Clase 4
Análisis del Error

4.1 Fórmula de propagación del Error

En la sección anterior vimos cómo se propaga el error cuando realizamos **operaciones aritméticas básicas como sumar, restar, multiplicar o dividir.**

Nos podemos plantear el caso general, cómo se propaga el error cuando aplicamos una función f a una cantidad x con error, es decir, dada una función cualquiera f y una cantidad x, ¿podemos tener una idea de cuánto nos equivocamos cuando hacemos $fl\left(f\left(fl\left(x\right)\right)\right)$? o sea, ¿cuánto vale: $\left|fl\left(f\left(fl\left(x\right)\right)\right)-f\left(x\right)\right|$?

La **fórmula de propagación del error** intenta responder a la cuestión anterior.

Definición 1:

Sea f una función de clase $\mathcal{C}^1$ y x un valor con error absoluto $e_a\left(x\right)$. En este caso tenemos

$$\left|f\left(x+e_a\left(x\right)\right)-f\left(x\right)\right|\approx\left|f'\left(x\right)\right|\cdot e_a\left(x\right)$$

es decir, el error en la función $f\left(x\right)$ es aproximadamente el valor de la derivada en x multiplicado por $e_a\left(x\right)$, de hecho

$$\left|f\left(x+e_a\left(x\right)\right)-f\left(x\right)\right|\leq\underset{t\in(x-e_a(x),x+e_a(x))}{\text{máx}}\left|f'\left(t\right)\right|\cdot e_a\left(x\right)$$

Por lo tanto, el comportamiento de la función derivada es clave para ver cómo se propaga el error cuando aplicamos una función f

Demostración

Si aplicamos el Teorema del Valor Medio a la función f en el intervalo $(x - e_a(x), x + e_a(x))$, tenemos que existe un punto $c \in (x - e_a(x), x + e_a(x))$ tal que:

$$f(x + e_a(x)) - f(x) = f'(c) \cdot e_a(x)$$

Si aproximamos $f'(c)$ por $f'(c) \approx f'(x)$ y considerando valores absolutos a la expresión anterior tenemos la fórmula de propagación del error.

Ejemplo 1:

Queremos computar $\left(\sqrt{2}-1\right)^6$ utilizando $x = 1.4$ valor aproximado de $\sqrt{2}$. ¿Cuál de las expresiones siguientes es la más apropiada desde el punto de vista numérico?

$$\frac{1}{\left(\sqrt{2}+1\right)^6}, \quad \left(3-2\sqrt{2}\right)^3, \quad \frac{1}{\left(3+2\sqrt{2}\right)^3}$$

Sean f_1, f_2 y f_3 las funciones siguientes:

$$f_1 = \frac{1}{(x+1)^6}$$
$$f_2 = (3-2x)^3$$
$$f_3 = \frac{1}{(3+2x)^3}$$

Para entender de donde salieron estas expresiones vamos a realizar lo siguiente:

- En la expresión $\left(\sqrt{2}-1\right)^6$ multiplicamos numerador y denominador por $\sqrt{2}+1$, así

$$= \left(\sqrt{2}-1\right)^6 \cdot \frac{\left(\sqrt{2}+1\right)^6}{\left(\sqrt{2}+1\right)^6}$$
$$= \frac{\left(\left(\sqrt{2}-1\right)\left(\sqrt{2}+1\right)\right)^6}{\left(\sqrt{2}+1\right)^6}$$
$$= \frac{(2-1)^6}{\left(\sqrt{2}+1\right)^6} = \frac{1}{(x+1)^6}$$

- Podemos expresar la expresión original como

$$\left(\left(\sqrt{2}-1\right)^2\right)^3 = \left(2-2\sqrt{2}+1\right)^3 = \left(3-2\sqrt{2}\right)^3$$

- En la expresión anterior $\left(3-2\sqrt{2}\right)^3$ se multiplica por el conjugado y se obtiene la expresión tres, así,

$$\left(3-2\sqrt{2}\right)^3 \cdot \frac{\left(3+2\sqrt{2}\right)^3}{\left(3+2\sqrt{2}\right)^3} = \frac{\left(9-4\cdot 2\right)^3}{\left(3+2\sqrt{2}\right)^3} = \frac{1}{\left(3+2\sqrt{2}\right)^3} = \frac{1}{\left(3+2x\right)^3}$$

De esta forma se da origen a las tres expresiones.

La derivada de las funciones anteriores en el punto 1.4 valen:

$$f_1'(x) = -\frac{6}{(x+1)^7} \Rightarrow |f_1'(1.4)| = 0.01308195$$
$$f_2'(x) = -6(3-2x)^2 \Rightarrow |f_2'(1.4)| = 0.24000000$$
$$f_3'(x) = -\frac{6}{(3+2x)^4} \Rightarrow |f_3'(1.4)| = 0.00530199$$

Se elige la derivada que evaluada en $x = 1.4$ la más pequeña es la correspondiente a la función f_3.

Como $e_r(x) \leq 5 \times 10^{-2}$ (el error relativo debe ser menor o igual a $5 \cdot 10^{-k}$ donde $k = 2$) ya que tiene dos cifras significativas, tenemos que

$$e_a(x) \approx |x| \cdot e_r(x) \leq 0.07$$
$$\approx |1.4| \cdot 5 \times 10^{-2} \leq 0.07$$

.

Los errores cometidos aplicando las funciones anteriores serán

aproximadamente;

$$\left|f_1\left(\sqrt{2}\right) - f_1\left(1.4\right)\right| \approx \left|f_1'\left(1.4\right)\right| \cdot e_a\left(x\right) \leq 0.01308195 \cdot 0.07 = 9.157 \times 10^{-4}$$
$$\left|f_2\left(\sqrt{2}\right) - f_2\left(1.4\right)\right| \approx \left|f_2'\left(1.4\right)\right| \cdot e_a\left(x\right) \leq 0.24000000 \cdot 0.07 = 1.68 \times 10^{-2}$$
$$\left|f_3\left(\sqrt{2}\right) - f_2\left(1.4\right)\right| \approx \left|f_3'\left(1.4\right)\right| \cdot e_a\left(x\right) \leq 0.00530199 \cdot 0.07 = 3.711 \times 10^{-4}$$

Observamos que la mejor función es la función f_3.

Ejercicio 1:

Construir un programa en **Python** que muestre los cálculos de este problema

Programa de Google Colab

Este programa se ha desarrollado usando Google Colab. Para ver el programa haz clic en el logo.

4.2 Formula de la Propagación del Error en n variables

Vamos a generalizar la fórmula de propagación del error para el caso en que tengamos n valores $x_1, \cdots, x_n$:

Definición 2:

Sea f una función de n variables de clase $\mathcal{C}^1$ y $x_1, \ldots, x_n$ n valores con errores absolutos $e_a(x_1), \ldots, e_a(x_n)$, respectivamente. En este caso, tenemos:

$$\begin{aligned} &|f(x_1 + e_a(x_1), \ldots, x_n + e_a(x_n)) - f(x_1, \ldots, x_n)| \\ &\approx \sum_{i=1}^{n} \left| \frac{\partial f}{\partial x_i}(x_1, \ldots, x_n) \right| \cdot e_a(x_i) \end{aligned}$$

es decir, el error en la función $f(x)$ es aproximadamente la suma de las derivadas parciales respecto cada variable x_1 multiplicando por $e_a(x_i)$, para $i = 1, \ldots, n$

Ejemplo 2:

Calcular el área de un círculo donde $e_a(\pi)$ y $e_a(r)$ son los errores absolutos de los valores π y el radio r.

El área del círculo se representa mediante la siguiente función

$$A(\pi, r) = \pi \cdot r^2$$

Utilizando la fórmula de propagación del error para dos variables, tenemos:

$$\begin{aligned} &|A(\pi + e_a(\pi), r + e_a(r)) - A(\pi, r)| \\ &\approx \left| \frac{\partial A}{\partial \pi} \right| \cdot e_a(\pi) + \left| \frac{\partial A}{\partial r} \right| \cdot e_a(r) \\ &= r^2 \cdot e_a(\pi) + 2\pi r \cdot e_a(\pi) \end{aligned}$$

Si por ejemplo, $r = 1$ cm, con $e_a(r) \leq 0.001$ (suponiendo que el instrumento de medida del radio comete ese error) y que el valor de $e_a(\pi) \leq 0.001$, obtenemos:

$$\begin{aligned} &|A(\pi + e_a(\pi), r + e_a(r)) - A(\pi, r)| \\ &\leq (1 + 0.001)^2 (0.001) + 2(\pi + 0.001)(1 + 0.001)(0.001) \\ &= 0.00729428699999999 \end{aligned}$$

Ejercicio 2:

Crear un programa en Python en el que se calcule el valor del error y el área del círculo del ejercicio anterior

Programa de Google Colab

Este programa se ha desarrollado usando Google Colab. Para ver el programa haz clic en el logo.

4.3 Algoritmos y Convergencia

Los métodos numéricos están basados en algoritmos:

Definición 3: Definición de Algoritmo

Un algoritmo es un procedimiento que describe de forma **no ambigua** una secuencia finita de pasos en un orden específico.

Para describir los algoritmos usamos lo que denominamos **pseudocódigo**, que sería una forma de describir el algoritmo de una manera *taquigráfica*, es decir, escueta y entendible.

