pratica3b_integrales

November 27, 2022

Práctica 3b. Cálculo de integrales simbólicas con sympy

Paso 1.- Cargar paquete sympy, y comandos básicos

Paso 2.- Definir función f= …

Paso 3.- usar comandos específicos, integrate(f), etc…

A. Cálculo simbólico de primitivas

[1]:
```
import sympy as sp
from sympy import symbols, integrate, pi, sin, cos, log, exp, sqrt
```

[3]:
```
x=symbols('x')
f=x**2

integrate(f)
```

[3]:
$$\frac{x^3}{3}$$

si no sabe calcular la primitiva simbólica, lo deja indicado, o expresa el resultado usando "funciones especiales"

[4]:
```
x=symbols('x')
f=x/sin(x)

integrate(f)
```

[4]:
$$\int \frac{x}{\sin\left(x\right)}\, dx$$

[38]:
```
x=symbols('x')
f=sin(x)/x

display(integrate(f))
integrate(f**2)
```

$\mathrm{Si}\left(x\right)$

[38]:
$$\mathrm{Si}\left(2x\right) + \frac{\cos\left(2x\right)}{2x} - \frac{1}{2x}$$

B. Integrales definidas (con cálculo simbólico)

[7]:
```
x=symbols('x')
f=x**2

integrate(f, (x, 0, 2))
```

[7]:
$$\frac{8}{3}$$

[14]:
```
from sympy import oo as infty

x=symbols('x')
f=sin(x)/x

integrate(f, (x, 0, infty))
```

[14]:
$$\frac{\pi}{2}$$

[16]:
```
#si integrate no sabe calcular la primitiva simbólica, deja la expresión sin␣
 ↪resolver

x=symbols('x')
f=x/sin(x)

integrate(f, (x, 0, pi/2))
```

[16]:
$$\int_0^{\frac{\pi}{2}} \frac{x}{\sin\left(x\right)}\, dx$$

C. Integración numérica.

Para integración numérica es preferible usar el paquete scipy.integrate, y alguno de los métodos numéricos de integración: simpson, trapezoid, romberg, quad,...

[2]:
```
import numpy as np
from scipy import integrate

def f(x):
    return x/np.sin(x)

display(integrate.romberg(f, 0.0001, np.pi/2))
integrate.romberg(f, 0.0001, np.pi/2, show=True)

# Nota: en estos ejemplos a veces da error si empezamos la integral en x=0...
```

```
1.8318311883617717
```

```
Romberg integration of <function vectorize1.<locals>.vfunc at
0x0000015BDE652430> from [0.0001, 1.5707963267948966]
```

```
 Steps  StepSize   Results
     1  1.570696  2.018970
     2  0.785348  1.881799  1.836076
     4  0.392674  1.844584  1.832179  1.831919
     8  0.196337  1.835037  1.831855  1.831833  1.831832
    16  0.098169  1.832634  1.831833  1.831831  1.831831  1.831831
    32  0.049084  1.832032  1.831831  1.831831  1.831831  1.831831  1.831831

The final result is 1.8318311883617717 after 33 function evaluations.
```

[2]:
```
1.8318311883617717
```

[3]:
```
import numpy as np
from scipy import integrate

def f(x):
    return x/np.sin(x)

x = np.linspace(0.0001, np.pi/2, 100)

integrate.simpson(f(x), x)
```

[3]:
```
1.8318315023310685
```

[39]:
```
import numpy as np
from scipy import integrate

def f(x):
    return x/np.sin(x)

integrate.quad(f, 0, np.pi/2)

# quad es un método muy eficiente, que devuelve también un margen de precisión
```

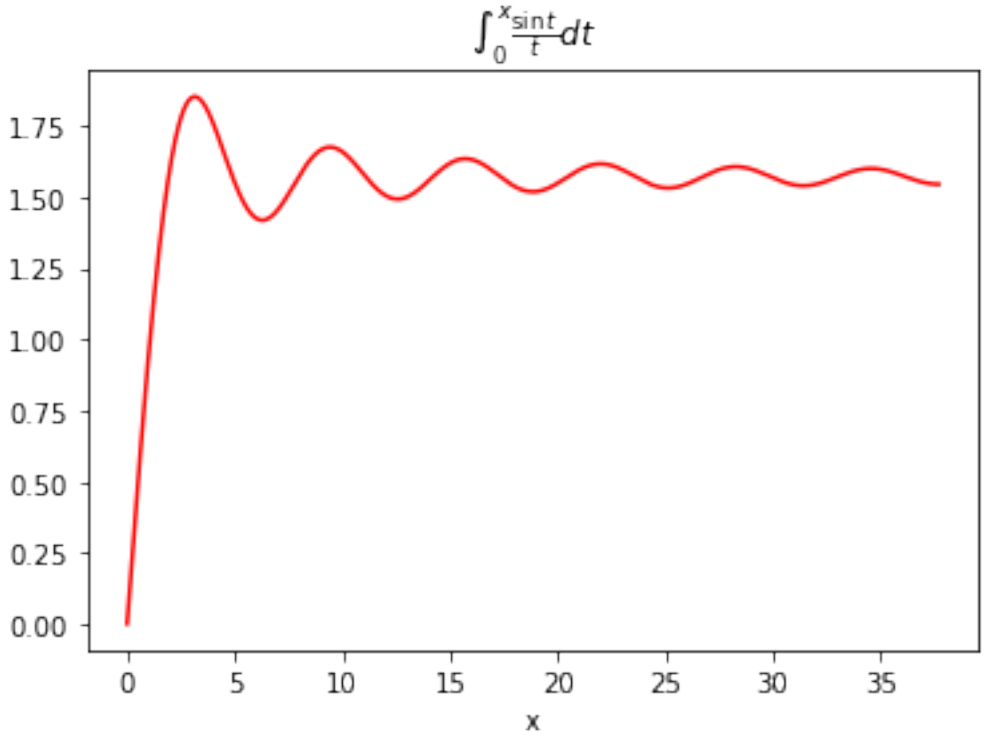
[39]:
```
(1.831931188354438, 2.0338521848400216e-14)
```

D. Gráficas de primitivas

[44]:
```
# damos un ejemplo de cómo definir y dibujar la primitiva de sin(x)/x...

import numpy as np
from scipy import integrate

def f(x):
    return np.sin(x)/x
def F(x):
    return integrate.quad(f, 0, x)
```

```
import matplotlib.pyplot as plt

x = np.linspace(0.001, 12*np.pi, 10000)

y = []
for i in x:
    y.append(F(i)[0])

plt.plot(x, y, 'r')

# adornos

plt.title(r'$\int_0^x\frac{\sin t}{t}dt$')
plt.xlabel('x')
```

[44]: Text(0.5, 0, 'x')



[ ]: