

Análisis Numérico: Parcial 3

2 de marzo de 2025

Universidad Nacional de Colombia

Jorge Mauricio Ruiz Vera

Andrés David Cadena Simons
Sandra Natalia Florez Garcia

acadenas@unal.edu.co
sflorezga@unal.edu.co

Problema 1:

Pregunta

Solución:

Solución

Problema 2:

Pregunta

Solución:

Solución

Problema 3:

Pregunta

Solución:

Solución

Problema 4:

Pregunta

Solución:

Solución

Problema 5:

Observe que

$$I = \int_0^1 \frac{4}{1+x^2} = \pi$$

1. Use las reglas compuestas del punto medio, del trapecio y de Simpson para aproximar I para varios tamaños de paso de integración $h = \frac{1}{n}$, $n = 10, 50, 100, 250, 500, 1000, 1500, 2000$. Grafique el logaritmo del error absoluto versus n para cada paso. Describa el efecto de redondeo de los errores cuando $i \rightarrow 0$.

Solución:

Veamos lo que sucede para cada caso:

- Punto Medio.

Note que al aplicar el algoritmo para la integral obtenemos la siguiente gráfica de errores:

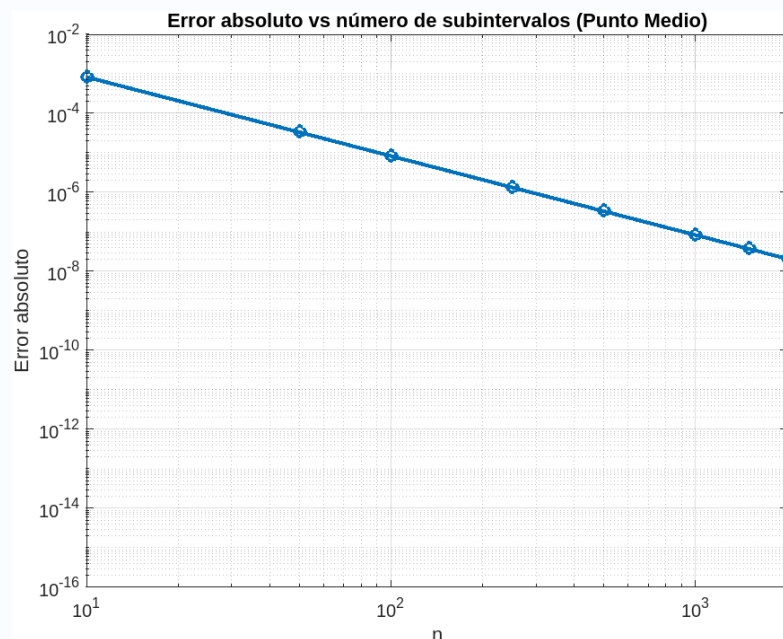


Figura 1: Error absoluto en el método del punto medio.

Aquí cabe recordar que el método del punto medio tiene convergencia de orden $O(h^2)$ y su implementación es bastante sencilla, la podremos ver en el siguiente código que aplicando el método se tomó 0,001995 segundos.

```
1  format rational
2
3  % Creamos la función f
4  f = @(x) 4 ./ (1 + x.^2);
5
6  % Parámetros de la integral y el método
7  a = 0;
8  b = 1;
9  valores_n = [10, 50, 100, 250, 500, 1000, 1500, 2000];
10 I_exacto = pi;
11
12 % Se crea un vector para guardar los errores
13 errores = zeros(size(valores_n));
14
15 % Método del punto medio
16 tic;
17 for k = 1:length(valores_n)
18     n = valores_n(k);
19     % Aquí se encuentra el tamaño del intervalo
20     h = (b - a) / n;
21     % Se calculan los puntos medios y los guarda en el vector
22     x_medios = a + ((2*(0:1:n-1)+1)*h)/2;
23     % Regla del punto medio compuesta
24     I_punto_medio = h * sum(f(x_medios));
25     % Calculo del error
26     errores(k) = abs(I_punto_medio - I_exacto);
27 end
28 tiempo = toc
29
30 % Graficar log-log del error vs n
31 figure;
32 loglog(valores_n, errores, '-o', 'LineWidth', 2);
33 xlabel('n');
34 ylabel('Error absoluto');
35 ylim([1e-16, 1e-2]);
36 title('Error absoluto vs número de subintervalos (Punto
37       Medio)');
37 grid on;
```

■ Trapecio.

Note que al aplicar el algoritmo para la integral obtenemos la siguiente gráfica de errores:

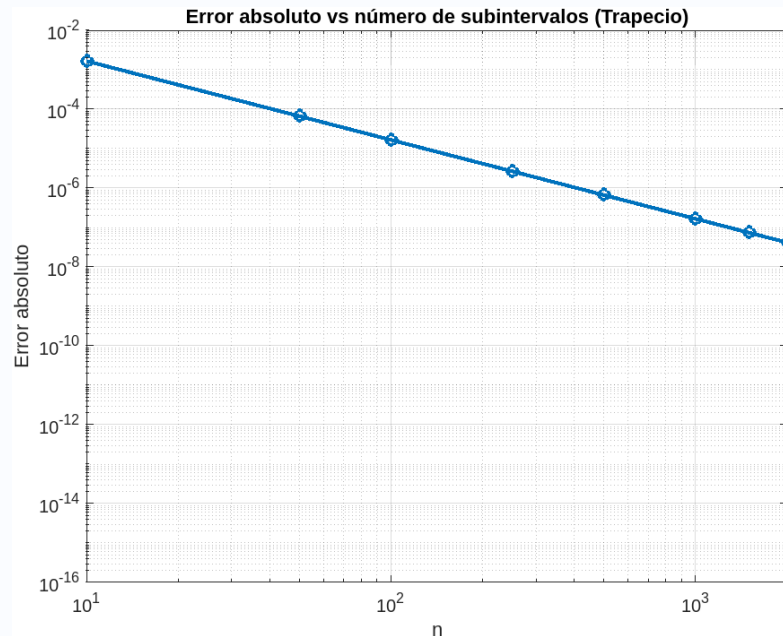


Figura 2: Error absoluto en el método del trapecio.

Aquí cabe recordar que el método del trapecio tiene convergencia de orden $O(h^2)$ y su implementación es (en cuestiones de operaciones) sencilla, la podremos ver en el siguiente código que aplicando el método se tomó 0,002719 segundos.


```
1  format rational
2
3  % Creamos la función f
4  f = @(x) 4 ./ (1 + x.^2);
5
6  % Parámetros de la integral y el método
7  a = 0;
8  b = 1;
9  valores_n = [10, 50, 100, 250, 500, 1000, 1500, 2000];
10 I_exacto = pi;
11
12 % Se crea un vector para guardar los errores
13 errores_trapecio = zeros(size(valores_n));
14
15 % Método del trapecio
16 for k = 1:length(valores_n)
17     n = valores_n(k);
18     h = (b - a) / n;
19     % Extremos de la partición
20     x_extremos = a:h:b;
21     m = length(x_extremos);
22
23     % Aplicar la regla del trapecio compuesta
24     I_trapecio = (h/2) * (f(a) + 2*sum(f(x_extremos(2:m-1)))
25         + f(b));
26
27     % Calculo del error
28     errores_trapecio(k) = abs(I_trapecio - I_exacto);
29 end
30
31 % Graficar error del método del trapecio
32 figure;
33 loglog(valores_n, errores_trapecio, '-o', 'LineWidth', 2);
34 xlabel('n');
35 ylabel('Error absoluto');
36 ylim([1e-16, 1e-2]);
37 title('Error absoluto vs número de subintervalos (Trapecio)');
38 grid on;
```

- Simpson.

Note que al aplicar el algoritmo para la integral obtenemos la siguiente gráfica de errores:

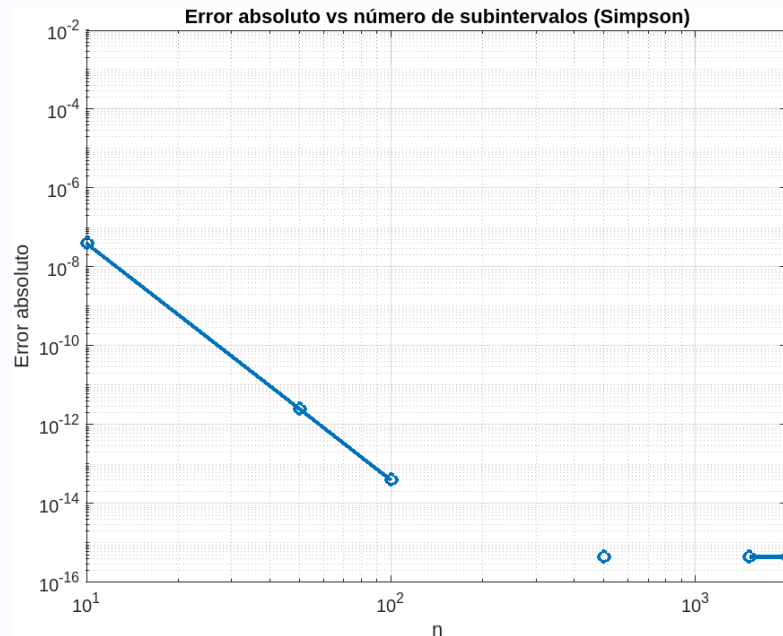


Figura 3: Error absoluto en el método de Simpson.