4.3.1 Pseudocódigo

El pseudocódigo usa los símbolos siguientes:

- **Programa:** Se indica con la palabra reservada **program**
- **Inicio y Fin de una secuencia de comandos:** Se usan las palabras reservadas **begin** y **end**
- **Un punto y coma**: separa las tareas dentro de un mismo paso.
- **Leer Valores:** Para leer valores se usa la palabra reservada InputInteger e InputReal, ejemplo:

```
1 n := InputInteger(edNumero);
2 r := InputReal(edNumero);
3
```

```
{edNumero es una caja de texto donde se escribe el
  valor numérico}
```

- **Mostrar texto y valores:** se usa la función console.log(seccion, variable, valor), ejemplo:

```
console.log('Cálculo', 'Número', 3.141592654);
```

- **Bucle For**: se indican con la palabra reservada **for,** ejemplo:

```
for i:=1 to 10 do
begin
  // Pasos a ejectuar
end;
```

- **Definición de Variables:** se usan dos puntos para definir una variable. Las variables se crean en una sección al inicio del programa mediante la palabra reservada **var**, ejemplo:

```
var
  n: integer;
  r: real;
  s: string;
  b: boolean;
```

- **Tipos de Variables:** como se observó anteriormente los tipos para definir una variable son:
  - **integer**: Número entero.
  - **real**: Número real.
  - **string**: Cadena de texto.
  - **boolean**: Valor booleano.
  - **array[***start* **..** *end***]**: **of type**; donde ***start*** se refiere al inicio de la indexación y ***end*** al final de la indexación del arreglo. ***of type*** es uno de los antes mencionados.

  ejemplo:

```
numero: integer;
tolerancia: real;
nombre: string;
esPrimo: boolean;
```

```
coeficientes: array[1..10] of real;
```

- **Asignación de variables**: se usa dos puntos y un igual para asignar un valor. **Ejemplo:** $x := 2$
- **Ciclo While:** se indica mediante la palabra reservada **while**, ejemplo:

```
while (condicion) do
begin
  // Pasos a ejectuar
end;
```

- **Condicional if:** se indica mediante las palabras reservadas **if-then-else-end,** ejemplo:

```
if (condicion) then
begin
  // Tareas a realizar si cumple la condición
end
else
  // Tareas a realizar si no se cumple la condición
end;
```

- **Finalizar Ciclo:** se usa la palabra reservada **Break;**
- **Finalizar un Programa:** se usa la palabra reservada **Exit;**
- **Punto**: indica que el programa terminó.
- **Comentario**: Para escribir comentarios en el pseudocódigo se usan {}, ejemplo:

```
{ Esto es un comentario entre llaves }
```

4.3.2 Historia de Pascal

Pascal es un lenguaje de programación imperativo, cuyo nombre se debe al matemático y filósofo francés Blaise Pascal. Fue desarrollado por el científico suizo Niklaus Wirth, quien lo dio a conocer en 1970.

Wirth inició el desarrollo de Pascal tras su paso por la Universidad de Stanford, donde había participado en el diseño del lenguaje ALGOL.

Su objetivo era crear un lenguaje sencillo para la **enseñanza de la programación.** Así, Pascal se convirtió rápidamente en uno de los lenguajes más utilizados en las universidades para la enseñanza de la programación.

A pesar de que **Pascal** no es un lenguaje de programación de propósito general, ha sido ampliamente utilizado en diferentes campos. Por ejemplo, fue el lenguaje en que se desarrolló la primera versión de la **interfaz gráfica de usuario de Apple, Lisa**. También fue muy utilizado en el desarrollo de videojuegos durante la década de 1980.

La primera versión del compilador de Pascal fue escrita en lenguaje ensamblador para la máquina CDC 6000. Wirth dirigió posteriormente un equipo que desarrolló un compilador para la máquina PDP-11. En 1973 publicó un informe titulado "El lenguaje de programación Pascal".

En la década de 1980, Pascal fue estandarizado por el Instituto de Ingenieros Eléctricos y Electrónicos (IEEE) y por la Organización Internacional de Normalización (ISO).

Ejemplo 3:

Determinar si un número es un número primo

El pseudocódigo siguiente nos permite identificar si un número es primo o no:

```
1  var
2    numero, divisor: integer;
3  begin
4    { Obtener el número a evaluar }
5    numero := InputInteger(edNumero);
6
7    { Inicializar el divisor en 2}
8    divisor := 2;
9
10   { Crear un ciclo condicional para determinar si el
11     número ingresado tiene más de un divisor}
12   while (divisor <= Sqrt(numero)) do
```

```
13  begin
14    {Preguntar si el numero es divisible entre divisor}
15    if (numero mod divisor) = 0 then
16    begin
17      console.log('Resultado', 'Mensaje', 'El número no
          es primo');
18      console.log('Número', numero);
19      exit;
20    end
21    else
22    begin
23      { Incrementar el valor del divisor }
24      divisor := divisor + 1;
25    end;
26  end;
27
28  {Si se cumple todo el ciclo el número es primo}
29  console.log('Resultado', 'Mensaje', 'El número es
      primo');
30  console.log('Número', numero);
```

Ejercicio 3:

Construir un *programa* en **Python** para determinar si un número dado es primo

Programa de Google Colab

Este programa se ha desarrollado usando Google Colab. Para ver el programa haz clic en el logo.