H2D2 Spectroscopy

James Amidei, Zach Stedman, And Max Markgraf

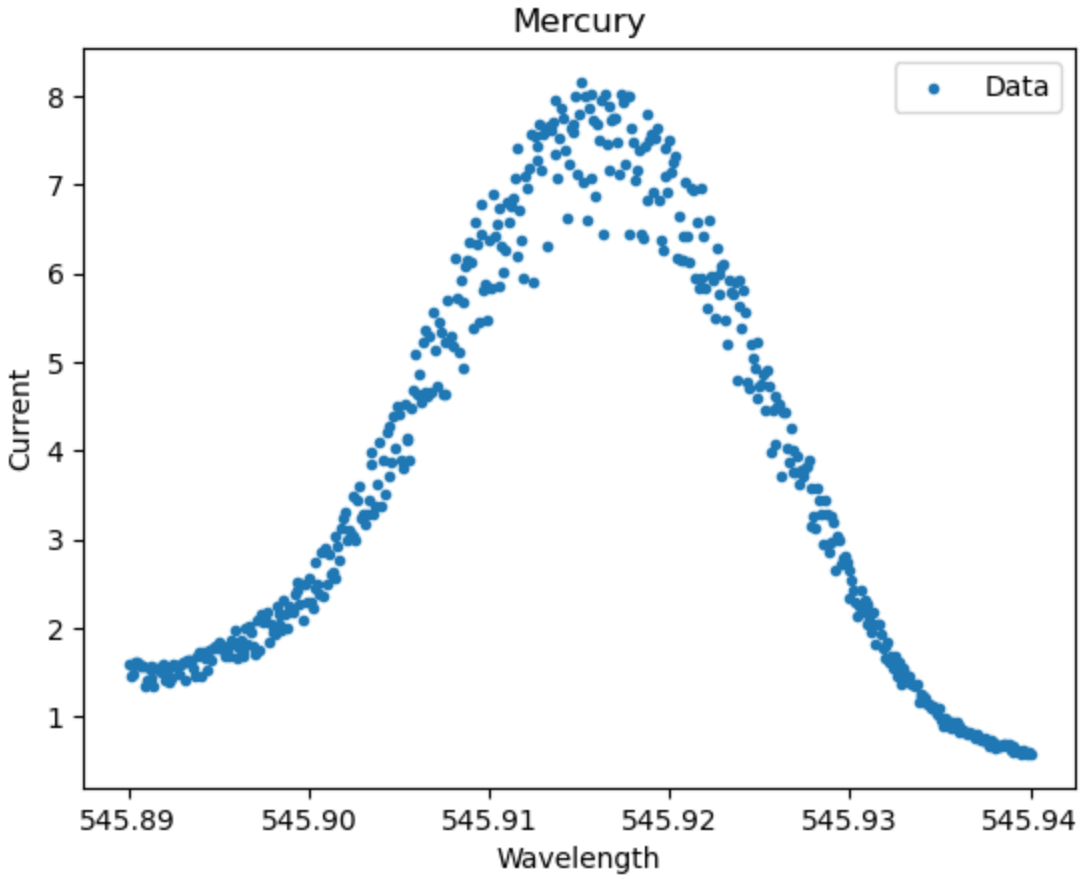
In [81]:

```
import numpy as np
import pylab as py
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit
import scipy.stats as stats

def fgaussian(x, A, B, C, D):
    return A * np.exp(-((x - B) ** 2) / (2 * C ** 2)) + D

def ftwogaussian(x, A1, A2, B1, B2, C1, C2, D):
    return (A1 * np.exp(-((x - B1) ** 2) / (2 * C1 ** 2))
            + A2 * np.exp(-((x - B2) ** 2) / (2 * C2 ** 2)) + D)

def is_float(string):
    try:
        float(string)
        return True
    except ValueError:
        return False

# pull data
#data1 = np.genfromtxt('MercuryData.csv', delimiter=',', skip_header=22, d
datam = np.genfromtxt('DataFiles/MercuryData.csv', delimiter=',', dtype=st
datam_copy = np.genfromtxt('DataFiles/MercuryDataCopy.csv', delimiter=',',

data1 =  np.genfromtxt('DataFiles/H2D2Data1.csv', delimiter=',', dtype=str
data1_peak1 =  np.genfromtxt('DataFiles/H2D2Data1Peak1.csv', delimiter=','
data1_peak2 =  np.genfromtxt('DataFiles/H2D2Data1Peak2.csv', delimiter=','

data2 =  np.genfromtxt('DataFiles/H2D2Data2.csv', delimiter=',', dtype=str
data2_peak1 =  np.genfromtxt('DataFiles/H2D2Data2Peak1.csv', delimiter=','
data2_peak2 =  np.genfromtxt('DataFiles/H2D2Data2Peak2.csv', delimiter=','

data3 =  np.genfromtxt('DataFiles/H2D2Data3.csv', delimiter=',', dtype=str
data3_peak1 =  np.genfromtxt('DataFiles/H2D2Data3Peak1.csv', delimiter=','
data3_peak2 =  np.genfromtxt('DataFiles/H2D2Data3Peak2.csv', delimiter=','

data4 =  np.genfromtxt('DataFiles/H2D2Data4.csv', delimiter=',', dtype=str
data4_peak1 =  np.genfromtxt('DataFiles/H2D2Data4Peak1.csv', delimiter=','
data4_peak2 =  np.genfromtxt('DataFiles/H2D2Data4Peak2.csv', delimiter=','

# split columns from data into x and y values
x_data_m = [float(row[0]) if is_float(row[0]) else np.nan for row in datam
y_data_m = [float(row[1]) if is_float(row[1]) else np.nan for row in datam
x_data_m_Peak1 = [float(row[0]) if is_float(row[0]) else np.nan for row in
y_data_m_Peak1 = [float(row[1]) if is_float(row[1]) else np.nan for row in

x_data_1 = [float(row[0]) if is_float(row[0]) else np.nan for row in data1
y_data_1 = [float(row[1]) if is_float(row[1]) else np.nan for row in data1
x_data_1_peak1 = [float(row[0]) if is_float(row[0]) else np.nan for row in
y_data_1_peak1 = [float(row[1]) if is_float(row[1]) else np.nan for row in
x_data_1_peak2 = [float(row[0]) if is_float(row[0]) else np.nan for row in
y_data_1_peak2 = [float(row[1]) if is_float(row[1]) else np.nan for row in

x_data_2 = [float(row[0]) if is_float(row[0]) else np.nan for row in data2
y_data_2 = [float(row[1]) if is_float(row[1]) else np.nan for row in data2
x_data_2_peak1 = [float(row[0]) if is_float(row[0]) else np.nan for row in
```

```
y_data_2_peak1 = [float(row[1]) if is_float(row[1]) else np.nan for row in
x_data_2_peak2 = [float(row[0]) if is_float(row[0]) else np.nan for row in
y_data_2_peak2 = [float(row[1]) if is_float(row[1]) else np.nan for row in

x_data_3 = [float(row[0]) if is_float(row[0]) else np.nan for row in data3
y_data_3 = [float(row[1]) if is_float(row[1]) else np.nan for row in data3
x_data_3_peak1 = [float(row[0]) if is_float(row[0]) else np.nan for row in
y_data_3_peak1 = [float(row[1]) if is_float(row[1]) else np.nan for row in
x_data_3_peak2 = [float(row[0]) if is_float(row[0]) else np.nan for row in
y_data_3_peak2 = [float(row[1]) if is_float(row[1]) else np.nan for row in

x_data_4 = [float(row[0]) if is_float(row[0]) else np.nan for row in data4
y_data_4 = [float(row[1]) if is_float(row[1]) else np.nan for row in data4
x_data_4_peak1 = [float(row[0]) if is_float(row[0]) else np.nan for row in
y_data_4_peak1 = [float(row[1]) if is_float(row[1]) else np.nan for row in
x_data_4_peak2 = [float(row[0]) if is_float(row[0]) else np.nan for row in
y_data_4_peak2 = [float(row[1]) if is_float(row[1]) else np.nan for row in
```

