

## Cálculo Numérico (521230) - Laboratorio 7

### *Integración Numérica*

El objetivo de este laboratorio es aprender técnicas de integración numérica, así como el uso correcto de funciones como parámetros.

En muchas ocasiones resulta sumamente útil poder pasar una función como parámetro. Un caso típico es cuando se quiere escribir un programa para calcular la integral de una función cualquiera mediante una regla de integración numérica en particular. Así, por ejemplo, si se escribe un programa MATLAB del tipo `function` llamado `trap` que calcula la aproximación de la integral de una función dada en un intervalo genérico  $[a, b]$  por la regla de los trapecios con  $N$  subintervalos, uno querría poder llamar a ese programa con las sentencias como las siguientes:

```
>> trap(f,0,1,10) % Calcula la integral de f(x) en [0,1]
                    % con 10 subintervalos
>> trap(g,-1,1,16) % Calcula la integral de g(x) en [-1,1]
                    % con 16 subintervalos
>> trap('sin',0,1,10) % Calcula la integral de sen(x) en [0,pi]
                    % con 8 subintervalos
```

Para poder hacer esto, el programa `trap` debe recibir la función pasada como parámetro ( $f(x)$ ,  $g(x)$ , o  $\sin(x)$ , respectivamente) en una variable alfabética (es decir, una variable cuyos valores son nombres y no números) y, cada vez que haga falta evaluar esa función, debe utilizarse el comando `feval`, cuya sintaxis es como en el ejemplo siguiente

```
function int=trap(funcnt,a,b,N)
. . .
y=feval(funcnt,a) % Evalua la funcion funcnt en el valor a
```

En este ejemplo, `funcnt` es la variable alfabética que contiene el nombre de la función que se quiere evaluar,  $a$  es el valor de la variable donde se quiere evaluar la función e `y` devuelve el valor calculado. En algunos casos también se puede evaluar la función directamente:

```
y=funcnt(a) % Evalua la funcion funcnt en el valor a
```

Por otra parte, debe haber un programa del tipo `function` (`f.m` en el primer ejemplo o `g.m` en el segundo) en el que se defina la función que se quiere integrar. Esto no es necesario cuando se trata de una de las funciones de biblioteca de MATLAB, como el caso de  $\sin(x)$  en el tercer ejemplo.

Cuando una función  $f(x)$  es muy simple y se quiere evitar tener que hacer un programa `f.m` para evaluarla, puede utilizarse el comando `inline` para definirla. Por ejemplo, para  $f(x) = e^{-x^2}$ :

```
>> f=inline('exp(-x.^2)'); % Define la funcion f(x)=exp(-x^2)
>> trap(f,0,1,10) % Calcula la integral de f(x) en [0,1]
                    % con 10 subintervalos
>> f(0) % Evalua exp(-0^2)
>> feval(f,0) % Evalua exp(-0^2)
```

### Ejercicio 1.

- 1.1 Escriba en un archivo `trap.m` un programa tipo `function` que calcule la aproximación de la integral de una función dada en un intervalo genérico  $[a, b]$  por la regla de los trapecios con  $N$  subintervalos.
- 1.2 Testee su programa con las siguientes integrales y  $N = 10, 20, 40, 80$ :

$$\int_0^3 x^2 dx \quad \int_{-1}^1 e^{-x^2} dx \quad \int_1^2 \log(x) dx \quad \int_0^1 \sqrt{x} dx$$

- 1.3 Haga un programa semejante para la regla de Simpson. Guárdelo en un archivo `simpson.m` y testeelo con las mismas funciones.

**Ejercicio 2.** MATLAB tiene dos comandos, `quad` y `quadl`, que permiten calcular de manera automática integrales de funciones suaves con error menor que una tolerancia dada. El primero es un método de bajo orden basado en la regla de Simpson, mientras que el segundo es un método de alto orden.

- 2.1 Utilice el comando `help` de MATLAB para conocer la sintaxis de estos comandos.
- 2.2 Calcule mediante `quad` las integrales  $\int_{-1}^1 e^{-x^2} dx$  y  $\int_0^1 \sqrt{x} dx$  del Ejercicio 1.2, primero con la tolerancia prefijada por el comando y después con error menor que  $10^{-6}$ .

**Ejercicio 3.** El fin de este ejercicio es verificar experimentalmente que el error del método de los trapecios aplicado a un integrando con derivadas acotadas en el intervalo de integración es de orden  $\mathcal{O}(h^2)$ .

- 3.1 Haga un programa que calcule la integral  $\int_0^1 e^{-x} dx$  por la regla de los trapecios con  $N = 10, 20, 30, \dots, 100$  subintervalos y almacene los errores respectivos. En este caso para calcular el error utilice el valor verdadero de la integral, el cual puede calcularse exactamente.
- 3.2 Grafique en escala logarítmica estos errores versus  $N$  y la función  $f(N) = (1/N)^2$ .  
**Sugerencia:** para graficar en escala logarítmica utilice el comando `loglog` en lugar de `plot`; la sintaxis de ambos comandos es la misma.
- 3.3 Explique por qué el gráfico anterior muestra que el error de la regla de los trapecios es en este caso  $\mathcal{O}(h^2)$ .

**Ejercicio 4.** El fin de este ejercicio es verificar experimentalmente que el error del método de los trapecios aplicado a un integrando con derivadas no acotadas en el intervalo de integración no es de orden  $\mathcal{O}(h^2)$ .

- 4.1 Repita el ejercicio anterior con la integral  $\int_0^1 \sqrt{x} dx$ .
- 4.2 Para estimar el orden de convergencia del método en este caso, determine por cuadrados mínimos las constantes  $C$  y  $\alpha$  que mejor ajustan los errores mediante el modelo

$$\text{error} \approx Ch^\alpha.$$

4.3 Grafique en escala logarítmica los errores versus  $N$  y la función  $f(N) = (1/N)^\alpha$ .

**Ejercicio 5.** Los pares  $(x, y)$  que satisfacen

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$$

pertenecen a la elipse con centro en el origen de coordenadas, semieje horizontal de longitud  $a$  y semieje vertical de longitud  $b$ . La ecuación paramétrica de esta elipse es:

$$(x(t), y(t)) = (a \cos(t), b \sin(t)), \quad t \in [-\pi, \pi]$$

Escriba un rutero MATLAB en el que:

- Grafique 200 puntos sobre la elipse de ecuación paramétrica  $(3 \cos(t), 5 \sin(t))$ ,  $t \in [-\pi, \pi]$ .
- Encuentre, con ayuda del comando `quad` o del comando `quadl` una aproximación al perímetro de la elipse antes graficada. Tenga en cuenta que éste es igual a

$$\int_{-\pi}^{\pi} \sqrt{\left(\frac{dx}{dt}\right)^2 + \left(\frac{dy}{dt}\right)^2} dt$$