Integration (Symbolic & Numeric) (Mr. P Solver)

Video Link: [https://youtu.be/2I44Y9hfQ4Q (https://youtu.be/2I44Y9hfQ4Q)](https://youtu.be/2I44Y9hfQ4Q)

Codes: https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqbTZ5MTBwU0JGRW9IeHI2VW1EdHh2ZTV3MzM5d3x (https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqbTZ5MTBwU0JGRW9IeHI2VW1EdHh2ZTV3MzM5d3x

In [1]:
```
import numpy as np
import scipy as sp
import sympy as smp
import matplotlib.pyplot as plt
```

In [2]:
```
from scipy.integrate import quad
from scipy.integrate import cumulative_trapezoid
```

Symbolic Case

We know the function. Here we have 2 options:

1. The integral can be solved analytically.
2. The integral cannot be solved analytically.

Part 1: Solvable Integrals

Example: $\int \sin^3(x) e^{-5x} dx$

In [3]:
```
x = smp.symbols('x', real = True)
```

In [4]:
```
f1 = smp.sin(x)**3 * smp.exp(-5*x)
smp.integrate(f1,x)
```

Out[4]:

$$-\frac{40e^{-5x}\sin^3(x)}{221} - \frac{21e^{-5x}\sin^2(x)\cos(x)}{221} - \frac{15e^{-5x}\sin(x)\cos^2(x)}{442} - \frac{3e^{-5x}\cos^3(x)}{442}$$

Example: $\int \cos(bx) e^{-ax} dx$

In [5]:
```
a, b = smp.symbols('a b', real=True, positive=True)
f2 = smp.cos(b*x) * smp.exp(-a*x)
smp.integrate(f2,x).simplify()
```

Out[5]:

$$\frac{(-a\cos(bx) + b\sin(bx))e^{-ax}}{a^2 + b^2}$$

Example: $\int \frac{(1+\sqrt{x})^{1/3}}{\sqrt{x}} dx$

In [6]:
```python
f3 = (1+smp.sqrt(x))**smp.Rational(1,3) / smp.sqrt(x)
smp.integrate(f3,x).simplify()
```

Out[6]:

$$\frac{3\left(\sqrt{x}+1\right)^{\frac{4}{3}}}{2}$$

Example: $\int_0^{\ln(4)} \frac{e^x}{\sqrt{e^{2x}+9}} dx$

In [7]:
```python
f4 = smp.exp(x)/ smp.sqrt(smp.exp(2*x) + 9)
smp.Integral(f4,(x,0,smp.log(4))).doit()
```

Out[7]:

$$-\operatorname{asinh}\left(\frac{1}{3}\right)+\operatorname{asinh}\left(\frac{4}{3}\right)$$

Example: $\int_0^{\infty} \frac{16\tan^{-1}(x)}{1+x^2} dx$

In [8]:
```python
f5 = 16*smp.atan(x)/(1+x**2)
smp.integrate(f5,(x,0,smp.oo))
```

Out[8]: $2\pi^2$

Part 2: Unsolvable Integrals

In Sympy it keeps running without giving the result until we interrupt it by **Kernel ---> Interrupt**. So, don't run these integrals in Sympy.

We will use **quad** function of Scipy to integrate numerically.

Example: $\int_1^2 e^{-\sin(x)} dx$

In [9]:
```python
f6 = lambda x: np.exp(-np.sin(x))
quad(f6,1,2)
```

Out[9]:
```
(0.3845918142796868, 4.2698268729567035e-15)
```

Example: $\int_0^{2\pi} \frac{1}{(a-\cos(x))^2+(b-\sin(x))^2} dx$

Here we need to choose a and b before integration.

In [10]:
```python
f7 = lambda x : 1/((a-np.cos(x))**2 + (b-np.sin(x))**2)
a, b = 1, 2
quad(f7, 0, 2*np.pi)
```

Out[10]:
```
(1.5707963267948961, 6.710624173315513e-09)
```

Solution for different values of a and b:

In [11]:
```python
def f8(x,a,b):
    return 1/((a-np.cos(x))**2 + (b-np.sin(x))**2)
```

In [12]:
```python
a_array = np.arange(1,5,1)
b_array = np.arange(1,5,1)
```

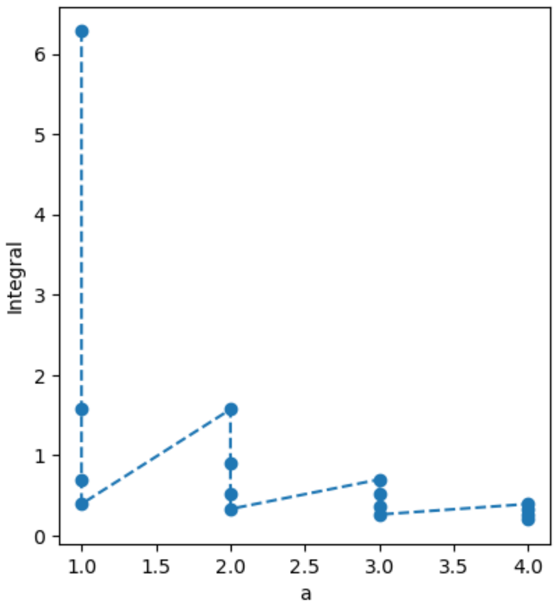
In [13]:
```python
integrals = [[a,b, quad(f8, 0, 2*np.pi, args=(a,b))[0]]
             for a in a_array for b in b_array]
```

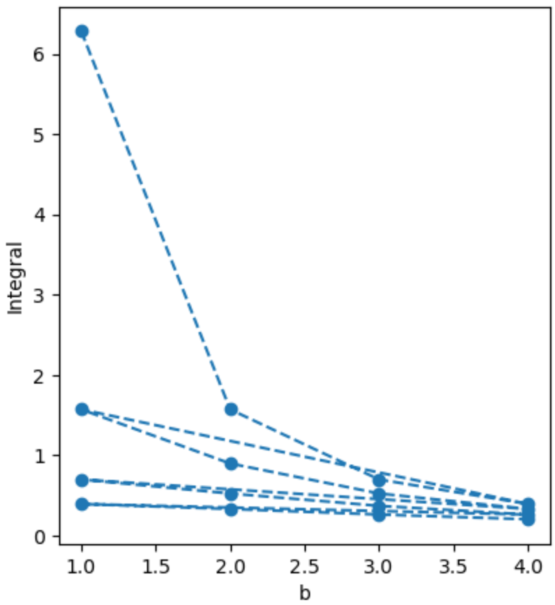
In [14]:
```python
integrals  # [a,b,integration]
```

Out[14]:
```
[[1, 1, 6.283185307179586],
 [1, 2, 1.5707963267948961],
 [1, 3, 0.6981317007977318],
 [1, 4, 0.3926990816987241],
 [2, 1, 1.5707963267948952],
 [2, 2, 0.8975979010256552],
 [2, 3, 0.5235987755982989],
 [2, 4, 0.3306939635357684],
 [3, 1, 0.6981317007977317],
 [3, 2, 0.5235987755982988],
 [3, 3, 0.36959913571644665],
 [3, 4, 0.26179938779914935],
 [4, 1, 0.39269908169872425],
 [4, 2, 0.3306939635357676],
 [4, 3, 0.26179938779914946],
 [4, 4, 0.2026833970057931]]
```

In [15]:
```python
ap = np.array(integrals).T[0]
bp = np.array(integrals).T[1]
I = np.array(integrals).T[2]
```

In [16]:
```python
# TRY 3D GRAPH
fig, axes = plt.subplots(1,2,figsize=(10,5))
axa = axes[0]
axa.plot(ap,I,'o--')
axa.set_xlabel('a')
axa.set_ylabel('Integral')
axb = axes[1]
axb.plot(bp,I,'o--')
axb.set_xlabel('b')
axb.set_ylabel('Integral')
plt.show()
```

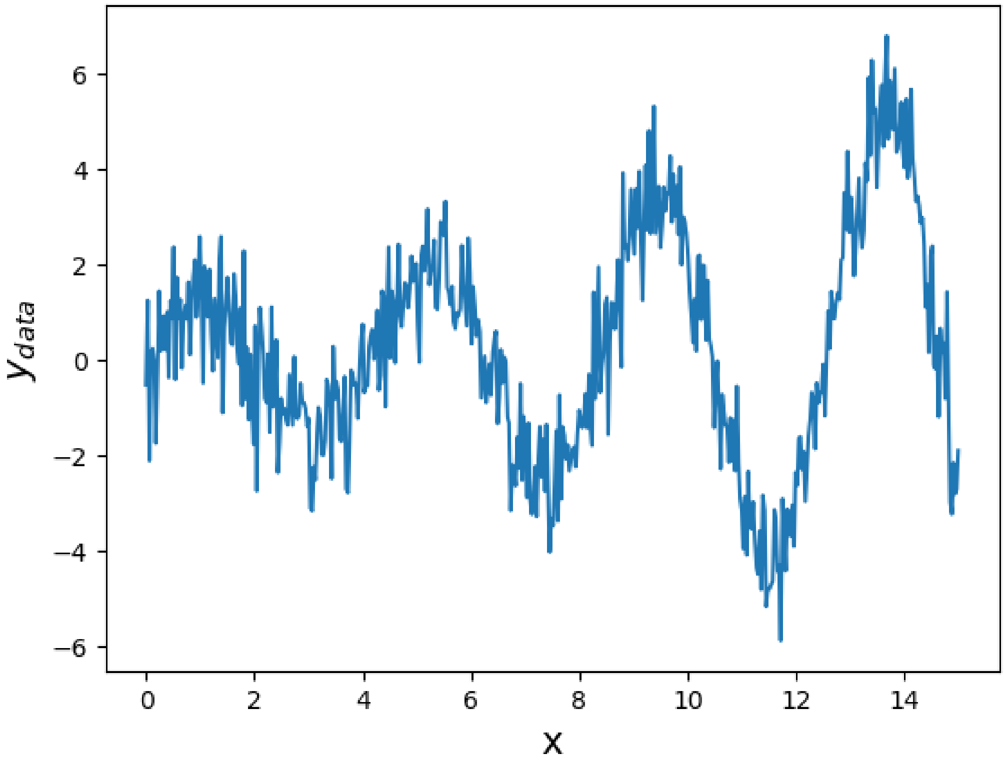




Numerical Case

In [17]:
```
xdata = np.linspace(0.001,15,500)
ydata = np.exp(xdata/8)*np.sin(1.5*xdata) +0.9*np.random.randn(len(xdata))
plt.plot(xdata,ydata)
plt.xlabel('x', fontsize=15)
plt.ylabel('$y_{data}$', fontsize=15)
```

Out[17]:
```
Text(0, 0.5, '$y_{data}$')
```

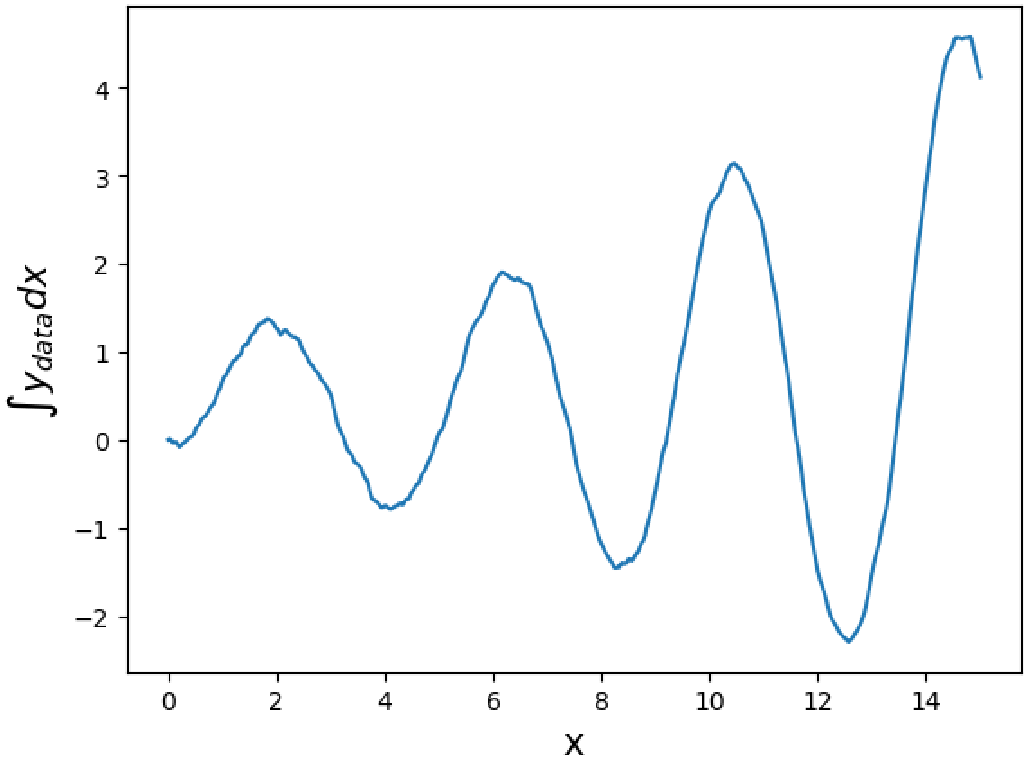


Function used: `cumulative_trapezoid`

In [18]:
```
inty = cumulative_trapezoid(ydata, xdata, initial=0)

plt.plot(xdata, inty)
plt.xlabel('x', fontsize=15)
plt.ylabel(r'$\int y_{data} dx$', fontsize=15)
```

Out[18]:
```
Text(0, 0.5, '$\\int y_{data} dx$')
```



In [ ]: