Homework Week 3

September 20, 2021

1 Homework Week 3

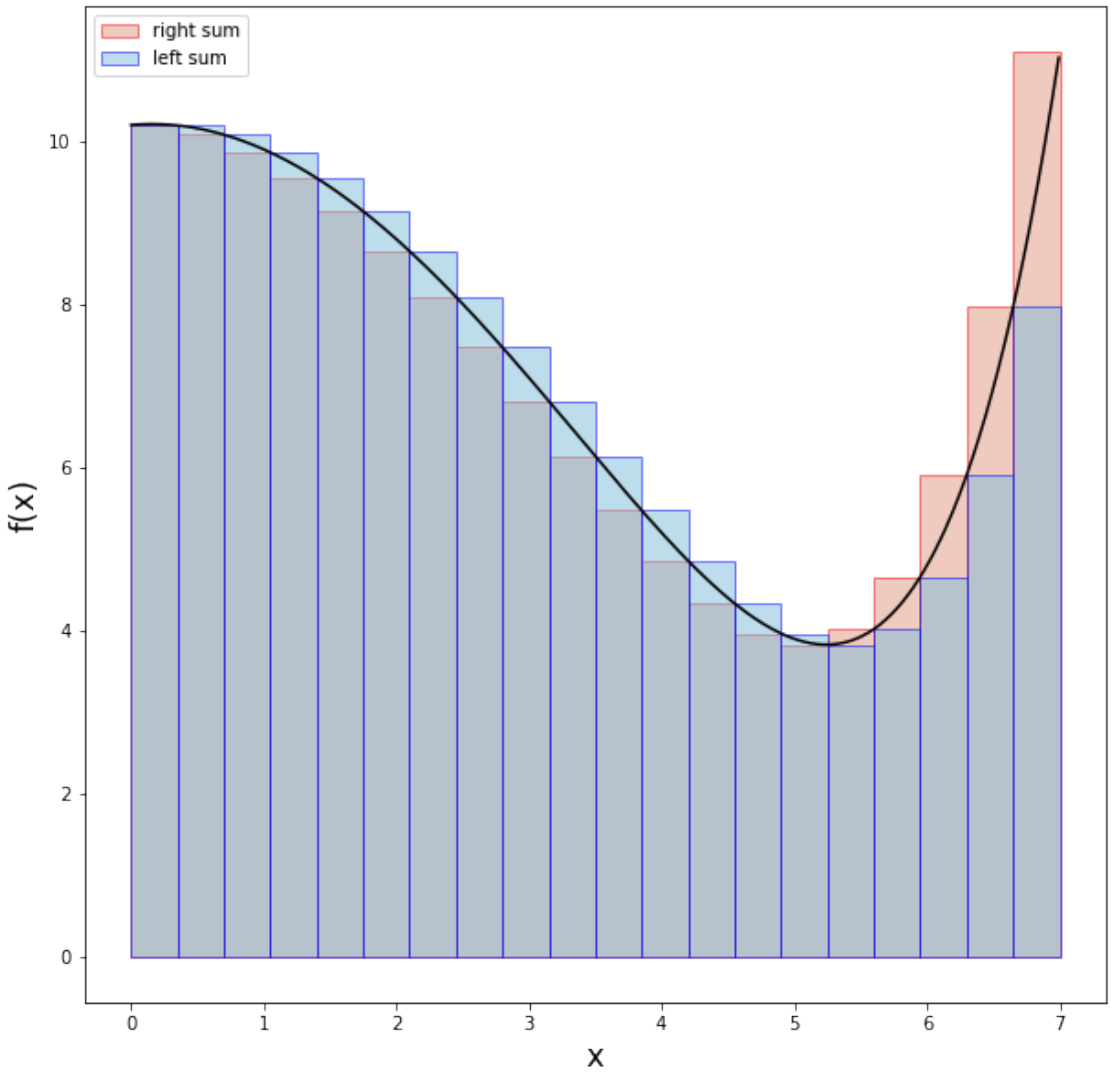
1.1 Left and Right Right Riemann Sums

[21]:
```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches

plt.figure(figsize=[10,10])

# inputs
functionText = input("input function (python syntax, in terms of x):")
n            = int(input("number of divisions (n):"))
a, b         = eval(input("interval a,b :"))

# process inputs
function     = lambda x: eval(functionText)
increment    = (b - a) / n

# domain and range for smooth function
XSmooth = np.arange(a, b, (b - a) / 1000)
YSmooth = [function(x) for x in XSmooth]

# domain and range for increment points
X = np.arange(a+increment, b+increment, increment)

# right riemann sum
total = 0
for x in X:
    # track height
    height = function(x)

    # track total sum
    total += height * increment

    # draw rectangles
```

```
    plt.gca().add_patch(mpatches.
 ↪Rectangle([x-increment,0],increment,height,color='#de9681',ec='red',alpha=0.
 ↪5))

print("right sum:",total)

# domain and range for increment points
X = np.arange(a, b, increment)

# left riemann sum
total = 0
for x in X:
    # track height
    height = function(x)

    # track total sum
    total += height * increment

    # draw rectangles
    plt.gca().add_patch(mpatches.
 ↪Rectangle([x,0],increment,height,color='#7fbddb',ec='blue',alpha=0.5))

print("left sum:",total)

plt.plot(XSmooth,YSmooth,color='black')
plt.xlabel("x",size='xx-large')
plt.ylabel("f(x)",size='xx-large')
red  = mpatches.Rectangle([0,0],1,1,color='#de9681',ec='red',alpha=0.5)
blue = mpatches.Rectangle([0,0],1,1,color='#7fbddb',ec='blue',alpha=0.5)
plt.legend([red,blue],['right sum','left sum'])
plt.show()
```

```
input function (python syntax, in terms of x): 0.2*2**x - 0.5*x**2 + 10
number of divisions (n): 20
interval a,b : 0,7
```

```
right sum: 49.74338027398858
left sum: 49.42838027398857
```

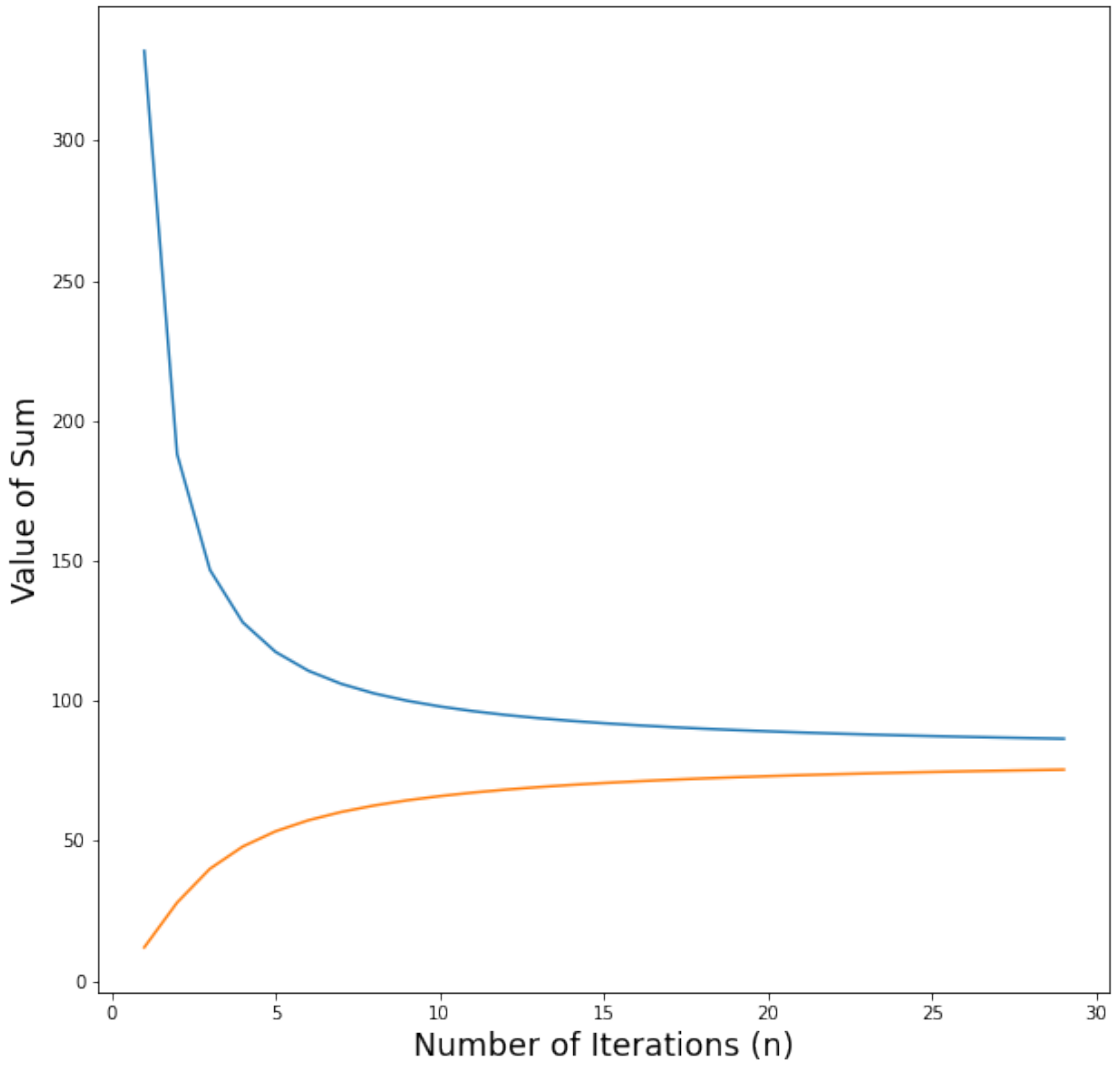


1.2 Convergence of Left and Right sums

[20]:
```
import numpy as np
import matplotlib.pyplot as plt

plt.figure(figsize = [10,10])

# same idea as before, but this time we are calculating several left/right sums␣
 ↪and graphing their convergence
functionText = input("input function (python syntax, in terms of x):")
a, b         = eval(input("interval a,b :"))
RBConverge   = eval(input("right bound for convergence function: "))
function     = lambda x: eval(functionText)
```

```
def rightSum(function, n):
    increment    = (b - a) / n
    X = np.arange(a+increment, b+increment, increment)
    total = 0
    for x in X:
        total += function(x) * increment
    return total


def leftSum(function, n):
    increment    = (b - a) / n
    X = np.arange(a, b, increment)
    total = 0
    for x in X:
        total += function(x) * increment
    return total

X = range(1,RBConverge)
YRightSum = []
YLeftSum  = []
for n in X:
    YRightSum.append(rightSum(function,n))
    YLeftSum.append(leftSum(function,n))

plt.plot(X,YRightSum)
plt.plot(X,YLeftSum)
plt.xlabel("Number of Iterations (n)",size='xx-large')
plt.ylabel("Value of Sum",size='xx-large')
plt.show()
```

```
input function (python syntax, in terms of x): 3**x + 2
interval a,b : 0,4
right bound for convergence function:  30
```



1.3 Double Integral Approximator

[22]:
```
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches

# inputs
functionText = input("input multivar function (python syntax, in terms of x,y):
 ↪")
increment    = float(input("increment (square, dy = dx):"))
a, b         = eval(input("interval a, b:"))
c, d         = eval(input("interval c, d:"))

# process the text input into a function
```

```
function = lambda x,y: eval(functionText)

# for each "square" increment, by height function to get total volume of a␣
 ↪cuboid
# add up cuboid volumes to get approximate volume
totalVolume = 0
for x in np.arange(a,b,increment):
    for y in np.arange(c,d,increment):
        area = increment * increment
        height = function(x,y)
        totalVolume += area*height

print("The integral evaluates to approximately:", totalVolume)
```

```
input multivar function (python syntax, in terms of x,y): x**2 + y**2
increment (square, dy = dx): 0.005
interval a, b: 0,5
interval c, d: 0,5
```

```
The integral evaluates to approximately: 416.04187499998716
```