{mín}[t > 0 : 1 + t > 1]$$

El épsilon absoluto se define comparando con cero:

$$[\epsilon_{abs} = \text{mín}[t > 0 : t \neq]0$$

En realidad el épsilon depende de la máquina pero también del sistema operativo, del compilador y del tipo de números utilizados.

Distribución de números reales en un computadora IBM PC



Primer código en Python

```
t = 1.0;
while 1+t != 1:
  eps = t
  t = t/2
print 'el cero de la maquina es: ', eps
```

Segunda versión del código

```
t=1.0
uno = 1
t1 = uno + t
while uno != t1:
    eps = t
    t = t/2
    t1 = uno + t
print 'el epsilon de la maquina (codigo version 2) es = ',eps
```

Error de truncamiento

Se originan al emplear al número finito de términos para calcular un valor que requiere un número infinito de términos.

Por ejemplo, una expresión que permite determinar de forma exacta el valor del número de Euler (base de los logaritmos naturales) a través de una serie de MacLaurin es:

$$e^x = \sum \frac{x_i}{i!}$$

Sin embargo, una aproximación a dicho valor, puede obtenerse a través de su expresión finita:

$$e^{x} \simeq \sum_{i=0}^{k} \frac{x_{i}}{i!} \qquad k < \infty$$

Es claro que esta expresión finita es manejable computacionalmente hablando, al contrario que la fórmula expresada en su forma infinita.

Error por redondeo

Se origina por el hecho de que una computadora sólo puede representar un número finito de términos. Para expresar una cantidad con un desarrollo decimal infinito, se tiene que prescindir de la mayoría de ellos.

Por ejemplo, el número $\pi = 3.14159265\ldots$, tiene un desarrollo decimal infinito no periódico. Por lo tanto, para fines de cálculo, sólo se toman algunos de sus dígitos.

Estrategias

- Redondeo. Prescinde de cierto número de cifras significativas y realiza un ajuste, sobre la última cifra no descartada: $\pi = 3.1416$
- Corte o poda: Prescinde de cierto número de cifras significativas sin realizar un ajuste sobre la última cifra no descartada: $\pi = 3.1415$

Nuevos conceptos

Una vez que se ha establecido la clasificación del error, definimos los conceptos de error absoluto verdadero, error relativo verdadero, error relativo aproximado, todos ellos como una suma o consecuencia de los errores de redondeo y truncamiento.

Error absoluto verdadero

Supóngase que $\widehat{p}$ es una aproximación a p.

El error absoluto verdadero se define con la siguiente expresión:

$$E_v = |p - \widehat{p}|$$

Esta definición de error, lo cuantifica en términos brutos. No obstante, una medida que puede describir con mayor detalle o proporción el error, es aquella que lo expresa en términos porcentuales. Para ello se emplea el error verdadero relativo.

Error relativo verdadero

Supóngase que $\widehat{p}$ es una aproximación a p. El error relativo verdadero se calcula con la siguiente expresión:

$$e_v = \frac{|p - \widehat{p}|}{p}$$

El resultado suele expresarse en términos porcentuales.

Error relativo aproximado

El error relativo aproximado, mide el error de un método numérico, determinando el error de la iteración actual respecto el error surgido en la iteración anterior:

$$e_a = \frac{\left| \widehat{x}_i - \widehat{x}_{i-1} \right|}{\widehat{x}_i}$$

Donde x_i es la aproximación actual a x y x_{i-1} es la aproximación anterior a x.

En métodos numéricos suele establecerse una tolerancia porcentual como criterio de paro, tal que el error relativo aproximado de un método, no exceda dicha tolerancia.

$$e_a < t$$

donde t, es tolerancia fijada de antemano. A menor tolerancia se tiene mayor precisión en la aproximación al valor verdadero, sin embargo esto implica un aumento en el número de iteraciones requeridas para detener el método.

Un error en un cálculo numérico "contamina las sucesivas evaluaciones.

Esta propagación puede describirse en términos de dos conceptos relacionados: los de estabilidad y condición.

Condición

La condición de una función $f(x)$ mide la sensibilidad de los valores de $f(x)$ a pequeños cambios de x, se define como:

$$C = \left| \frac{E_{rel}(f(x))}{E_{rel}(x)} \right|$$

Del teorema del valor medio en cálculo, podemos expresar

$$f(x_T) - f(x_A) \approx f'(x_t)(x_T - x_A) \rightarrow E_{rel}(f(x)) \approx$$
$$\approx \frac{f'(x_T)}{f(x_T)}(x_T - x_A)$$

luego

$$C \approx \left| x_T \frac{f'(x_T)}{f(x_T)} \right|$$

Se utilizará ésta definición como definición de condición para funciones $f(x)$ de una variable real.

Entonces los números de condición serán

$$C = \left| x \frac{f'(x)}{f(x)} \right|$$

1. Para un x dado $0 < C(x) < 1$ se dirá que el problema está bien condicionado, y cuanto menor sea C, mejor condicionado.
2. Si $C(x) > 1$, el problema estará mal condicionado.
3. Si $C(x) = 1$, el error relativo se mantiene.

Ejemplos

Las siguientes funciones están bien condicionadas?

1. $f(x) = \sqrt{x}$ $\qquad C(x) = ?$
2. $g(x) = x^2 - 1$ $\qquad C(x) = ?$

Estabilidad

La estabilidad de un algoritmo describe la sensibilidad de un método numérico específico respecto a los inevitables errores de redondeo cometidos durante su ejecución en aritmética de precisión finita.

Consideremos la siguiente función:

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

Su número de condición es:

$$C(x) = \left| x \frac{f'(x)}{f(x)} \right| = \frac{x}{2\sqrt{x}\sqrt{x+1}}$$

Vemos que $C(x) < \frac{1}{2}$ para $x > 0$, por lo que la función está bien condicionada pero ...

El algoritmo para calcular x de tal forma que se vayan realizando las operaciones, es:

1. obtener x
2. $y = x + 1$
3. $f = sqrt(y)$
4. $g = sqrt(x)$
5. $h = f - g$

es inestable para x grandes, dado el paso 5, por lo que debemos de re-estructurar la función.

Eficiencia

Todo algoritmo debe evitar ser inestable. Si existieran varios métodos para evaluar una misma función, entonces conviene utilizar aquel que sea más eficiente, es decir, más rápido.

Debemos de aprovechar al máximo los recursos: hardware, software, algoritmos, para resolver problemas más complejos y no para resolver peor problemas simples.

Eficiencia

Todo algoritmo debe evitar ser inestable. Si existieran varios métodos para evaluar una misma función, entonces conviene utilizar aquel que sea más eficiente, es decir, más rápido.

Debemos de aprovechar al máximo los recursos: hardware, software, algoritmos, para resolver problemas más complejos y no para resolver peor problemas simples.

Por ejemplo, para calcular $x**4$ para un x dado, no es buena idea calcular $x**4.0$ (exponente en punto flotante).

La mejor idea consiste en economizar el cálculo en dos pasos:

$$\begin{aligned} x2 &= x * x \\ x4 &= x2 * x2 \end{aligned}$$

y no un producto $x4 = x * x * x * x$

Ejemplo: Evaluación de polinomios

Supongamos que queremos evaluar el polinomio:

$$P(x) = 2 + 4x - 5x^2 + 2x^3 - 6x^4 + 8x^5 + 10x^6$$

Contando con que cada potencia de exponente k entero como $k - 1$ productos, tendríamos que el total de productos para evaluar en forma directa es:

$$1 + 2 + 3 + 4 + 5 + 6 = 21$$

Además de seis sumas.

Una mejora en el algoritmo , es calcular primero las potencias de forma sucesiva:

$$
\begin{aligned}
x^2 &= x * x \\
x^3 &= x * x^2 \\
x^4 &= x * x^3 \\
x^5 &= x * x^4 \\
x^6 &= x * x^6
\end{aligned}
$$

De tal forma que se añade un producto por cada potencia, para un total de productos:

$$1 + 2 + 2 + 2 + 2 + 2 = 11$$

Con el polinomio

$$P(x) = 2 + 4x - 5x^2 + 2x^3 - 6x^4 + 8x^5 + 10x^6$$

podemos mejorar el algoritmo de la siguiente manera

$$P(x) = 2 + x\{4 + x(-5 + x[2 + x(-6 + x\{8 + x * 10\})])\}$$

Para evaluar un polinomio de grado n en el que ninguno de los coeficientes es cero, se necesitan

$\dfrac{n(n+1)}{2}$	Productos para el primer método
$2n-1$	para el segundo métodos
n	para el tercero

Antes de escribir una línea de código, hay que revisar la manera en que podemos optimizar la solución del problema.

Algoritmo de Horner

Dado el polinomio

$$P(x) = a_0 + a_1 x + \ldots + a_n x^n \qquad a_n \neq 0$$

La evaluación de $P(x)$ para cierto valor de $x = z$ se puede realizar en n pasos mediante:

$$\begin{aligned}
b_n &= a_n \\
b_{n-1} &= a_{n-1} + z * b_n \\
b_{n-2} &= a_{n-2} + z * b_{n-1} \\
&\vdots \\
b_0 &= a_0 + z * b_1
\end{aligned}$$

Pseudocódigo

1. $b_n = a_n$
2. repetir mientras $n > 0$
3. $n = n - 1$
4. $b = a_n + z * b$
5. volver al paso 2
6. $p(z) = b$

Completa la siguiente tabla

$$P(x) = 2 + 4x - 5x^2 + 2x^3 - 6x^4 + 8x^5 + 10x^6$$

Evalúa el polinomio $P(x)$ en:

x	P(x)
-1.5	
-0.65	
0.1	
1.4	
2.87	