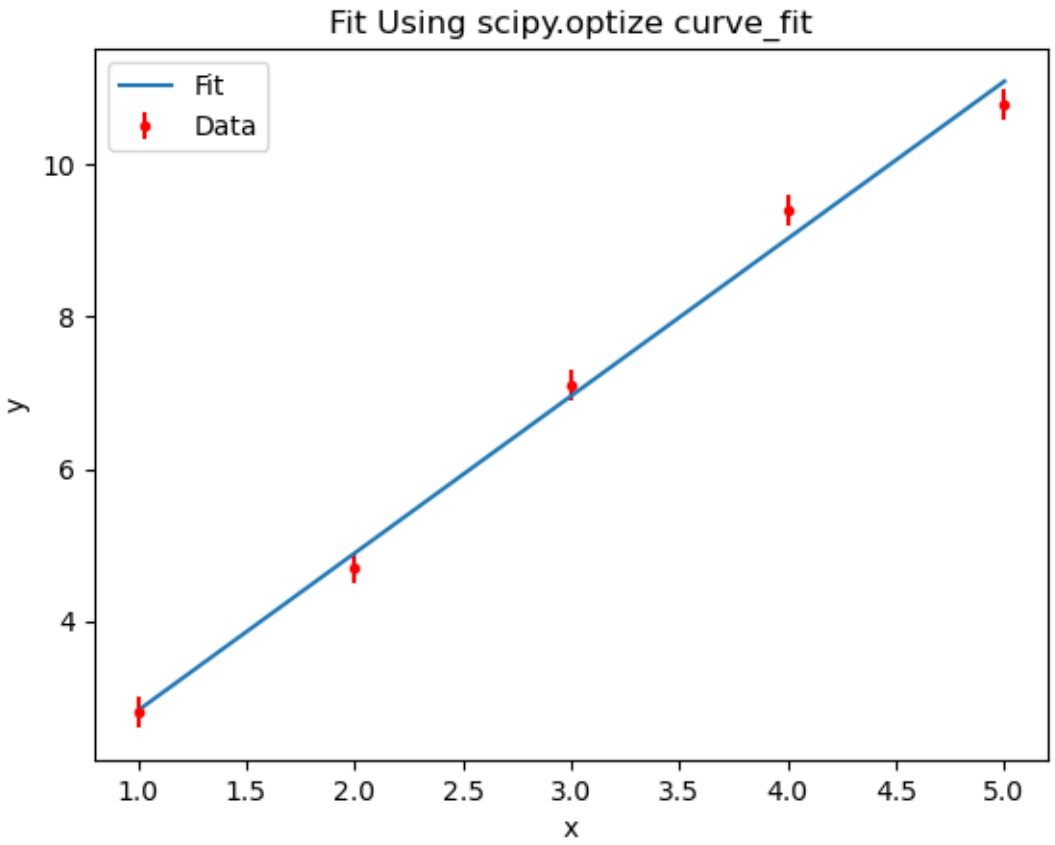
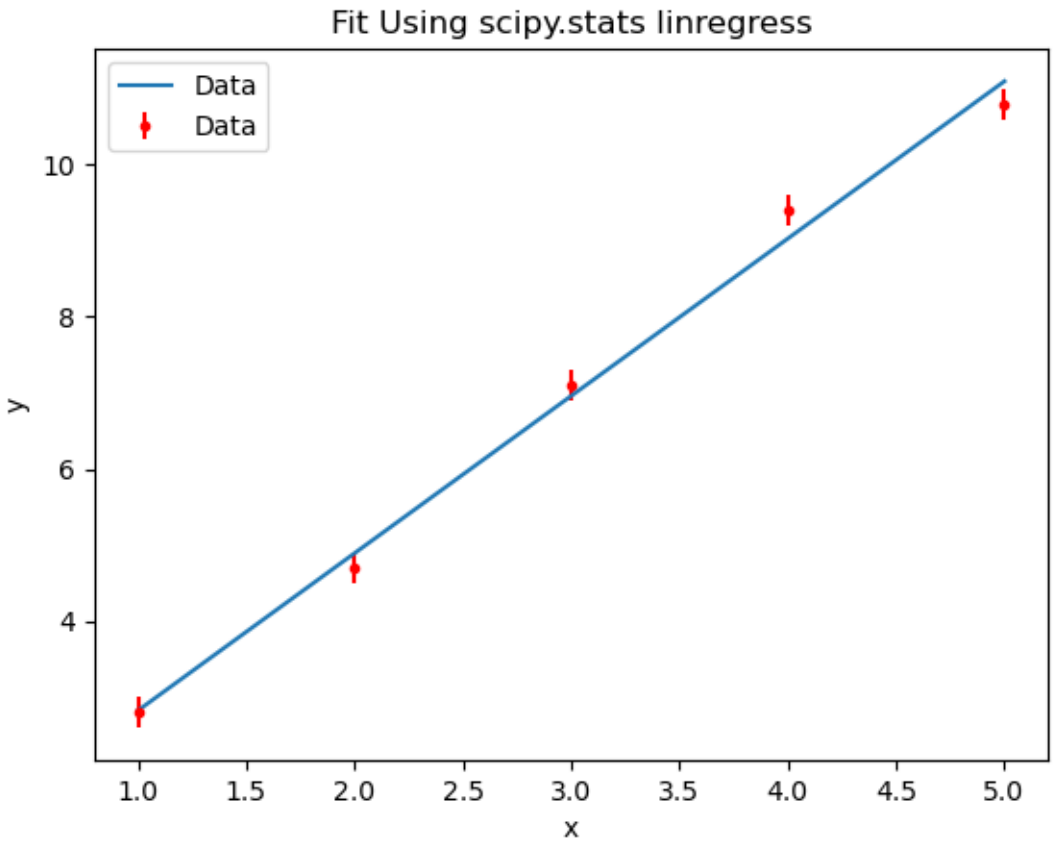
# Best_Fit_w_Python

February 14, 2024

[38]:
```
import matplotlib.pyplot as plt
import numpy as np
from scipy.optimize import curve_fit
from scipy.stats import linregress

def linearFunc(x,intercept,slope):
    """This function defines the function to be fit. In this case a linear
    function.

    Parameters
    ----------
    x : independent variable
    slope : slope
    intercept : intercept

    Returns
    -------
    y : dependent variable
    """
    y = intercept + slope * x
    return y


x=np.array([1.00,2.00,3.00,4.00,5.00])
y=np.array([2.8,4.7,7.1,9.4,10.8])
d=np.array([.2,.2,.2,.2,.2])

a_fit,cov=curve_fit(linearFunc,x,y,sigma=d,absolute_sigma=True)

inter = a_fit[0]
slope1 = a_fit[1]
d_inter = np.sqrt(cov[0][0])
d_slope = np.sqrt(cov[1][1])

# Create a graph showing the data.
plt.errorbar(x,y,yerr=d,fmt='r.',label='Data')
```

```
# Compute a best fit line from the fit intercept and slope.
yfit = inter + slope*x

yfit2=linregress(x,y)
yfit2=yfit2.intercept+yfit2.slope*x
# Create a graph of the fit to the data. We just use the ordinary plot
# command for this.
plt.title('Fit Using scipy.optize curve_fit')
plt.plot(x,yfit,label='Fit')

# Display a legend, label the x and y axes and title the graph.
plt.legend()
plt.xlabel('x')
plt.ylabel('y')

# Show the graph in a new window on the users screen.
plt.show()


#Shows fit using scipy.stats linregress
plt.title('Fit Using scipy.stats linregress')
plt.errorbar(x,y,yerr=d,fmt='r.',label='Data')
plt.plot(x,yfit2, label='Data')
plt.legend()
plt.xlabel('x')
plt.ylabel('y')
plt.show()

print(f'{inter:.4f} , {slope1:.4f} , {d_inter:.4f} , {d_slope:.4f}')
```


Fit Using scipy.optize curve_fit
Fit
Data
10
8
6
4
y
1.0
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
x




```
0.7500 , 2.0700 , 0.2098 , 0.0632
```

```
[ ]:
```