

Integración numérica

- [Fórmulas de cuadratura](#)
 - [Ejercicio 1 plot](#)
- [Fórmulas de cuadratura de Newton-Cotes simples](#)
 - [Ejercicio 1a](#)
 - [Ejercicio 1b](#)
 - [Ejercicio 1c](#)
- [Fórmulas de cuadratura de Newton-Cotes compuestas](#)
 - [Ejercicio 1d](#)
 - [Ejercicio 1e](#)
 - [Ejercicio 1f](#)
- [Fórmulas de cuadratura gaussianas](#)
 - [Ejercicio 2](#)
- [Grado de precisión de las fórmulas de cuadratura](#)
 - [Ejercicio 3](#)
- [Ejercicios propuestos](#)
 - [Integración numérica con órdenes python](#)
 - [Integración de Montecarlo](#)

Fórmulas de cuadratura

Las fórmulas de integración numérica o de cuadratura son de la forma:

$$\int_a^b f(x) dx \approx \omega_0 f(x_0) + \omega_1 f(x_1) + \cdots + \omega_N f(x_N)$$

donde x_0, x_1, \dots, x_N (nodos) son $N + 1$ puntos distintos pertenecientes al intervalo $[a, b]$ y $\omega_0, \omega_1, \dots, \omega_N$ (pesos) son números reales.

Fórmulas interpolatorias

Si P_N es el polinomio que interpola a f en los puntos distintos $x_0, x_1, \dots, x_N \in [a, b]$ y

$$\int_a^b f(x) dx \approx \int_a^b P_N(x) dx = \omega_0 f(x_0) + \omega_1 f(x_1) + \cdots + \omega_N f(x_N)$$

decimos que la fórmula de cuadratura es de tipo interpolatorio.

Fórmulas de cuadratura simples y compuestas

Las fórmulas de cuadratura se llaman simples si la aproximación se hace en el intervalo completo $[a, b]$, y compuestas si, antes de aplicar la fórmula, dividimos el intervalo $[a, b]$, en n subintervalos.

Grado de precisión

Una fórmula de cuadratura tiene grado de precisión r si es exacta para

$$f(x) = 1, \quad f(x) = x, \quad f(x) = x^2, \dots, \quad f(x) = x^r$$

pero no es exacta para $f(x) = x^{r+1}$

Fórmulas de cuadratura de Newton-Cotes simples

Son fórmulas de cuadratura de tipo interpolatorio, eligiendo los puntos de interpolación (nodos de la fórmula) igualmente separados de una de las dos formas siguientes:

- *Fórmulas cerradas* Los límites de integración a y b son nodos de la fórmula.



- *Fórmulas abiertas* Ninguno de los límites de integración es nodo de la fórmula.



Ejercicio 1 plot

Escribir la función `dibujo(f,a,b,nodos)` que tiene como argumentos de entrada la función lambda `f` a integrar, los extremos del intervalo de integración `a` y `b`, y los nodos con los que vamos a construir la fórmula interpolatoria en un array numpy `nodos` y que dibuja

- El área calculada por la integral exacta.
- Los nodos.
- El área aproximada calculada por la fórmula interpolatoria que usa los nodos.

Probar la fórmula con los siguientes datos:

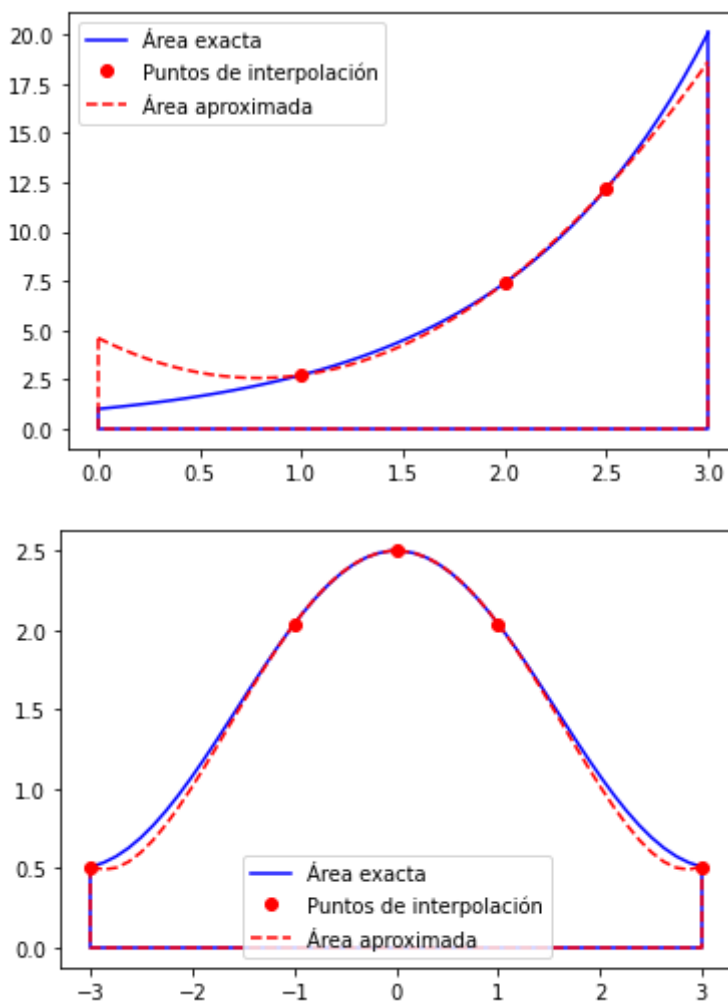
1. $f(x) = e^x$, $[a, b] = [0, 3]$ y `nodos = np.array([1,2,2.5])`
2. $f(x) = \cos(x) + 1.5$, $[a, b] = [-3, 3]$ y `nodos = np.array([-3., -1, 0, 1, 3])`

Nota

- Si queremos dibujar una poligonal que une tres puntos (x_0, y_0) , (x_1, y_1) y (x_2, y_2) con una línea roja discontinua

```
plt.plot([x0,x1,x2],[y0,y1,y2], 'r--')
```

```
%run Ejercicio1plot
```



Calculemos primero una integral de forma exacta (simbólica) para comparar los sucesivos resultados numéricos

Cargamos los paquetes `numpy` y `sympy`

```
import numpy as np
import sympy as sym
```

La integral

$$\int_1^3 \ln(x) dx \quad (1)$$

se calcula de forma exacta

```
x = sym.Symbol('x', real=True)
f_sim = sym.log(x)
I_exacta = sym.integrate(f_sim,(x,1,3))
print(I_exacta)
```

```
-2 + 3*log(3)
```

Que también podemos escribir

```
I_exacta = float(I_exacta)
print(I_exacta)
```

1.2958368660043291

Fórmula del punto medio

La fórmula del punto medio es

$$\int_a^b f(x) dx \approx (b - a) f\left(\frac{a + b}{2}\right)$$

Usar la fórmula del punto medio para integrar una función en un intervalo, equivale a sustituir, dentro de la integral, la función a integrar por el polinomio de interpolación de grado cero, es decir, una recta horizontal que pasa por el punto de la curva que corresponde al punto medio del intervalo $[a, b]$. Sustituimos la función por una recta e integramos. Estamos entonces calculando el área de un rectángulo.

Ejercicio 1a

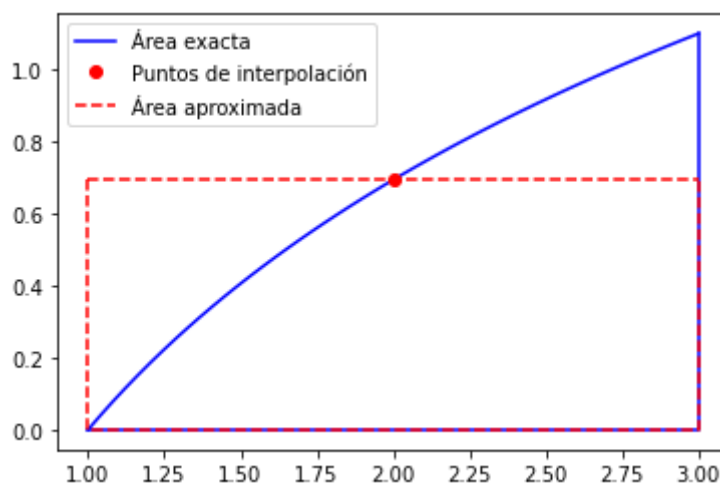
Escribir la función `punto_medio(f,a,b)` que tiene como argumentos de entrada la función `f` a integrar y los extremos del intervalo de integración `a` y `b`, y que devuelve el valor aproximado de la integral utilizando la Regla del Punto Medio.

Calcular la integral

$$\int_1^3 \ln(x) dx$$

Escribir el valor exacto (calculado antes) y el valor aproximado.

```
%run Ejercicio1a.py
```



El valor aproximado es 1.3862943611198906

El valor exacto es 1.2958368660043291

Fórmula de los trapecios

La fórmula de los trapecios simple es

$$\int_a^b f(x)dx \approx \frac{b-a}{2} (f(a) + f(b))$$

Usar la fórmula de los trapecios para integrar una función en un intervalo, equivale a sustituir, dentro de la integral, la función a integrar por el polinomio de interpolación de grado uno, que pasa por los puntos de la función de los extremos del intervalo. Es decir, sustituimos la función por una recta e integramos. Estamos entonces calculando el área de un trapecio.

Ejercicio 1b

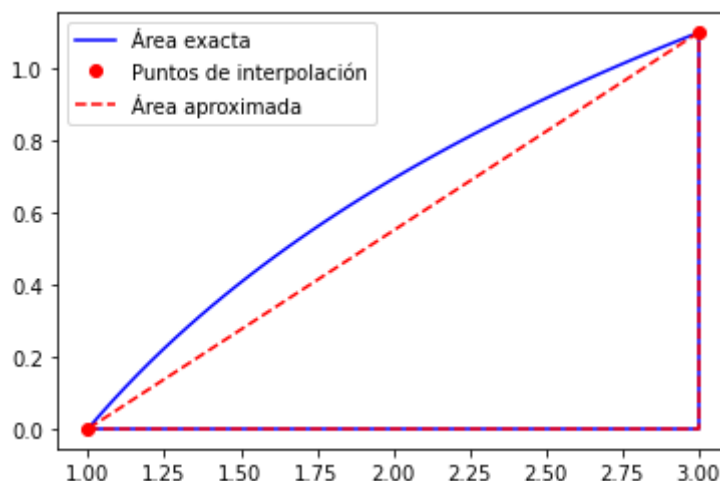
Escribir la función `trapecio(f,a,b)` del ejemplo anterior que tiene como argumentos de entrada la función f a integrar y los extremos del intervalo de integración a y b , y que devuelve el valor aproximado de la integral utilizando la Regla del Trapecio.

Calcular la integral

$$\int_1^3 \ln(x) dx$$

Escribir el valor exacto y el valor aproximado.

```
%run Ejercicio1b.py
```



El valor aproximado es 1.0986122886681098

El valor exacto es 1.2958368660043291

Fórmula de Simpson

La fórmula de Simpson simple es

$$\int_a^b f(x)dx \approx \frac{b-a}{6} \left(f(a) + 4f\left(\frac{a+b}{2}\right) + f(b) \right)$$

Usar la fórmula de Simpson para integrar una función en un intervalo, equivale a sustituir, dentro de la integral, la función a integrar por el polinomio de interpolación de grado dos, que pasa por los puntos de la función de los extremos y el punto medio del intervalo. Es decir, sustituimos la función por una parábola e integramos.

Ejercicio 1c

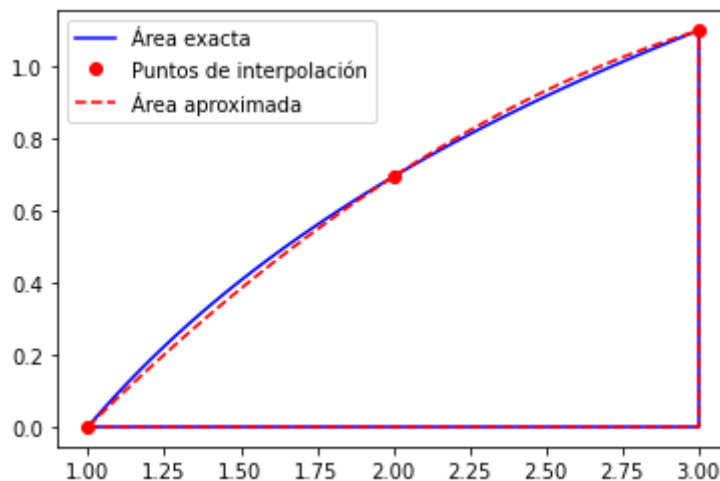
Escribir la función `simpson(f,a,b)` del ejemplo anterior que tiene como argumentos de entrada la función `f` a integrar y los extremos del intervalo de integración `a` y `b`, y que devuelve el valor aproximado de la integral utilizando la Regla del Simpson.

Calcular la integral

$$\int_1^3 \ln(x) dx$$

Escribir el valor exacto y el valor aproximado.

```
%run Ejercicio1c.py
```



El valor aproximado es 1.290400336969297

El valor exacto es 1.2958368660043291

Fórmulas de cuadratura de Newton-Cotes compuestas

Una forma de disminuir el error de las fórmulas anteriores es aumentar el número de nodos utilizando las fórmulas compuestas. Estas se obtienen dividiendo el intervalo $[a, b]$ en n subintervalos y aplicando a cada uno de estos subintervalos una fórmula de cuadratura sencilla.

Si dividimos el intervalo $[a, b]$ en n subintervalos de igual longitud usando los nodos x_0, x_1, \dots, x_n ,

la longitud de un subintervalo es

$$h = \frac{b - a}{n}$$

los nodos son

$$x_i = a + i h \quad i = 0, 1, \dots, n$$

y el punto medio de un intervalo es

$$\bar{x}_i = \frac{x_{i-1} + x_i}{2}$$

entonces, las fórmulas compuestas se pueden escribir:

Fórmula del punto medio compuesta

$$\int_a^b f dx \approx h \sum_{i=1}^n f(\bar{x}_i)$$

Ejercicio 1d

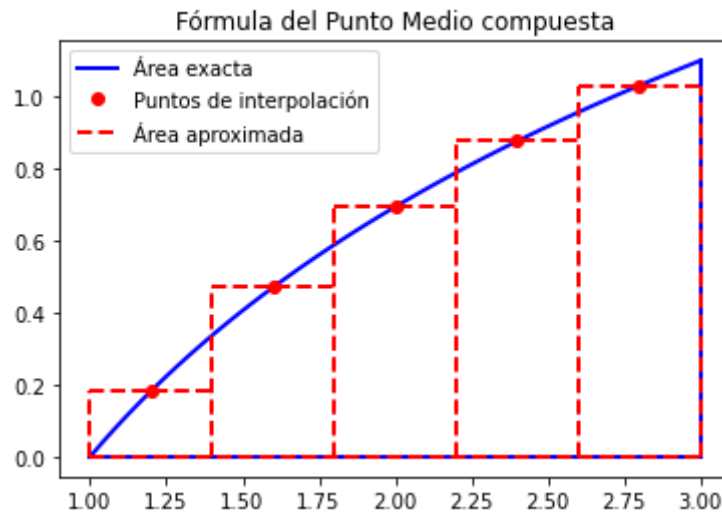
Escribir la función `punto_medio_comp(f,a,b,n)` que tiene como argumentos de entrada la función `f` a integrar, los extremos del intervalo de integración `a` y `b`, y el número de subintervalos que vamos a usar en la fórmula compuesta `n` y devuelve el valor aproximado utilizando la Regla del Trapecio compuesta.

Calcular, con `n = 5` subintervalos, la integral

$$\int_1^3 \ln(x) dx$$

Escribir el valor exacto y el valor aproximado.


```
%run Ejercicio1d.py
```



El valor aproximado es 1.3002242084538775

El valor exacto es 1.2958368660043291

Regla de los trapecios compuesta

$$\int_a^b f(x) dx \approx \frac{h}{2}(f(a) + f(b)) + h \sum_{i=1}^{n-1} f(x_i)$$

Ejercicio 1e

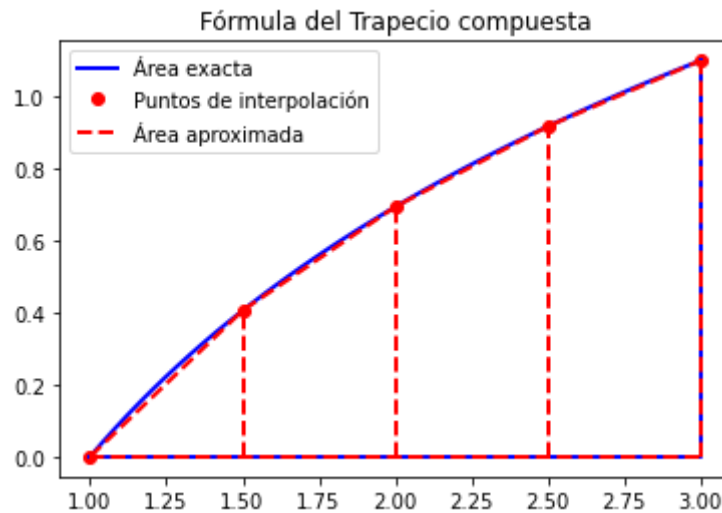
Escribir la función `trapezio_comp(f,a,b,n)` que tiene como argumentos de entrada la función `f` a integrar, los extremos del intervalo de integración `a` y `b` y el número de subintervalos que vamos a usar en la fórmula compuesta `n` y devuelve el valor aproximado utilizando la Regla del Trapecio compuesta.

Calcular, con `n = 4` subintervalos, la integral

$$\int_1^3 \ln(x) dx$$

Escribir el valor exacto y el valor aproximado.

```
%run Ejercicio1e.py
```



El valor aproximado es 1.2821045824381598

El valor exacto es 1.2958368660043291

Fórmula de Simpson Compuesta

$$\int_a^b f dx \approx \frac{h}{6} \sum_{i=1}^n (f(x_{i-1}) + 4f(\bar{x}_i) + f(x_i))$$

Ejercicio 1f

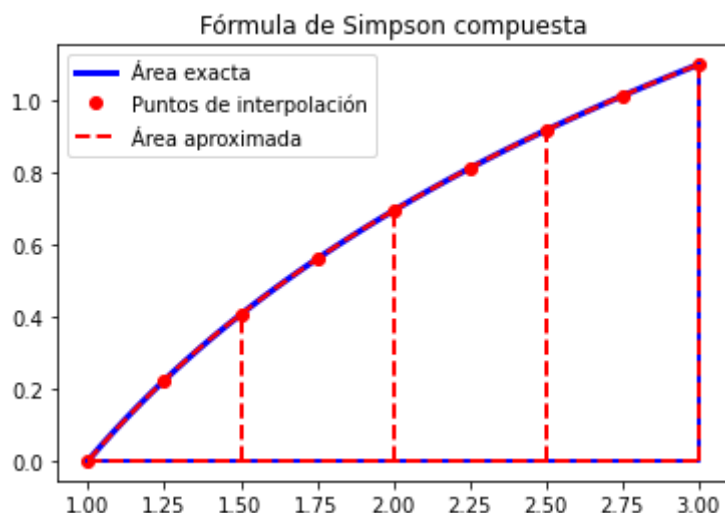
Escribir la función `simpson_comp(f,a,b,n)` que tiene como argumentos de entrada la función `f` a integrar, los extremos del intervalo de integración `a` y `b` y el número de subintervalos que vamos a usar en la fórmula compuesta `n` y devuelve el valor aproximado utilizando la Regla del Punto Medio compuesta.

Calcular, con `n = 4` subintervalos, la integral

$$\int_1^3 \ln(x) dx$$

Escribir el valor exacto y el valor aproximado.

```
%run Ejercicio1f.py
```



El valor aproximado es 1.295798349860867

El valor exacto es 1.2958368660043291

Fórmulas de cuadratura gaussianas

En la fórmula de cuadratura:

$$\int_a^b f(x)dx \approx \omega_0 f(x_0) + \omega_1 f(x_1) + \cdots + \omega_N f(x_N)$$

¿Es posible calcular los pesos ω_i y los nodos x_i de forma que la precisión de la fórmula sea lo mayor posible? Sí, pero entonces los nodos no estarán equiespaciados.

Si $[a, b] = [-1, 1]$ los pesos y nodos son

n	w_i	x_i
1	2.000000	0.000000
2	1.000000	± 0.577350
3	0.555556	± 0.774597
	0.888889	0.000000
4	0.347855	± 0.861136
	0.652145	± 0.339981
5	0.236927	± 0.906180
	0.478629	± 0.538469
	0.568889	0.000000

Estos nodos y pesos de la fórmula de Gauss-Legendre con nodos se pueden obtener con `np.polynomial.legendre.leggauss(n)`. Por ejemplo, para $n = 1$:

```
n = 1
[x, w] = np.polynomial.legendre.leggauss(n)
print('w\n',w)
print('x\n',x)
```

```
w
[2.]
x
[0.]
```

Así, por ejemplo, la fórmula gaussiana para un punto es

$$\int_{-1}^1 f(x) dx \approx 2 f(0)$$

Para dos puntos

```
n = 2
[x, w] = np.polynomial.legendre.leggauss(n)
print('w\n',w)
print('x\n',x)
```

```
w
[1. 1.]
x
[-0.57735027  0.57735027]
```

$$\int_{-1}^1 f(x) dx \approx f(-0.57735027) + f(0.57735027)$$

Para tres puntos

```
n = 3
[x, w] = np.polynomial.legendre.leggauss(n)
print('w\n',w)
print('x\n',x)
```

```
w
[0.55555556 0.88888889 0.55555556]
x
[-0.77459667 0.          0.77459667]
```

$$\int_{-1}^1 f(x) dx \approx 0.55555556 f(-0.77459667) + 0.88888889 f(0) + 0.55555556 f(0.77459667)$$

Y así sucesivamente.

Estos resultados se pueden generalizar a cualquier intervalo $[a, b]$ cambiando los x_i por y_i de acuerdo con la fórmula

$$y_i = \frac{b-a}{2} x_i + \frac{a+b}{2}$$

Y entonces la fórmula de cuadratura es

$$\int_a^b f(x) dx \approx \frac{b-a}{2} (\omega_0 f(y_0) + \omega_1 f(y_1) + \dots + \omega_n f(y_n))$$



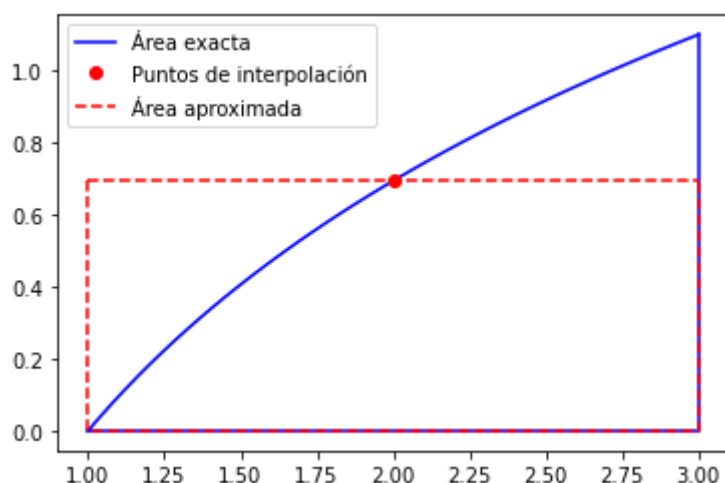
Ejercicio 2

Escribir la función `gauss(f,a,b,n)` que tiene como argumentos de entrada la función `f` a integrar, los extremos del intervalo de integración `a` y `b` y el número de nodos `n` y devuelve el valor aproximado utilizando la fórmulas gaussianas con `n = 1`, `n = 2` y `n = 3` nodos la integral

$$\int_1^3 \ln(x) dx$$

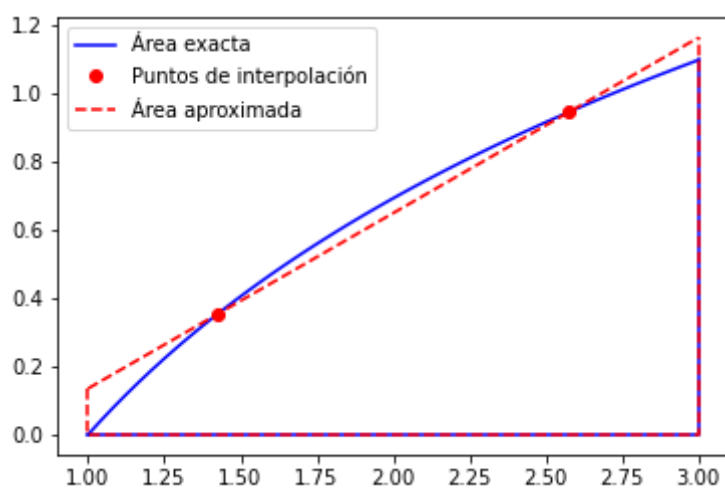
Escribir el valor exacto y el valor aproximado.

```
%run Ejercicio2.py
```



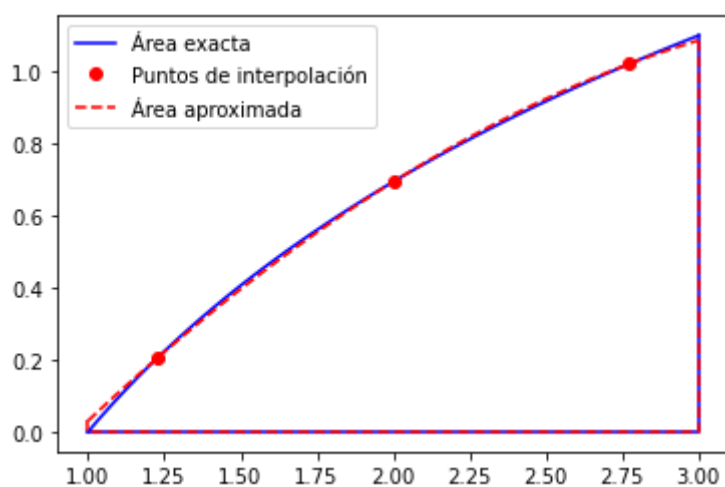
El valor aproximado es 1.3862943611198906

El valor exacto es 1.2958368660043291



El valor aproximado es 1.2992829841302609

El valor exacto es 1.2958368660043291



El valor aproximado es 1.2960060669544604

El valor exacto es 1.2958368660043291

Grado de precisión de las fórmulas de cuadratura

Grado de precisión

Una fórmula de cuadratura tiene grado de precisión r si es exacta para

$$f(x) = 1, \quad f(x) = x, \quad f(x) = x^2, \dots, \quad f(x) = x^r$$

pero no es exacta para $f(x) = x^{r+1}$

Ejercicio 3

Escribir una función `newton_cotes(f,a,b,n)` que devuelva el valor de la integral aproximada de f en el intervalo $[a,b]$ con la función `punto_medio` si $n = 1$, `trapecio` si $n = 2$ y `Simpson` si $n = 3$ donde n es el número de nodos usado en la fórmula.

Escribir la función `grado_de_precision(formula,n)` que tiene como argumento de entrada la función `formula` que puede ser la función `newton_cotes` o `gauss` y donde n es el número de nodos que se usa en la construcción de la fórmula, que estudia el grado de precisión de cada una de estas fórmulas calculando su error para las integrales

$$\int_1^3 x^i dx \quad i = 0, 1, 2, \dots$$

y cuando el error es distinto de cero, para. Imprimir los errores para cada polinomio y el grado de precisión de la fórmula.

Notas:

- Debido a los errores de redondeo, considerar que el error es cero cuando es menor que 10^{-10} .

```
%run Ejercicio3.py
```


----- Fórmula del punto medio (n = 1) -----

f(x) = x⁰ error = 0.0
 f(x) = x¹ error = 0.0
 f(x) = x² error = 0.6666666666666661

El grado de precisión es 1

----- Fórmula del trapecio (n = 2) -----

f(x) = x⁰ error = 0.0
 f(x) = x¹ error = 0.0
 f(x) = x² error = 1.3333333333333334

El grado de precisión es 1

----- Fórmula de Simpson (n = 3) -----

f(x) = x⁰ error = 0.0
 f(x) = x¹ error = 0.0
 f(x) = x² error = 0.0
 f(x) = x³ error = 0.0
 f(x) = x⁴ error = 0.26666666666666657

El grado de precisión es 3

----- Fórmula Gauss n = 1 -----

f(x) = x⁰ error = 0.0
 f(x) = x¹ error = 0.0
 f(x) = x² error = 0.6666666666666661

El grado de precisión es 1

----- Fórmula Gauss n = 2 -----

f(x) = x⁰ error = 0.0
 f(x) = x¹ error = 0.0
 f(x) = x² error = 0.0
 f(x) = x³ error = 0.0
 f(x) = x⁴ error = 0.17777777777777715

El grado de precisión es 3

----- Fórmula Gauss n = 3 -----

f(x) = x⁰ error = 4.440892098500626e-16
 f(x) = x¹ error = 0.0
 f(x) = x² error = 1.7763568394002505e-15
 f(x) = x³ error = 7.105427357601002e-15
 f(x) = x⁴ error = 7.105427357601002e-15
 f(x) = x⁵ error = 1.4210854715202004e-14
 f(x) = x⁶ error = 0.04571428571415481

El grado de precisión es 5

---- Fórmula Gauss n = 4 ----

```
f(x) = x^0    error = 2.220446049250313e-16
f(x) = x^1    error = 4.440892098500626e-16
f(x) = x^2    error = 0.0
f(x) = x^3    error = 0.0
f(x) = x^4    error = 7.105427357601002e-15
f(x) = x^5    error = 0.0
f(x) = x^6    error = 5.684341886080802e-14
f(x) = x^7    error = 2.2737367544323206e-13
f(x) = x^8    error = 0.011609977324951615
```

El grado de precisión es 7

Ejercicios propuestos

Integración numérica con órdenes python

El módulo [scipy.integrate](https://docs.scipy.org/doc/scipy/reference/integrate.html) (<https://docs.scipy.org/doc/scipy/reference/integrate.html>) provee varias funciones con diferentes técnicas de integración. Entre sus funciones está la función de propósito general `quad`.

```
from scipy.integrate import quad

f = lambda x : np.log(x)
a = 1.; b = 3;

I = quad(f,a,b)

print('El valor aproximado es', I[0])
print('El valor exacto es      ', I_exacta)
```

```
El valor aproximado es 1.2958368660043291
El valor exacto es      1.2958368660043291
```

Integración de Montecarlo

Podemos calcular integrales aproximadas usando números aleatorios.

Consideremos, de momento, que la función es positiva en el intervalo de integración $[a, b]$. El valor de la integral de la función es igual al área bajo la curva. Para aproximar este área:

1. Generamos puntos aleatorios dentro del rectángulo $[a, b] \times [0, \max(f)]$.
2. Contamos el número de puntos por debajo de la curva.
3. La proporción del número de puntos por debajo de la curva relativa a los puntos totales, multiplicada por el área del anterior rectángulo nos da el área aproximada.

Ejercicio 4

Escribir la función `montecarlo(f,a,b,n)` que tiene como argumentos de entrada la función `f` a integrar, los extremos del intervalo de integración `a` y `b` y el número de puntos aleatorios `n` y devuelve el valor aproximado utilizando el método de Montecarlo.

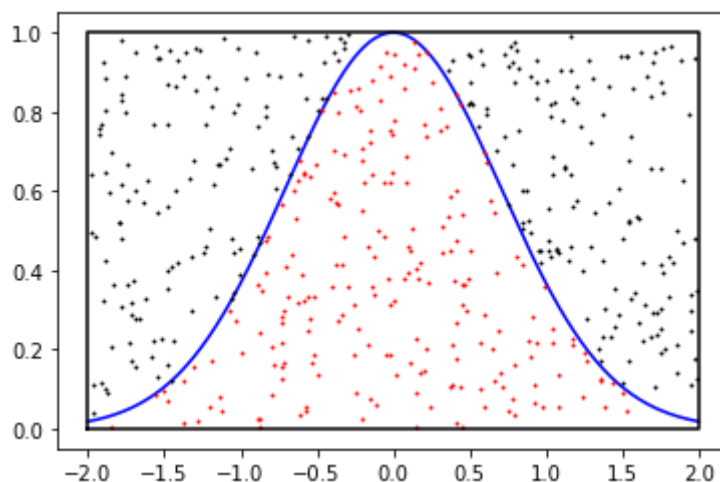
$$\int_{-2}^2 e^{-x^2} dx$$

Escribir el valor exacto y el valor aproximado.

Nota

Para calcular `n` puntos aleatorios distribuidos uniformemente en el intervalo $[a, b]$ podemos usar `np.random.rand(n) * (b-a) + a`.

```
%run Ejercicio4.py
```



El valor aproximado es 1.735972

El valor exacto es 1.764163

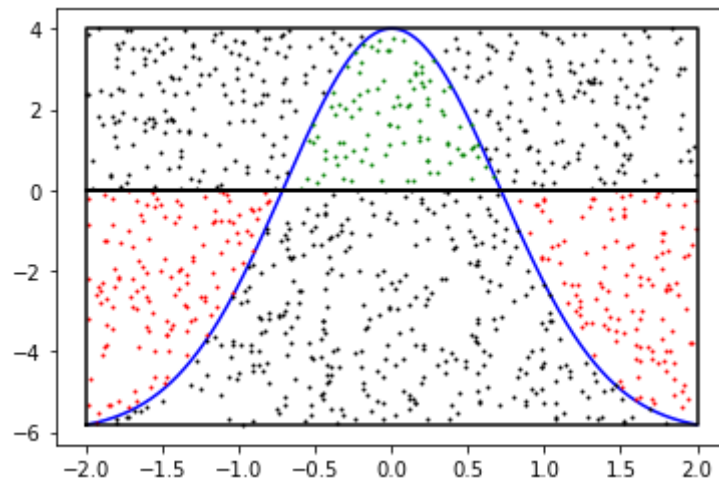
Ejercicio 5

Modificar la función anterior para que sea válida para funciones no positivas. Calcular la integral

$$\int_{-2}^2 (10e^{-x^2} - 6) dx$$

Escribir el valor exacto y el valor aproximado.

```
%run Ejercicio5.py
```



El valor aproximado es -6.243487
El valor exacto es -6.358372