Integración numérica

Tema 2 - Operaciones matemáticas básicas

M. en C. Gustavo Contreras Mayén

Facultad de Ciencias - UNAM

8 de marzo de 2018

Problema inicial

Dada la función $f(x)$, queremos calcular

$$\int_a^b f(x)dx$$

Introducción

La integración numérica (también conocida como cuadratura) es un procedimiento con mayor precisión que la diferenciación numérica.

Introducción

La cuadratura aproxima la integral definida

$$\int_a^b f(x)dx$$

mediante la suma

$$I = \sum_{i=0}^{n} A_i f(x_i)$$

donde las *abscisas nodales* x_i y los pesos A_i dependen de una regla en particular usada para la cuadratura.

Clasificación de las cuadraturas

Todas las reglas de cuadratura se dividen en dos grupos:

1. Fórmulas de Newton-Cotes.
2. Fórmulas de Cuadraturas Gaussianas.

Fórmulas de Newton-Cotes

Estas fórmulas se caracterizan por usar un espaciamiento uniforme y constante en las abscisas, aquí se consideran los métodos del trapecio y la regla de Simpson.

Fórmulas de Newton-Cotes

Son útiles si $f(x)$ se ha evaluado en intervalos iguales; dado que las fórmulas Newton-Cotes se basan en una interpolación local, se requiere de una porción del dominio para ajustarla al polinomio.

Fórmulas de Newton-Cotes

Consideremos la integral definida

$$\int_a^b f(x)dx$$

Dividimos el intervalo de integración $[a, b]$ en n intervalos de igual longitud $h = (b - a)/n$, y hacemos que las abscisas sean $x_0, x_1, \ldots, x_n$.

Aproximación polinomial de $f(x)$

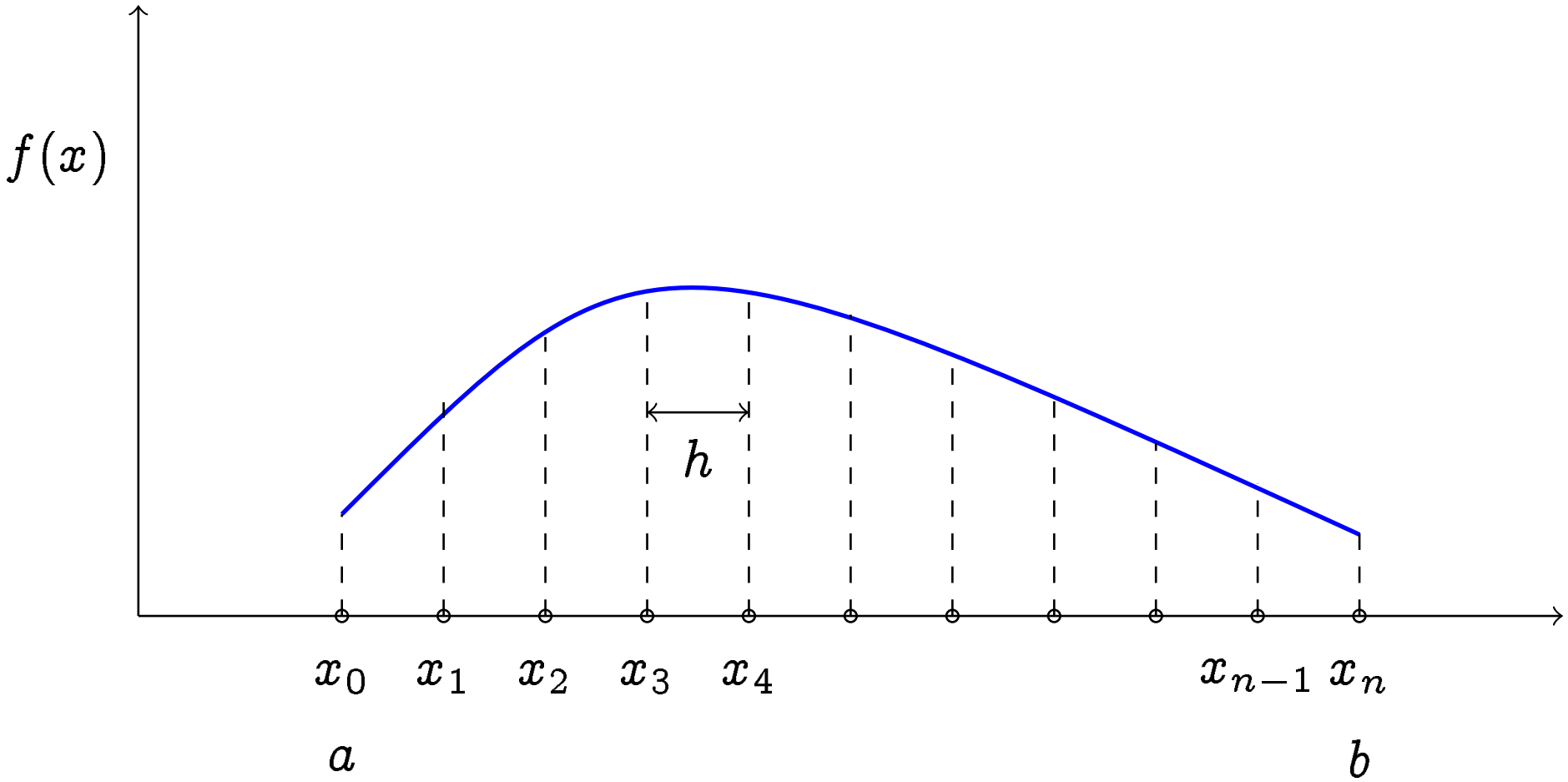


Figura 1: *Aproximación polinomial para la función.*

Aproximación polinomial de $f(x)$

Ahora aproximamos $f(x)$ con un polinomio de orden n que intersecta todos los nodos. La expresión para el polinomio de Lagrange es:

$$P_n(x) = \sum_{i=0}^{n} f(x_i)\mathcal{L}_i(x)$$

donde $\mathcal{L}_i(x)$ son las funciones definidas en el tema de interpolación.

Aproximación polinomial de $f(x)$

Por tanto, un aproximación a la integral es

$$I = \int_a^b P_n(x)dx = \sum_{i=0}^{n} \left[f(x_i) \int_a^b \mathcal{L}_i(x)dx \right] = \sum_{i=0}^{n} A_i f(x_i)$$

donde

$$A_i = \int_a^b \mathcal{L}_i dx, \qquad i = 0, 1, \ldots, n$$

Fórmulas de Newton-Cotes

Las ecuaciones

$$I = \int_a^b P_n(x)dx = \sum_{i=0}^{n} \left[f(x_i) \int_a^b \mathcal{L}_i(x)dx \right] = \sum_{i=0}^{n} A_i f(x_i)$$

Fórmulas de Newton-Cotes

Las ecuaciones

$$I = \int_a^b P_n(x)dx = \sum_{i=0}^{n} \left[f(x_i) \int_a^b \mathcal{L}_i(x)dx \right] = \sum_{i=0}^{n} A_i f(x_i)$$

se conocen como las fórmulas de Newton-Cotes.
Siendo los casos:

1. $n = 1$, Regla del trapecio.

Fórmulas de Newton-Cotes

Las ecuaciones

$$I = \int_a^b P_n(x)dx = \sum_{i=0}^{n} \left[f(x_i) \int_a^b \mathcal{L}_i(x)dx \right] = \sum_{i=0}^{n} A_i f(x_i)$$

se conocen como las fórmulas de Newton-Cotes.
Siendo los casos:

1. $n = 1$, Regla del trapecio.
2. $n = 2$, Regla de Simpson.

Fórmulas de Newton-Cotes

Las ecuaciones

$$I = \int_a^b P_n(x)dx = \sum_{i=0}^{n} \left[f(x_i) \int_a^b \mathcal{L}_i(x)dx \right] = \sum_{i=0}^{n} A_i f(x_i)$$

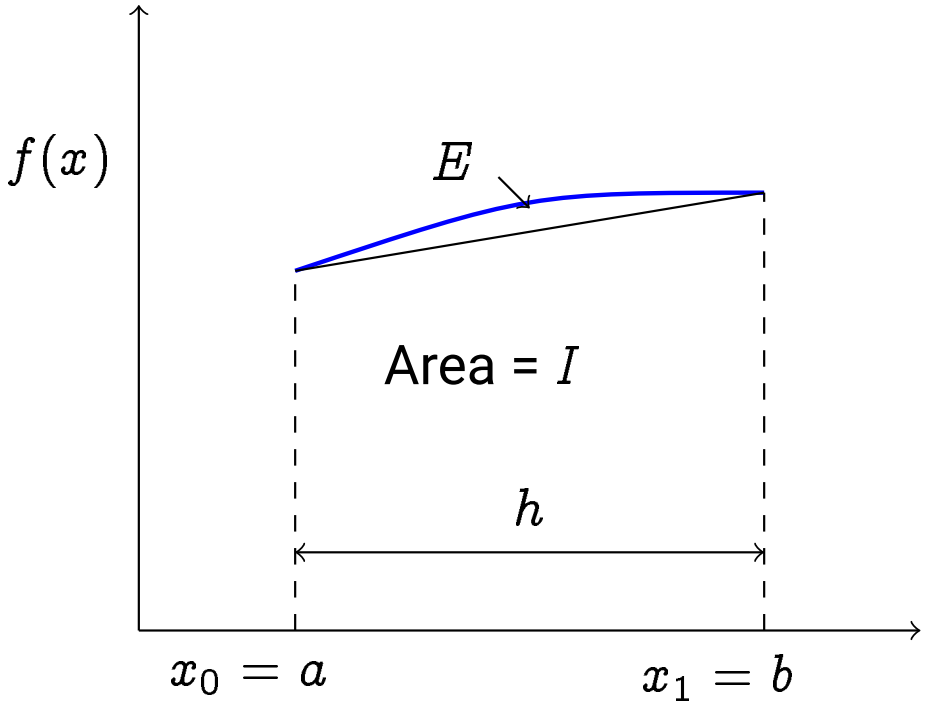
se conocen como las fórmulas de Newton-Cotes.
Siendo los casos:

1. $n = 1$, Regla del trapecio.
2. $n = 2$, Regla de Simpson.
3. $n = 3$, Regla de Simpson de $3/8$.

Fórmulas de Newton-Cotes

La más importante es la regla del trapecio, ya que se puede combinar con la extrapolación de Richardson, en un algoritmo eficiente llamado: Integración de Romberg.

Regla del trapecio



Si $n = 1$ (un bloque), tenemos que
$l_0 = (x - x_1)/(x_0 - x_1) = (x - b)/h$ por tanto:

$$A_0 = \frac{1}{h}\int_a^b (x-b)dx = \frac{1}{2h}(b-a)^2 = \frac{h}{2}$$

Regla del trapecio

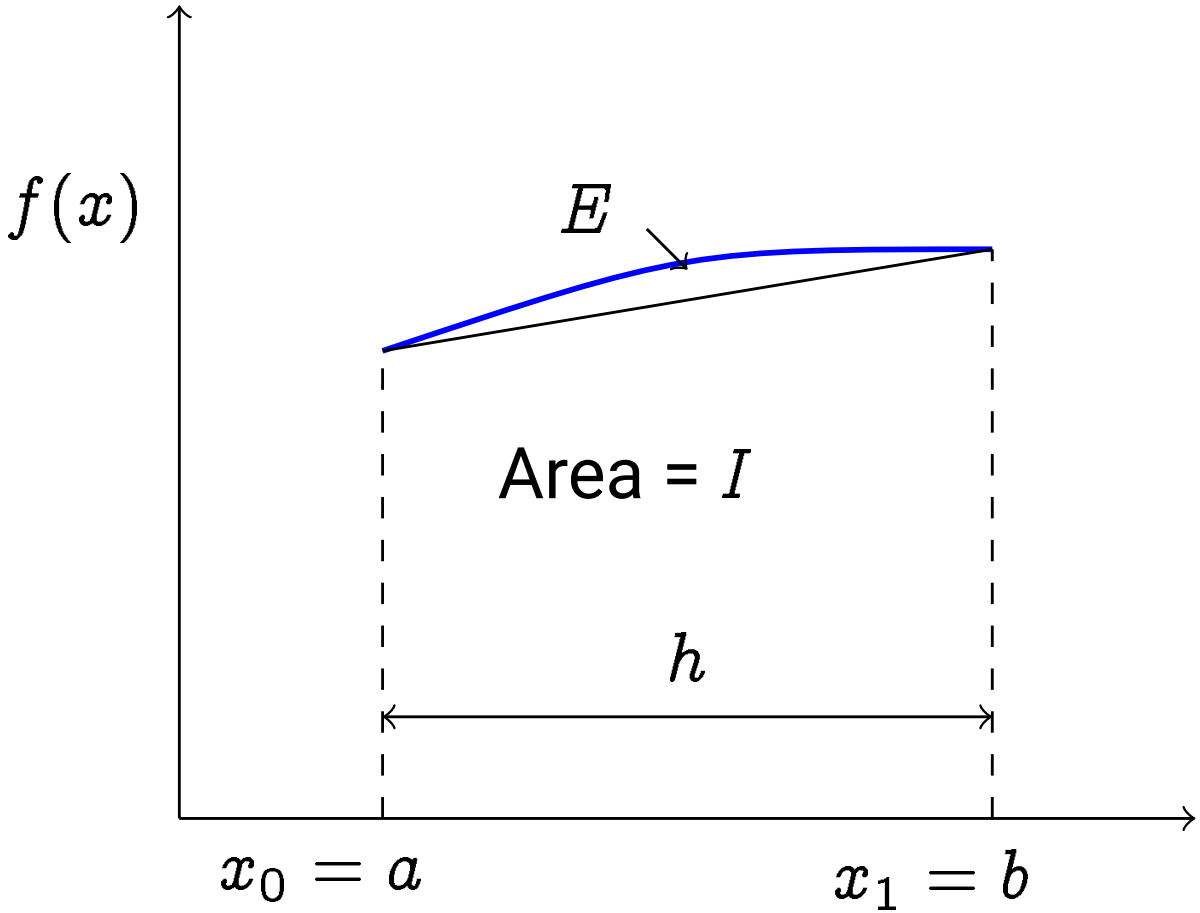
Para $l_1 = (x - x_0)/(x_1 - x_0) = (x - a)/h$ tenemos

$$A_1 = \frac{1}{h}\int_a^b (x-a)dx = \frac{1}{2h}(b-a)^2 = \frac{h}{2}$$

Sustituyendo:

$$I = [f(a) + f(b)]\,\frac{h}{2}$$

Regla del trapecio



Que resulta ser la regla del trapecio, y representa el área del trapecio que se muestra en la figura anterior.

Error en la regla del trapecio

El error viene dado por

$$E = \int_a^b f(x)dx - I$$

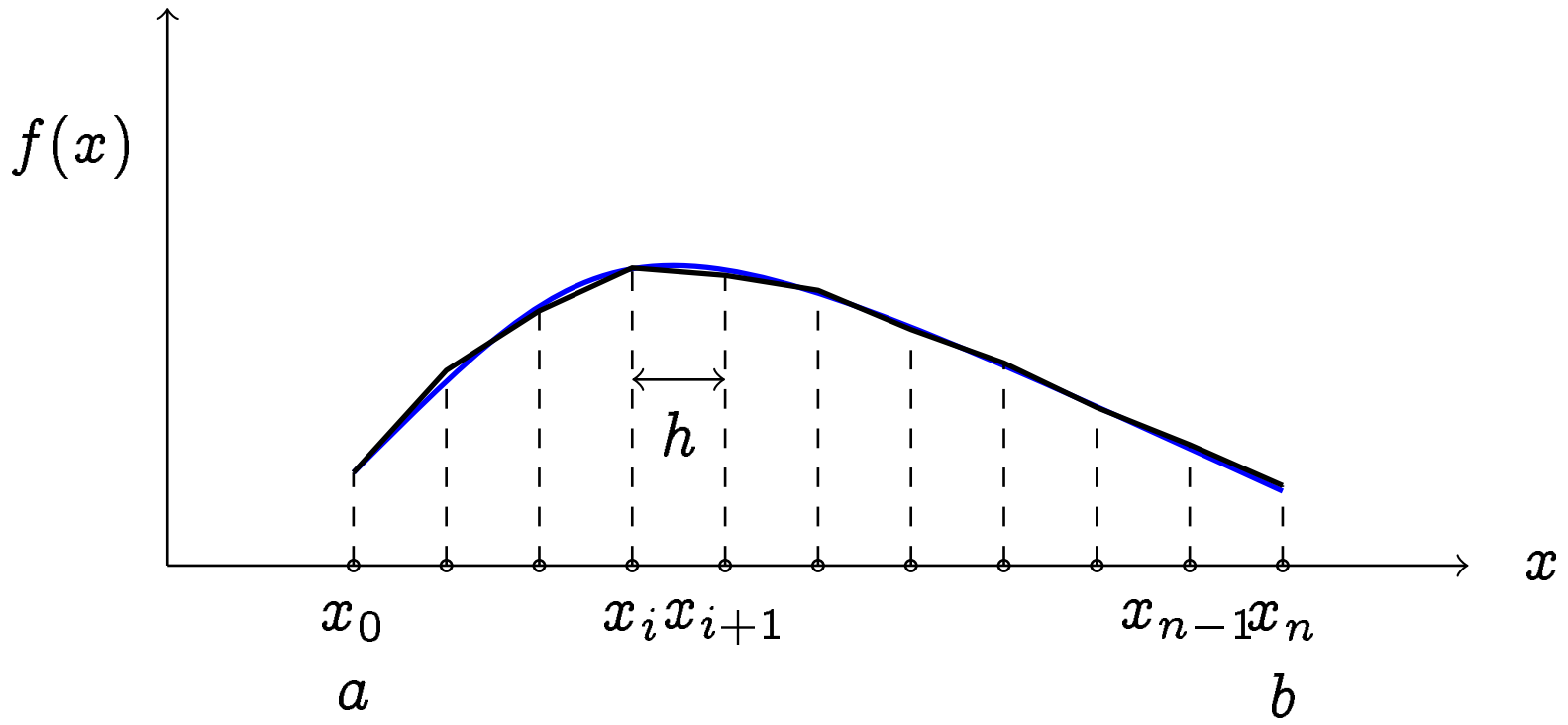
que es diferencia entre el área debajo de la curva de $f(x)$ y el la integral obtenida.

Integrando el error de interpolación:

$$\begin{aligned} E =& \frac{1}{2!} \int_a^b (x - x_0)(x - x_1)\, f''(\xi)\, dx \\ =& \frac{1}{2} f''(\xi) \int_a^b (x - a)(x - b)\, dx = \\ =& -\frac{1}{12}(b - a)^3 f''(\xi) \\ =& -\frac{h^3}{12} f''(\xi) \end{aligned}$$

Regla extendida del trapecio

En la práctica la regla del trapecio se usa con una división en el dominio. La siguiente figura muestra la región $[a, b]$ dividida en n bloques, de longitud h.



Regla extendida del trapecio

La función $f(x)$ se integrará con una aproximación lineal en cada panel.

De la regla del trapecio, sabemos que para el i-ésimo panel:

$$I_i = [f(x_i) + f(x_{i+1})]\frac{h}{2}$$

Regla extendida del trapecio

El área total queda representada por la integral:

$$I \simeq \sum_{i=0}^{n-1} [f(x_0) + 2f(x_1) + 2f(x_2) + \ldots + 2f(x_{n-1}) + f(x_n)] \frac{h}{2}$$

que es la regla del extendida del trapecio.

Regla recursiva del trapecio

Sea I_k la integral evaluada con la regla compuesta del trapecio, usando 2^{k-1} bloques. Con la notación $H = b - a$,de la regla compuesta del trapecio, para $k = 1, 2, 3$

$$k = 1 \text{ (1 bloque)} :$$

$$I_1 = [f(a) + f(b)]\frac{H}{2}$$

Regla recursiva del trapecio

$k = 2$ (2 bloques) :

$$I_2 = \left[f(a) + 2\, f\left(a + \frac{H}{2}\right) + f(b) \right] \frac{H}{4}$$

$$= \frac{1}{2}\, I_1 + f\left(a + \frac{H}{2}\right)\, \frac{H}{2}$$

Regla recursiva del trapecio

$$k = 3 \text{ (4 bloques)} :$$

$$I_3 = \left[f(a) + 2\, f\left(a + \frac{H}{4}\right) + 2\, f\left(a + \frac{H}{2}\right) + \right.$$
$$\left. + 2\, f\left(a + \frac{3\,H}{4}\right) + f(b) \right] \frac{H}{8}$$
$$= \frac{1}{2}\, I_2 \left[f\left(a + \frac{H}{4}\right) + f\left(a + \frac{3\,H}{4}\right) \right] \frac{H}{4}$$

Regla recursiva del trapecio

Para un $k > 1$ arbitrario, tenemos

$$I_k = \frac{1}{2} I_{k-1} + \frac{H}{2^{k-1}} \sum_{i=1}^{2^{k-2}} f\left[a + \frac{(2\,i-1)H}{2^{k-1}}\right], \qquad k = 2, 3, \ldots$$

Otra forma de la misma ecuación es:

$$I(h) = \frac{1}{2}\, I(2\,h) + h \sum f(x_{\text{nuevo}})$$

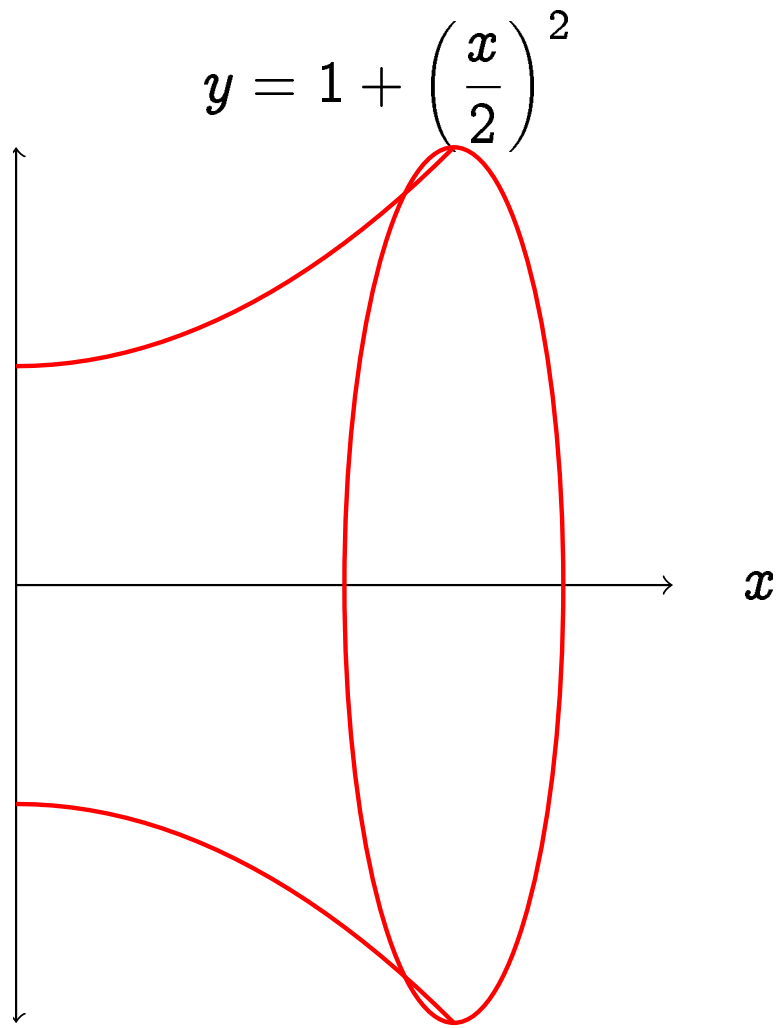
Ejercicio

El cuerpo de revolución que se muestra en la siguiente figura, se obtiene al girar la curva dada por

$$y = 1 + \left(\frac{x}{2}\right)^2, \quad 0 \leq x \leq 2$$

en torno al eje x.

Figura para el ejercicio



Ejercicio

Calcula el volumen del sólido, usando la regla extendida del trapecio con $N = 2, 4, 8, 16, 32, 64, 128$.

El valor exacto del volumen es es $I = 11.7286$. Evalúa el error para cada N.

Resolviendo el problema

Hay que definir inicialmente la función que queremos integrar, por tanto

$$I = \int_a^b f(x)dx$$

donde

$$f(x) = \pi \left(1 + \left(\frac{x}{2}\right)^2\right)^2$$

Código 1: Código para la función trapecios

```
1  def trapecios(f, a, b, n):
2      h = (b - a)/float(n)
3      x = a
4      suma = 0
5
6      for i in range(1, n):
7          x = x + h
8          suma = suma + funcion(x)
9
10     return (h/2.) * (funcion(a) + funcion
       (b) + 2 * suma)
```

Declaramos la función

Código 2: Código para la función trapecios

```
1 def funcion(x):
2     return pi * (1 + (x/2)**2)**2
```

Nota: debes de usar adecuadamente la librería para el manejo de `pi`, es decir, usa la notación `libreria.pi` donde `libreria.` puede ser **`math`**, **`numpy`** o **`scipy`**.

Ejercicio completo

Código 3: Completamos el código y evaluamos el error

```
1 for i in range(1, 8):
2     n = 2**i
3     integral = trapecios(funcion(i), 0,
    2, n)
4     print ('{:3d} \t {:2.9f} \t  {:1.8e}
    '.format(n, integral, \
5                         (100*(abs(11.7286 -
    integral))/11.7286)))
```

Observación

Nótese que la manera en que incrementamos el valor de N, es más dinámica: si aumentamos el número de elementos, tendríamos que ajustar “a mano” el contenido de una lista.

Hacerlo con un incremento en la potencia, nos permite entonces, manejar cualquier cambio en el número de elementos sin problema ni ajustes manuales.

i	Integral	Error
2	12.762720155	$8.81708094e-02$
4	11.989593838	$2.22527700e-02$
8	11.794011288	$5.57707550e-03$
16	11.744971839	$1.39589033e-03$
32	11.732702989	$3.49827695e-04$
64	11.729635215	$8.82641387e-05$
128	11.728868236	$2.28702562e-05$

Librería Scipy

`SciPy` (Scientific Python) es una librería matemática con funciones que extienden la librería **`numpy`** para `python`.

Se le proporciona al usuario un poder significativo mediante el uso de comandos de alto nivel y clases, para la manipulación y visualización de datos.

Organización de Scipy

La librería **SciPy** está organizada en sub-paquetes que cubren diferentes áreas de computación científica. Estos se resumen en la siguiente tabla:

Subpaquete	Descripción
cluster	Algortimos para clusters
constants	Constantes físicas y matemáticas
fftpack	Rutinas para la Transformada Rápida de Fourier
integrate	Integración y EDO
interpolate	Interpolación y uso de splines
io	Rutinas de entrada y salida

Subpaquete	Descripción
linalg	Algebra lineal
ndimage	Procesamiento N-dimensional de imagenes
odr	Regresión de distancias ortogonales
optimize	Optimizació y rutinas para encontrar raíces
signal	Procesamiento de señales
sparse	Matrices sparse y rutinas asociadas
spatial	Estructura de datos espaciales
special	Funciones especiales
stats	Distribuciones estadísticas
weave	Integración con C/C++

Integración (`scipy.integrate`)

El subpaquete **`scipy.integrate`** proporciona varias técnicas de integración.

quad	Integración en general.
dblquad	Integración doble en general.
tplquad	Integración triple en general.
fixed-quad	Integración de f(x) usando cuadraturas gaussianas de orden n.
quadrature	Integra con tolerancia dada usando cuadratura gaussiana.
romberg	Integra una función mediante la integración de Romberg.

Usando `integrate.quad`

Para comparar el resultado que nos devuelve la función `arg`scipy.integrate.quad, veamos cómo implementar la solución del problema del sólido de revolución.

Usando `integrate.quad`

Código 4: Integración con scipy

```
1 from numpy import pi
2 from scipy.integrate import quad
3
4 def f(x):
5     return pi*(1 + (x/2)**2)**2
6
7 print(quad(f, 0, 2))
```

Usando `integrate.quad`

Código 5: Integración con scipy

```
1 from numpy import pi
2 from scipy.integrate import quad
3
4 def f(x):
5     return pi*(1 + (x/2)**2)**2
6
7 print(quad(f, 0, 2))
```

El resultado que nos devuelve es:
$(11.7286125734018 93, 1.302137572589889e - 13)$.

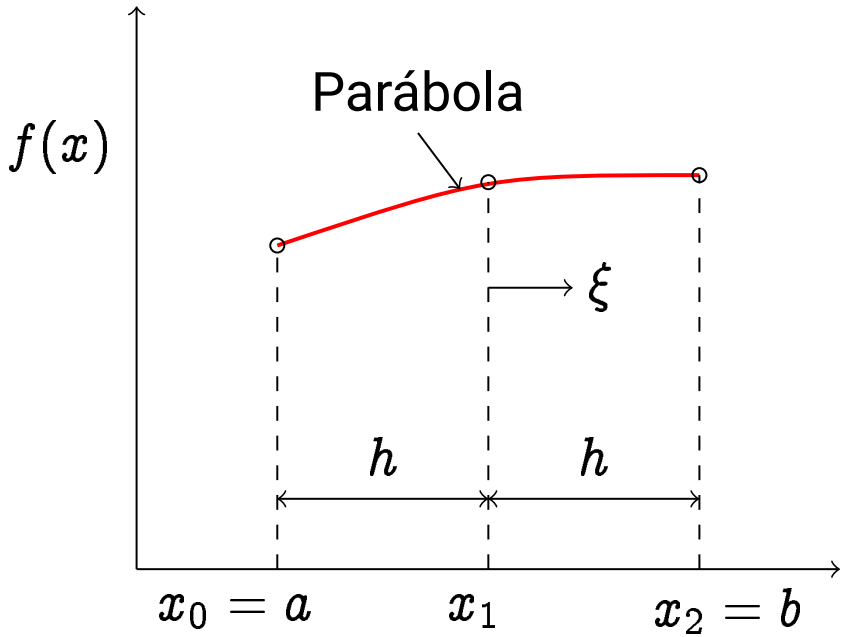
Usando `integrate.quad`

El valor posterior al resultado de la integral es el error asociado al algoritmo que usa **`integrate.quad`**, para que no lo reporte en el resultado, basta con indicar que queremos sólo el primer elemento de la lista:

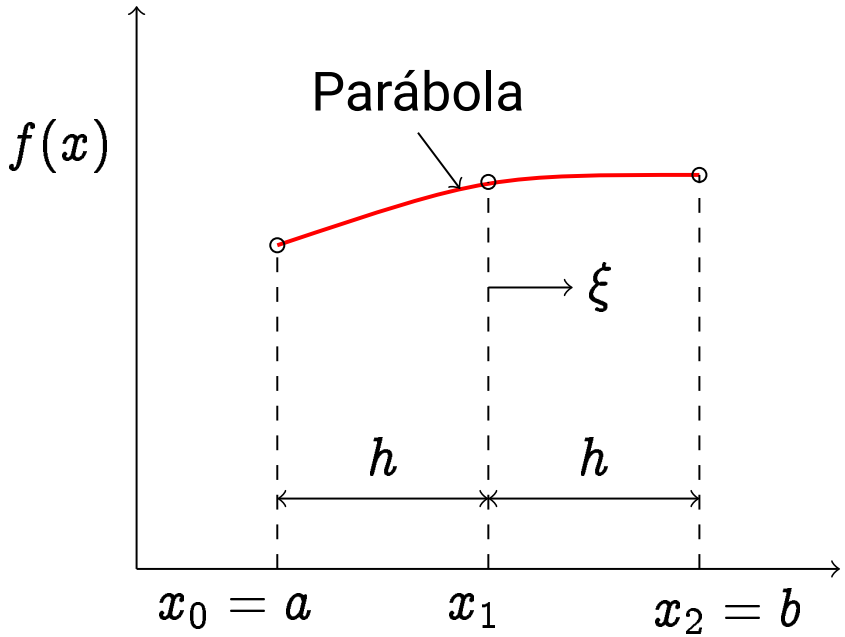
```
print(quad(f, 0, 2)[0])
```

Regla de $1/3$ de Simpson

La regla de $1/3$ de Simpson se obtiene de las fórmulas de Newton-Cotes con $n = 2$, es decir, haciendo una interpolación con una parábola a través de tres nodos, como se muestra en la siguiente figura:



Regla de 1/3 de Simpson



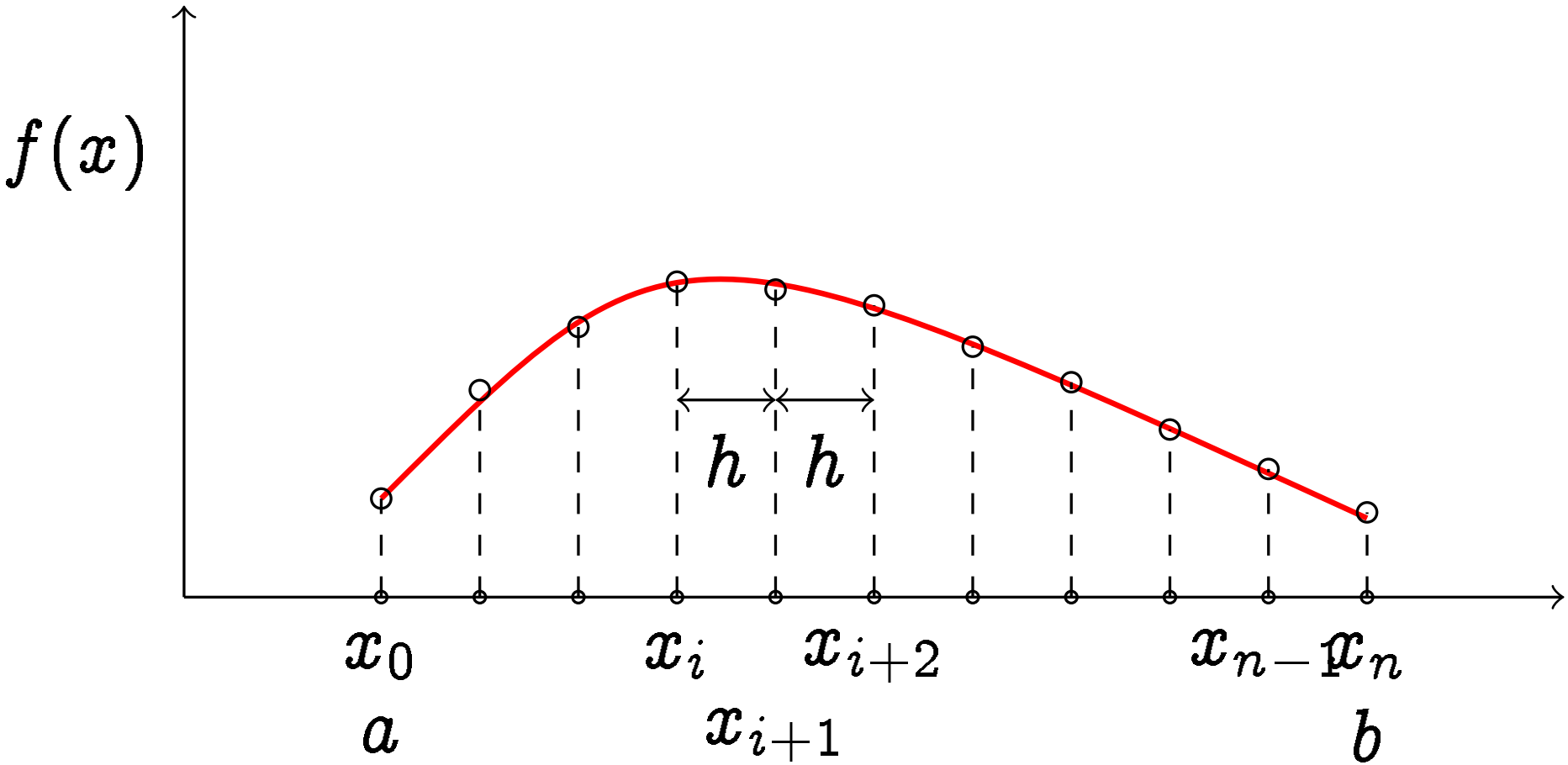
El área debajo de la curva representa una aproximación a la integral $\int_a^b f(x)$:

$$I = \left[f(a) + 4\, f\left(\frac{a+b}{2} \right) + f(b) \right] \frac{h}{3}$$

Regla compuesta de $1/3$ de Simpson

Para obtener la regla compuesta de $1/3$ de Simpson, se divide el intervalo de integración $[a, b]$ en n bloques (n par) de ancho $h = (b - a)/n$

Regla compuesta de $1/3$ de Simpson



Regla compuesta de $1/3$ de Simpson

Aplicando la fórmula anterior a dos bloques adyacentes, tenemos:

$$\int_{x_i}^{x_{i+2}} f(x)\,dx \simeq \left[f(x_i) + 4\,f(x_{i+1}) + f(x_{i+2})\right]\,\frac{h}{3}$$

sustituyendo la ecuación a todo el intervalo

$$\int_a^b f(x)\,dx = \int_{x_0}^{x_m} f(x)\,dx = \sum_{i=0,2,\ldots}^{n} \left[\int_{x_i}^{x_{i+2}} f(x)\,dx\right]$$

Regla compuesta de $1/3$ de Simpson

Por lo que la aproximación a la integral resulta ser:

$$\int_a^b f(x)dx \simeq I = [f(x_0) + 4\, f(x_1) + 2\, f(x_2) + \ldots$$

$$\ldots + 2\, f(x_{n-2}) + 4\, f(x_{n-1}) + f(x_n)]\ \frac{h}{3}$$

Regla compuesta de $1/3$ de Simpson

Por lo que la aproximación a la integral resulta ser:

$$\int_a^b f(x)dx \simeq I = [f(x_0) + 4\, f(x_1) + 2\, f(x_2) + \ldots$$
$$\ldots + 2\, f(x_{n-2}) + 4\, f(x_{n-1}) + f(x_n)]\ \frac{h}{3}$$

y es quizás el método más conocido de integración numérica.

Regla compuesta de $1/3$ de Simpson

Aunque su reputación es algo inmerecido, ya que la regla del trapecio es más robusta, y la integración de Romberg es más eficiente.

El error en la regla de $1/3$ de Simpson

El error en la regla compuesta de Simpson viene dado por:

$$E = \frac{(b-a)h^4}{180} f^{(4)}(\xi)$$

de donde inferimos que la integral obtenida por el método, es exacta si el polinomio es de grado tres o menor.

Regla de $3/8$ de Simpson

La regla de $1/3$ de Simpson necesita que el número de bloques n sea par.

Si la condición no se cumple, podemos integrar sobre los primeros (o últimos) tres bloques con la regla de $3/8$ de Simpson:

$$I = [f(x_0) + 3\, f(x_1) + 3\, f(x_2) + f(x_3)]\frac{3\, h}{8}$$

y aplicar la regla de $1/3$ de Simpson en los bloques restantes.

Ejemplo

Estimar la integral

$$\int_0^{2.5} f(x)dx$$

a partir de los siguientes datos:

x	0	0.5	1.0	1.5	2.0	2.5
$f(x)$	1.5000	2.0000	2.0000	1.6364	1.25000	0.9565

Solución

Usaremos las reglas de Simpson:

1. Dado que el número de bloques es impar, calculamos la integral sobre los primeros tres bloques con la regla $3/8$ de Simpson.

Solución

Usaremos las reglas de Simpson:

1. Dado que el número de bloques es impar, calculamos la integral sobre los primeros tres bloques con la regla $3/8$ de Simpson.
2. Usamos la regla de $1/3$ de Simpson en los dos últimos bloques.

Solución

Usaremos las reglas de Simpson:

1. Dado que el número de bloques es impar, calculamos la integral sobre los primeros tres bloques con la regla $3/8$ de Simpson.
2. Usamos la regla de $1/3$ de Simpson en los dos últimos bloques.

$$I = [f(0) + 3f(0.5) + 3f(1.0) + f(1.5)]\frac{3(0.5)}{8}$$

Solución

Usaremos las reglas de Simpson:

1. Dado que el número de bloques es impar, calculamos la integral sobre los primeros tres bloques con la regla $3/8$ de Simpson.
2. Usamos la regla de $1/3$ de Simpson en los dos últimos bloques.

$$I = [f(0) + 3f(0.5) + 3f(1.0) + f(1.5)]\frac{3(0.5)}{8}$$
$$+ [f(1.5) + 4f(2.0) + f(2.5)]\frac{0.5}{3}$$

Solución

Usaremos las reglas de Simpson:

1. Dado que el número de bloques es impar, calculamos la integral sobre los primeros tres bloques con la regla $3/8$ de Simpson.
2. Usamos la regla de $1/3$ de Simpson en los dos últimos bloques.

$$
\begin{aligned}
I =& [f(0) + 3f(0.5) + 3f(1.0) + f(1.5)]\frac{3(0.5)}{8} \\
&+ [f(1.5) + 4f(2.0) + f(2.5)]\frac{0.5}{3} \\
=& 2.8381 + 1.2655 = 4.1036
\end{aligned}
$$

Ejercicio

Evalúa la integral

$$\int_{-1}^{1} \cos(2\cos^{-1} x)dx$$

con la regla de Simpson de $1/3$ usando 2, 4 y 6 bloques.

Explica tus resultados.

Gráfica de la función

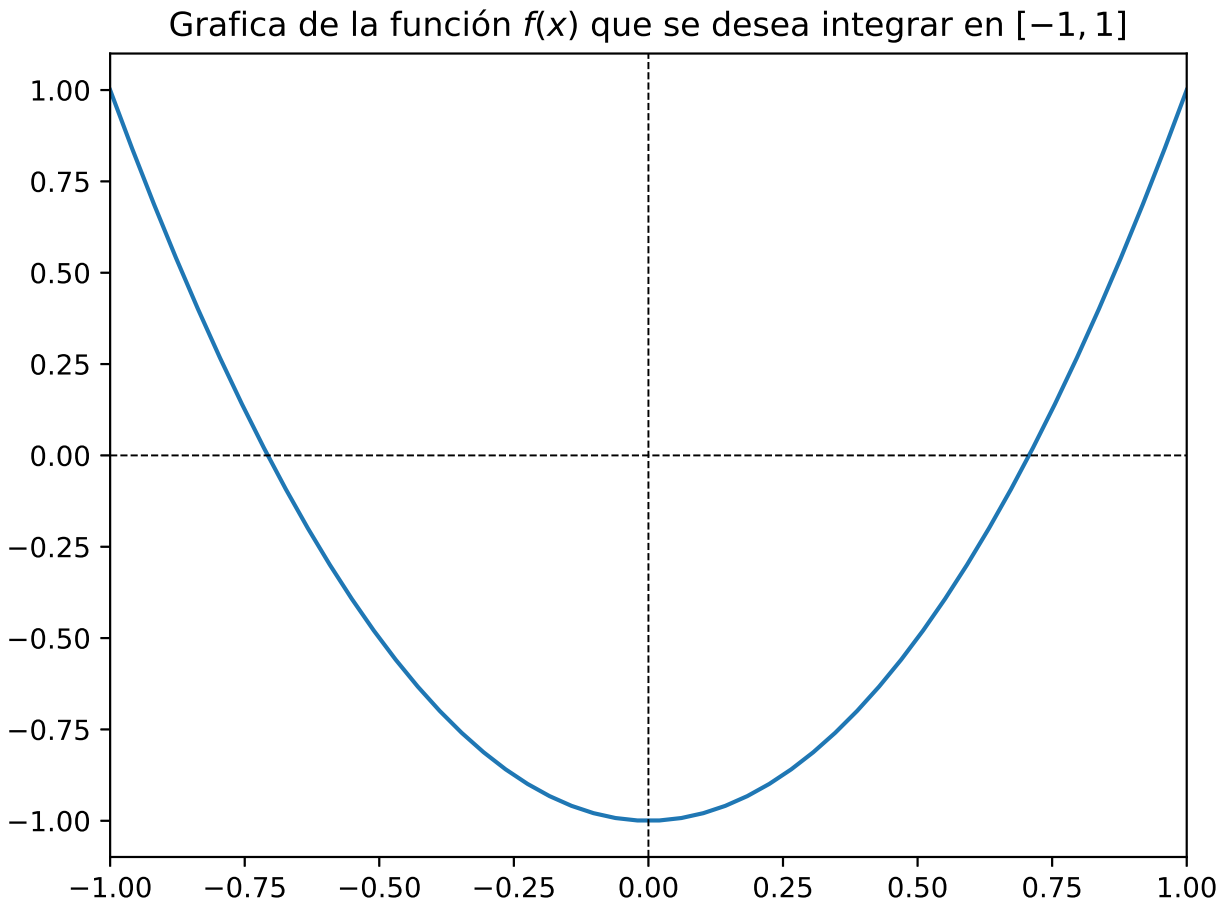


Figura 2: *Queremos calcular el valor del área debajo de la curva.*

Código 6: Propuesta de código

```
1  def Simpson13(f, x0, xf, n):
2
3      n = n - n%2 # truncar al numero par
       mas cercano
4      print ('Para n= ' + str(n))
5
6      if n <= 0:
7          n = 1
8
9      h = (xf - x0)/n
10     x = x0
11
12     suma = 0
```

```
13
14      for j in range(int(n/2)):
15          suma += f(x) + 4. * f(x + h) + f(x
        + 2 * h)
16          x += 2 * h
17      return (h/3.) * suma
```

Código 7: Solución al problema

```
1 def f(x):
2     return cos(2 * acos(x))
3
4 for i in range(1, 4):
5     print ('La integral I vale = ' + str
    (Simpson13(f, -1., 1., i * 2)) + '\n')
```

Solución en la terminal

```
Para n = 2
La integral I vale = -0.66666666666666

Para n = 4
La integral I vale = -0.66666666666665

Para n = 6
La integral I vale = -0.66666666666667
```