Aquí cabe recordar que el método tiene convergencia de orden $O(h^4)$ y su implementación es aunque un poco más compleja en la teoría, sigue siendo eficiente cuando hablamos computacionalmente, la podremos ver en el siguiente código que aplicando el método se tomó 0,003878 segundos.

```
1  format rational
2  % Creamos la función f
3  f = @(x) 4 ./ (1 + x.^2);
4  % Parámetros de la integral y el método
5  a = 0;
6  b = 1;
7  valores_n = [10, 50, 100, 250, 500, 1000, 1500, 2000];
8  I_exacto = pi;
9  % Se crea un vector para guardar los errores
10 errores_simpson = zeros(size(valores_n));
11 % Método de Simpson
12 tic;
13 for k = 1:length(valores_n)
14     n = valores_n(k);
15     % Aquí nos aseguramos que n sea par (Simpson necesita n
16     % par)
17     if mod(n, 2) == 1
18         n = n + 1;
19     end
20     h = (b - a) / n;
21     % Extremos de la partición
22     x_extremos = a:h:b;
23     m = length(x_extremos);
24     % Aplicar la regla de Simpson compuesta
25     I_simpson = (h/3) * (f(a) + 4*sum(f(x_extremos(2:2:m-1)))
26     + 2*sum(f(x_extremos(3:2:m-2))) + f(b));
27     % Calculo del error
28     errores_simpson(k) = abs(I_simpson - I_exacto);
29 end
30 tiempo = toc
31 % Graficar error del método de Simpson
32 figure;
33 loglog(valores_n, errores_simpson, '-o', 'LineWidth', 2);
34 xlabel('n');
35 ylabel('Error absoluto');
36 ylim([1e-16, 1e-2]);
37 title('Error absoluto vs número de subintervalos (Simpson)');
38 grid on;
```

Ahora sería interesante ver la comparación entre los 3 métodos en la siguiente figura y la tabla de tiempos:

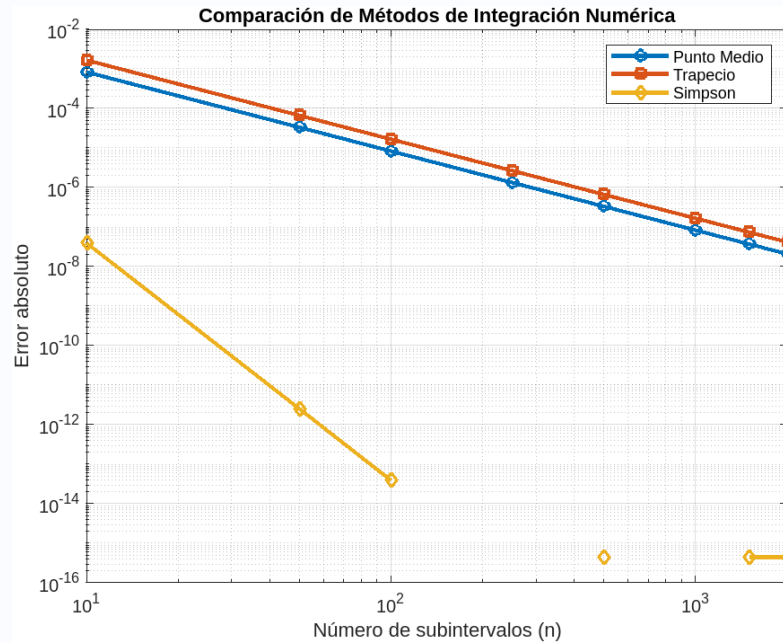


Figura 4: Comparación de los métodos.

Método	Tiempo (s)
Punto medio	0,001995
Trapecio	0,002719
Simpson	0,003878

De lo que podemos concluir lo siguiente:

- En la aplicación el método del punto medio aparenta ser superior al método del trapecio en cuestiones de errores y tiempo, no obstante la diferencia entre estos aparenta ser bastante baja para tenerlo en cuenta a la hora de realizar los cálculos y favorecer alguna de las 2, por lo que no se descarta que el uso del método del trapecio sea más efectivo en otro tipos de problemas en los que quizás sea mejor considerar los términos de borde en la función.
- Es claro que el método de Simpson es mucho más preciso al momento de aproximar el resultado, pues, desde la propia teoría (la convergencia en orden $O(h^4)$) vimos que al refinar la malla el error se reduciría bastante rápido, por lo que vemos que rápidamente con tomar $n = 250$ ya este había superado la precisión numérica de

maquina al ser tan cercano a 0 el error absoluto, situación que con los otros 2 métodos no se alcanzó a concluir de igual forma.

- Si bien en cuestión de tiempo vemos que aún podríamos permitirles a los métodos tener un refinamiento de malla aún más grande, es claro que al método de Simpson le fue suficiente con muy poco y que a los restantes 2 métodos aparenta aún faltarle mucho al refinamiento de la malla para alcanzar la precisión numérica que tiene el método de Simpson.

2. Implemente el método de integración de Romberg para calcular I . Grafique el logaritmo del error en los términos diagonales en la tabla de extrapolación versus $\log h$. Verifique sus resultados con la teoría.