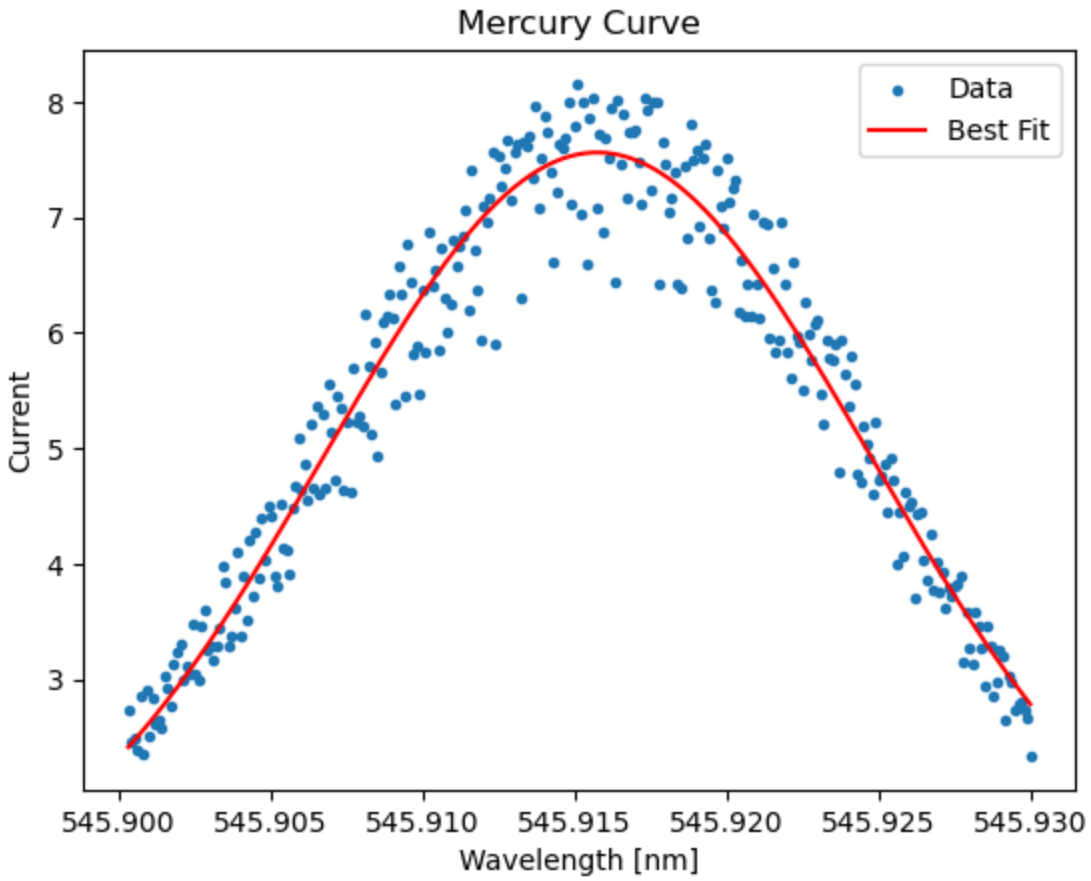
Mercury Calibration

In [82]:
```
# plot all data

plt.scatter(x_data_m, y_data_m, label='Data', marker='.')
plt.xlabel('Wavelength')
plt.ylabel('Current')
plt.title('Mercury')
plt.legend()
plt.show()
```



In [83]:

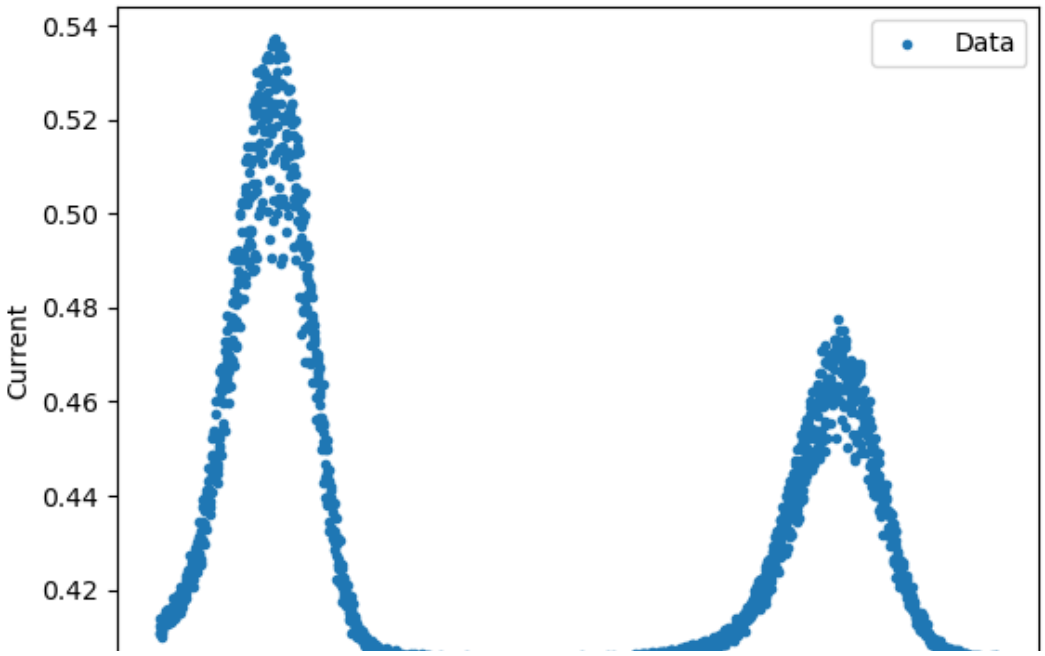
```
#x_min = 545.901
#x_max = 545.93
A1 = 7
B1 = 545.91
C1 = 0.01
D = 1

params, covariance = curve_fit(fgaussian, x_data_m_Peak1, y_data_m_Peak1,
                               p0=[A1, B1, C1, D])

A1_fit, B1_fit, C1_fit, D_fit = params
uncert = np.sqrt(np.diag(covariance))

plt.scatter(x_data_m_Peak1, y_data_m_Peak1, label='Data', marker='.')
plt.plot(x_data_m_Peak1, fgaussian(x_data_m_Peak1, *params), label='Best F
plt.xlabel('Wavelength [nm]')
plt.ylabel('Current')
plt.title('Mercury Curve')
plt.legend()
plt.show()

print('Peak 1 (0.081 MeV):')
print()
print(f'A1 = {A1_fit:.8f} ± {uncert[0]:.8f}')
print(f'B1 = {B1_fit:.8f} ± {uncert[1]:.8f}')
print(f'C1 = {C1_fit:.8f} ± {uncert[2]:.8f}')
print(f'D = {D_fit:.8f} ± {uncert[3]:.8f}')
```



```
Peak 1 (0.081 MeV):

A1 = 6.68835828 ± 0.31518634
B1 = 545.91574142 ± 0.00006202
C1 = 0.00900771 ± 0.00038161
D = 0.87393753 ± 0.33408648
```

656.279 nm

In [84]:
```
plt.scatter(x_data_1, y_data_1, label='Data', marker='.')
plt.xlabel('Wavelength')
plt.ylabel('Current')
plt.title(' ')
plt.legend()
plt.show()
```



Peak 1

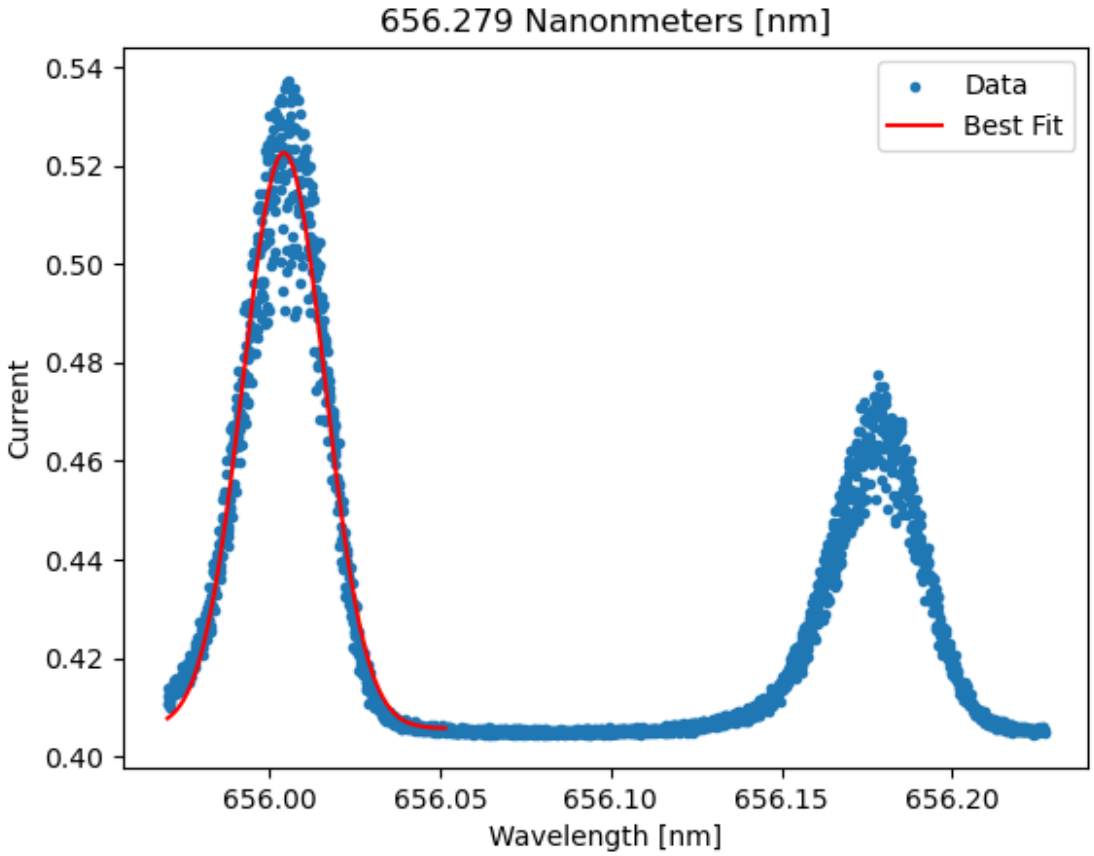
(655.95, 656.05)

Peak 2

(656.10, 656.25)

In [85]:

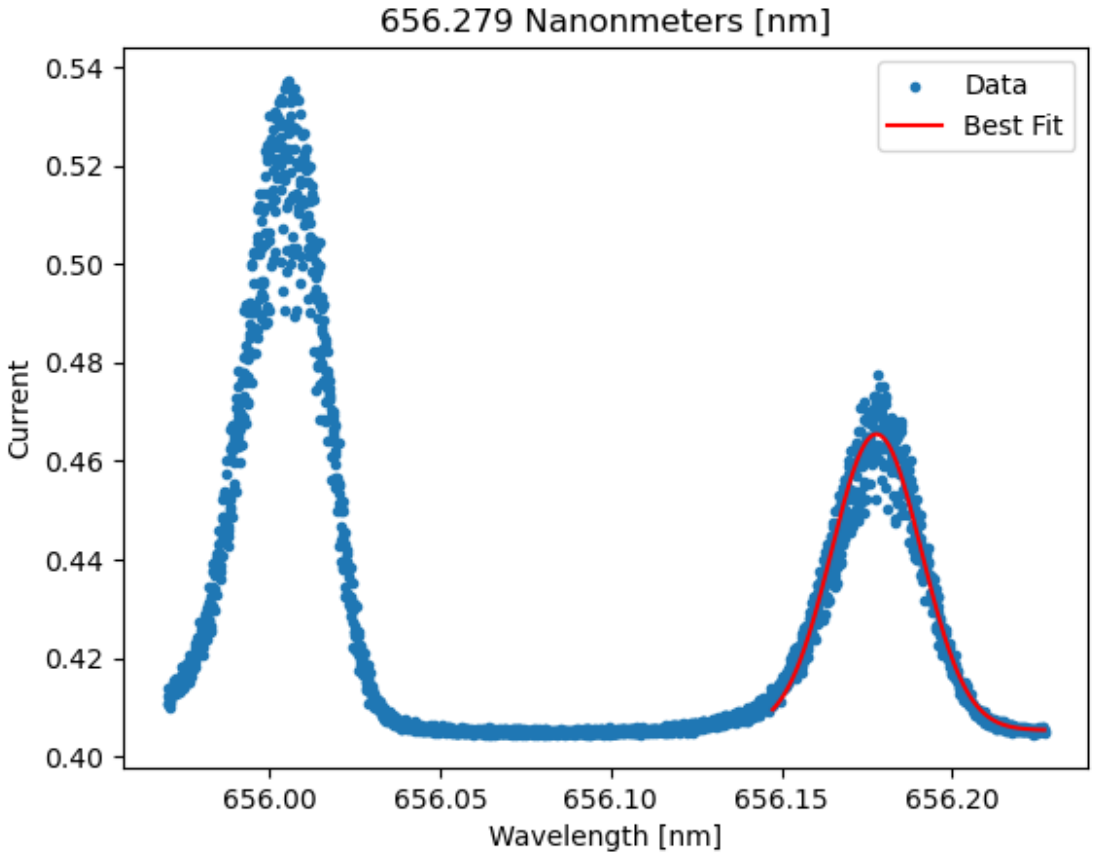
```
A1 = 0.54
B1 = 656
C1 = .01
D = 1

params, covariance = curve_fit(fgaussian, x_data_1_peak1, y_data_1_peak1,
                               p0=[A1, B1, C1, D])

A1_fit, B1_fit, C1_fit, D_fit = params
uncert = np.sqrt(np.diag(covariance))

plt.scatter(x_data_1, y_data_1, label='Data', marker='.')
plt.plot(x_data_1_peak1, fgaussian(x_data_1_peak1, *params), label='Best F
plt.xlabel('Wavelength [nm]')
plt.ylabel('Current')
plt.title('656.279 Nanonmeters [nm]')
plt.legend()
plt.show()

print('656.279 Nanonmeters [nm]')
print()
print(f'A1 = {A1_fit:.8f} ± {uncert[0]:.8f}')
print(f'B1 = {B1_fit:.8f} ± {uncert[1]:.8f}')
print(f'C1 = {C1_fit:.8f} ± {uncert[2]:.8f}')
```



```
656.279 Nanonmeters [nm]

A1 = 0.11682109 ± 0.00067805
B1 = 656.00413598 ± 0.00006824
C1 = 0.01199412 ± 0.00009666
```

In [71]:

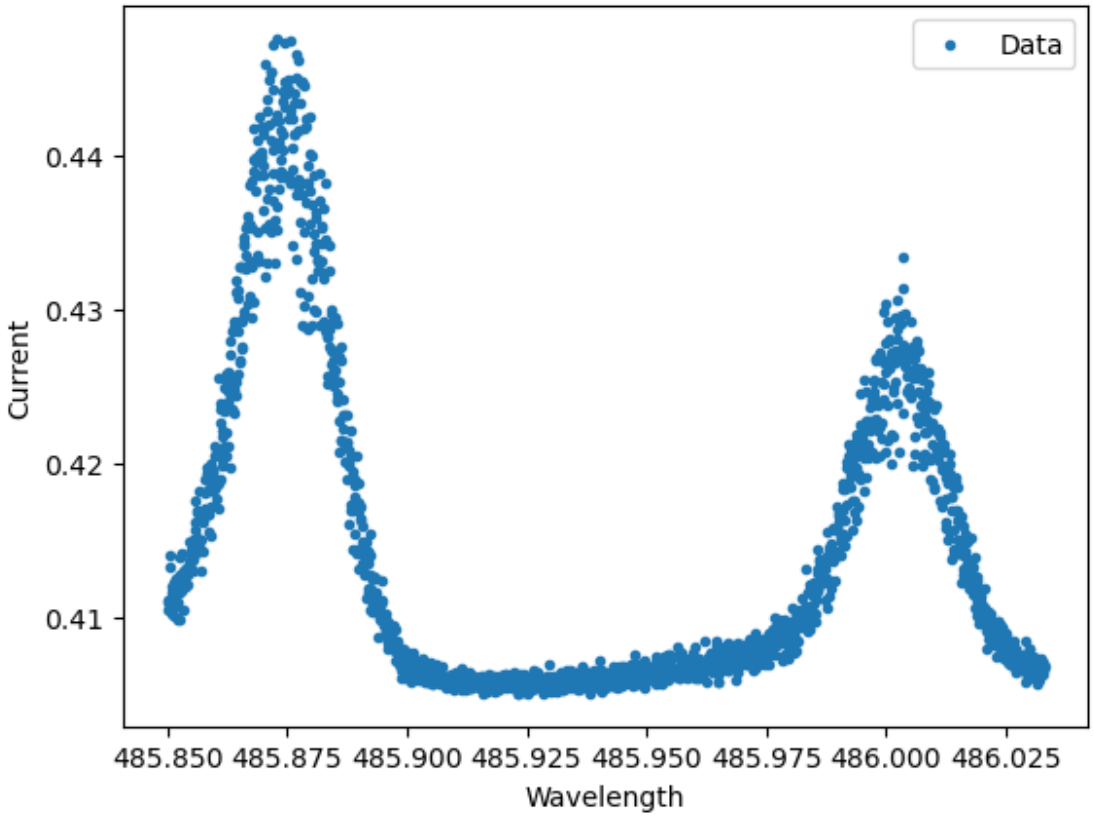
```
A1 = 0.54
B1 = 656.156
C1 = .05
D = 1

params, covariance = curve_fit(fgaussian, x_data_1_peak2, y_data_1_peak2,
                               p0=[A1, B1, C1, D])

A1_fit, B1_fit, C1_fit, D_fit = params
uncert = np.sqrt(np.diag(covariance))

plt.scatter(x_data_1, y_data_1, label='Data', marker='.')
plt.plot(x_data_1_peak2, fgaussian(x_data_1_peak2, *params), label='Best F
plt.xlabel('Wavelength [nm]')
plt.ylabel('Current')
plt.title('656.279 Nanonmeters [nm]')
plt.legend()
plt.show()

print('656.279 Nanonmeters [nm]')
print()
print(f'A1 = {A1_fit:.8f} ± {uncert[0]:.8f}')
print(f'B1 = {B1_fit:.8f} ± {uncert[1]:.8f}')
print(f'C1 = {C1_fit:.8f} ± {uncert[2]:.8f}')
```



```
656.279 Nanonmeters [nm]

A1 = 0.06007533 ± 0.00037762
B1 = 656.17789794 ± 0.00007609
C1 = 0.01316435 ± 0.00011315
```

486.135 nm

In [86]:
```
plt.scatter(x_data_2, y_data_2, label='Data', marker='.')
plt.xlabel('Wavelength')
plt.ylabel('Current')
plt.title('')
plt.legend()
plt.show()
```



Peak 1

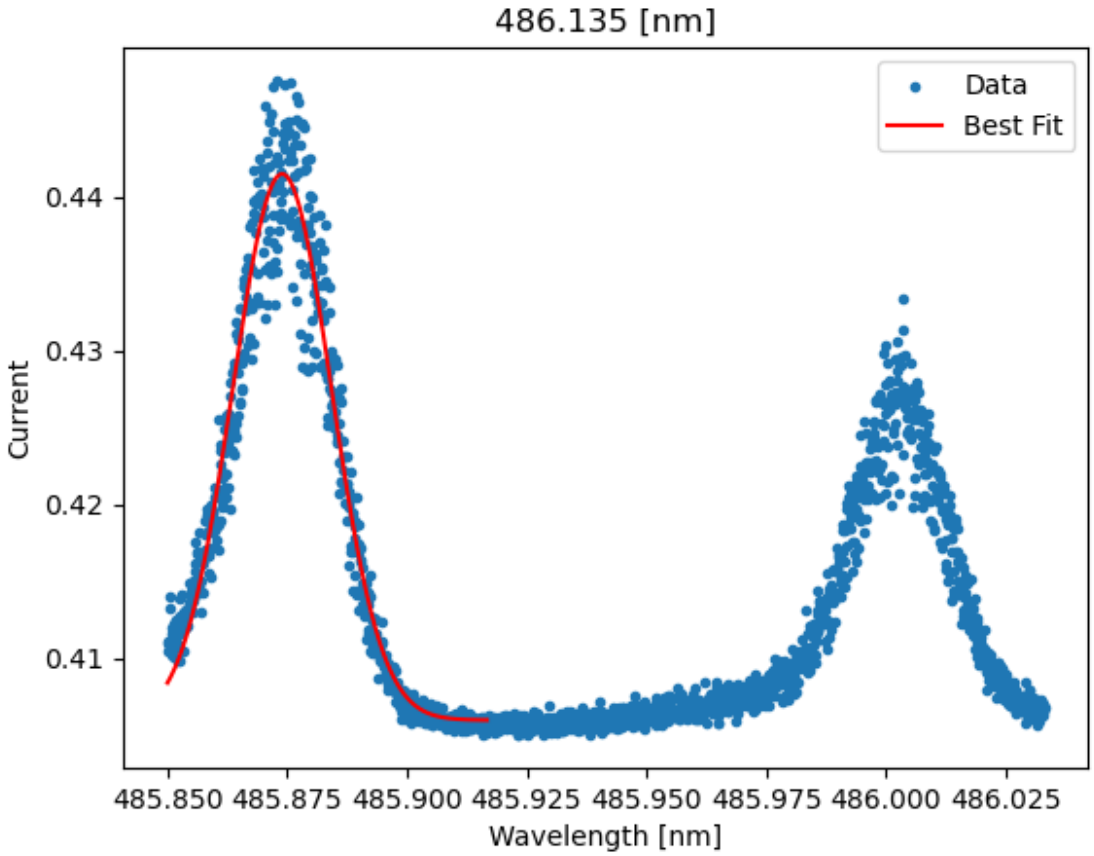
(485, 485.9)

Peak 2

(485.975, 486.025)

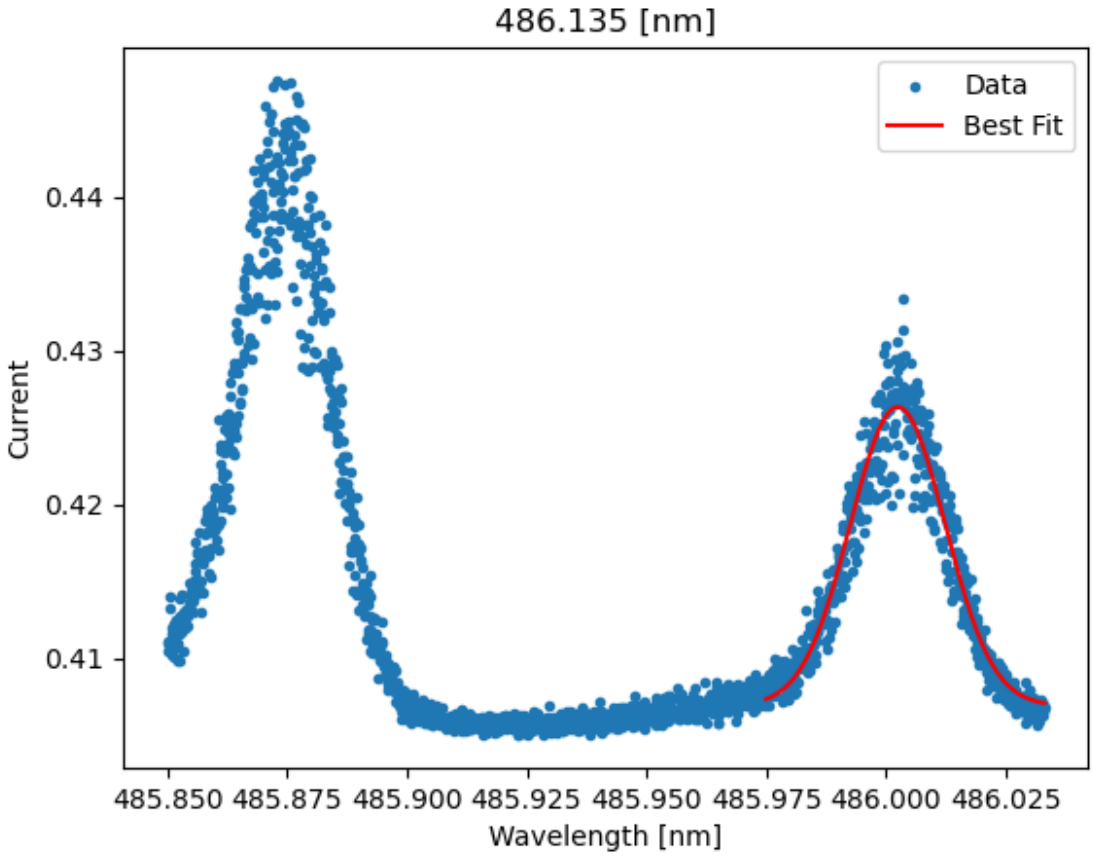
In [87]:

```
A1 = 0.45
B1 = 485.8
C1 = .05
D = 0

params, covariance = curve_fit(fgaussian, x_data_2_peak1, y_data_2_peak1,
                               p0=[A1, B1, C1, D])

A1_fit, B1_fit, C1_fit, D_fit = params
uncert = np.sqrt(np.diag(covariance))

plt.scatter(x_data_2, y_data_2, label='Data', marker='.')
plt.plot(x_data_2_peak1, fgaussian(x_data_2_peak1, *params), label='Best F
plt.xlabel('Wavelength [nm]')
plt.ylabel('Current')
plt.title('486.135 [nm]')
plt.legend()
plt.show()

print('486.135 [nm]')
print()
print(f'A1 = {A1_fit:.8f} ± {uncert[0]:.8f}')
print(f'B1 = {B1_fit:.8f} ± {uncert[1]:.8f}')
print(f'C1 = {C1_fit:.8f} ± {uncert[2]:.8f}')
```



```
486.135 [nm]

A1 = 0.03550120 ± 0.00025833
B1 = 485.87392625 ± 0.00007253
C1 = -0.01032817 ± 0.00010056
```

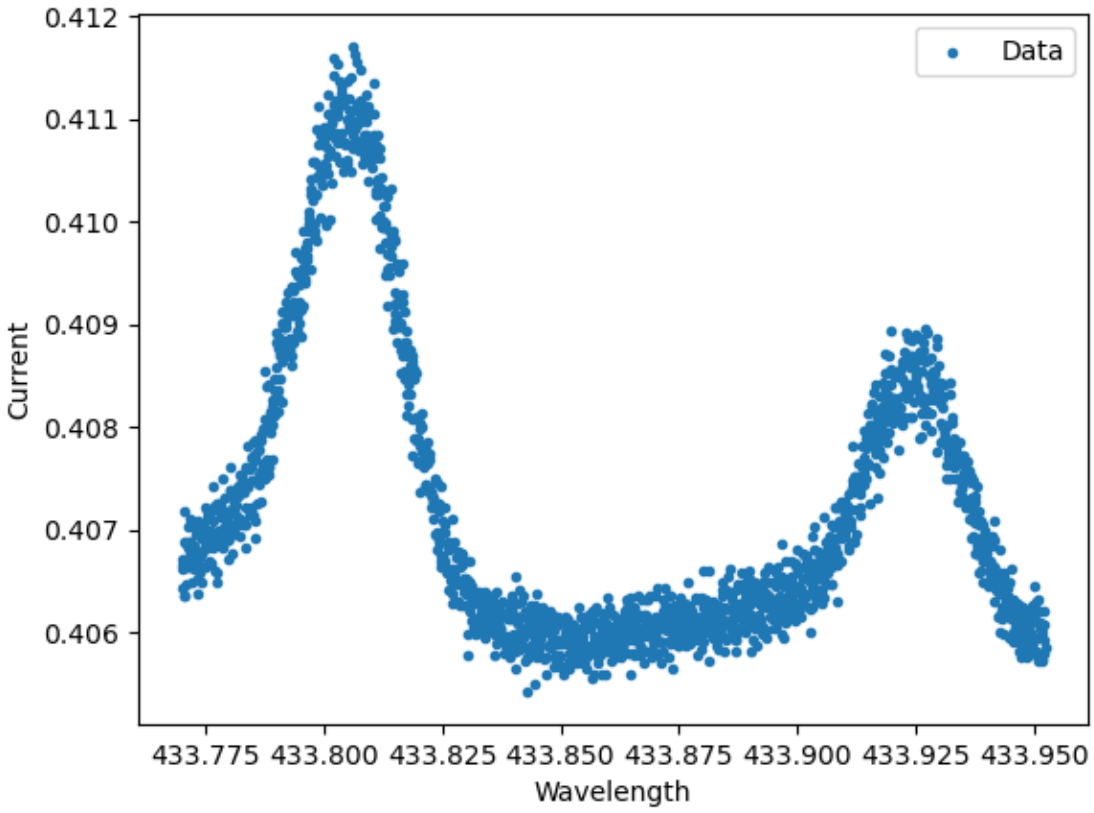
In [88]:
```
A1 = 0.43
B1 = 486
C1 = .02
D = 0

params, covariance = curve_fit(fgaussian, x_data_2_peak2, y_data_2_peak2,
                               p0=[A1, B1, C1, D])

A1_fit, B1_fit, C1_fit, D_fit = params
uncert = np.sqrt(np.diag(covariance))

plt.scatter(x_data_2, y_data_2, label='Data', marker='.')
plt.plot(x_data_2_peak2, fgaussian(x_data_2_peak2, *params), label='Best F
plt.xlabel('Wavelength [nm]')
plt.ylabel('Current')
plt.title('486.135 [nm]')
plt.legend()
plt.show()

print('486.135 [nm]')
print()
print(f'A1 = {A1_fit:.8f} ± {uncert[0]:.8f}')
print(f'B1 = {B1_fit:.8f} ± {uncert[1]:.8f}')
print(f'C1 = {C1_fit:.8f} ± {uncert[2]:.8f}')
```



```
486.135 [nm]

A1 = 0.01939941 ± 0.00020042
B1 = 486.00249349 ± 0.00008774
C1 = -0.00992384 ± 0.00015021
```

434.0472 nm

In [75]:
```
plt.scatter(x_data_3, y_data_3, label='Data', marker='.')
plt.xlabel('Wavelength')
plt.ylabel('Current')
plt.title('')
plt.legend()
plt.show()
```



Peak 1

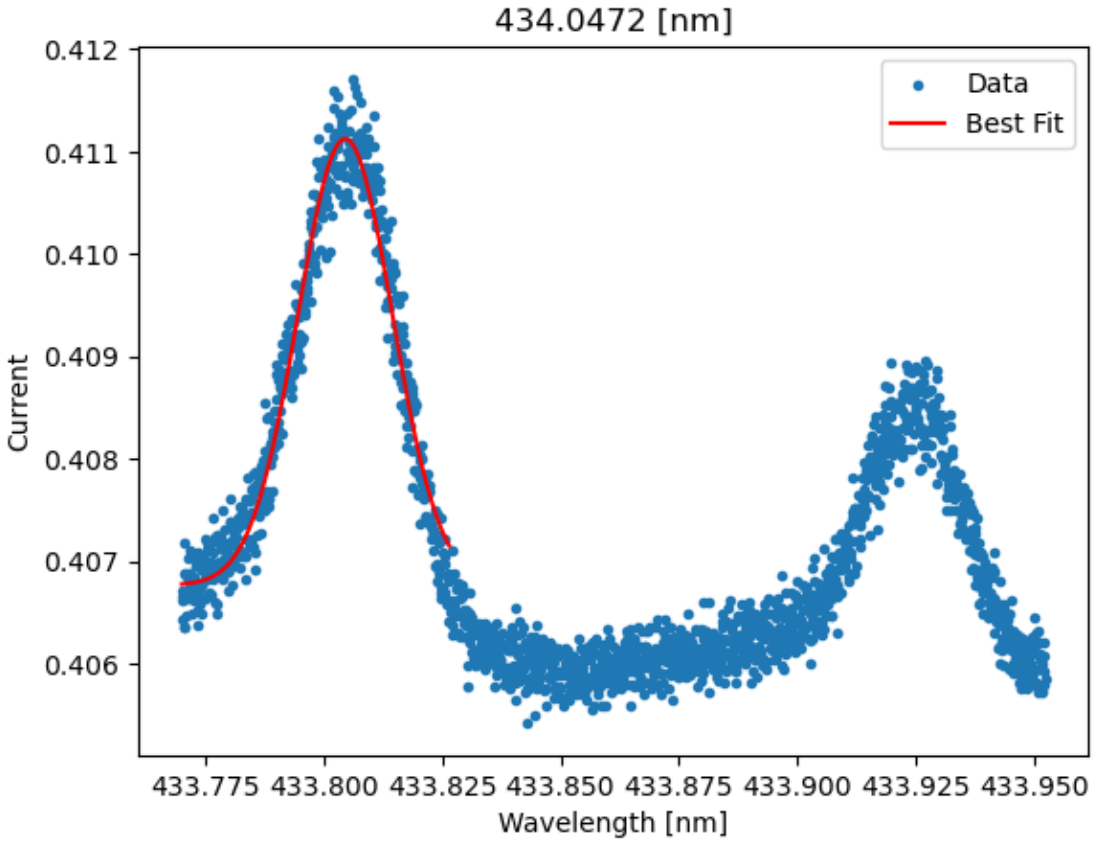
(433.775, 433.820)

Peak 2

(433.900, 433.950)

In [89]:

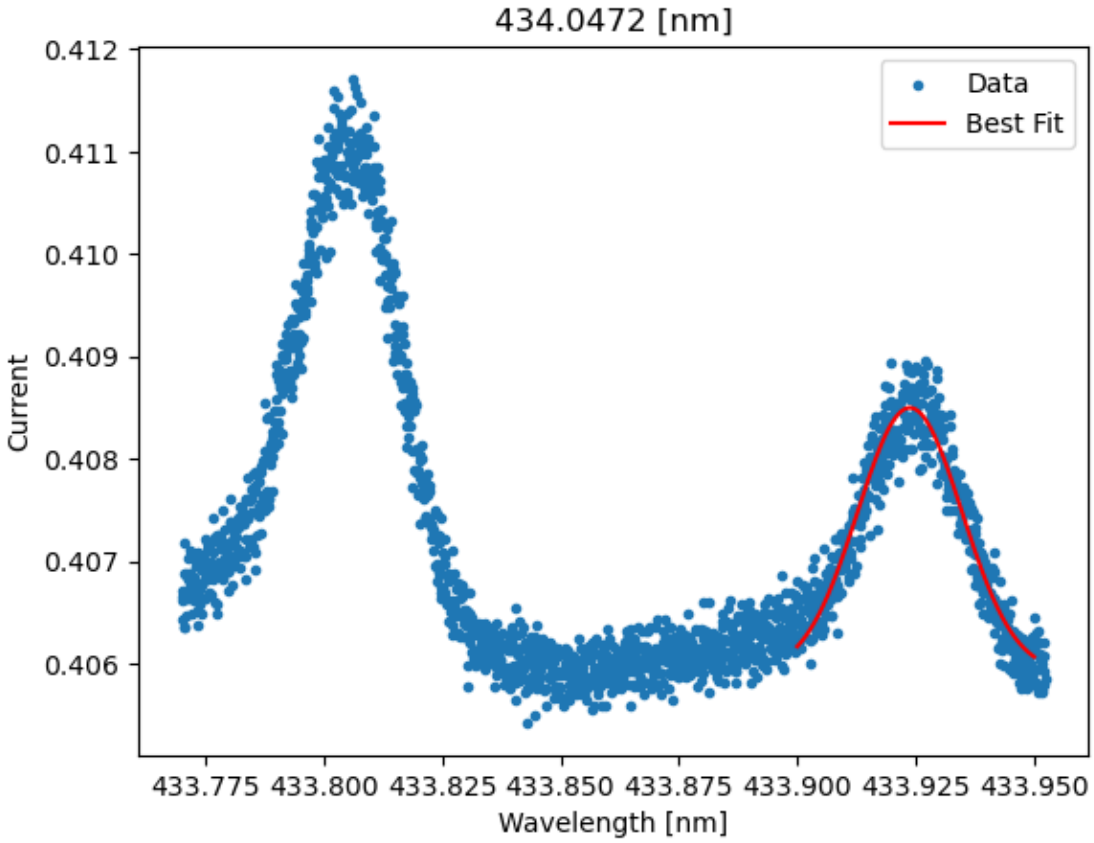
```
A1 = 0.412
B1 = 433.8
C1 = .1
D = 0

params, covariance = curve_fit(fgaussian, x_data_3_peak1, y_data_3_peak1,
                               p0=[A1, B1, C1, D])

A1_fit, B1_fit, C1_fit, D_fit = params
uncert = np.sqrt(np.diag(covariance))

plt.scatter(x_data_3, y_data_3, label='Data', marker='.')
plt.plot(x_data_3_peak1, fgaussian(x_data_3_peak1, *params), label='Best F
plt.xlabel('Wavelength [nm]')
plt.ylabel('Current')
plt.title('434.0472 [nm]')
plt.legend()
plt.show()

print('434.0472 [nm]')
print()
print(f'A1 = {A1_fit:.8f} ± {uncert[0]:.8f}')
print(f'B1 = {B1_fit:.8f} ± {uncert[1]:.8f}')
print(f'C1 = {C1_fit:.8f} ± {uncert[2]:.8f}')
```



```
434.0472 [nm]

A1 = 0.00435986 ± 0.00003734
B1 = 433.80452053 ± 0.00007416
C1 = 0.00997645 ± 0.00011759
```

In [90]:

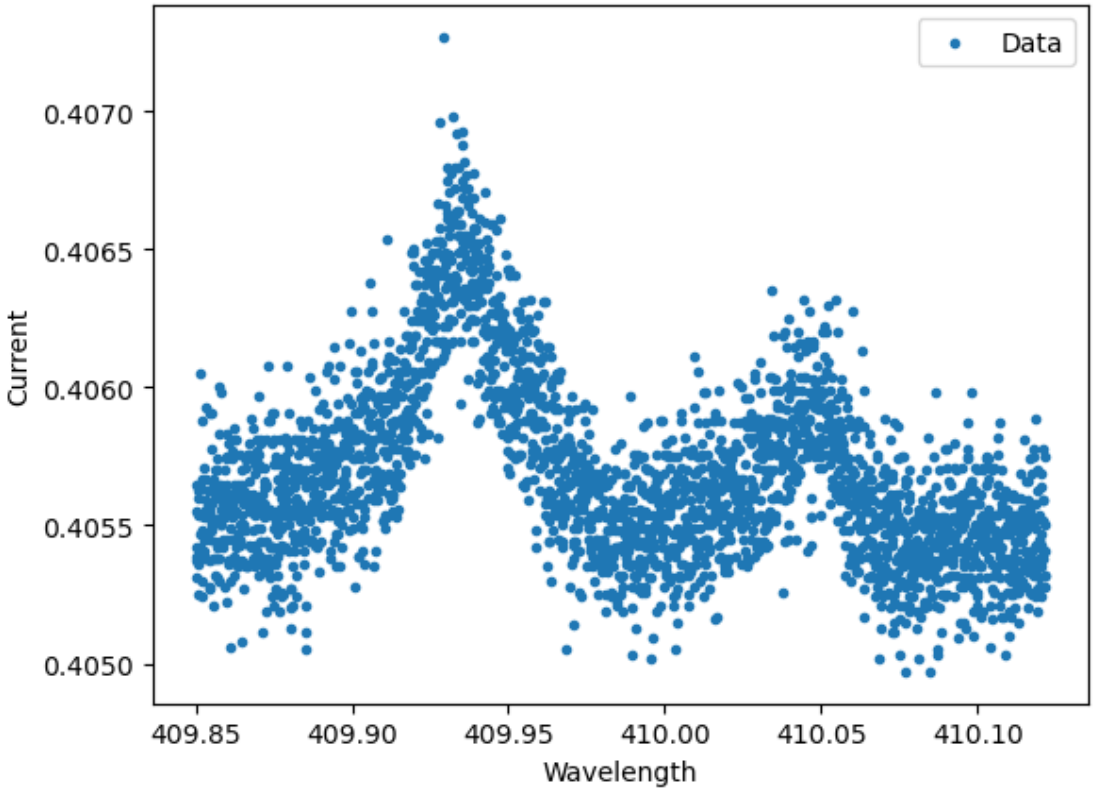
```
A1 = 0.43
B1 = 433.925
C1 = .02
D = 0

params, covariance = curve_fit(fgaussian, x_data_3_peak2, y_data_3_peak2,
                               p0=[A1, B1, C1, D])

A1_fit, B1_fit, C1_fit, D_fit = params
uncert = np.sqrt(np.diag(covariance))

plt.scatter(x_data_3, y_data_3, label='Data', marker='.')
plt.plot(x_data_3_peak2, fgaussian(x_data_3_peak2, *params), label='Best F
plt.xlabel('Wavelength [nm]')
plt.ylabel('Current')
plt.title('434.0472 [nm]')
plt.legend()
plt.show()

print('434.0472 [nm]')
print()
print(f'A1 = {A1_fit:.8f} ± {uncert[0]:.8f}')
print(f'B1 = {B1_fit:.8f} ± {uncert[1]:.8f}')
print(f'C1 = {C1_fit:.8f} ± {uncert[2]:.8f}')
```



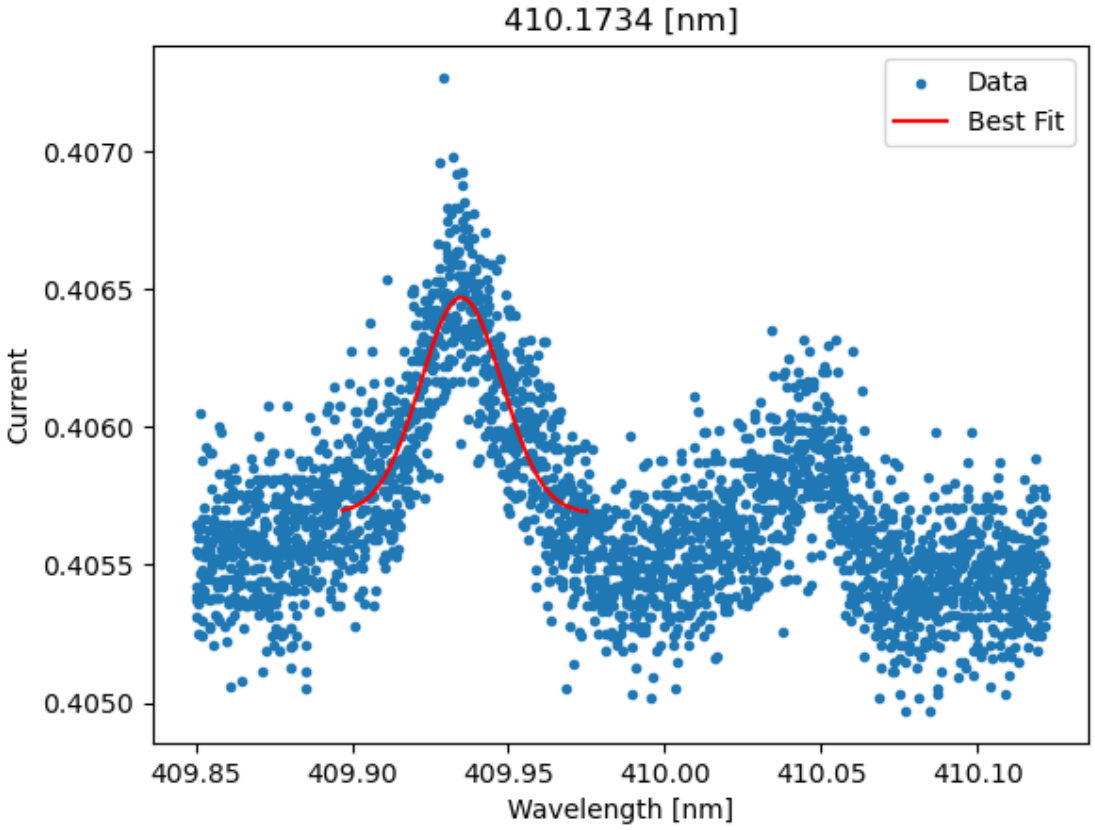
```
434.0472 [nm]

A1 = 0.00258533 ± 0.00005235
B1 = 433.92375671 ± 0.00010745
C1 = 0.01102500 ± 0.00029061
```

410.1734 nm

In [78]:
```
plt.scatter(x_data_4, y_data_4, label='Data', marker='.')
plt.xlabel('Wavelength')
plt.ylabel('Current')
plt.title('')
plt.legend()
plt.show()
```

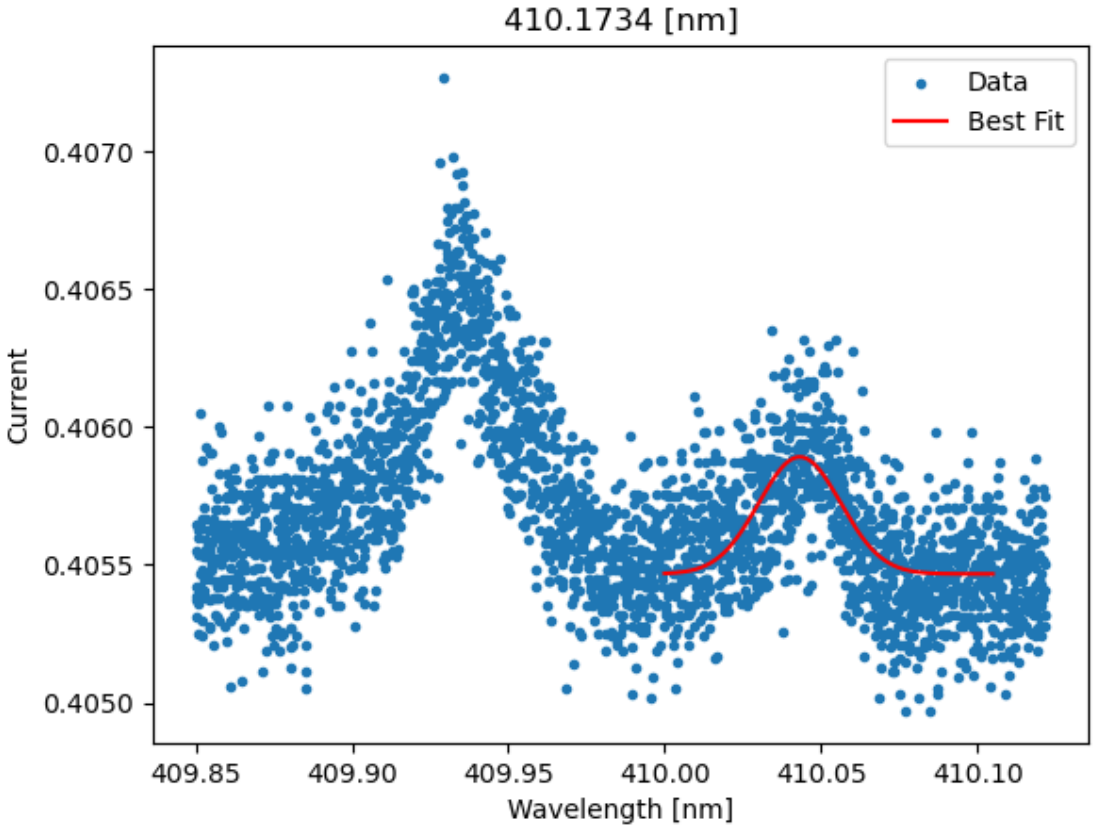


In [92]:
```
A1 = 0.407
B1 = 409.925
C1 = .01
D = 0

params, covariance = curve_fit(fgaussian, x_data_4_peak1, y_data_4_peak1,
                               p0=[A1, B1, C1, D])

A1_fit, B1_fit, C1_fit, D_fit = params
uncert = np.sqrt(np.diag(covariance))

plt.scatter(x_data_4, y_data_4, label='Data', marker='.')
plt.plot(x_data_4_peak1, fgaussian(x_data_4_peak1, *params), label='Best F
plt.xlabel('Wavelength [nm]')
plt.ylabel('Current')
plt.title('410.1734 [nm]')
plt.legend()
plt.show()

print('410.1734 [nm]')
print()
print(f'A1 = {A1_fit:.8f} ± {uncert[0]:.8f}')
print(f'B1 = {B1_fit:.8f} ± {uncert[1]:.8f}')
print(f'C1 = {C1_fit:.8f} ± {uncert[2]:.8f}')
```




```
410.1734 [nm]

A1 = 0.00078410 ± 0.00002220
B1 = 409.93494744 ± 0.00032382
C1 = 0.01299440 ± 0.00053964
```

In [91]:
```
A1 = 0.43
B1 = 410.05
C1 = .01
D = 0

params, covariance = curve_fit(fgaussian, x_data_4_peak2, y_data_4_peak2,
                               p0=[A1, B1, C1, D])

A1_fit, B1_fit, C1_fit, D_fit = params
uncert = np.sqrt(np.diag(covariance))

plt.scatter(x_data_4, y_data_4, label='Data', marker='.')
plt.plot(x_data_4_peak2, fgaussian(x_data_4_peak2, *params), label='Best F
plt.xlabel('Wavelength [nm]')
plt.ylabel('Current')
plt.title('410.1734 [nm]')
plt.legend()
plt.show()

print('410.1734 [nm]')
print()
print(f'A1 = {A1_fit:.8f} ± {uncert[0]:.8f}')
print(f'B1 = {B1_fit:.8f} ± {uncert[1]:.8f}')
print(f'C1 = {C1_fit:.8f} ± {uncert[2]:.8f}')
```




```
410.1734 [nm]

A1 = 0.00042482 ± 0.00001670
B1 = 410.04316769 ± 0.00053846
C1 = 0.01310418 ± 0.00068792
```

In [ ]:

In [ ]: