UNIVERSIDAD DE CONCEPCION
FACULTAD DE CIENCIAS
FISICAS Y MATEMATICAS
DEPARTAMENTO DE INGENIERIA MATEMATICA

Cálculo Numérico (521230) - Laboratorio 7

INTEGRACION NUMERICA

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, pór 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 sen(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(funct,a,b,N)
. . .
y=feval(funct,a) % Evalua la funcion funct en el valor a
```

En este ejemplo, `funct` 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 fución e `y` devvuelve el valor claculado. En algunos casos también se puede evaluar la función directamente:

```
y=funct(a) % Evalua la funcion funct 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 sen(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)
```

También puede utilizar `trap` de la siguiente forma:

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

La instrucción `@(x)` indica que la función `f` es una entrada de `trap` con respecto a la variable `x`. Esto es útil cuando se definen funciones de más de una veriable. Por ejemplo:

```
>> f=inline('exp(-x.^2)*cos(y)'); % Define la funcion f(x,y)=exp(-x.^2)*cos(y)
>> trap(@(x)f(x,1),0,1,10) % Calcula la integral de f(x,y) c/r a x en [0,1], con y=1
                           % y con 10 subintervalos
>> trap(@(x)f(2,y),0,1,10) % Calcula la integral de f(x,y) c/r a y en [0,1], con x=2
                           % y con 10 subintervalos
```

Ejercicio 1.

1.1 Escriba en un archivo `trap.m` un programa tipo `function` que clacule 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 $N = 10, 20, 40, 80$:

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

1.3 Haga un programa semejante para la regla de Simpson. Guerdelo 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. Escriba un programa tipo `function` en MATLAB que reciba como entrada una función de 2 variables $f(x, y)$, los valores a, b, c, d y N y cuya salida sea el valor de la integral:

$$\int_a^b \int_c^d f(x, y)\, dy\, dx$$

donde la integral con respecto a y sea resuelta mediante la regla de los trapecios y la integral con respecto a x sea resuelta con la regla de Simpson.

Ejercicio 4. 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 $\mathcal{O}(h^2)$.

4.1 Haga un programa que calcule la integral $\int_0^1 e^{-x}\, dx$ por la regla de los trapecios con $N = 10, 20, 30, \ldots, 100$ 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.

4.2 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.

4.3 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 5. 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 $\mathcal{O}(h^2)$.

5.1 Repita el ejercicio anterior con la integral $\int_0^1 \sqrt{x}\, dx$.

5.2 Para estimar el orden de convergencia del método en este caso, determine por cuadrados mínimos las constantes C y α que mejor ajustan los errores mediante el modelo

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

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

Ejercicio 6. Una **Serie de Fourier** para f una función definida en el intervalo $[a, b]$ es una serie de la forma

$$SF\{f(x)\} = \frac{a_0}{2} + \sum_{n=1}^{\infty} \left(a_n \cos\left(\frac{2\pi n x}{b-a}\right) + b_n \operatorname{sen}\left(\frac{2\pi n x}{b-a}\right)\right),$$

con

$$a_0 = \frac{2}{b-a}\int_a^b f(x)\, dx, \quad a_n = \frac{2}{b-a}\int_a^b f(x)\cos\left(\frac{2\pi n x}{b-a}\right) dx \quad \text{y} \quad b_n = \frac{2}{b-a}\int_a^b f(x)\operatorname{sen}\left(\frac{2\pi n x}{b-a}\right) dx,$$

la cual converge puntualmente a f, cuando f es continua por pedazos en $[a, b]$, es decir $SF\{f(x)\} = f(x)$.

Para $N \in \mathbb{N}$, se define la **Suma Parcial de Fourier** como

$$SF_N\{f(x)\} = \frac{a_0}{2} + \sum_{n=1}^{N} \left(a_n \cos\left(\frac{2\pi n x}{b-a}\right) + b_n \operatorname{sen}\left(\frac{2\pi n x}{b-a}\right)\right).$$

6.1) Escriba una funcion en MATLAB cuyas entradas sean f, el intervalo $[a, b]$ y N. Esta función debe dibujar en un mismo gráfico f y la N-ésima Suma Parcial de Fourier en el intervalo $[a, b]$.

6.2) Pruebe su función para distintos valores $N = 1, 2, 5, 10$ y las siguientes funciones:

- $f(x) = 1$, $x \in [0, 1]$.
- $g(x) = x$, $x \in [-1, 2]$.
- $h(x) = \begin{cases} 1 & \text{si } x \in [-2, 0[\\ x^2 & \text{si } x \in [0, 1] \end{cases}$

Observe que ocurre a medida que crecen los valores de N.

6.3) Modifique la función anterior para que calcule la Suma Parcial de Fourier en el intervalo $[a, b]$, pero que grafique dicha suma en el intervalo $[a - L, b + L]$, con $L = b - a$. ¿Qué se puede observar?

Sugerencia: Es muy útil el uso de `@(x)` en el cálculo de las integrales, considando que se integra una función que depende de a, b, n y la variable x.

Ejercicio 7. 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 coordenas, semieje mayor de longitud a y semieje menor de longitud b. La ecuación paramétrica de esta elipse es:

$$(x(t), y(t)) = (a\cos(t), b\,\mathrm{sen}(t)), \quad t \in [-\pi, \pi]$$

Escriba un rutero MATLAB en el que:

- Grafique 200 puntos sobre la elipse de ecuación paramétrica $(3\cos(t), 5\,\mathrm{sen}(t))$, $t \in [-\pi, \pi]$.
- 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

$$\int_{-\pi}^{\pi} \sqrt{\left(\frac{\mathrm{d}x}{\mathrm{d}t}\right)^2 + \left(\frac{\mathrm{d}y}{\mathrm{d}t}\right)^2}\, dt$$