OPO_greenbeam_characterization

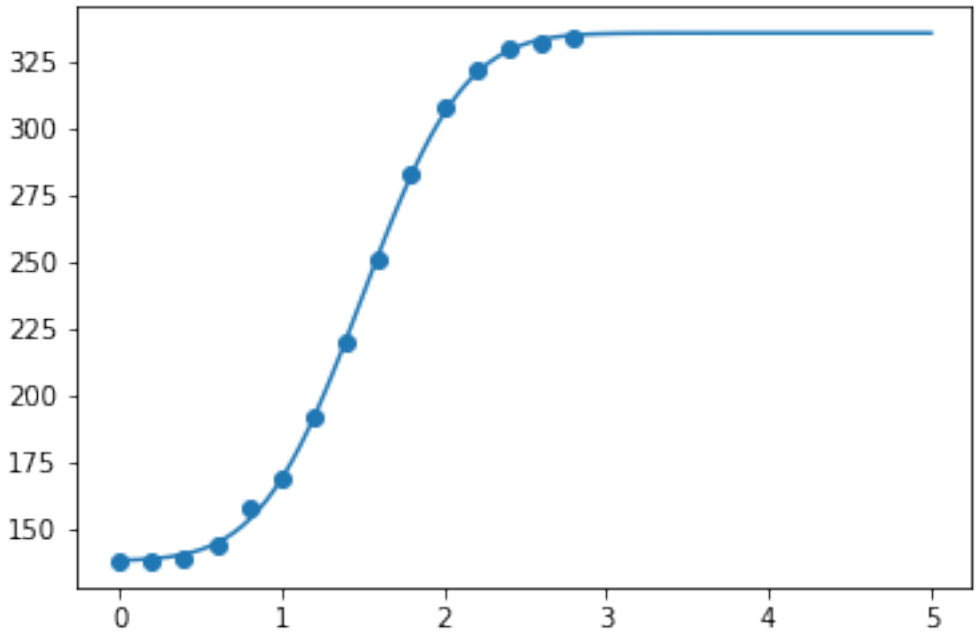
October 22, 2018

In [2]:
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit
from scipy.special import erf
```

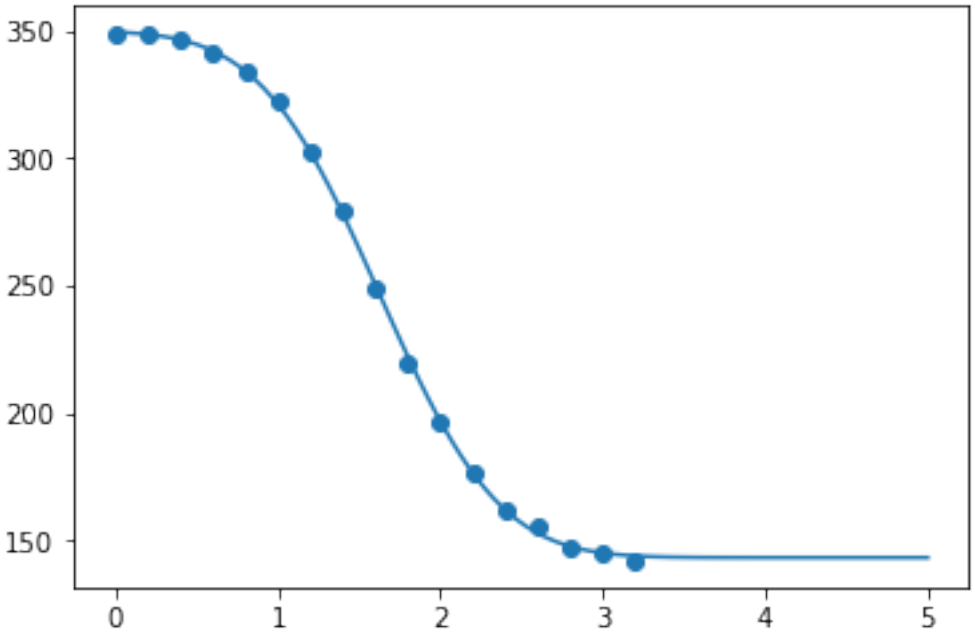
In [3]:
```
def errorfun(x, a, x0, w, o):
    return a*erf(np.sqrt(2)*(x-x0)/w)+o
```

In [18]:
```
# T1 is 12.5 cm from OPO
# position T5p is at 17 cm from T4. position 9 is 2 hole from position 7 and position
#from position 9 all the other are separated by 1 hole

T1 = np.linspace(0,2.8,15)
H1 = np.array([138,138,139,144,158,169,192,220,251,283,308,322,330,332,334], dtype=np
T2 = np.linspace(0,3.2,17)
H2 = np.array([349,349,347,341,334,322,303,279,249,220,197,177,162,156,147,145,142],d
T3 = np.linspace(0,3.4,18)
H3 = np.array([349,347,341,333,323,308,289,267,242,218,200,180,167,156,150,146,143,140
T4 = np.linspace(0,4,22)
H4 = np.array([345,345,344,343,339,335,328,317,304,290,267,247,224,206,189,173,160,153
                144,144,144],dtype=np.float64)
T5p= np.linspace(0,7.8,18)
H5p= np.array([343,343,339,329,317,301,280,254,229,205,181,165,153,144,141,139,138,138
T6 = np.linspace(0,9.6,24)
H6= np.array([346,346,346,345,343,341,339,331,320,300,278,254,229,205,185,169,157,152
              143,141,140,140],dtype=np.float64)
T7 = np.linspace(0,10.6,12)
H7 = np.array([343,343,335,324,301,265,219,178,154,143,142,142],dtype=np.float64)
T9 = np.linspace(0,10,10)
H9 = np.array([338,335,327,296,247,197,162,148,142,141],dtype=np.float64)
T11 = np.linspace(0,11,11)
H11 = np.array([342,341,334,318,288,243,200,167,146,140,138],dtype=np.float64)
#plt.xlabel('blade position [mm]')
#plt.ylabel('Power[a.u.]')
#plt.scatter(T11,H11)
#plt.plot(T11, 100*erf(-T11+5.4)+200)
#plt.show()
```

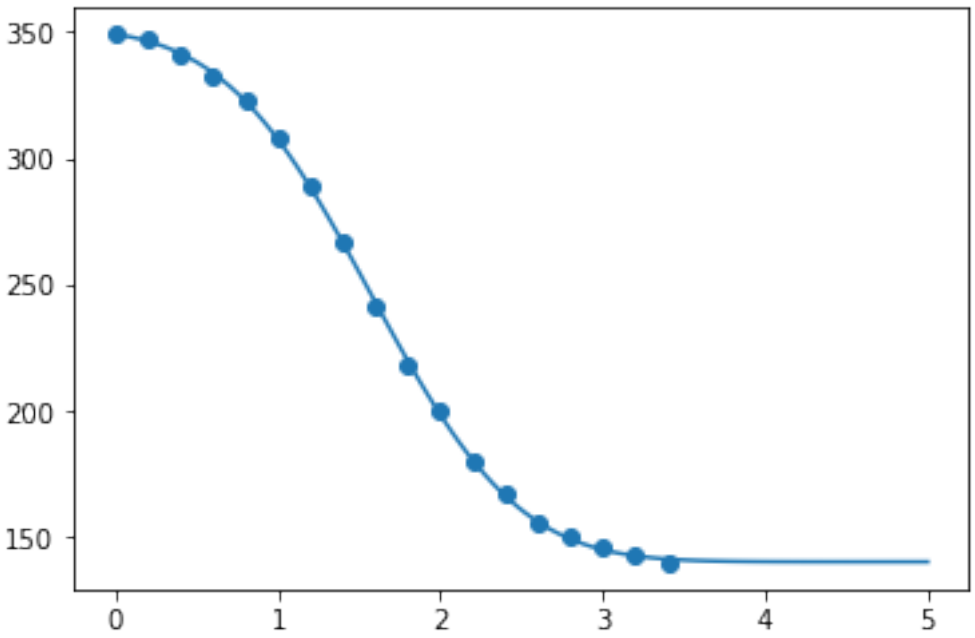
In [5]:
```
# position 1 (12.5 cm from OPO)

X = T1
Y = H1
popt1, pcov = curve_fit(errorfun, X, Y, bounds=([50, 0, 0, 150], [180, 4, 10, 290]))
#perr = np.sqrt(np.diag(pcov))
t = np.linspace(0, 5, 100)
plt.scatter(X, Y)
plt.plot(t, errorfun(t,*popt1))
plt.show()
```



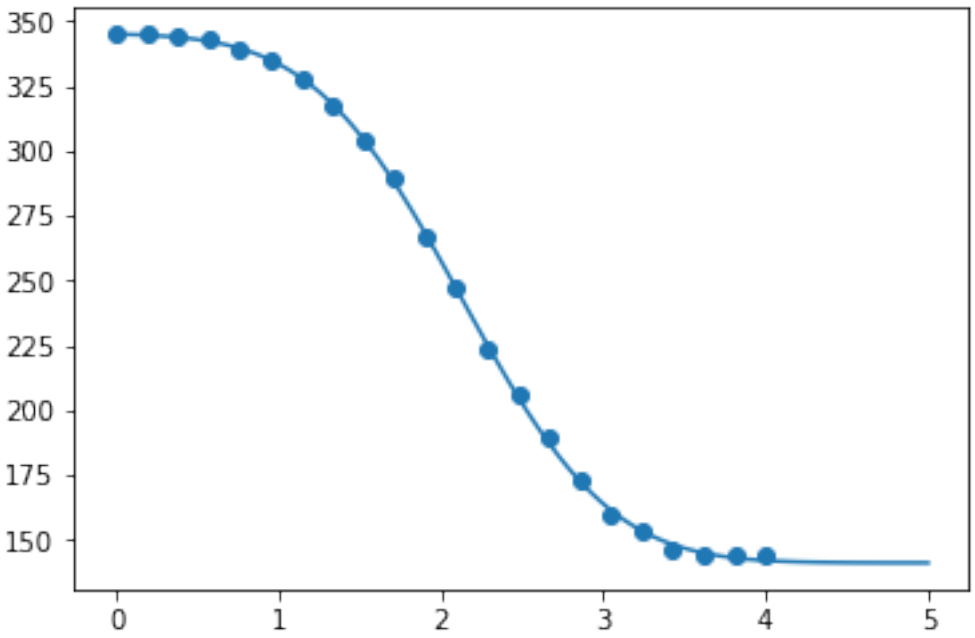
In [6]:
```
# position 2

X = T2
Y = H2
popt2, pcov = curve_fit(errorfun, X, Y, bounds=([50, 0, -10, 150], [180, 4, 0, 290]))
#perr = np.sqrt(np.diag(pcov))
t = np.linspace(0, 5, 100)
plt.scatter(X, Y)
plt.plot(t, errorfun(t,*popt2))
plt.show()
```



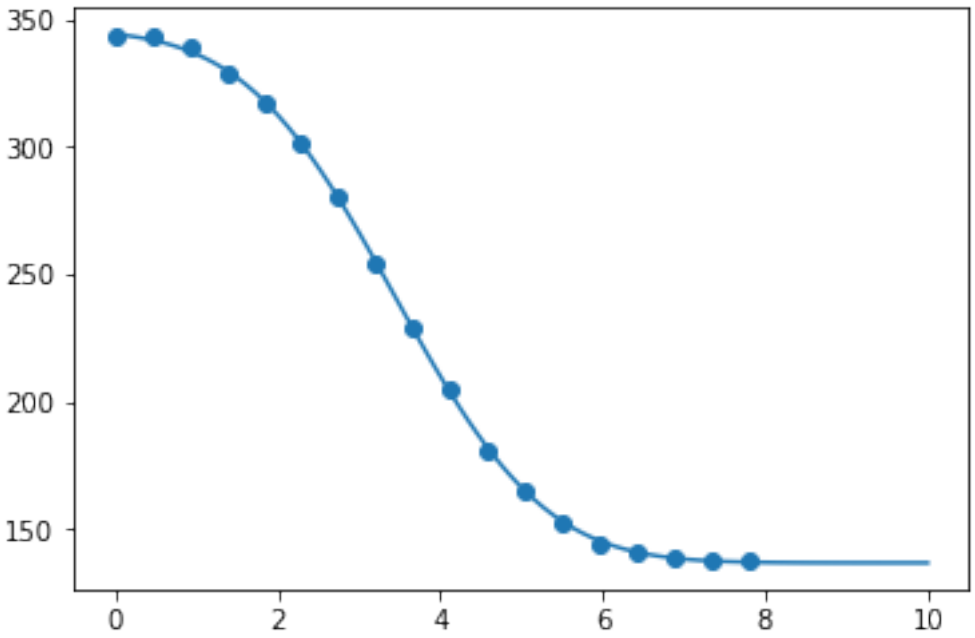
In [7]:
```
# position 3

X = T3
Y = H3
popt3, pcov = curve_fit(errorfun, X, Y, bounds=([50, 0, -10, 150], [180, 4, 0, 290]))
#perr = np.sqrt(np.diag(pcov))
t = np.linspace(0, 5, 100)
plt.scatter(X, Y)
plt.plot(t, errorfun(t,*popt3))
plt.show()
```



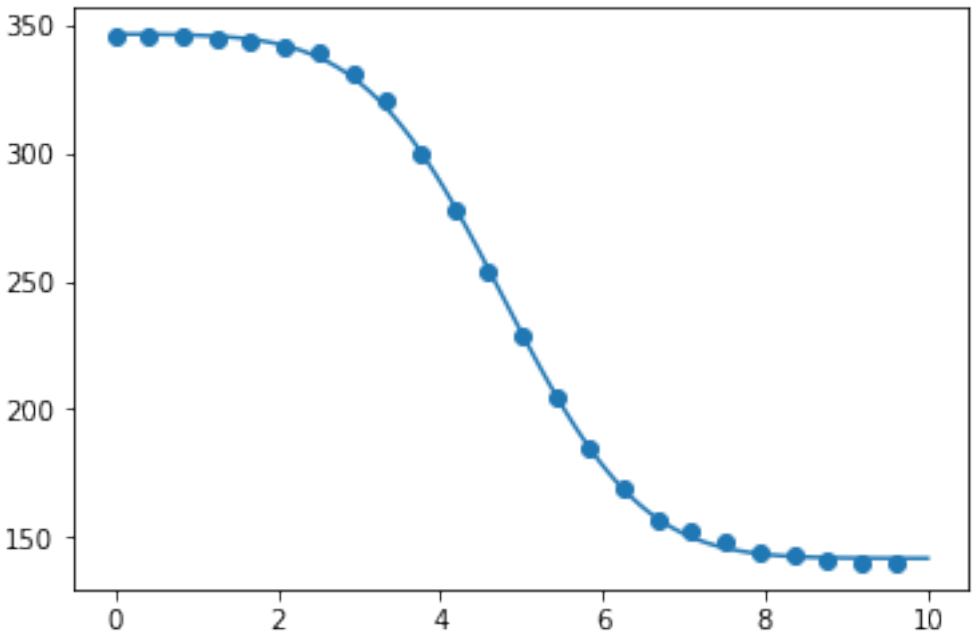
In [8]:
```
# position 4

X = T4
Y = H4
popt4, pcov = curve_fit(errorfun, X, Y, bounds=([50, 0, -10, 150], [180, 4, 0, 290]))
#perr = np.sqrt(np.diag(pcov))
t = np.linspace(0, 5, 100)
plt.scatter(X, Y)
plt.plot(t, errorfun(t,*popt4))
plt.show()
```



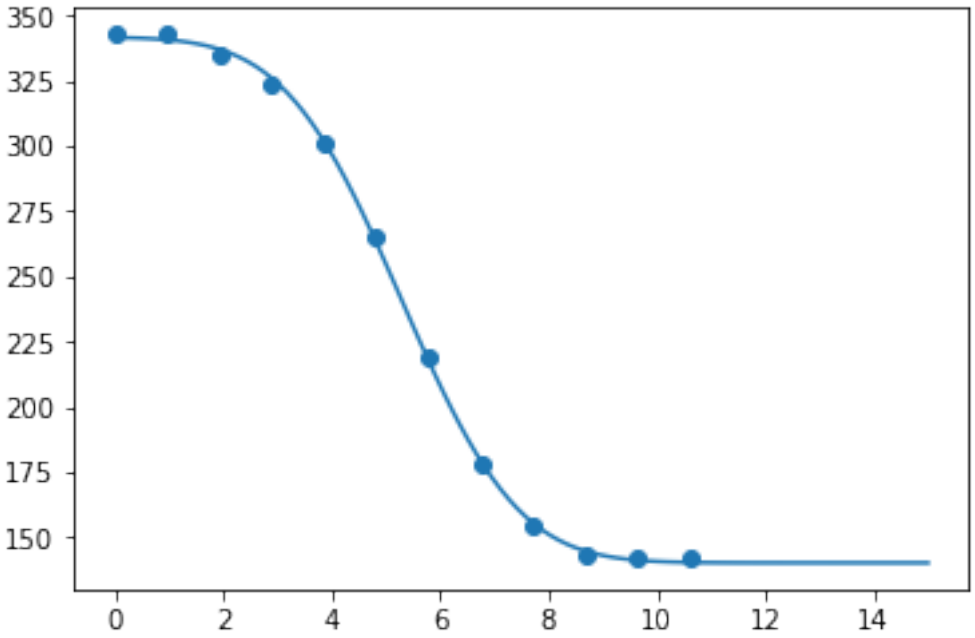
In [9]:
```
# position 5

X = T5p
Y = H5p
popt5, pcov = curve_fit(errorfun, X, Y, bounds=([50, 0, -10, 150], [180, 4, 0, 290]))
perr = np.sqrt(np.diag(pcov))
t = np.linspace(0, 10, 100)
plt.scatter(X, Y)
plt.plot(t, errorfun(t,*popt5))
plt.show()
```



In [10]:
```
# position 6

X = T6
Y = H6
popt6, pcov = curve_fit(errorfun, X, Y, bounds=([50, 0, -10, 150], [180, 10, 0, 290]))
perr = np.sqrt(np.diag(pcov))
t = np.linspace(0, 10, 100)
plt.scatter(X, Y)
plt.plot(t, errorfun(t,*popt6))
plt.show()
```

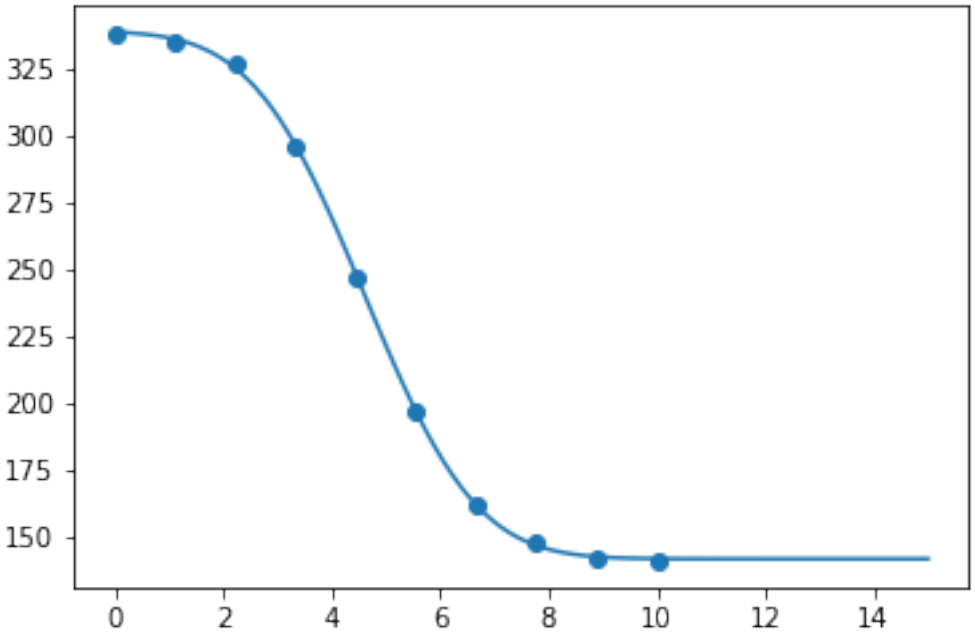


In [11]:
```
# position 7

X = T7
Y = H7
popt7, pcov = curve_fit(errorfun, X, Y, bounds=([50, 0, -10, 150], [180, 10, 0, 290]))

perr = np.sqrt(np.diag(pcov))
t = np.linspace(0, 15, 100)
plt.scatter(X, Y)
plt.plot(t, errorfun(t,*popt7))
plt.show()
```

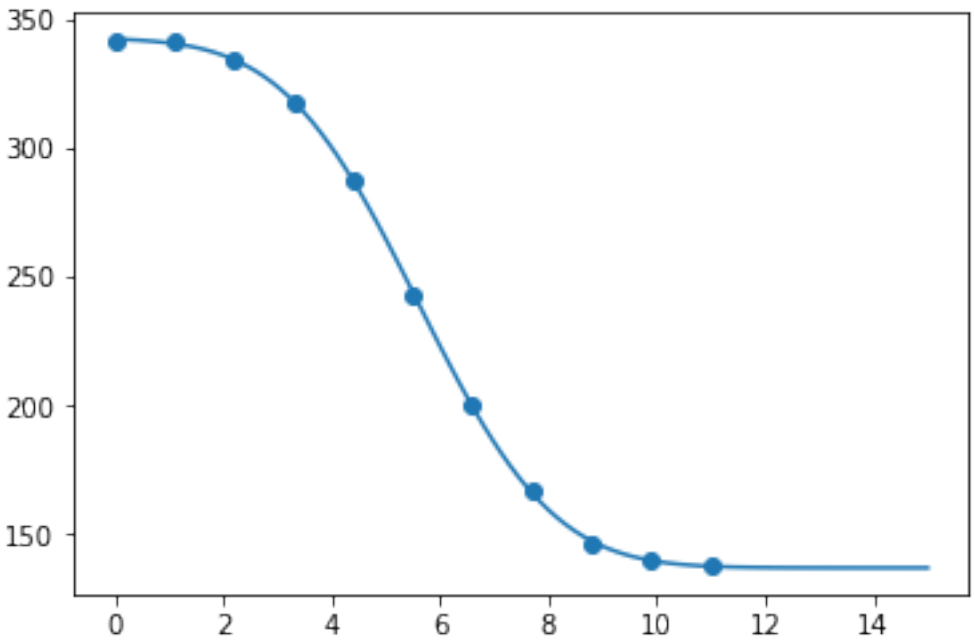


In [12]:
```
# position 9

X = T9
Y = H9
popt9, pcov = curve_fit(errorfun, X, Y, bounds=([50, 0, -10, 150], [180, 10, 0, 290]))
print(popt9)
perr = np.sqrt(np.diag(pcov))
t = np.linspace(0, 15, 100)
plt.scatter(X, Y)
plt.plot(t, errorfun(t,*popt9))
plt.show()
```

```
[ 98.61936788   4.60324945  -3.24447519 240.33905286]
```



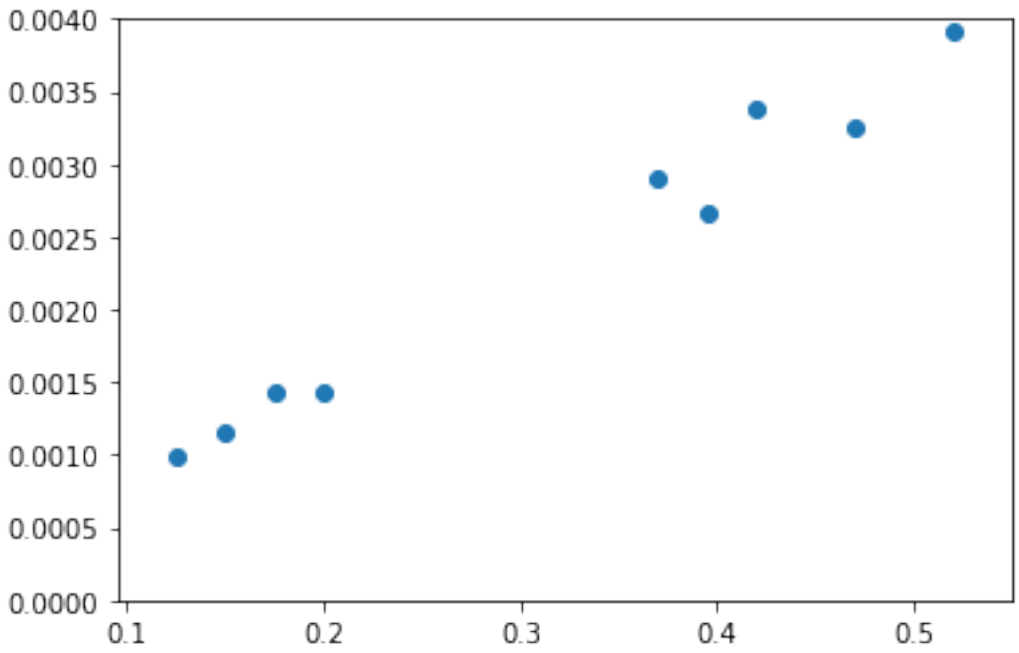
In [13]:
```
# position 11

X = T11
Y = H11
popt11, pcov = curve_fit(errorfun, X, Y, bounds=([50, 0, -10, 150], [180, 10, 0, 290])
perr = np.sqrt(np.diag(pcov))
t = np.linspace(0, 15, 100)
plt.scatter(X, Y)
plt.plot(t, errorfun(t,*popt11))
plt.show()
```



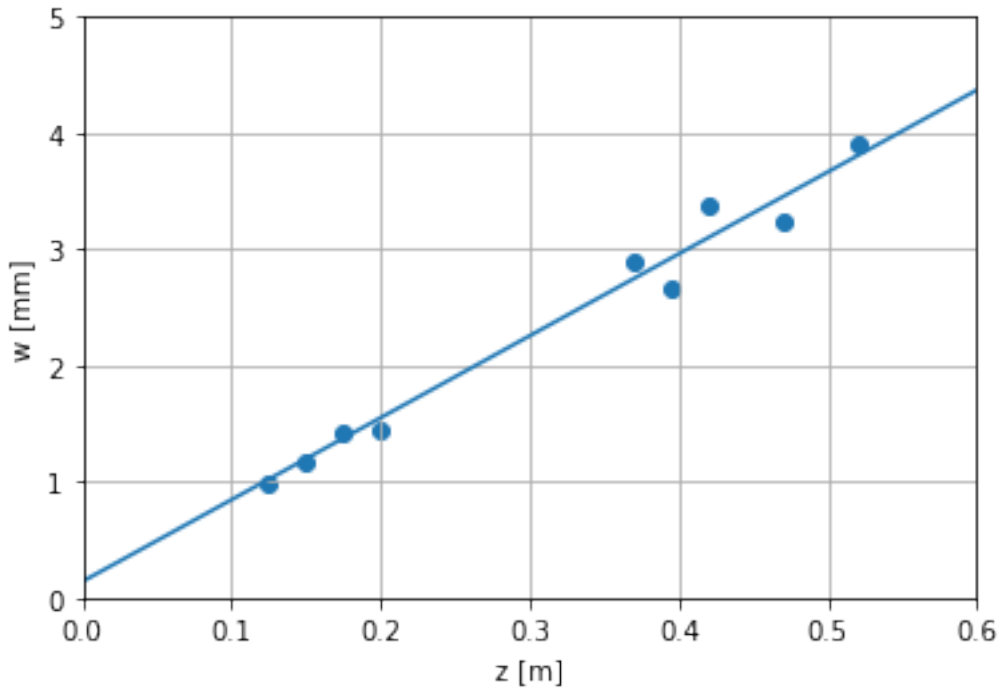

```
In [14]: waist = -np.array([-popt1[2],popt2[2],popt3[2],popt4[2],popt5[2],popt6[2],popt7[2],po
         waist = waist/1000
         print(waist)
         z = np.array([0.125, 0.15, 0.175, 0.20, 0.37, 0.395, 0.42, 0.47, 0.52])
         plt.ylim(0,0.004)
         plt.scatter(z, waist)
```

```
[0.00098238 0.00115781 0.00142657 0.00143289 0.00290379 0.00266076
 0.00338225 0.00324448 0.00390693]
```

```
Out[14]: <matplotlib.collections.PathCollection at 0x101449b358>
```




```
In [15]: l = 532*10**(-9)
         def gaussian(x, z0, w0):
             return w0*np.sqrt(1+((x-z0)/(np.pi*w0**2/l))**2)

In [17]: popt, pcov = curve_fit(gaussian, z, waist, bounds=([-100, 0.000001], [-0.01, 3]))
         zy = np.linspace(0, 0.6, 100)
         perr = np.sqrt(np.diag(pcov))
         print(popt)
         plt.scatter(z, waist*1000)
         plt.plot(zy, gaussian(zy,*popt)*1000)
         #plt.plot(z, gaussian(z,-0.016, 36e-6/(np.sqrt(2)))*1000)
         #wg = 36e-6/(np.sqrt(2))
         print(perr)
         plt.ylim(0,5)
         plt.xlim(0,0.6)
         plt.xlabel('z [m]')
         plt.ylabel('w [mm]')
         plt.grid()
         plt.show()

[-1.97536348e-02  2.40191620e-05]
[2.25846321e-02 1.49626697e-06]
```



In [ ]:

In [ ]: