MATH 152 Lab 4

Blake Stewart and Samuel Molero

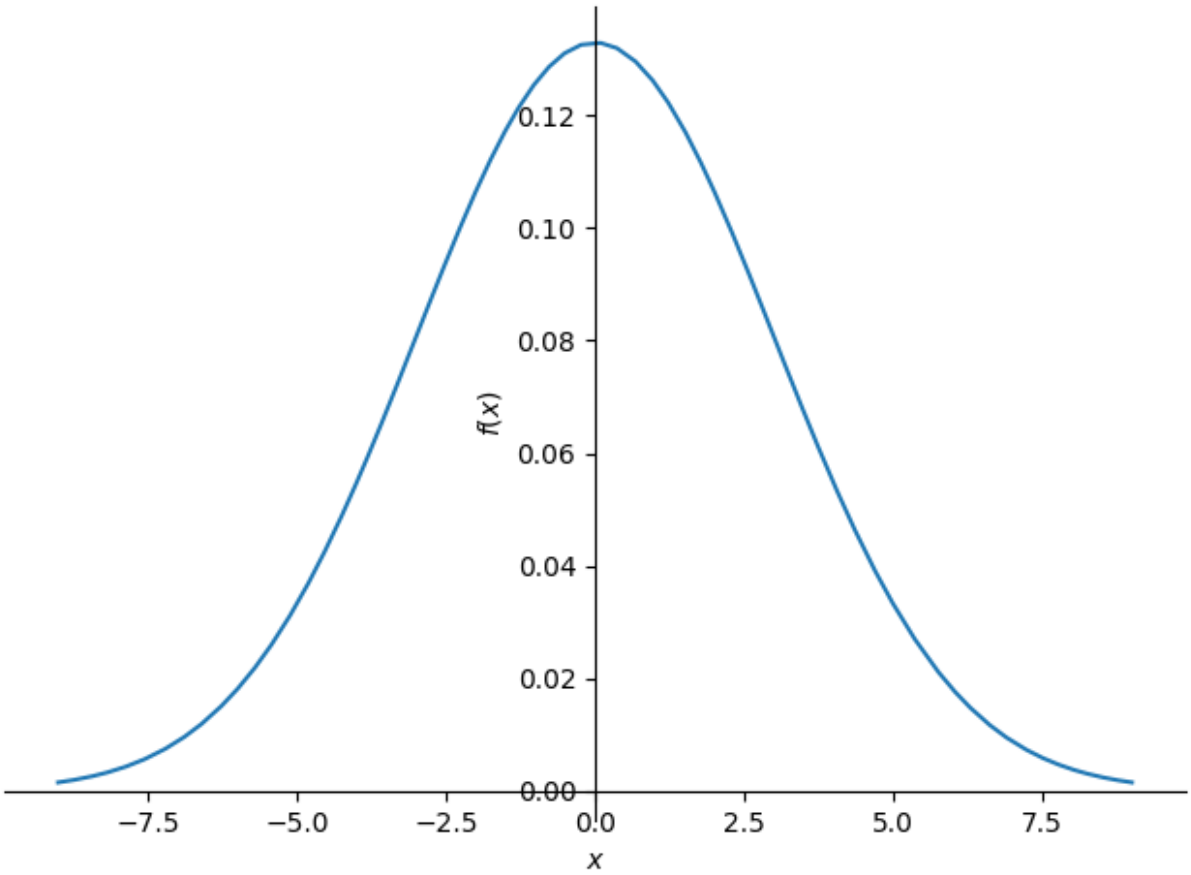
In [3]:
```
import numpy as np
import sympy as sp
from sympy.plotting import (plot, plot_parametric)
```

Question 1

1a

In [4]:
```
# Given the normal distribution, plot.
x = sp.symbols('x')
f_x = (1 / (3*sp.sqrt(2*sp.pi)))*sp.exp((-1/2)*(x/3)**2)


p1 = plot(f_x,(x,-9,9), show = False)
p1.show()
```



1b Left Endpoint Approximation

In [5]:
```
n = 500
a = -6
b = 6

delta_x = (b-(a))/n
x_i = np.arange(a,b,delta_x)

y_i = (1 / 3*sp.sqrt(2*sp.pi)) * (sp.E**((-1/2)*(x_i/3)**2))

L_500 =sum(y_i) * delta_x
print('The Left endpoint Approximation from -6 to 6 is: ', L_500.evalf())
```

```
The Left endpoint Approximation from -6 to 6 is:  5.99729148058229
```

Question 2 Right Endpoint Approximation

In [6]:
```
dx = (b-(a)) / n
x_i = np.arange(a+dx,b+dx,dx)
y_i = (1 / 3*sp.sqrt(2*sp.pi)) * (sp.E**((-1/2)*(x_i/3)**2))

R_500 =sum(y_i) * delta_x
print("The right hand sidde approximation from [-6,6] is:", R_500.evalf())
```

```
The right hand sidde approximation from [-6,6] is: 5.99729148058229
```

Question 3 Midpoint Approximation

3a

In [7]:
```
dx = (b-(a)) / n
xi = np.arange((a + (dx/2)), (b+(dx/2)), dx)
y_i = (1 / 3*sp.sqrt(2*sp.pi)) * (sp.E**((-1/2)*(xi/3)**2))
M500 = sum(y_i) * dx
print('The midpoint approximation is ', M500.evalf())
```

```
The midpoint approximation is  5.99730233609575
```

3b

In [8]:
```
average = (R_500 + L_500 )/ 2

if average  == M500:
    print("The are the same")
else:
    print("they are not the same")
```

```
they are not the same
```

Question 4 Trapezoid Approximation

4a

In [9]:
```
from scipy.integrate import trapz
n= 500
a = -6
b = 6
dx = (b- (a)) / n
xi = np.arange(a,b+dx,dx)
y_i = (1 / 3*sp.sqrt(2*sp.pi)) * (sp.E**((-1/2)*(xi/3)**2))

approximation = trapz(y_i,xi)
print("The Trapezoid approximation is:", approximation.evalf())
```

```
/tmp/ipykernel_244480/2063998966.py:9: DeprecationWarning: 'scipy.integrate.
trapz' is deprecated in favour of 'scipy.integrate.trapezoid' and will be re
moved in SciPy 1.14.0
  approximation = trapz(y_i,xi)
The Trapezoid approximation is: 5.99729148058230
```

4b

In [10]:
```
if approximation == average:
    print("It is equal to the average")
else:
    print("It is not equal to the average")
```

```
It is not equal to the average
```

Question 5 Simpson's Rule

In [11]:
```
from scipy.integrate import simps

xi = np.arange(-6,6+dx,dx)
yi = (1 / 3*sp.sqrt(2*sp.pi)) * (sp.E**((-1/2)*(xi/3)**2))
simpson = simps(yi,xi)
print('The simpsons approximation', simpson.evalf())
```

```
/tmp/ipykernel_244480/3215813395.py:5: DeprecationWarning: 'scipy.integrate.
simps' is deprecated in favour of 'scipy.integrate.simpson' and will be remo
ved in SciPy 1.14.0
  simpson = simps(yi,xi)
The simpsons approximation 5.99729871756232
```

Question 6 Errors

In [17]:
```
actual = 0.954499736103642
left = abs(actual - L_500)
print('The error in the leftendpoit', left.evalf())
right = abs(actual - R_500)
print('The error in the rightpoint', right.evalf())
middle = abs(actual - M500)
print('The error in the midpoint', middle.evalf())
trap = abs(actual - approximation)
print('The error in the Trapezoid approximation', trap.evalf())
```

```
simp = abs(actual - simpson)
print('The error in the Simpson approximation', simp.evalf())
```

```
The error in the leftendpoit 5.04279174447865
The error in the rightpoint 5.04279174447865
The error in the midpoint 5.04280259999210
The error in the Trapezoid approximation 5.04279174447866
The error in the Simpson approximation 5.04279898145867
```

In [ ]: