In [1]:

```
import numpy as np
from scipy.integrate import odeint
from scipy.optimize import curve_fit
from scipy.optimize import differential_evolution
%matplotlib inline
import matplotlib.pyplot as plt
from matplotlib.backends.backend_pdf import PdfPages
import pandas as pd
```

In [2]:

```
from endpoint_maker import sqres
from endpoint_maker import func
from endpoint_maker import t
from endpoint_maker import new
from endpoint_maker import lightdata
from endpoint_maker import rawdata
```

In [3]:

```
if __name__ == '__main__':
    def initparams():
        #bounds = ([0.00001,10000],[0,4],[0.00001,1000],[0.00001,1000],[0.0000
        bounds = ([0.001,10],[0.001,10],[0.001,10],[0.001,10])
        result = differential_evolution(sqres,bounds,maxiter=500,popsize=20,po
        return result
```

In [4]:

```
initialp = initparams()
print(initialp)
```

```
 message: Optimization terminated successfully.
 success: True
     fun: 6.023437900134183
       x: [ 9.996e+00  1.000e-03  1.000e-03  6.766e+00]
     nit: 98
    nfev: 7920
```

In [5]:

```
#def tester(t,Kd,n,d2,k2,k3,i):
def tester(t,d2,k2,k3,k7,i):

    inivalues = [1,0,0,0,0,0,0]
    arrayvalues = np.asarray([])

    #for i in range(len(lightdata[:,0])):
    def I(t):
        tindex = t/5
        if tindex > 12241:
            tindex = 12240
        return lightdata[int(tindex)]

    #def odes(z,t,Kd,n,d2,k2,k3):
    def odes(z,t,d2,k2,k3,k7):
        Pu,Pb,Pa,mRNA,mCherry1,mCherry2,mCherry3 = z
        d1 = 0.019905
        k1 = 0.08299
        Kd = 90.41
        n = 0.964487
        #d2= 486.67
        #k2= 6.597
        #k3= 0.0539


        d3 = 0.000077
        k4 = 1.25
        d4 = 0.000031
        k5 = 0.00283
        k6 = 0.00283

        Pu = z[0]
        Pb = z[1]
        Pa = z[2]
        mRNA = z[3]
        mCherry1 = z[4]
        mCherry2 = z[5]
        mCherry3 = z[6]

        dPudt = d1*Pb - k1*I(t)**n/(Kd**n+I(t)**n)*Pu + k7*I(t)*Pa
        dPbdt = k1*I(t)**n/(Kd**n+I(t)**n)*Pu - k2*Pb - d1*Pb + d2*Pa
        dPadt = k2*Pb - d2*Pa - k7*I(t)*Pa
        dmRNAdt = k3*Pa - d3*mRNA
        dmCherry1dt = k4*mRNA-(d4 + k5)*mCherry1
        dmCherry2dt = k5*mCherry1-(d4+k6)*mCherry2
        dmCherry3dt = k6*mCherry2 - d4*mCherry3


        return [dPudt,dPbdt,dPadt,dmRNAdt,dmCherry1dt,dmCherry2dt,dmCherry3dt]

    #solver = odeint(odes,inivalues,t,args = (Kd,n,d2,k2,k3),hmax=0.1)
    solver = odeint(odes,inivalues,t,args = (d2,k2,k3,k7),hmax=0.1)

    mCherryout = solver[:,6]
    #mCherryout = mCherryout[0:24480:240]
    return mCherryout
```

In [6]:

```
print(initialp.x)
```

```
[9.99585123e+00 1.00009040e-03 1.00002075e-03 6.76552605e+00]
```

In [7]:

```
popt, covt = curve_fit(func,t,new,initialp.x,maxfev=1000000)

#popt, covt = curve_fit(func,t,newdata,initialp.x, maxfev=10000000, bounds=((0
#popt, covt = curve_fit(func,t,newdata,initialp.x, maxfev=1000000)

#Kd,n,d2,k2,k3 = popt
#print('Kd=',Kd,'n=',n,'d2=',d2,'k2=',k2,'k3=',k3)

#d3,k4,d4,k5,k6 = popt
#print('d3=',d3,'k4=',k4,'d4=',d4,'k5=',k5,'k6=',k6)

d2,k2,k3,k7 = popt
print('d2=',d2,'k2=',k2,'k3=',k3,'k7=',k7)
```

```
d2= 20.909436087854324 k2= 0.0009572834655778124 k3= 0.0005478874261473799 k7
= 3.250415333549934
```

In [8]:

```
import sys
import numpy
#params = [1,1,1,1,1,1,1]
params = [1,1,1,1]
numpy.set_printoptions(threshold=10)
#model1 = np.asarray(func(t,Kd,n,d2,k2,k3))
model1 = np.asarray(func(t,d2,k2,k3,k7))
print(len(model1))
#print(model1)

#a,b,c,d,e,f,g = params
a,b,c,d= params

ydata = np.asarray(new)
print(len(ydata))

ssr = np.sum((ydata-model1)**2)
#ssr2 = np.sum((ydata-model2)**2)
sst = np.sum((ydata-np.mean(ydata))**2)
R2=1-ssr/sst
print('R2 is:', R2)
```

```
51
51
R2 is: 0.9565305179888783
```

In [9]:

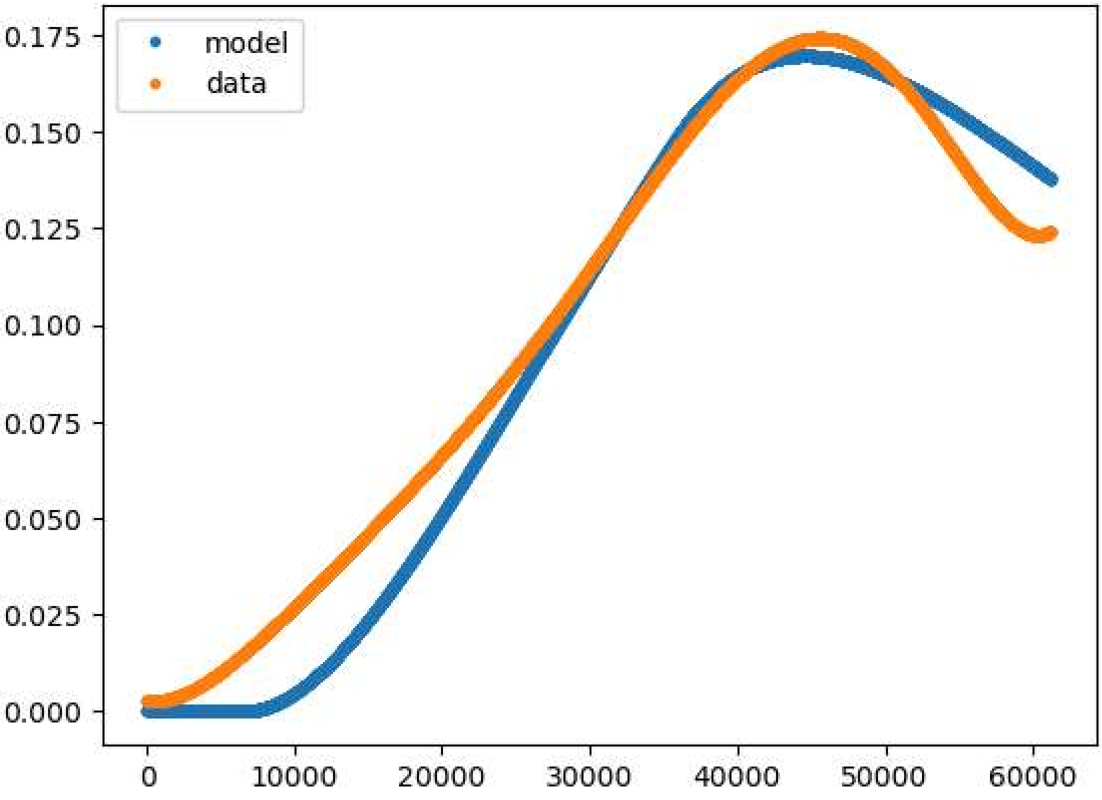
```python
pp = PdfPages('multipage.pdf')
ydata = np.asarray(new)

#condition = [1,2,3,4,5,6,7,8,9]
for i in range(1):
    #model = tester(t,Kd,n,d2,k2,k3,i)
    model = tester(t,d2,k2,k3,k7,i)

    #print(model)
    #a,b,c,d,e,f,g= params
    a,b,c,d= params

    plt.plot(t,model,'.', label = 'model')
    #print(model)
    #t = np.linspace(0,34800, num=6961)
    #raw = rawdata[0:13920:240]
    plt.plot(t,rawdata,'.',label = 'data')
    #print(rawdata[i])
    plt.legend()
    pp.savefig()
    plt.show()

pp.close
```



Out[9]:

```
<bound method PdfPages.close of <matplotlib.backends.backend_pdf.PdfPages obj
ect at 0x000001F89C021290>>
```

In [ ]:

In [ ]: