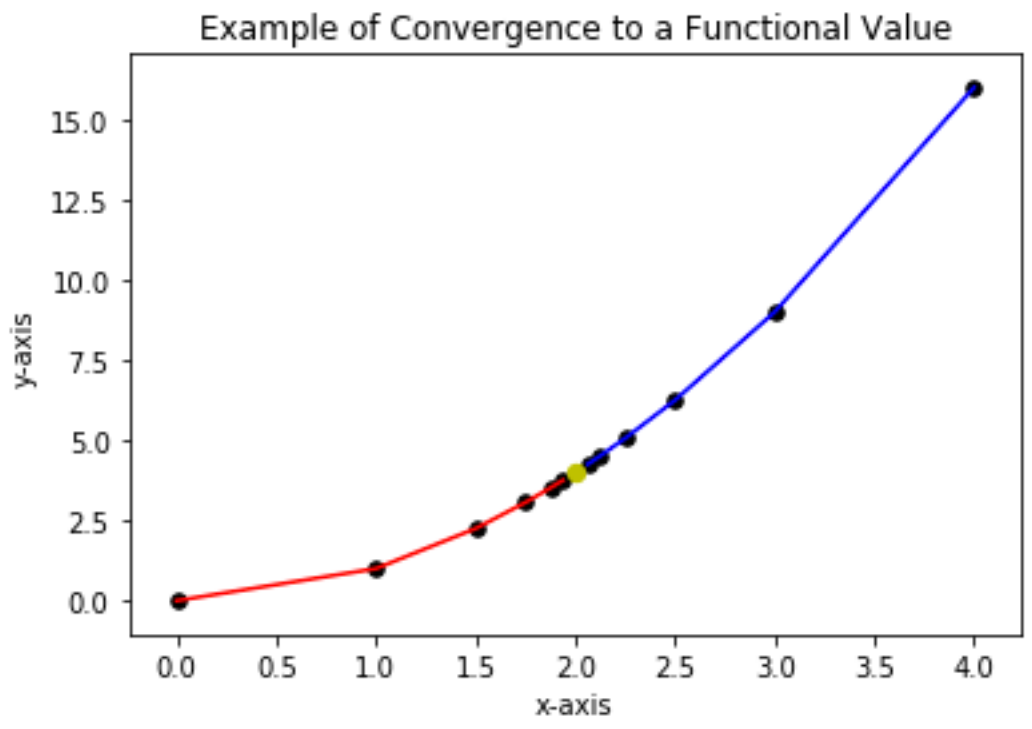
Module 6 Answers

Module 6 Practice 1

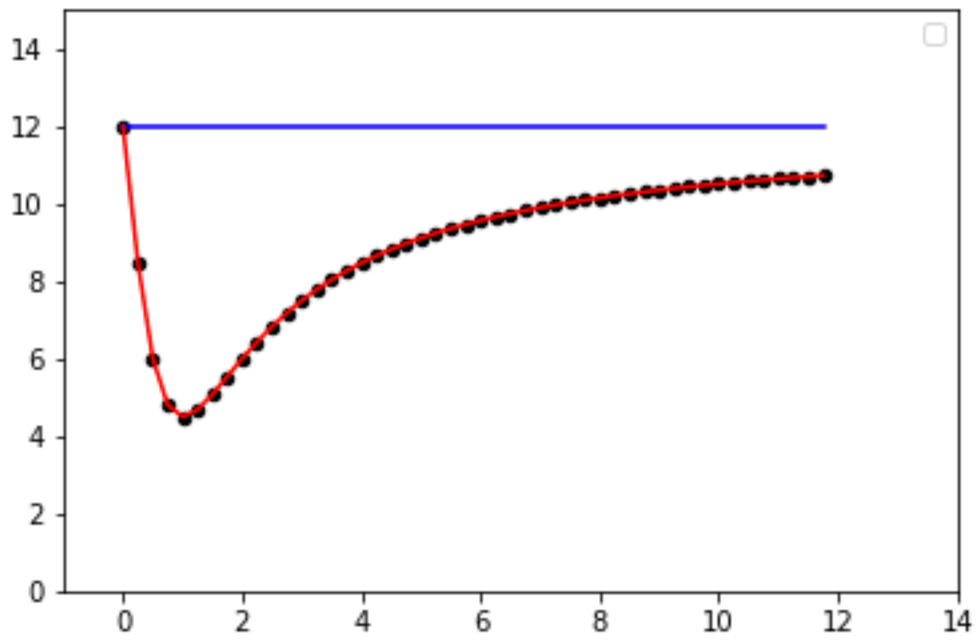
Exercise 1: Refer to Lial Section 11.1 Example 1. Generalize the code for the function indicated to determine a limit when x=2. Compare the code and the resulting plot to the answer sheet.

```
from numpy import arange
import matplotlib.pyplot as plt

def g(x):
        g = (x*x) #This is the function in Lial Section 11.1 Example 1.
        return g
n=5     # This determines the number of values calculated on each side of x=0.
powers=arange(0,n+1)
denominator=2.0**powers    # denominator contains exponentiated values of 2.0.
delta=2.0        # This is the interval used on either side of x=2.0.

# The following are values of x and f(x) trending to the limit at x=2.0.
# Delta is being divided by powers of 2 to reduce the distance from the limit.

x_r=2.0+delta/denominator #Approaching from the right.
y_r=g(x_r)
x_l=2.0-delta/denominator #Approaching from the left.
y_l=g(x_l)


# The following determine the vertical boundaries of the resulting plot.
ymax1=max(abs(y_r))+0.5
ymax2=max(abs(y_l))+0.5
ymax=max(ymax1,ymax2)
ymin1=min(abs(y_r))-0.5
ymin2=min(abs(y_l))-0.5
ymin=min(ymin1,ymin2)
plt.figure()
plt.xlim(2.0-delta-0.5,2.0+delta+0.5)
plt.ylim(ymin,ymax)
plt.plot(x_r,y_r, color='b')
plt.plot(x_l,y_l,color='r')
plt.scatter(x_r,y_r,color='k',s=30)
plt.scatter(x_l,y_l,color='k',s=30)
plt.scatter(2.0,g(2.0),c='y',s=40)
plt.title ('Example of Convergence to a Functional Value')
plt.xlabel('x-axis')
plt.ylabel('y-axis')

# The black points were computed. The yellow point marks the limit.
plt.show()
```



Exercise 2: Generalize the code which was used to determine a limiting value at infinity. Apply to Lial Section 11.1 Example 11.

```
from numpy import arange
import matplotlib.pyplot as plt

def f(x):
        f = (12.0*x**2-15.0*x+12.0)/(x**2+1.0)
        return f
# This code will be generalized using arrays and an increment of 0.25.
# For simplicity, equal intervals between calculated points will be used.

x = arange(0,12,0.25) #This defines the domain for x.
y=f(x)
z=12.0+0.0*x #This is defined to show a horizontal line at the limit of 12.0.
plt.figure()
plt.xlim(-1,14)
plt.ylim(0,15.0)
plt.plot(x,z, color='b', label='limit is 12.0')
plt.legend(['limit is 12.0'],'best')
plt.plot(x,y, color='r')
plt.scatter(x,y,color='k',s=20)
plt.scatter(number,y[-1],c='y',s=40)
plt.title ('Example of Convergence to a Limit at Infinity')
plt.xlabel('x-axis')
plt.ylabel('y-axis')
plt.show() # The black points were computed.
```



14
12
10
8
6
4
2
0
0
2
4
6
8
10
12
14

Module 6 Practice 2

Exercise 1: Refer to Lial Section 11.3 Examples 3-5. Using the function der(), calculate approximate slopes for the functions given in Lial Section 11.3 Examples 3(b), 4(c), & 5. Use a small value for delta and evaluate der() at the points used in the examples. Round to 4 places.

```
from scipy.misc import derivative

delta=0.00001
def f(x):
        f= 2.0*x**2-5*x+40.0
        return f
print (round(derivative(f,4,delta),4))
def f(x):
        f=100.0+15.0*x-x**2
        return f
print (round(derivative(f,1,delta),4))
print (round(derivative(f,5,delta),4))
```

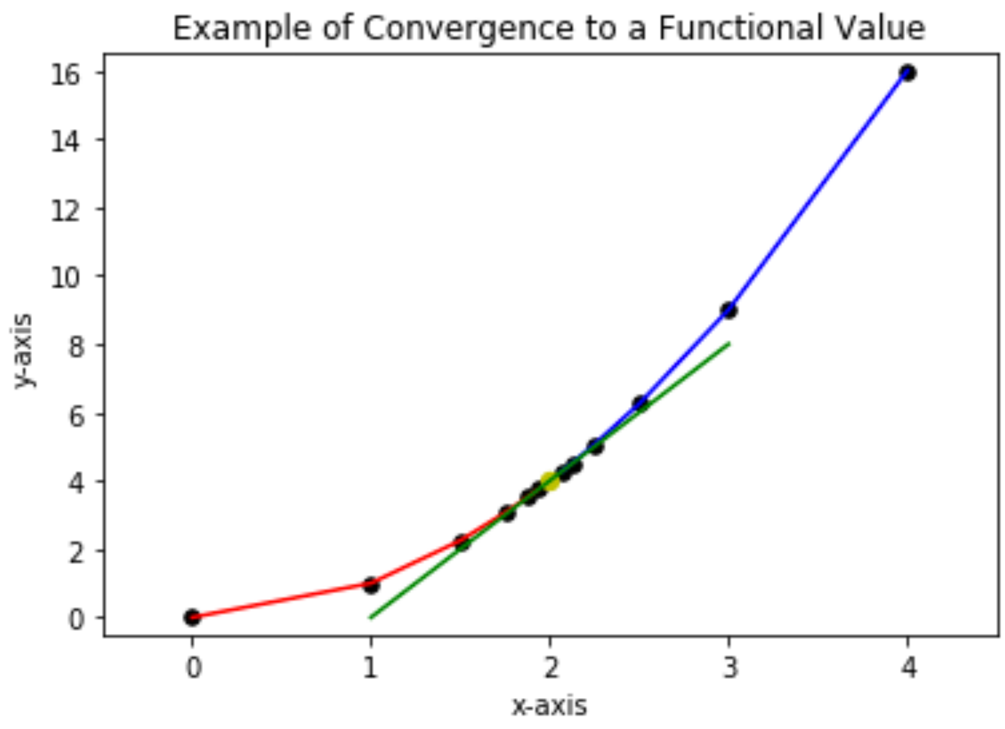
```
11.0
13.0
5.0
```

Exercise 2: Refer to Lial 11.1 Example 1. This example was used in Module 5 Practice 1. The solution code is presented in the answer sheet. Modify the code shown so that the tangent line at x=2 appears on the preceding plot. This necessitates determining the linear equation for the tangent line using y=mx+b and writing the statements to plot this line on the existing plot. Use arange().

Using the code solution for Lial Section 11.1 Example 1, add the following statements prior to the other plotting statements for the previous plot. The resulting plot with green tangent line is shown below.

```
from numpy import arange
import matplotlib.pyplot as plt
from scipy.misc import derivative #add this to compute the derivative

def g(x):
        g = (x*x)
        return g
n=5
powers=arange(0,n+1)
denominator=2.0**powers
delta=2.0



x_r=2.0+delta/denominator
y_r=g(x_r)
x_l=2.0-delta/denominator
y_l=g(x_l)
```

```
ymax1=max(abs(y_r))+0.5
ymax2=max(abs(y_l))+0.5
ymax=max(ymax1,ymax2)
ymin1=min(abs(y_r))-0.5
ymin2=min(abs(y_l))-0.5
ymin=min(ymin1,ymin2)
plt.figure()
plt.xlim(2.0-delta-0.5,2.0+delta+0.5)
plt.ylim(ymin,ymax)
plt.plot(x_r,y_r, color='b')
plt.plot(x_l,y_l,color='r')
plt.scatter(x_r,y_r,color='k',s=30)
plt.scatter(x_l,y_l,color='k',s=30)
plt.scatter(2.0,g(2.0),c='y',s=40)
plt.title ('Example of Convergence to a Functional Value')
plt.xlabel('x-axis')
plt.ylabel('y-axis')

#Additional code to compute and graph the derivative at x = 2.
def derivative(x,delta):
        delta = float(delta)
        if delta < (0.0000001):
                print ('Value chosen for delta is too small.')
                return 1/delta
        else:
                slope = (f(x + delta) - f(x))/delta
                return slope
def f(x):
        f=x*x
        return f
point=2.0
limit=derivative(2.0,0.00001)
w=arange(point-1.0,point+1.1,0.1)
t=f(point)+limit*(w-point)
plt.plot(w,t,color='g')
```


Example of Convergence to a Functional Value
y-axis
x-axis
0
2
4
6
8
10
12
14
16
0
1
2
3
4