In [1]:

```
import numpy as np
from scipy.integrate import odeint
%matplotlib inline
import matplotlib.pyplot as plt
from matplotlib.backends.backend_pdf import PdfPages
import sys
```

In [2]:

```
t = np.linspace(0,50395, num=10080)
lightdata = np.transpose(np.delete(np.genfromtxt('AllLightnosmooth.csv', delimiter=',')


def func(t):
    z0 = [1,0,0,0,0,0]
    #mCherry2 = np.empty((len(lightdata[:,0]), len(t)))
    out =  np.empty((len(lightdata[:,0]),len(t),6))
    arrayvalues = np.asarray([])
    #for i in range(42):
    for i in range(len(lightdata[:,0])):
        def I(t):
            tindex = t/5
            if tindex > 10079:
                tindex = 10079
            return lightdata[i][int(tindex)]

        def model(z,t):
            d1 = 0.017981
            k1 = 0.118682
            d2 = 2.296629
            k2 = 0.026267
            Kd = 72.69424
            n = 1.289105
            k3 = 0.000279
            d3 = 0.000544
            k4 = 1.25
            d4 = 0.0000924
            k5 = 0.00144



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

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

        z = odeint(model,z0,t)
        #mCherry2[i] = z[:,5]
        out[i] = z
    return out
```

In [3]:

```
model1 = np.asarray(func(t))
mCherry2 = model1[:,:,5]
Pu = model1[:,:,0]
Pb = model1[:,:,1]
Pa = model1[:,:,2]
mRNA = model1[:,:,3]

#total = Pu+Pb+Pa+Pi
#print(total)

#for i in range(42):
for i in range(len(lightdata[:,0])):
    last = mCherry2[i]
    ##total = Pi[i]+Pu[i]+Pb[i]+Pa[i]
    print(last[-1])
```

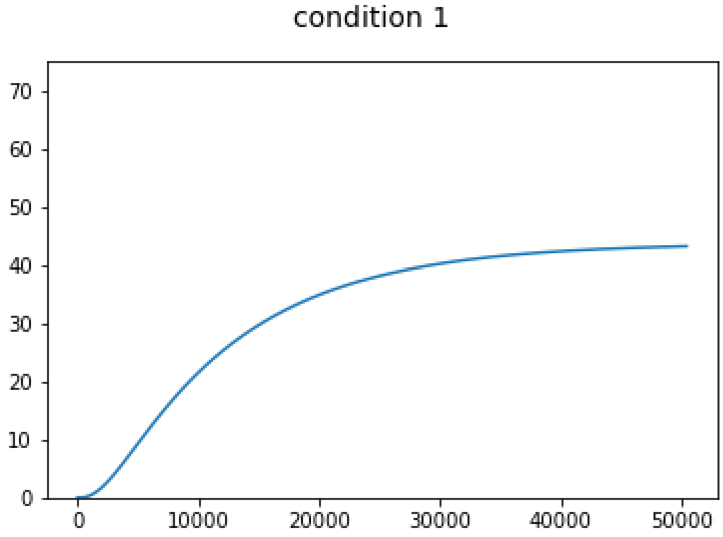
```
43.22293141410458
23.689923981188144
14.014047363170572
0.0032994740366985693
0.0010789760956070261
49.925948885080345
30.369498265834242
18.935711249958178
0.0031391697512623395
54.53280069258792
35.85561402959026
23.387410258424794
0.05657083253912293
0.001956649374544469
0.0019566168499721534
55.08787669294011
36.574788447412544
24.000968930287225
0.004117950005499563
0.0020193204988296883
35.50077055066495
12.212804044039466
6.109115538624184
2.0402445640644826
0.010109668145042442
39.62098201920765
13.659190409990634
6.832535275948372
2.2816557058910347
0.297616293728474
0.011305053160676547
42.28003967342267
14.594545775043274
7.300347467973042
2.437745887877762
0.012077858970030414
0.012077858970030414
42.59176793342165
14.704295050254556
7.355237273546147
2.456059231693949
0.04305261316811613
38.71582148856501
29.064081929176673
9.628342097612602
4.371288695149969
1.146992207779832
0.04095782139789479
```

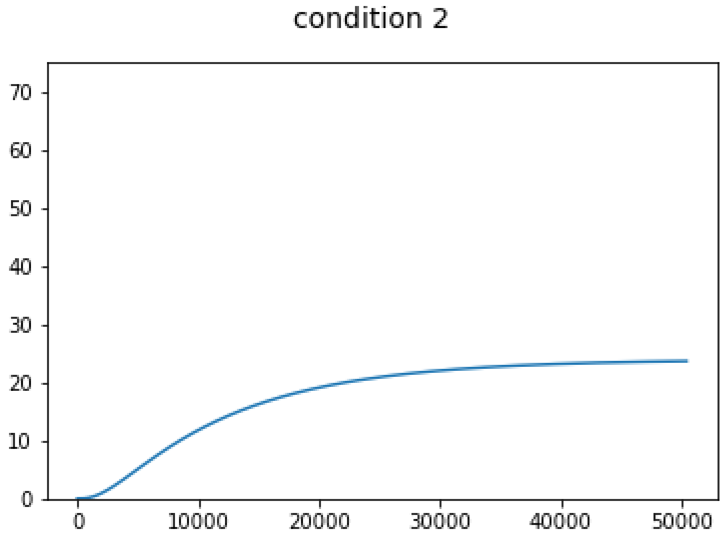
```
42.54198542607992
31.93712931437549
10.579360960453181
4.802719961228616
1.2601629491117838
0.044998989783221255
44.96645653886533
33.75770899301771
11.181926757206584
5.076044137086851
1.331856754820981
0.047559077039287787
45.248475239511166
33.96948477392988
11.252015570652805
5.107834929083276
1.34019542434712
0.047856840652944474
47.646234340492036
28.001153174643797
13.107816414347608
3.481093676766625
1.4002853977130612
0.12433661231659582
52.151169658343264
30.647468786984003
14.345385607975414
3.809636742211917
1.5324423055764758
0.13607133110535724
54.989311137285334
32.3145608409841
15.124906347539563
4.016569070536445
1.615681057919639
0.143462387748785 9
55.318637004582264
32.50799736159263
15.215350501866265
4.040577943084394
1.6253386579771134
0.14431989115063634
51.11683908384346
35.86407607624848
27.36274214313299
7.501008986052082
3.0191942419116553
0.26808999098443614
55.89084891823457
39.211610420591
29.915641625850643
8.20047772160795
3.300731197850051
0.293089096629429
58.89352936210272
41.31691960434343
31.521097387074647
8.64032515771292
3.4777699809558476
0.3088093124605335
59.241699252215184
41.56102817650059
31.707243478684436
8.691322123105962
3.498296277182579
```

```
0.31063192624347646
54.28163540753594
59.29057203318615
62.43585346909243
62.80030004132466
```

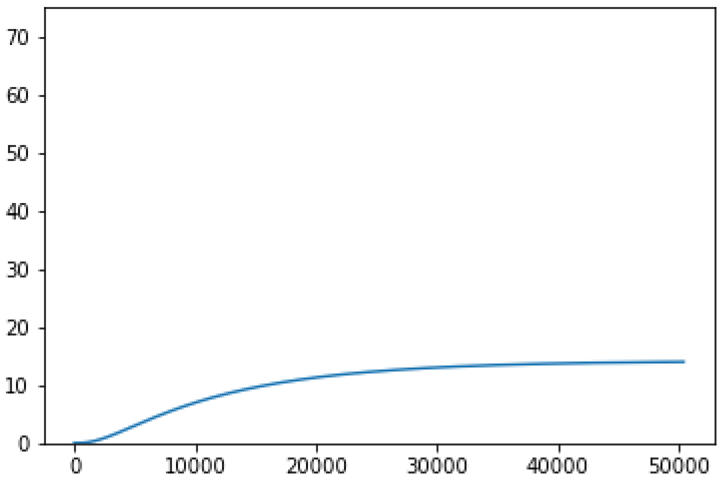
In [4]:

```
import csv
pp = PdfPages('multipage.pdf')

#for i in range(42):
for i in range(len(lightdata[:,0])):
#    mCherry2 = model1[:,:,6]
#    with open('dataout.csv','w') as csvfile:
#        csvwriter=csv.writer(csvfile)
#        for row in mCherry2:
#            csvwriter.writerow(row)
#    Pu = model1[:,:,0]
#    with open('Puout.csv','w') as csvfile:
#        csvwriter=csv.writer(csvfile)
#        for row in Pu:
#            csvwriter.writerow(row)
#    Pb = model1[:,:,1]
#    with open('Pbout.csv','w') as csvfile:
#        csvwriter=csv.writer(csvfile)
#        for row in Pb:
#            csvwriter.writerow(row)
#    Pi = model1[:,:,2]
#    with open('Piout.csv','w') as csvfile:
#        csvwriter=csv.writer(csvfile)
#        for row in Pi:
#            csvwriter.writerow(row)
#    Pa = model1[:,:,3]
#    with open('Paout.csv','w') as csvfile:
#        csvwriter=csv.writer(csvfile)
#        for row in Pa:
#            csvwriter.writerow(row)
#    mRNA = model1[:,:,4]
#    with open('mRNAout.csv','w') as csvfile:
#        csvwriter=csv.writer(csvfile)
#        for row in mRNA:
#            csvwriter.writerow(row)
    #U=U*100;
    #Pu=Pu*100;
    #Pb=Pb*100;
    #Pa=Pa*100;
    #t = np.linspace(0,50395, num=10080)
    #plt.plot(t,U[i])
    #plt.plot(t,Pu[i])
    #plt.plot(t,Pb[i])
    #plt.plot(t,Pa[i])
    #plt.legend(['U','Pu','Pb','Pa'])
    #plt.plot(t,lightdata[i])
    plt.plot(t,mCherry2[i])
    #plt.xlabel('Time [s]')
    #plt.ylabel('Fluoresence [a.u.]')
    plt. suptitle('condition '+str(i+1),fontsize = 14)
    plt.ylim((0,75))


    #def annot_max (t, mCherry2, ax=None):
    #    tmax = t[np.argmax(mCherry2[i])]
    #    ymax = mCherry2[i].max()
    #    text = "t={:.3f}, y={:.3f}".format(tmax,ymax)
    #    print(i+1, ymax)
    #    if not ax:
    #        ax=plt.gca()
    #    bbox_props = dict(boxstyle = "square,pad=0.3", fc="w", ec="k",lw=0.72)
    #    arrowprops = dict(arrowstyle="->", connectionstyle = "angle,angleA=0,angleB=60
    #    kw = dict(xycoords='data', textcoords="axes fraction", arrowprops=arrowprops,
    #    ax.annotate(text,xy=(tmax,ymax), xytext=(0.94,0.7), **kw)
    #annot_max(t,mCherry2)
    pp.savefig()
    plt.show()

pp.close()
```

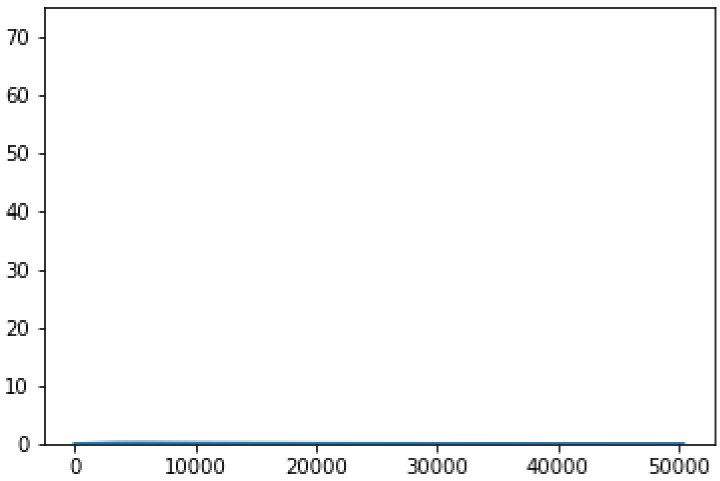




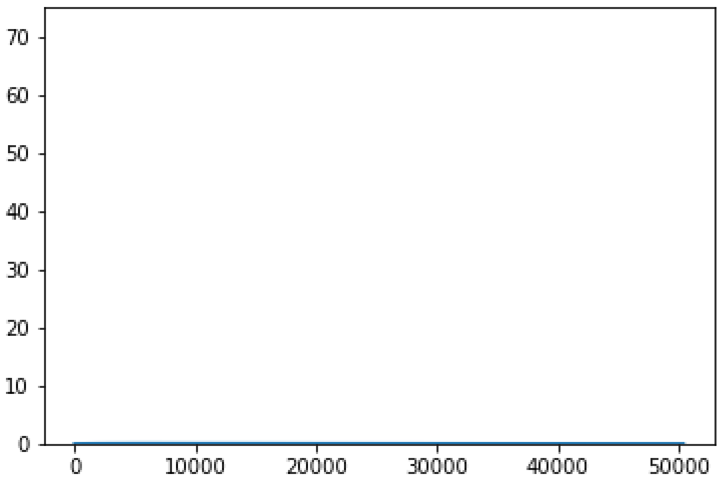
condition 3



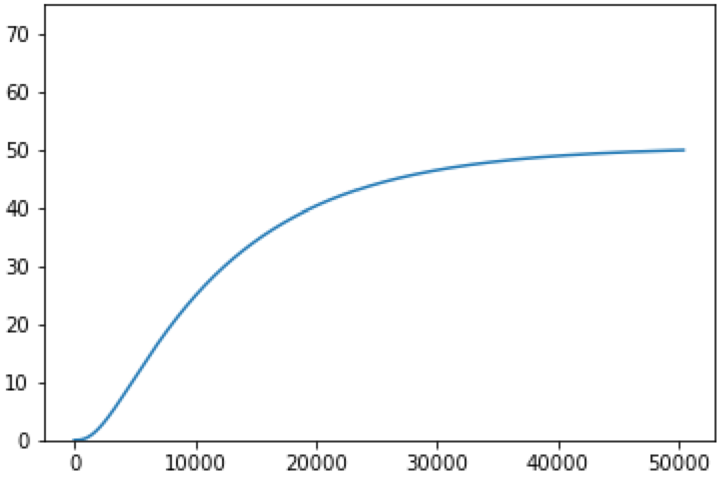
condition 4



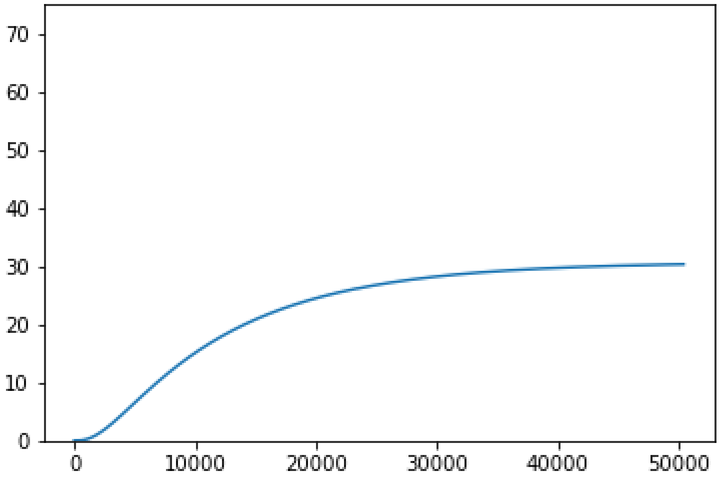
condition 5



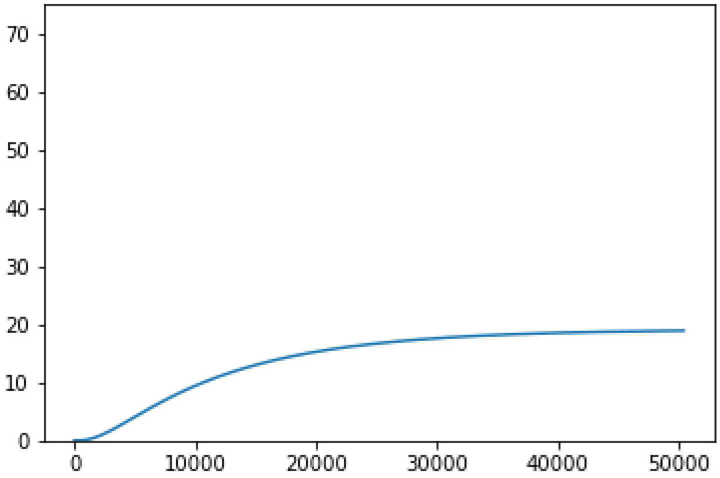
condition 6



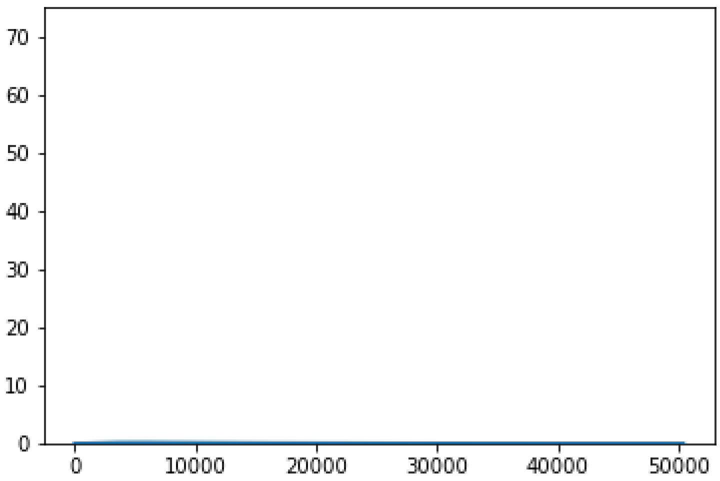
condition 7



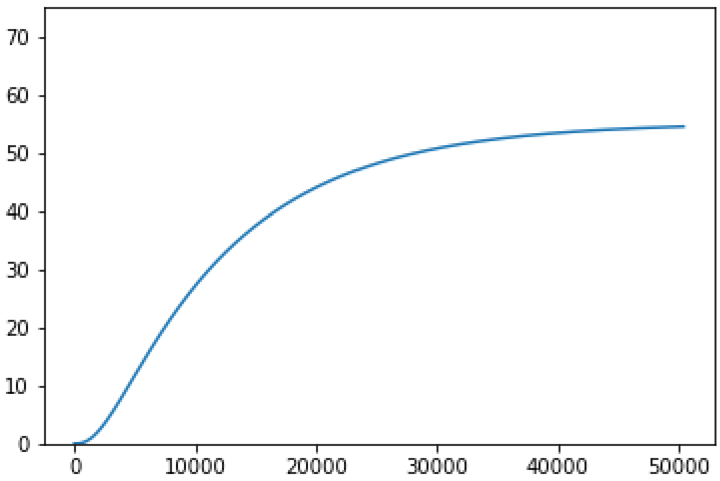
condition 8



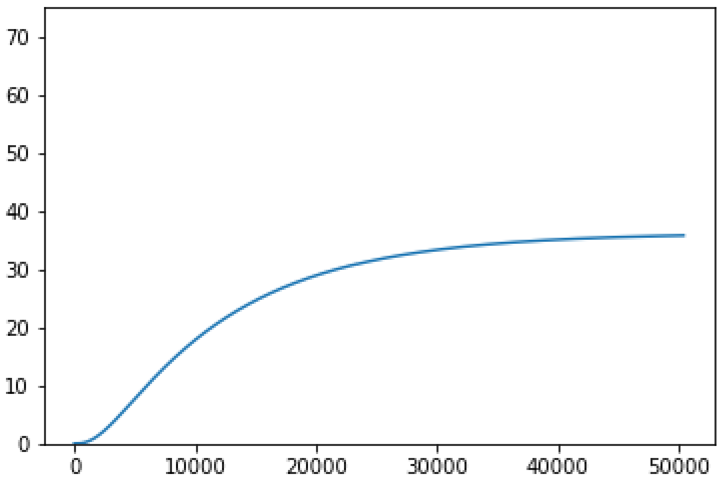
condition 9



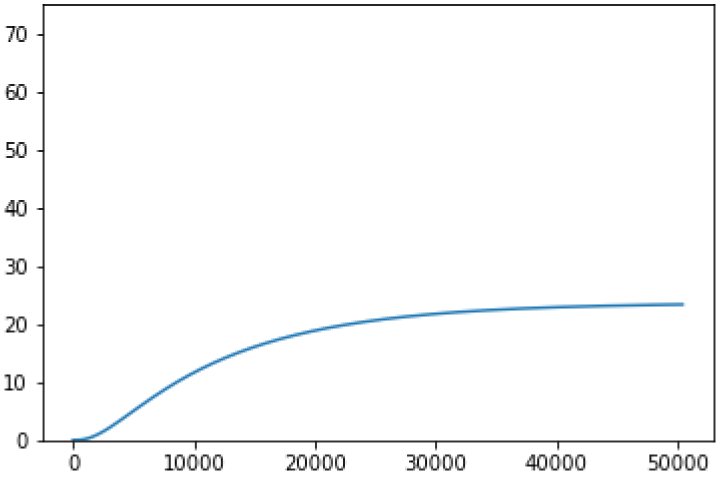
condition 10



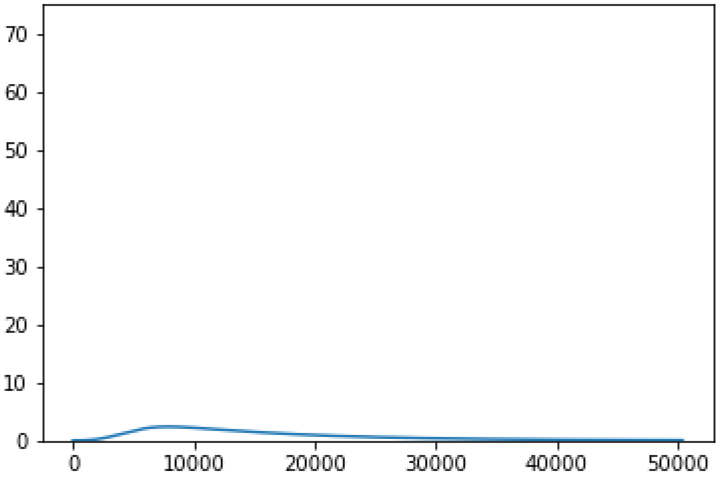
condition 11



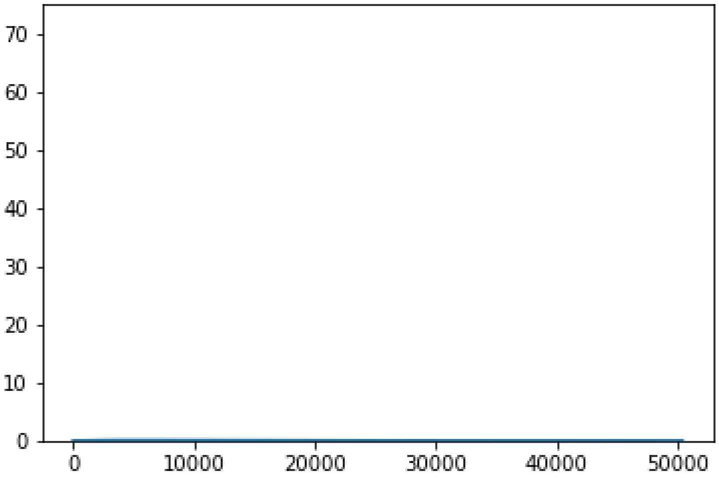
condition 12



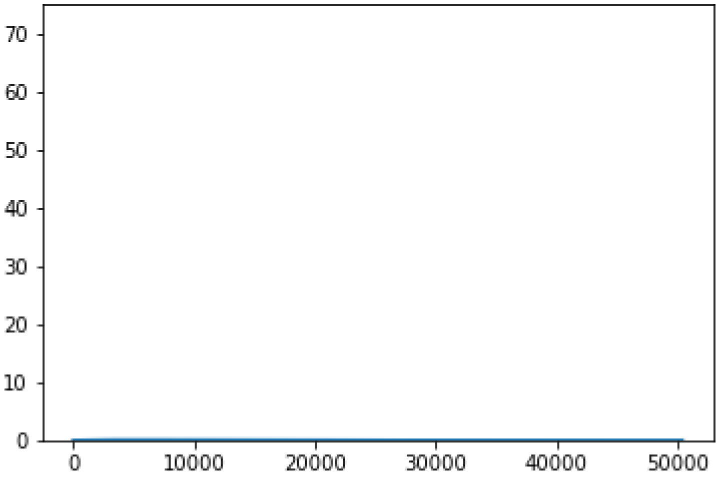
condition 13



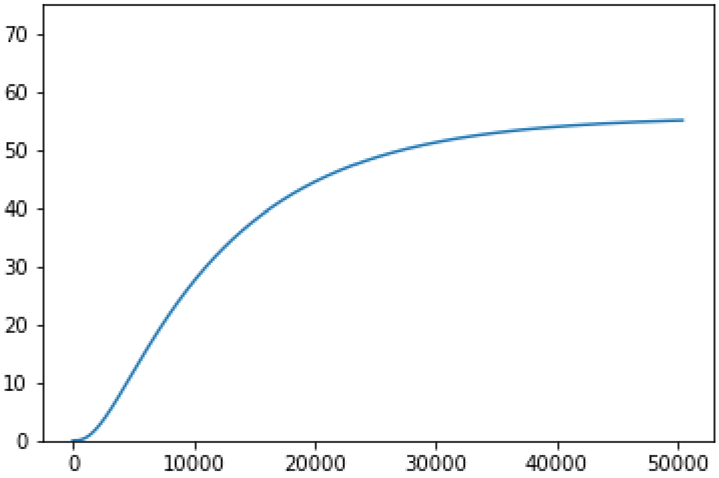
condition 14



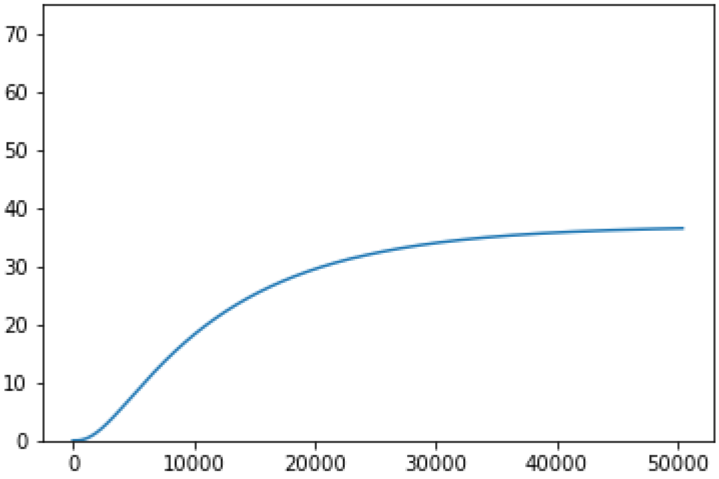
condition 15



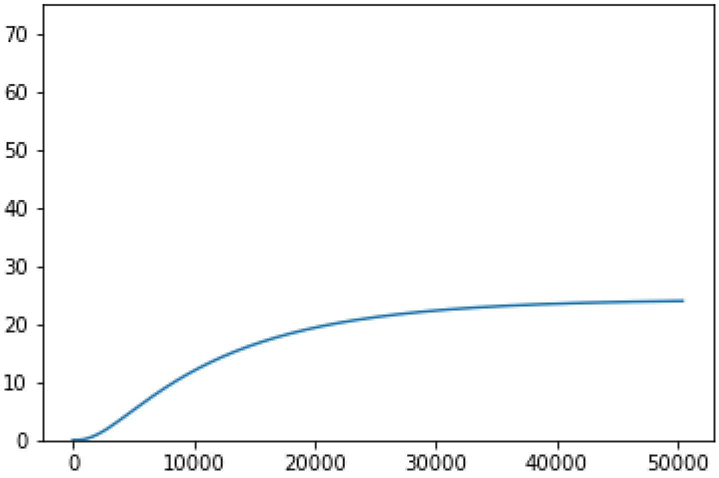
condition 16



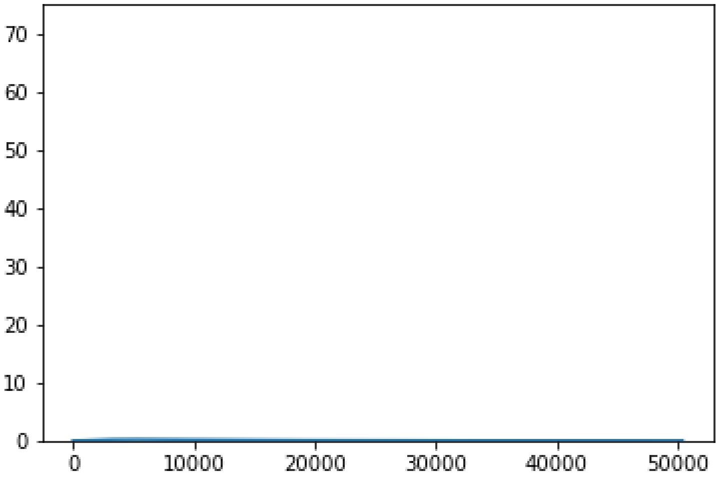
condition 17



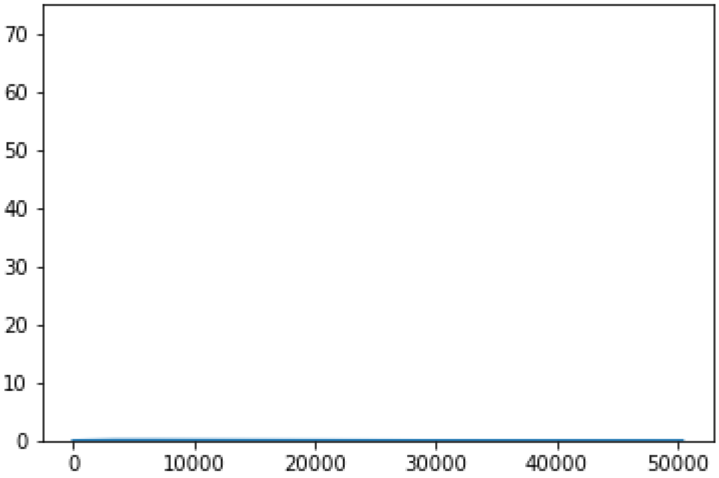
condition 18



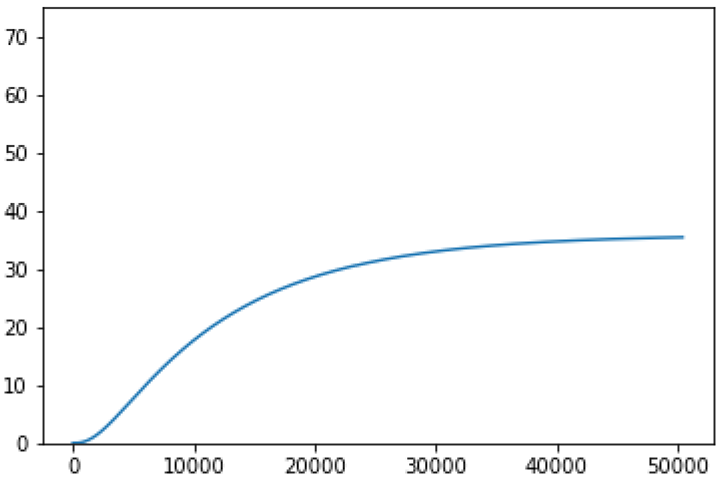
condition 19



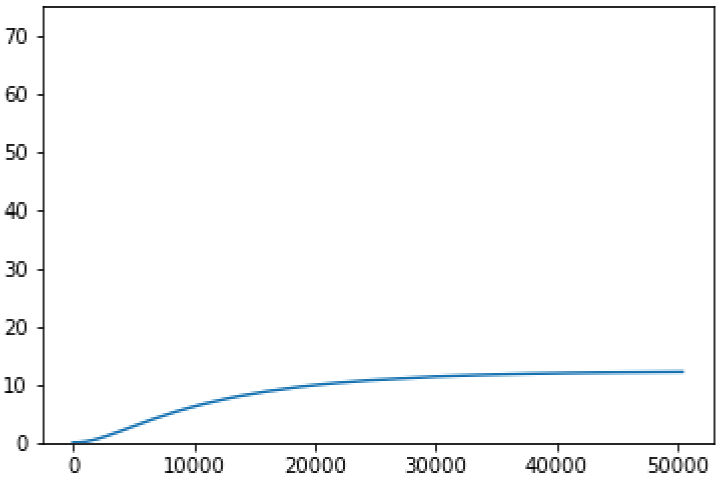
condition 20



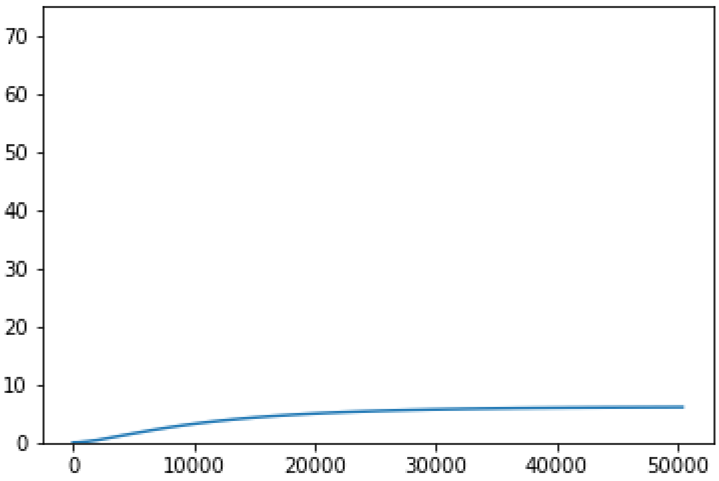
condition 21



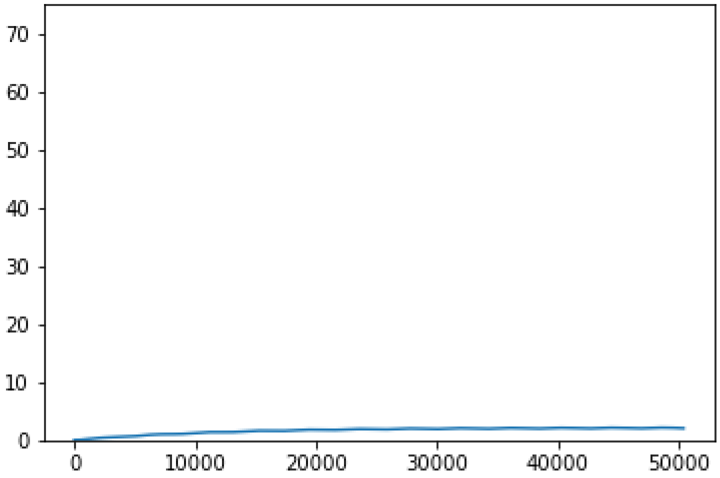
condition 22



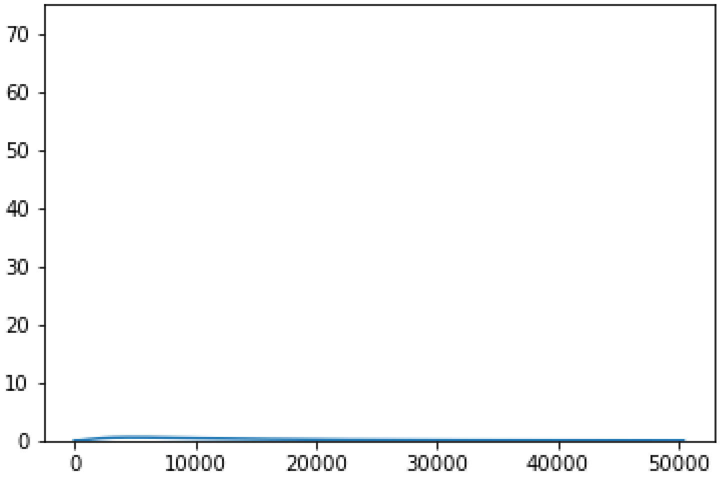
condition 23



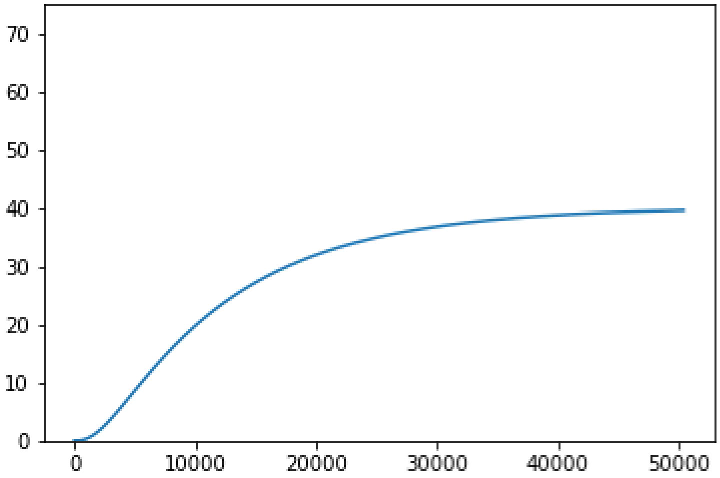
condition 24



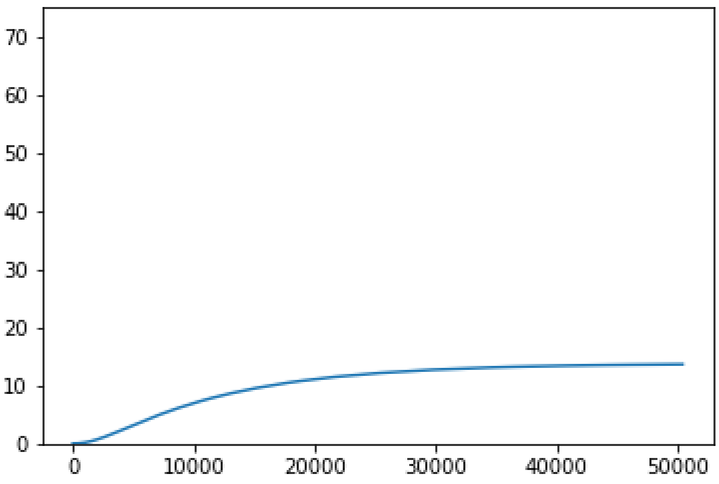
condition 25



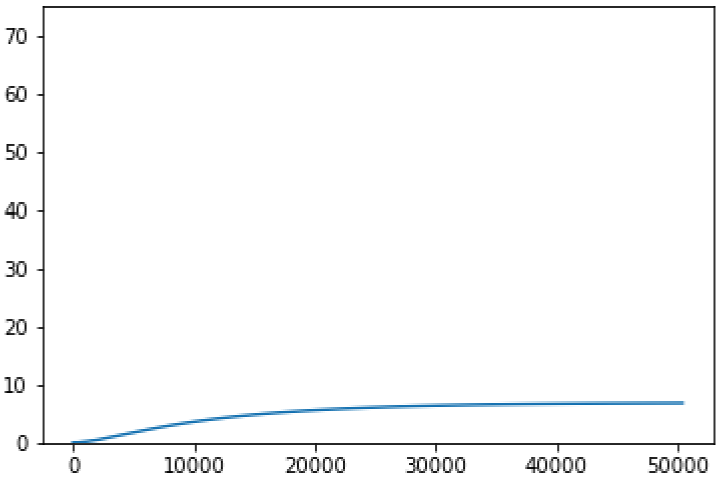
condition 26



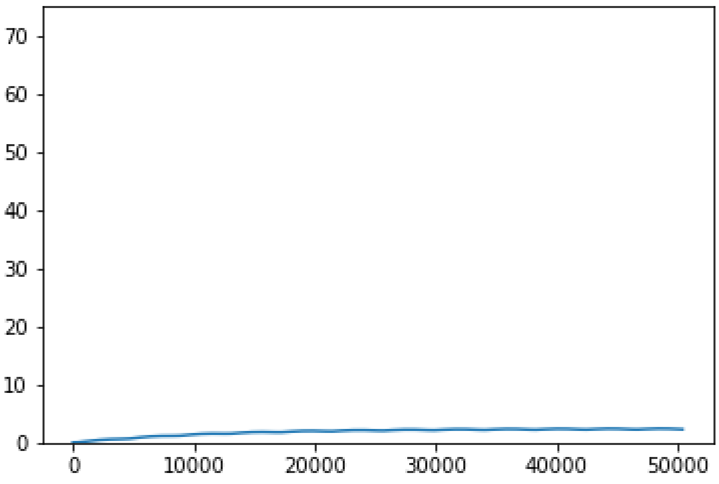
condition 27



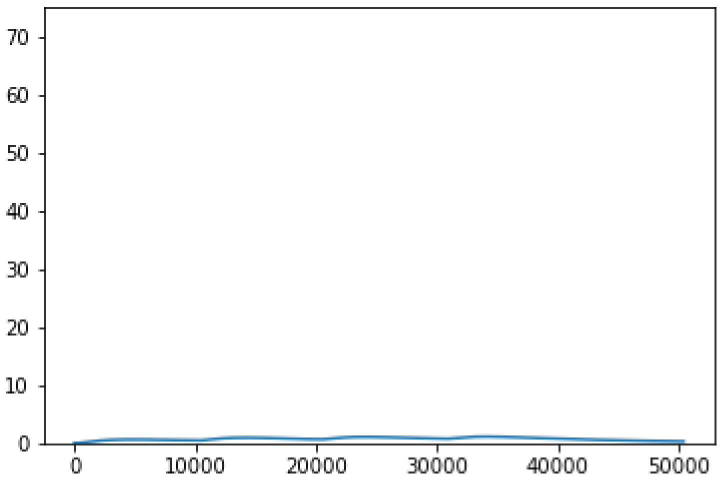
condition 28



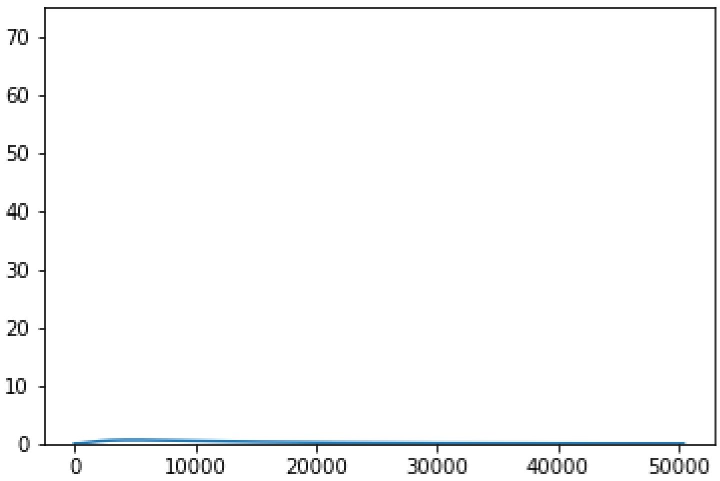
condition 29



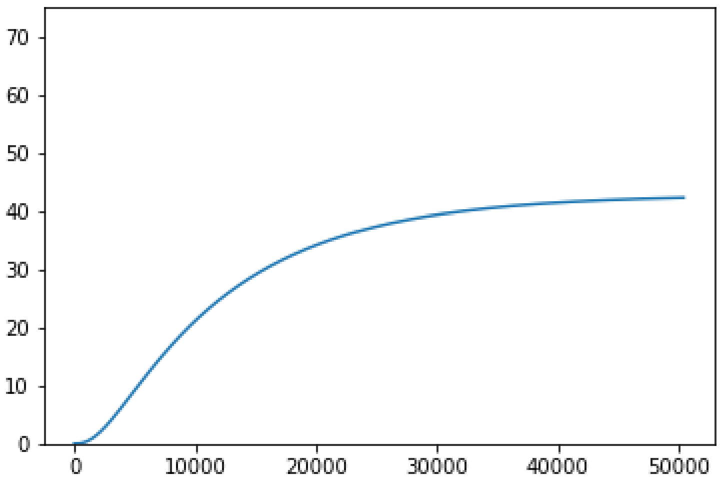
condition 30

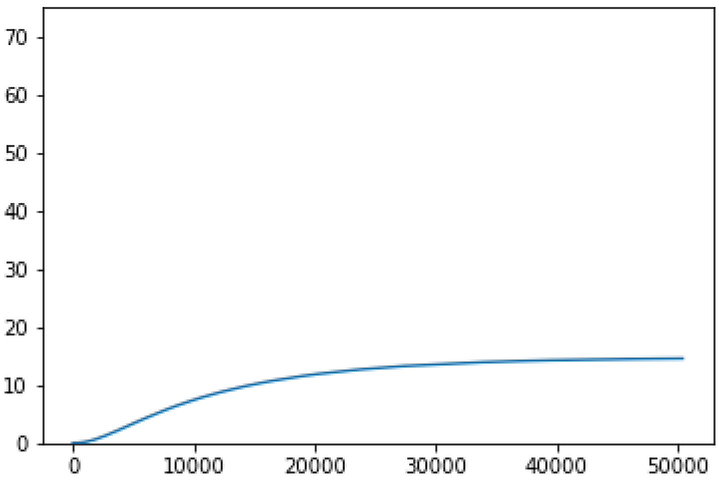


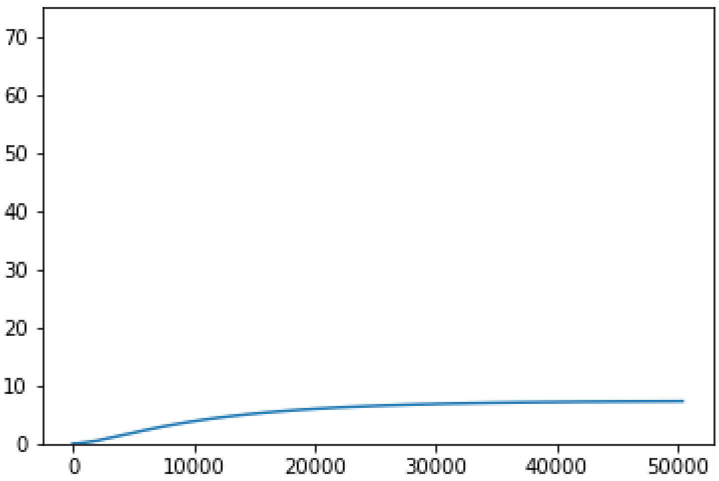
condition 31

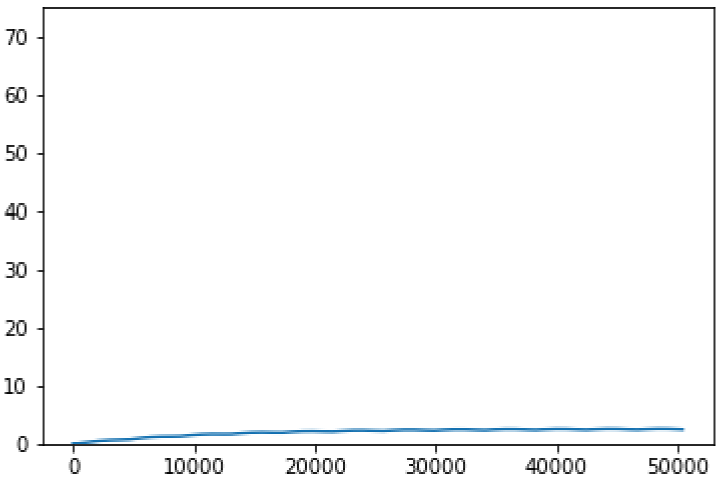


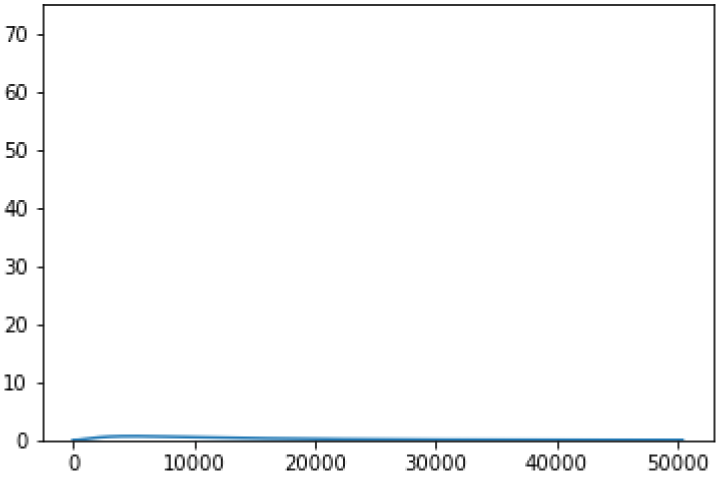
condition 32



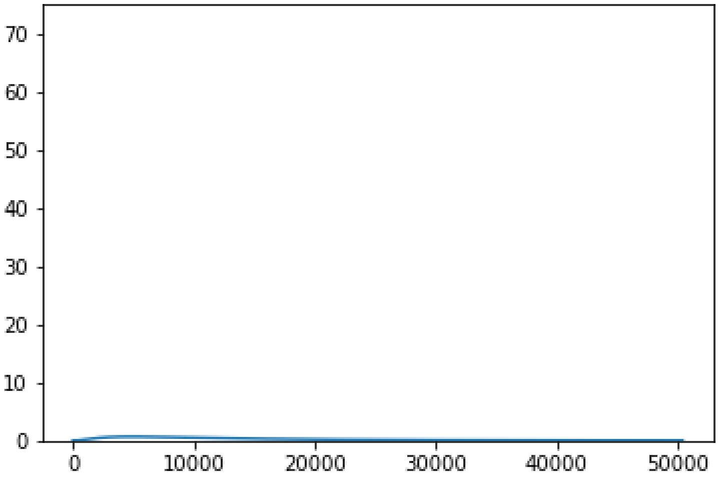

condition 33


condition 34

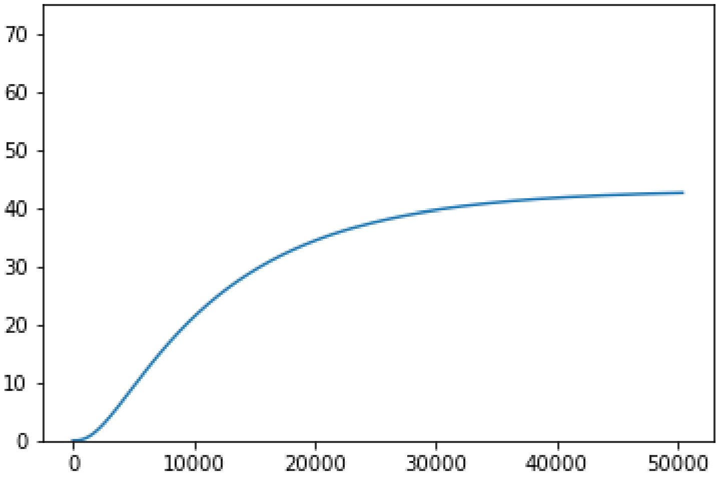

condition 35



condition 36

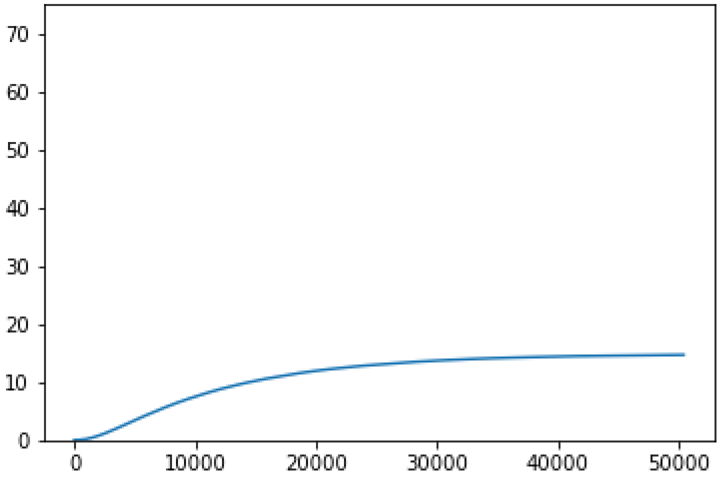


condition 37

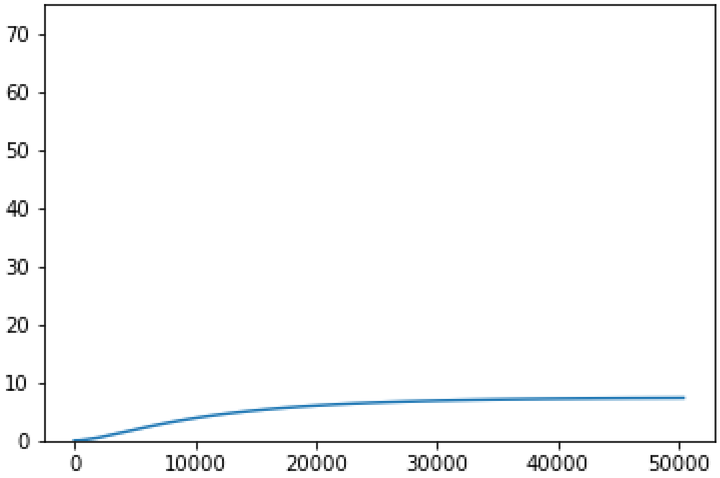


condition 38

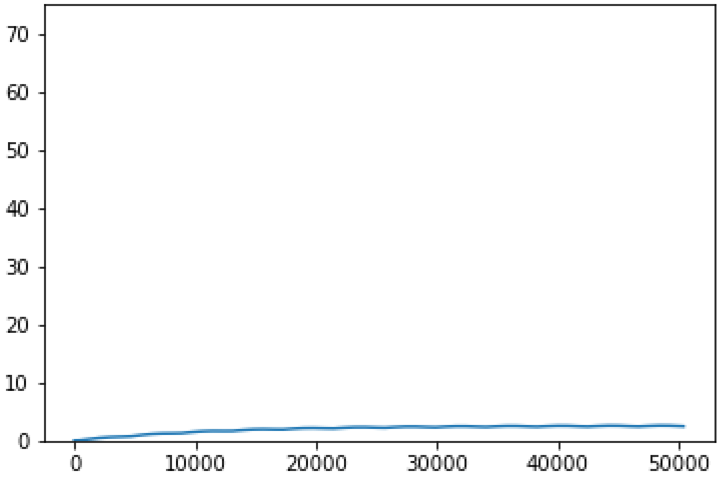
condition 39

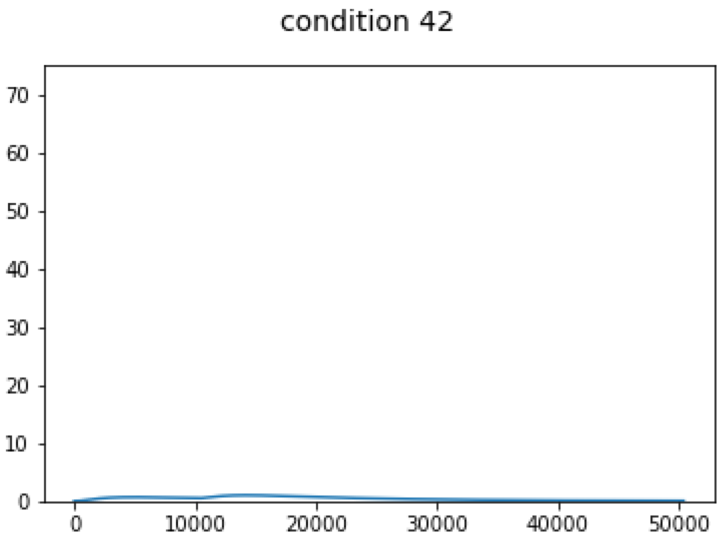


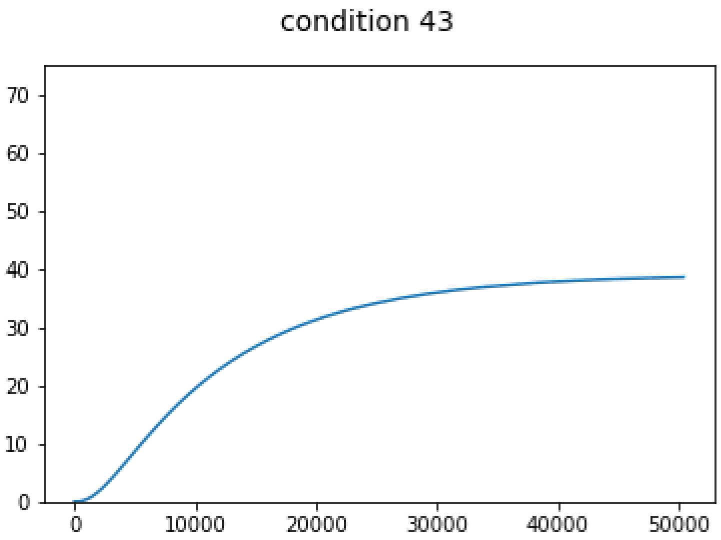
condition 40

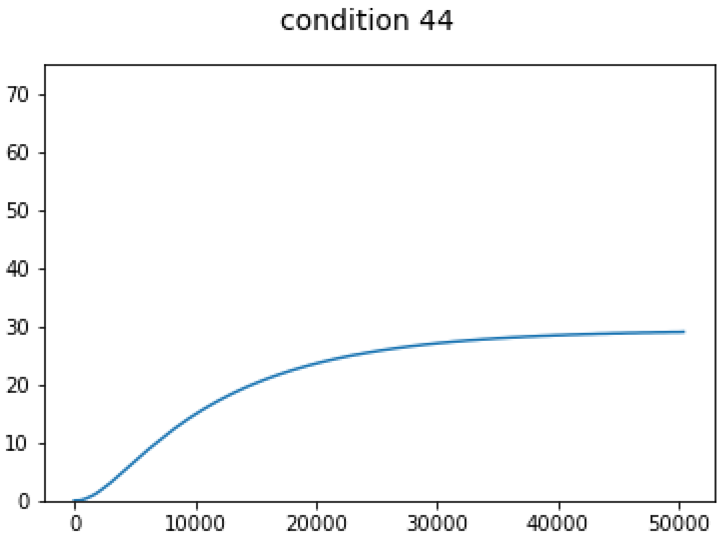


condition 41

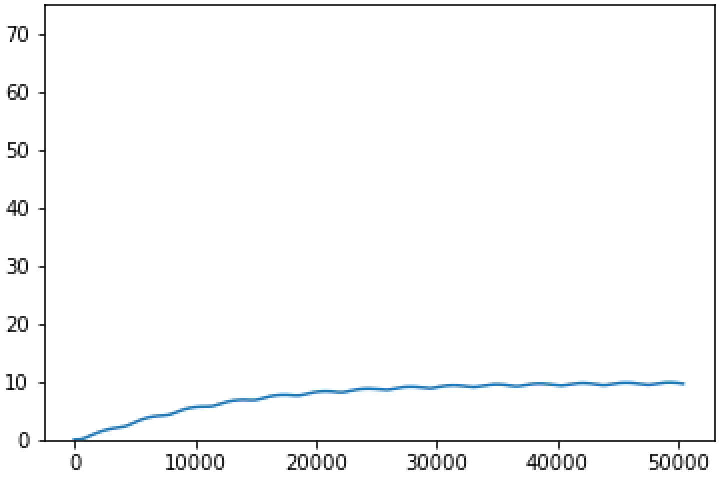



condition 42
70
60
50
40
30
20
10
0
0
10000
20000
30000
40000
50000

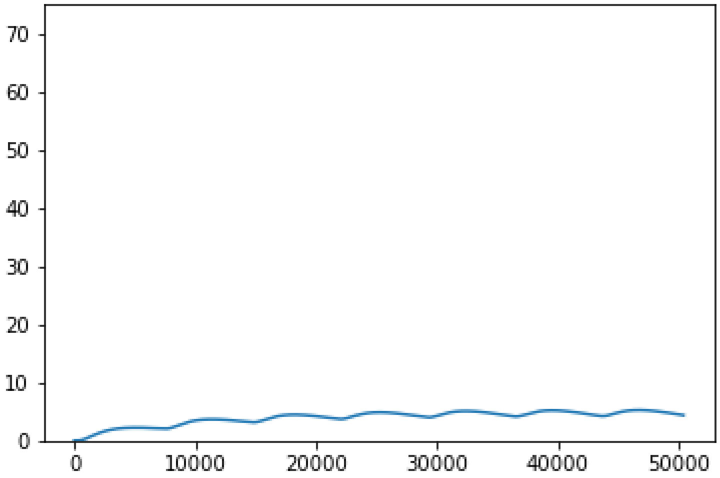

condition 43
70
60
50
40
30
20
10
0
0
10000
20000
30000
40000
50000


condition 44
70
60
50
40
30
20
10
0
0
10000
20000
30000
40000
50000

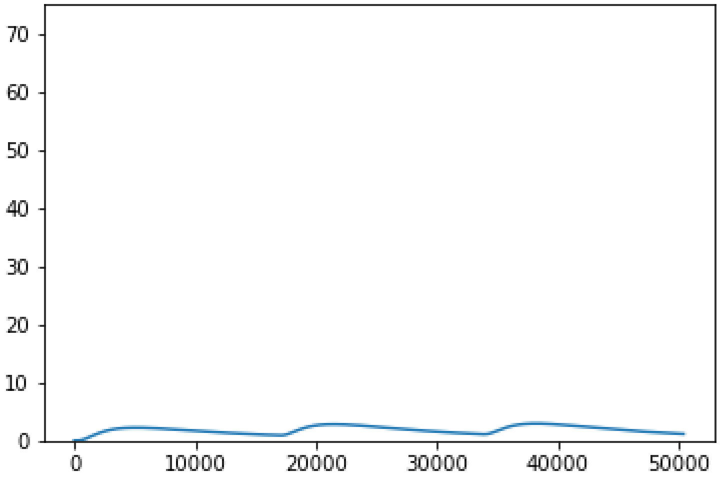
condition 45



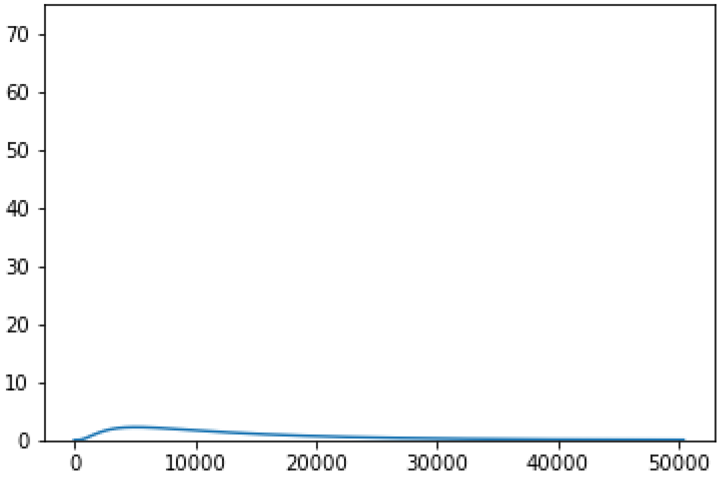
condition 46



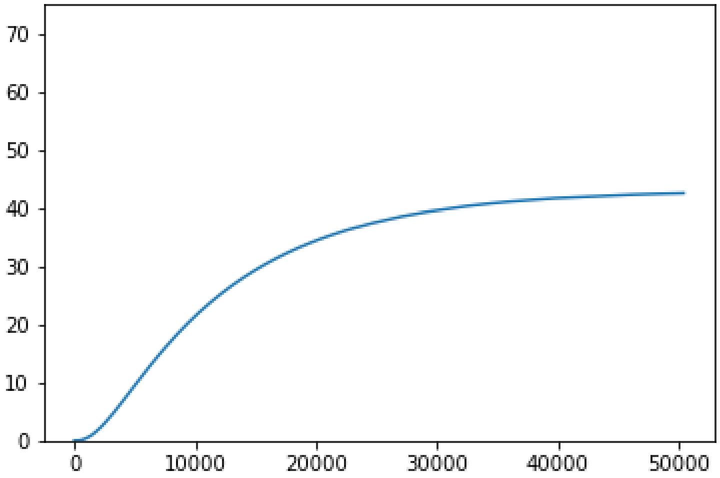
condition 47



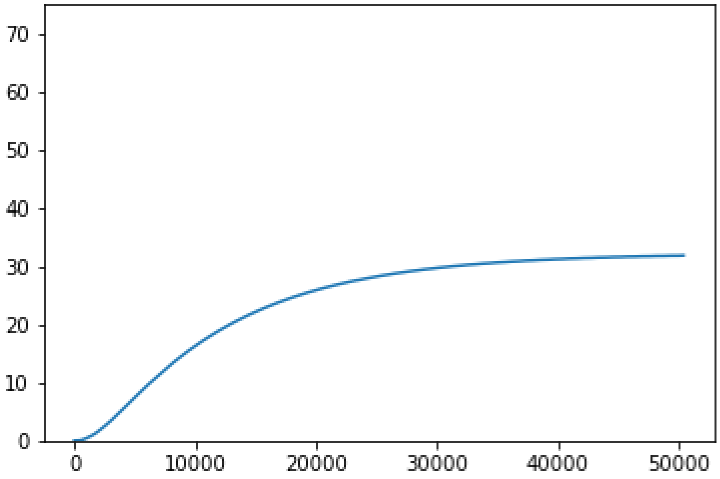
condition 48



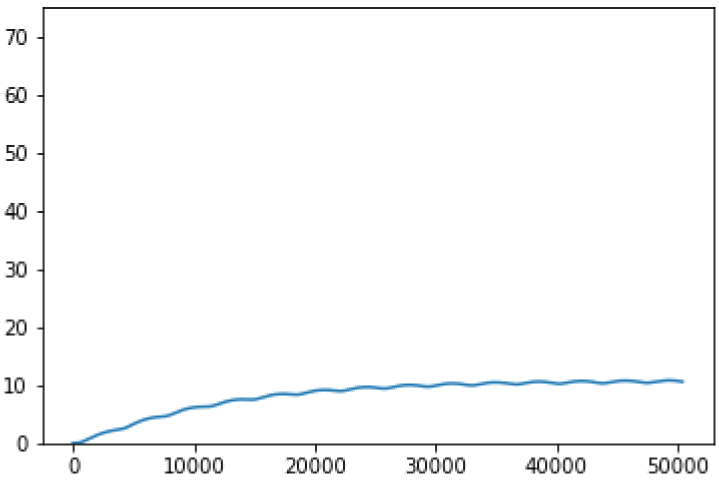
condition 49



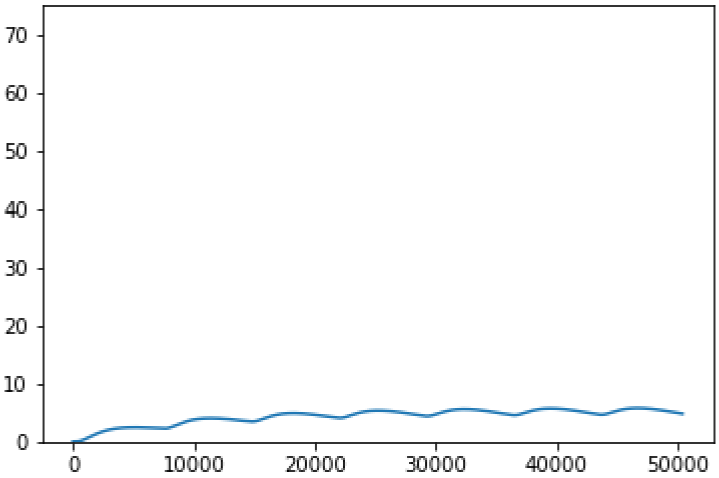
condition 50



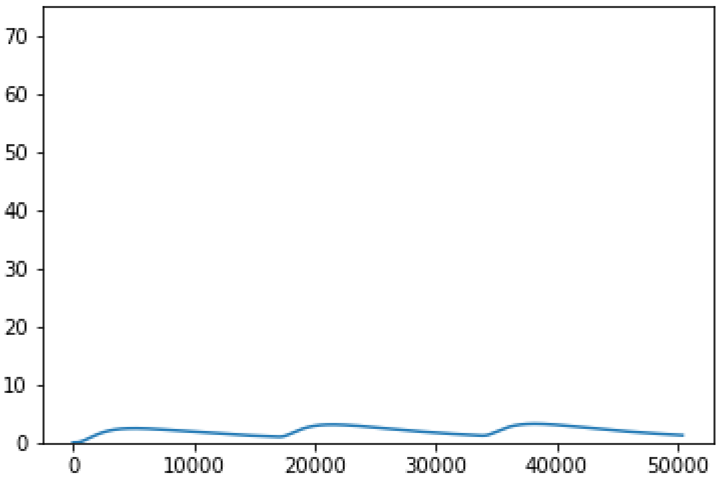
condition 51



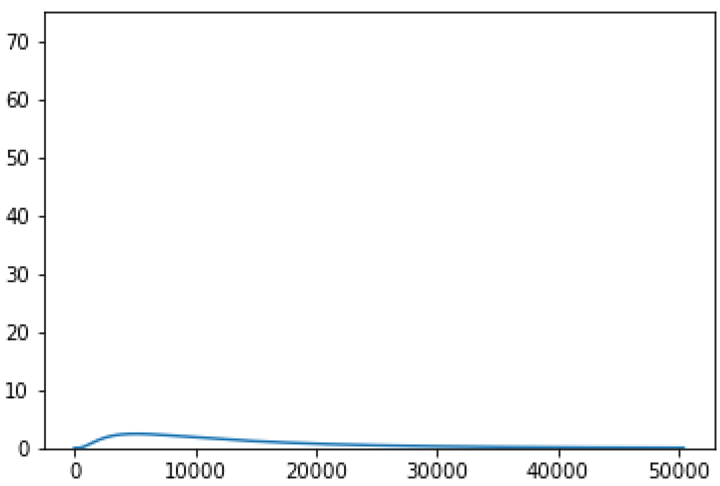
condition 52



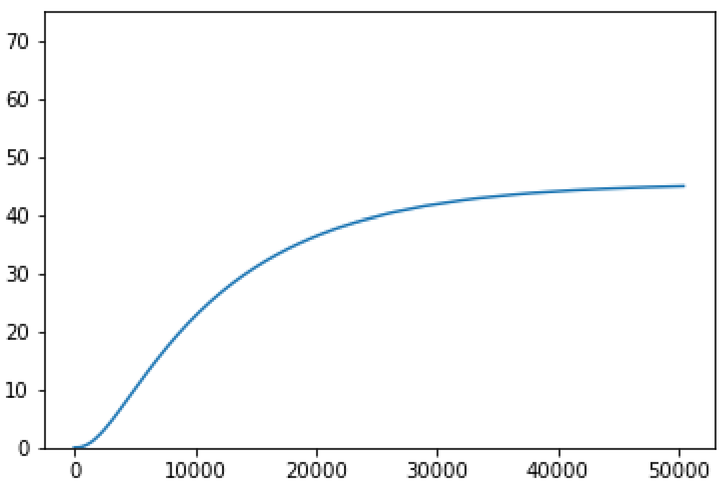
condition 53



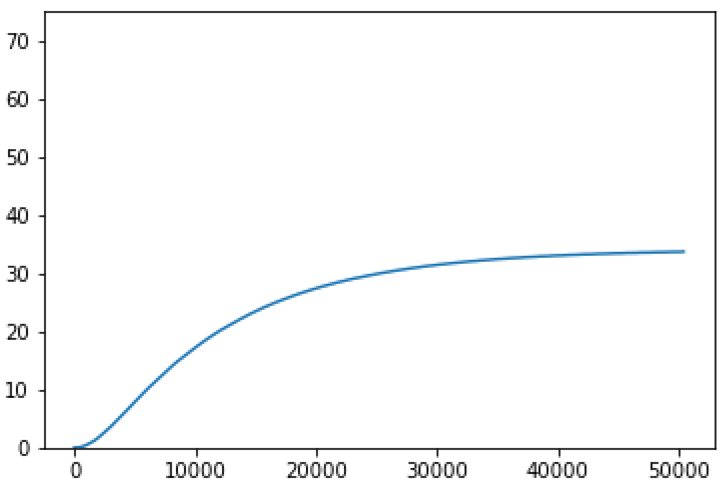
condition 54



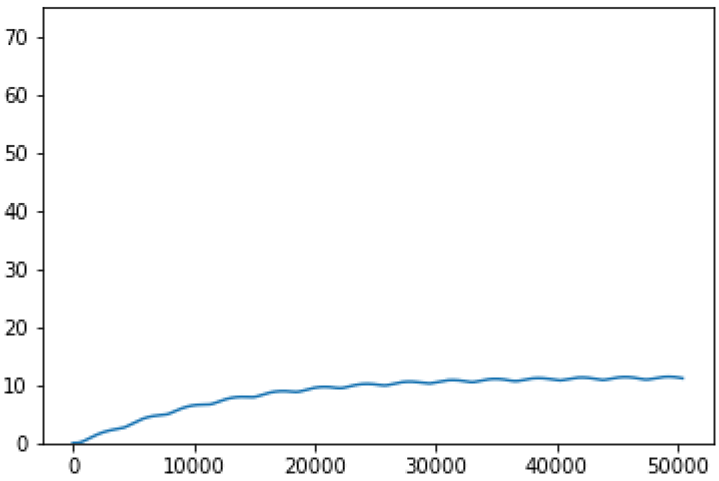
condition 55



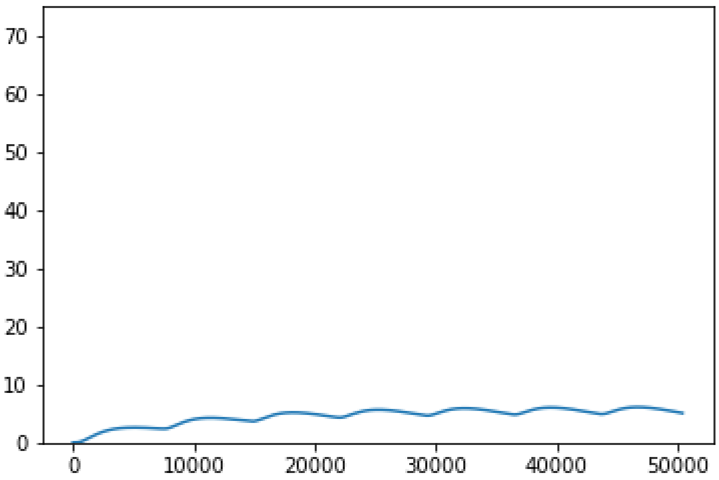
condition 56



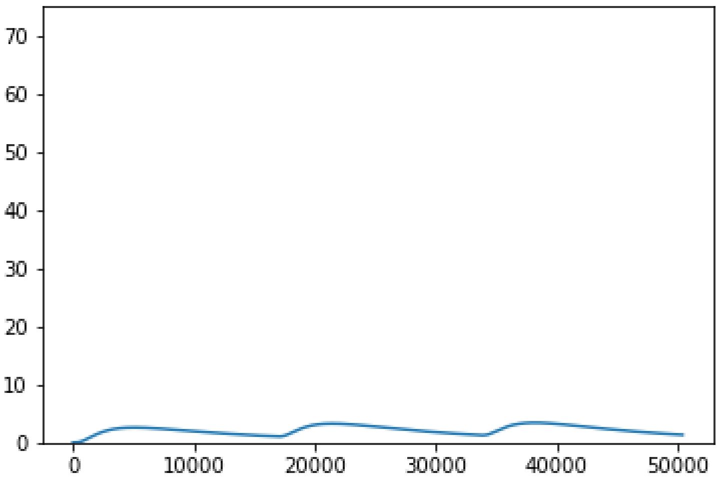
condition 57

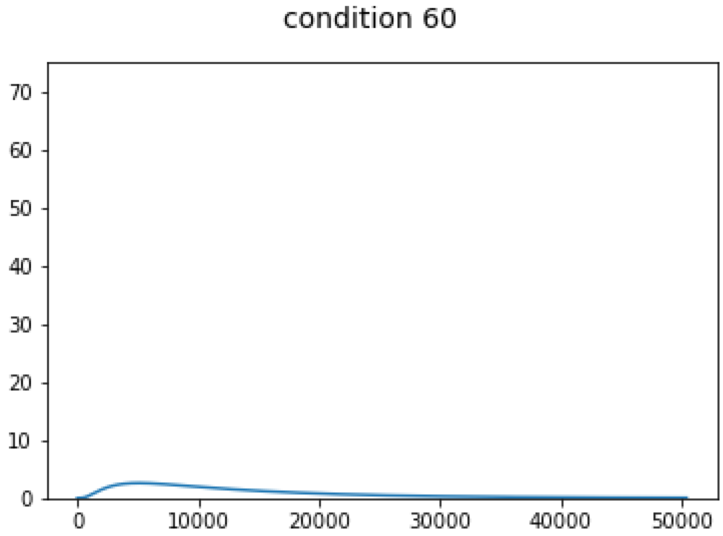


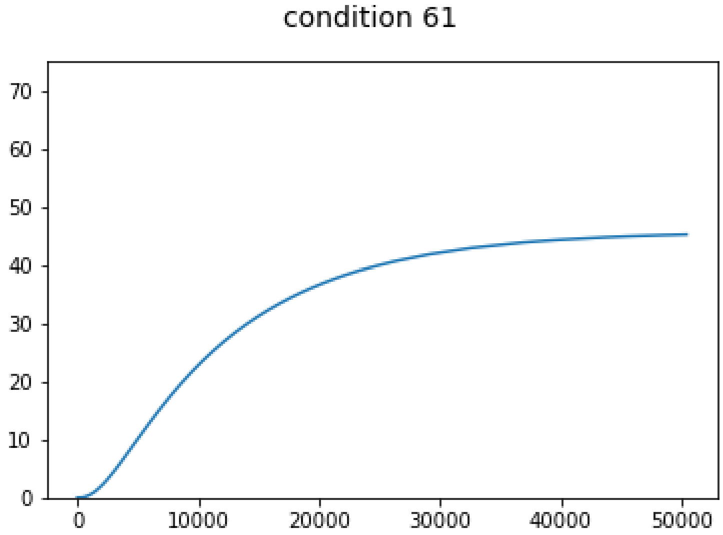
condition 58

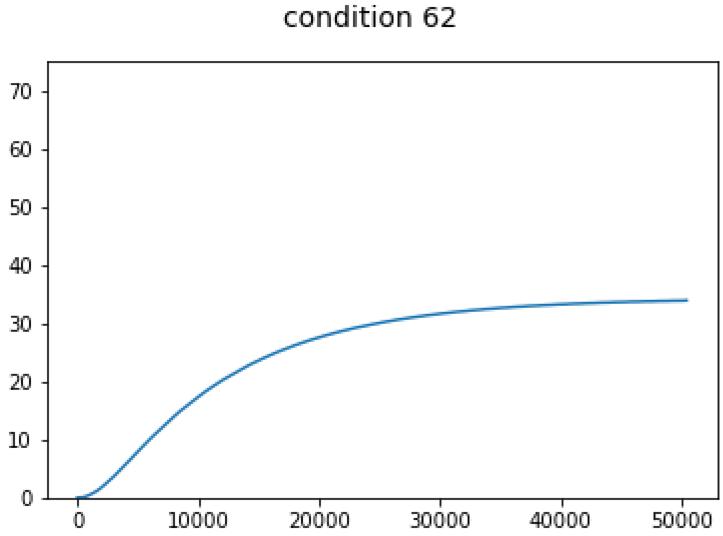


condition 59

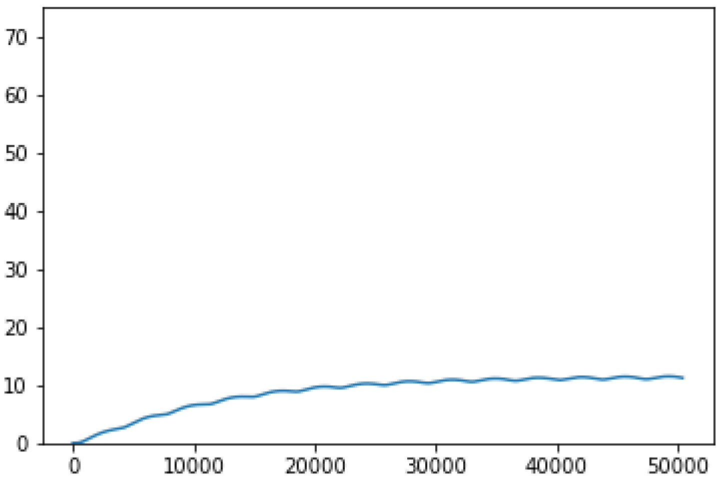



condition 60
70
60
50
40
30
20
10
0
0
10000
20000
30000
40000
50000

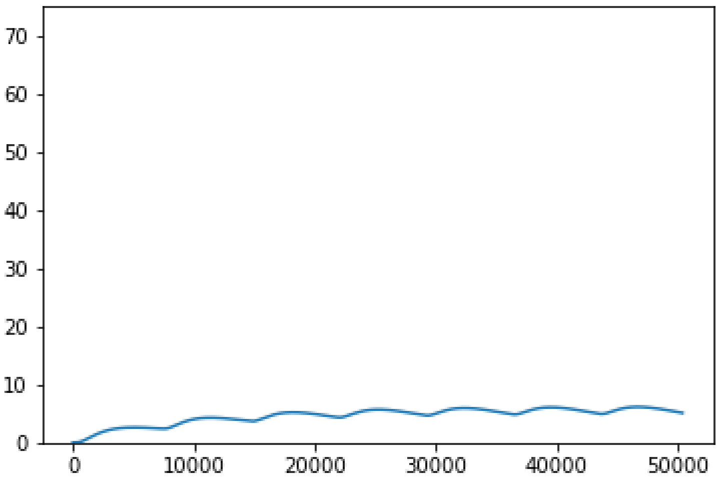

condition 61
70
60
50
40
30
20
10
0
0
10000
20000
30000
40000
50000


condition 62
70
60
50
40
30
20
10
0
0
10000
20000
30000
40000
50000

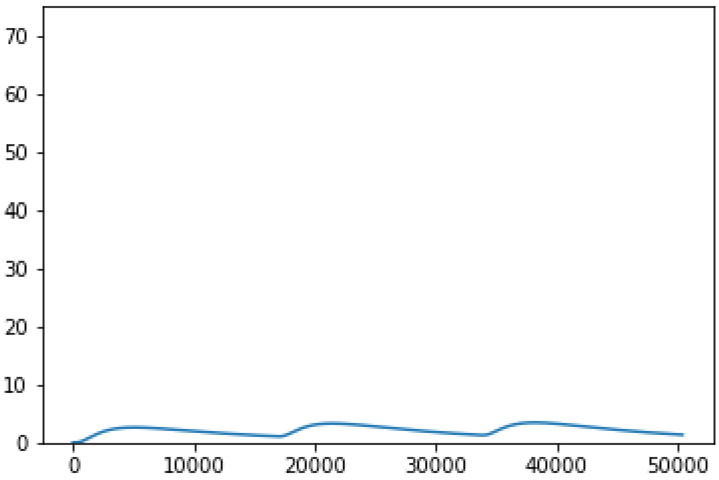
condition 63



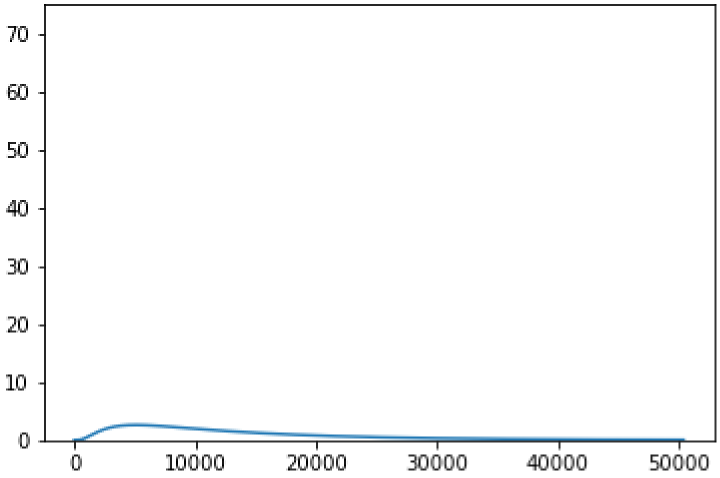
condition 64



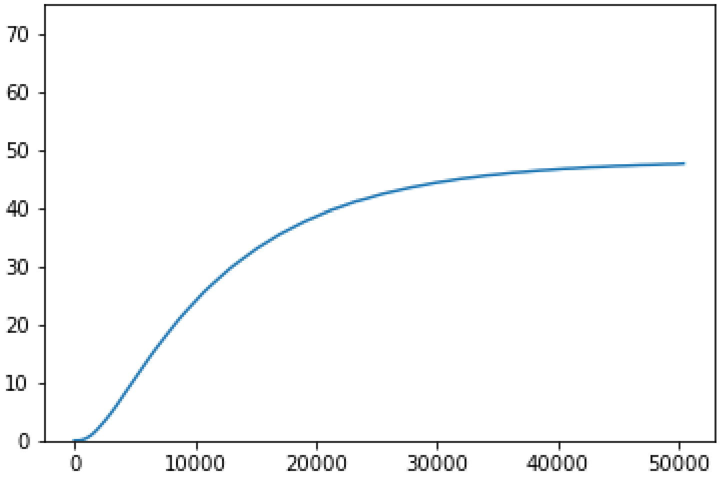
condition 65



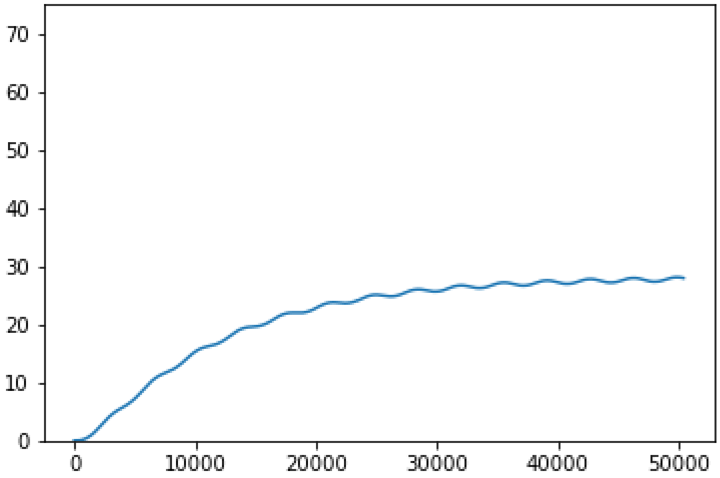
condition 66

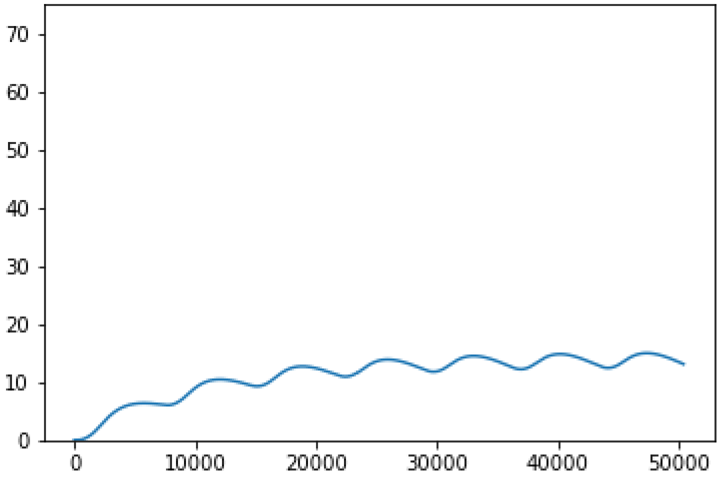


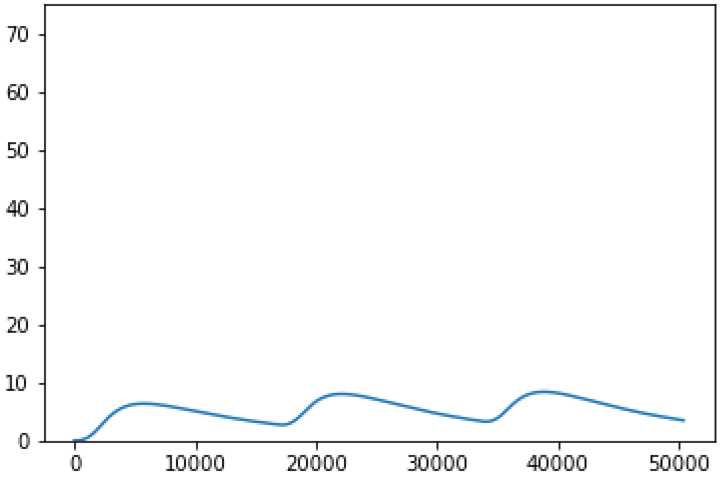
condition 67

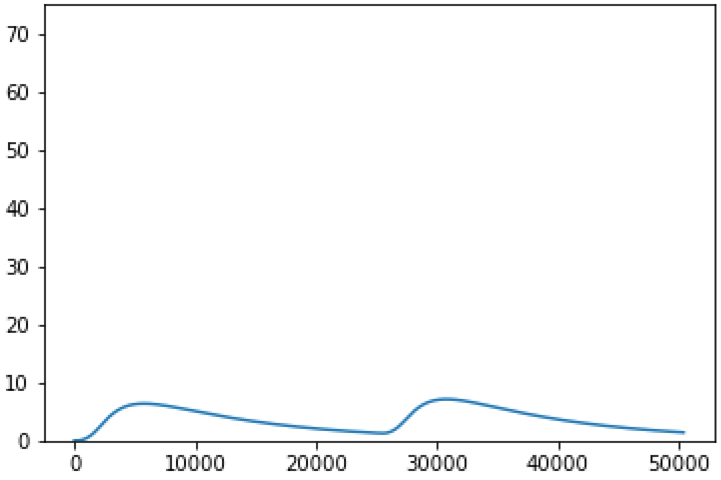


condition 68

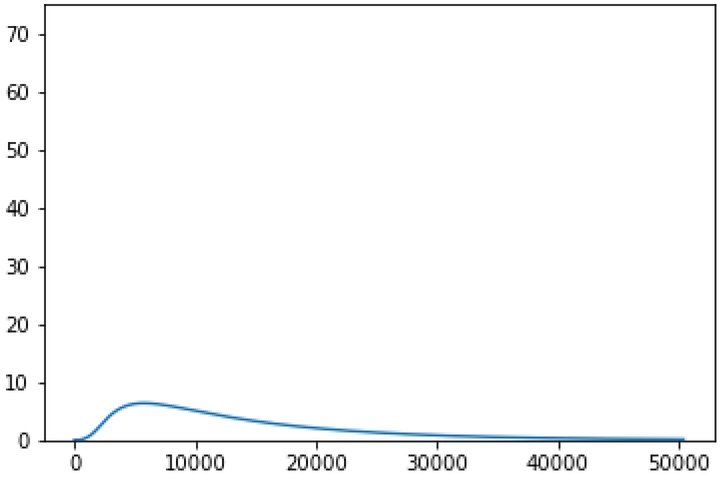



condition 69

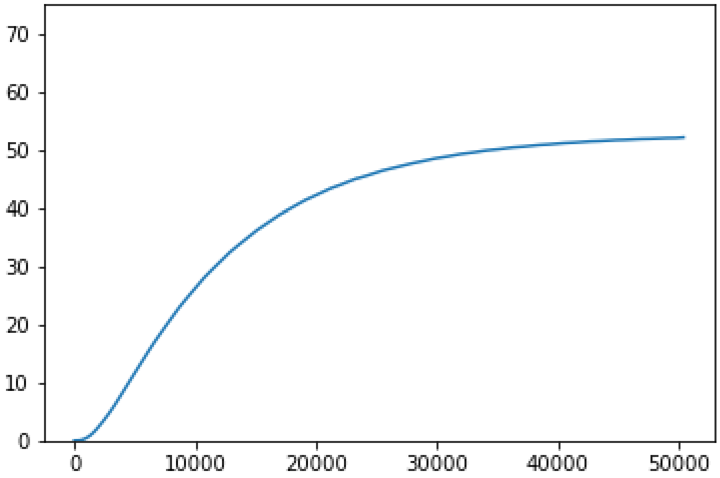

condition 70


condition 71

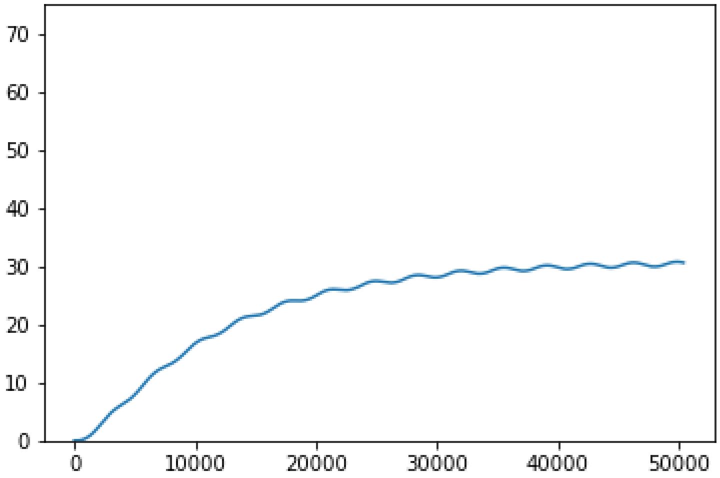
condition 72



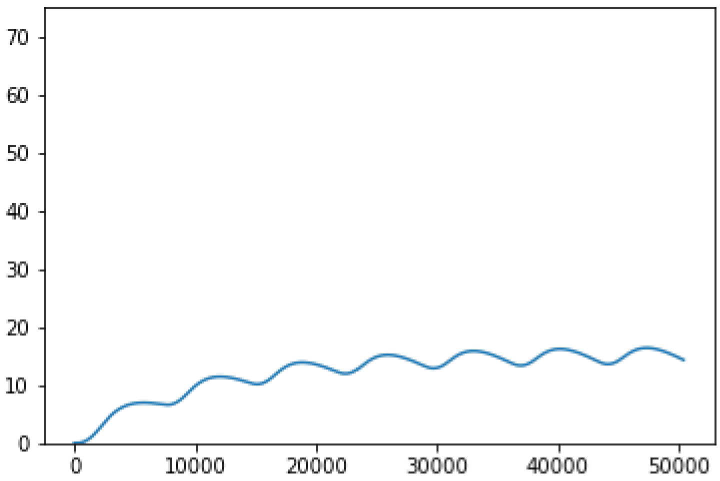
condition 73



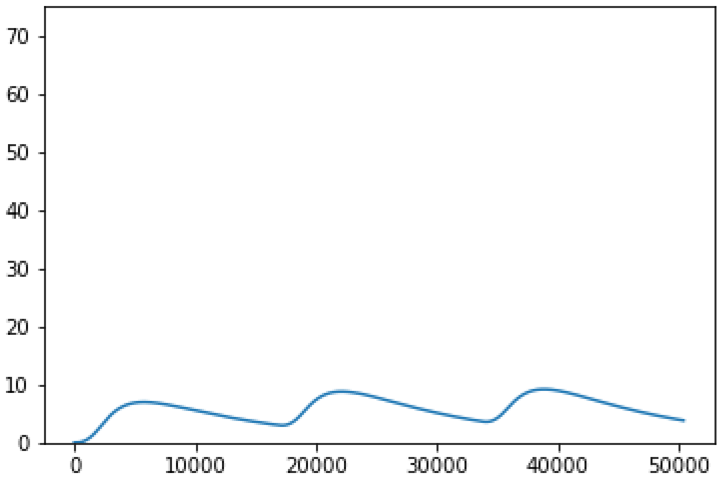
condition 74



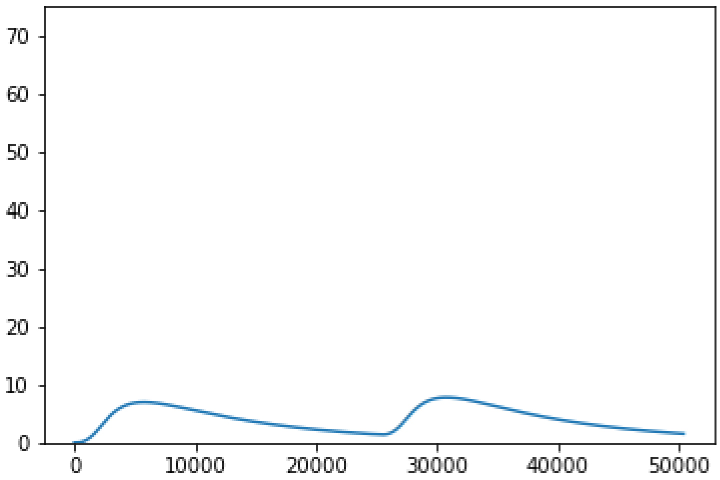
condition 75



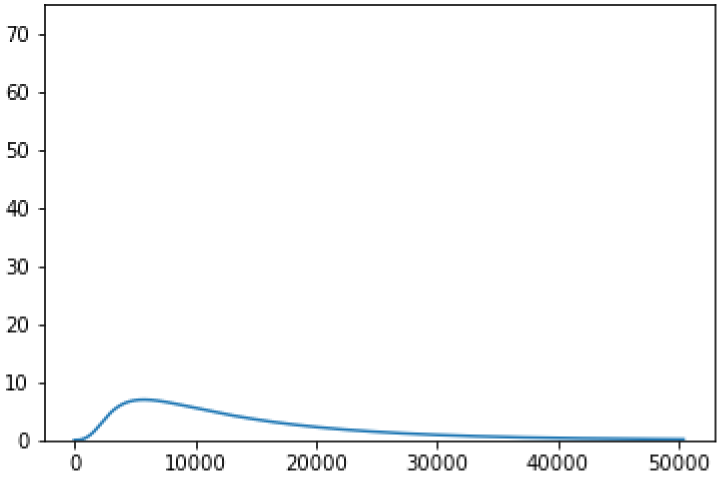
condition 76



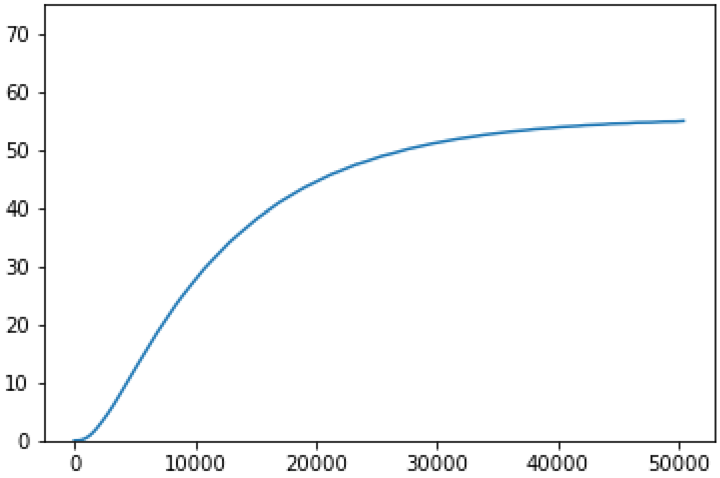
condition 77



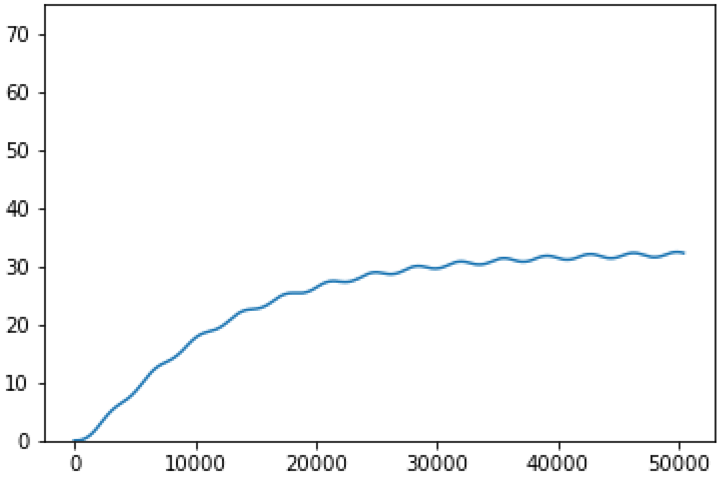
condition 78



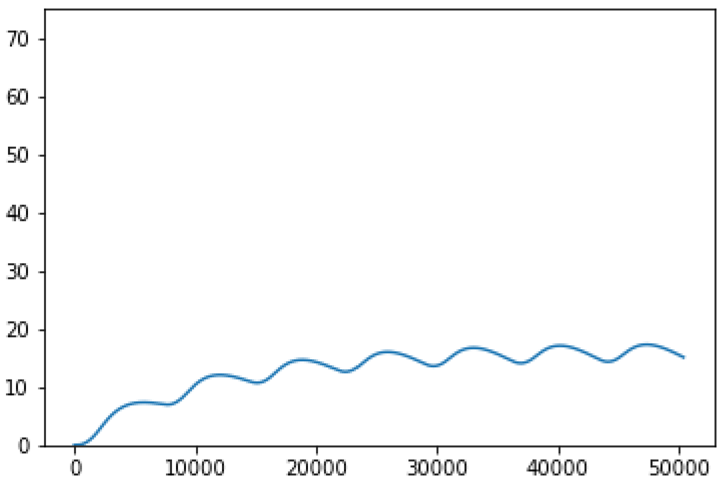
condition 79



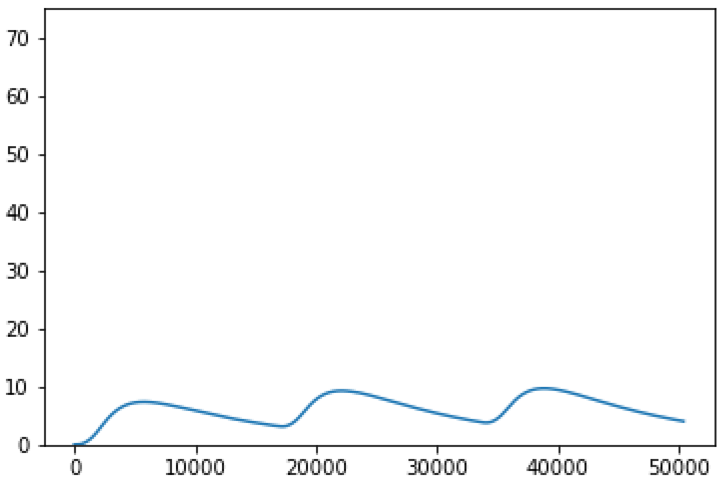
condition 80



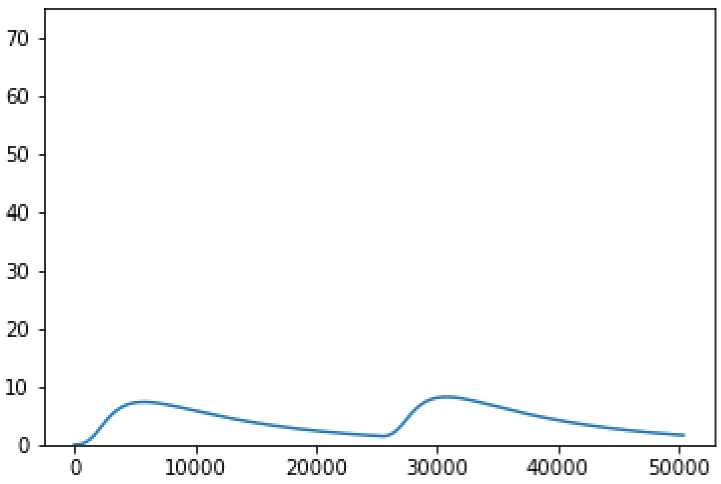
condition 81

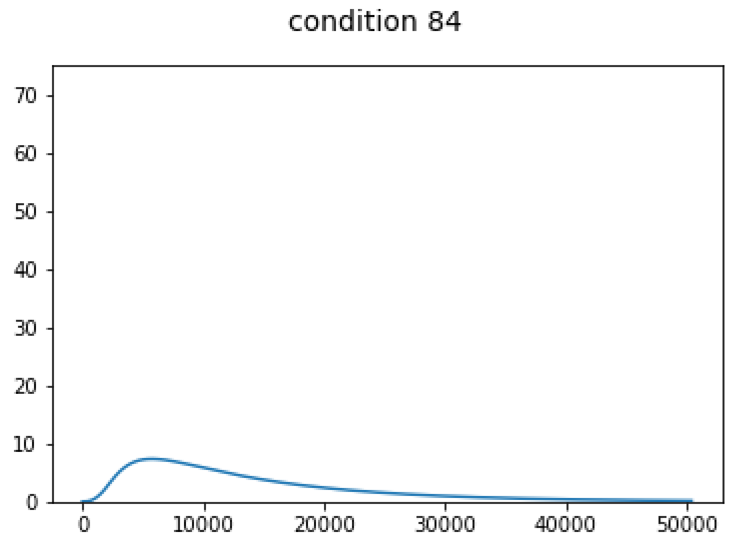


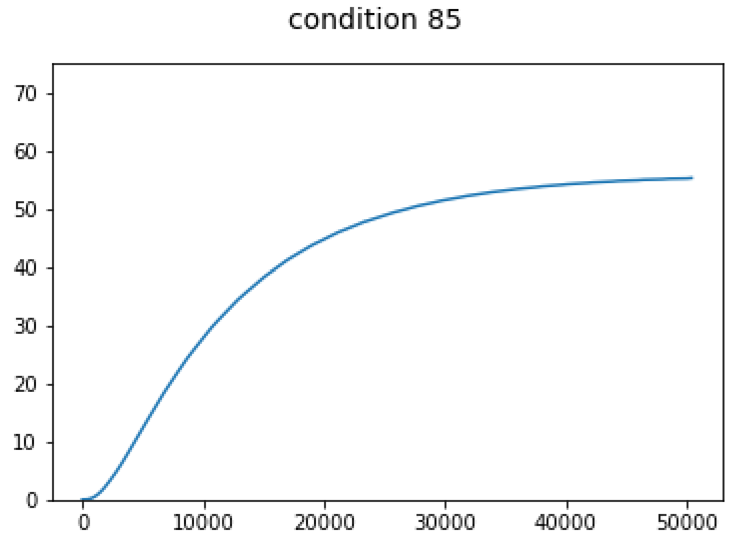
condition 82

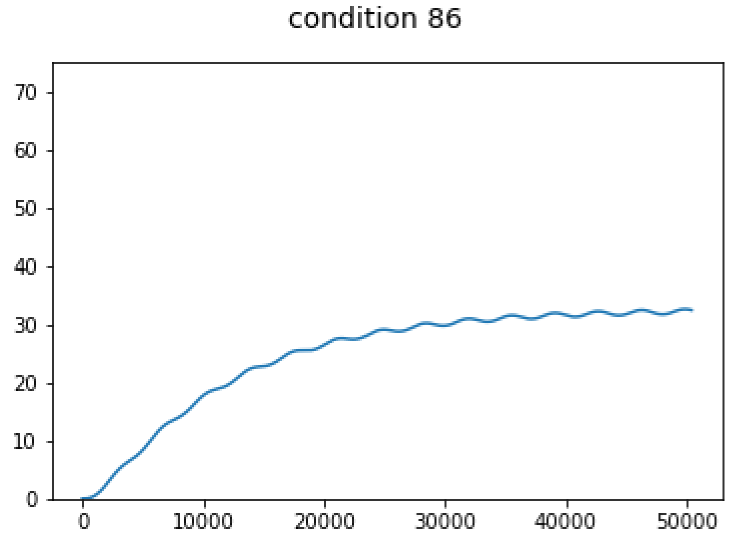


condition 83

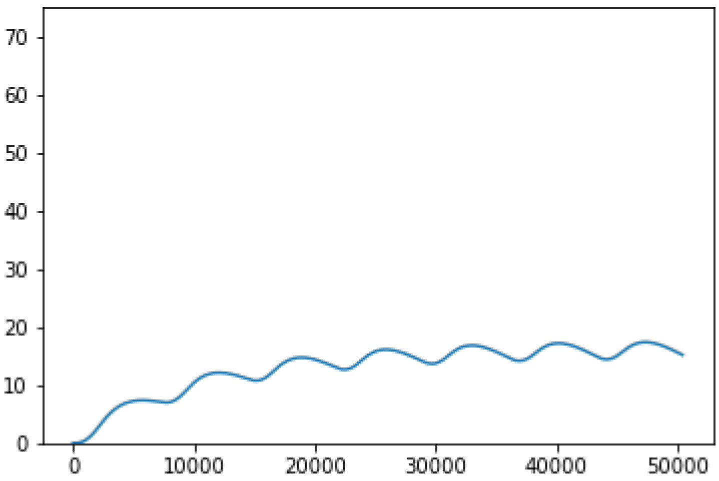



condition 84
70
60
50
40
30
20
10
0
0
10000
20000
30000
40000
50000

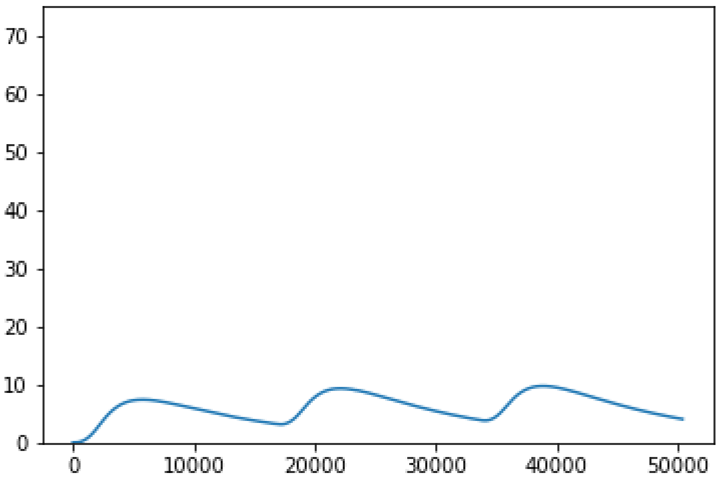

condition 85
70
60
50
40
30
20
10
0
0
10000
20000
30000
40000
50000


condition 86
70
60
50
40
30
20
10
0
0
10000
20000
30000
40000
50000

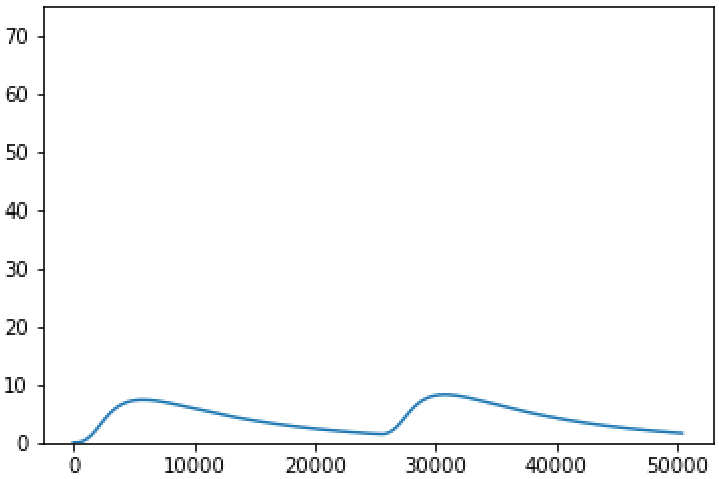
condition 87

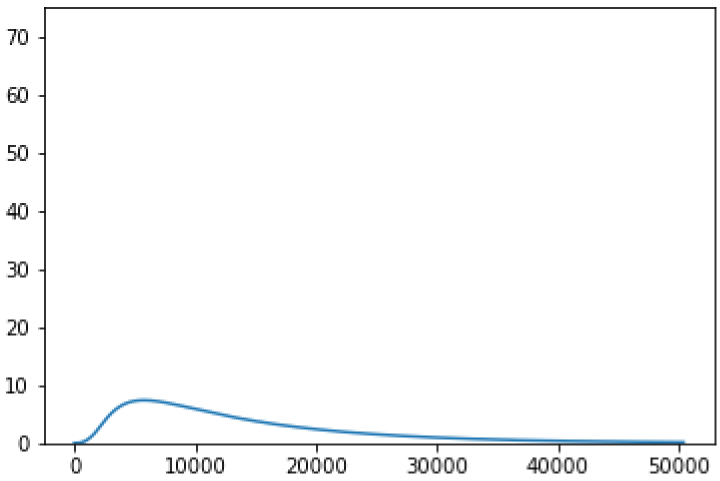


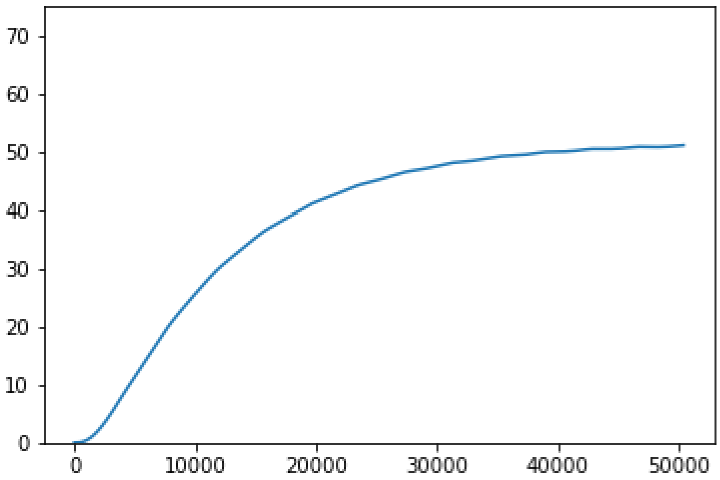
condition 88

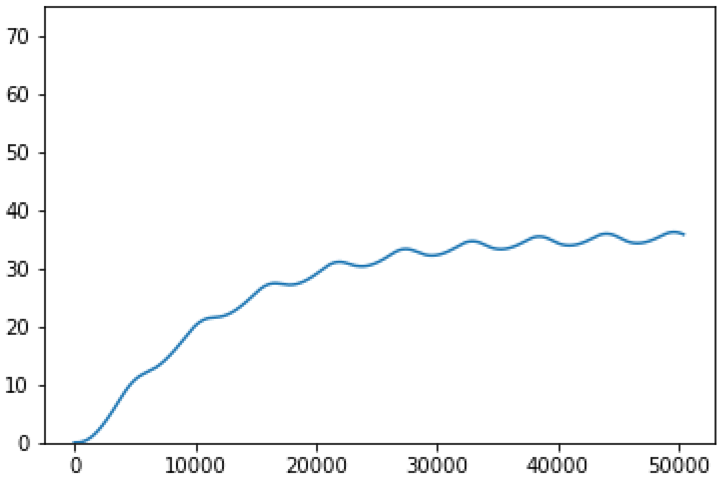


condition 89

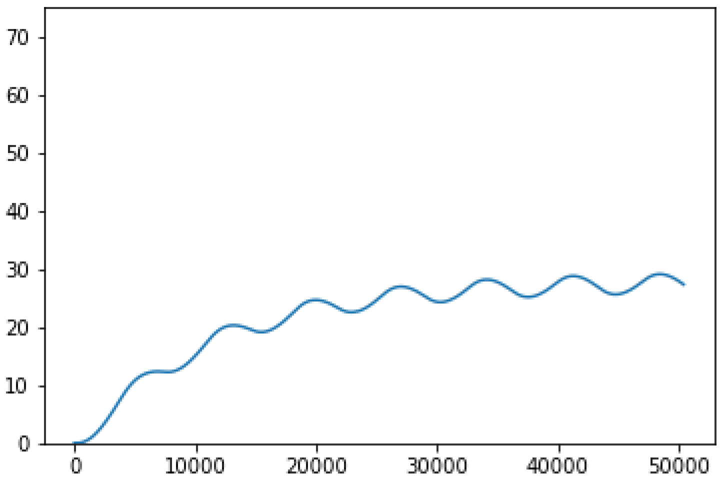



condition 90

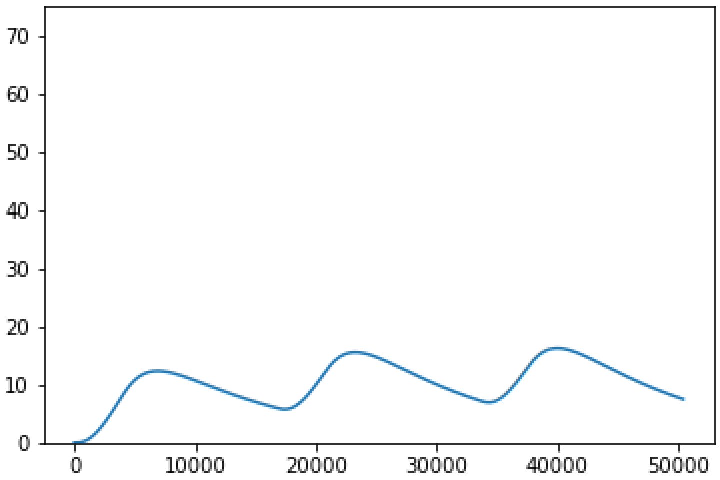

condition 91


condition 92

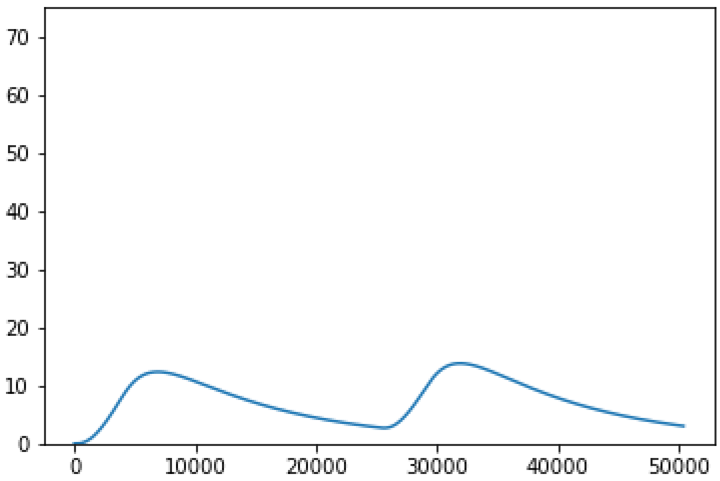
condition 93



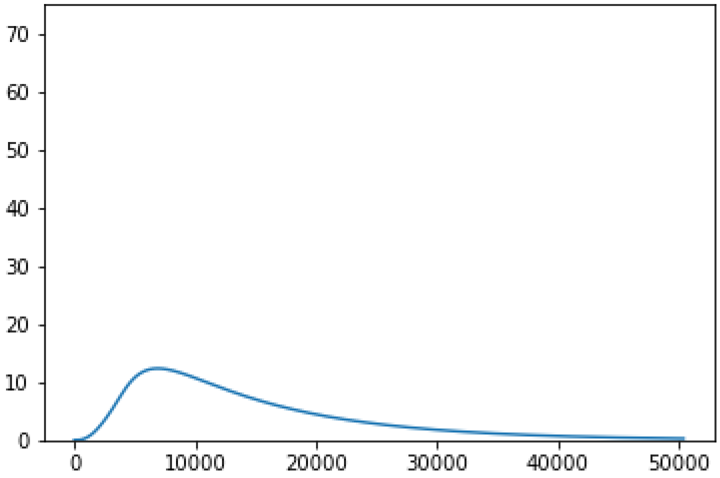
condition 94



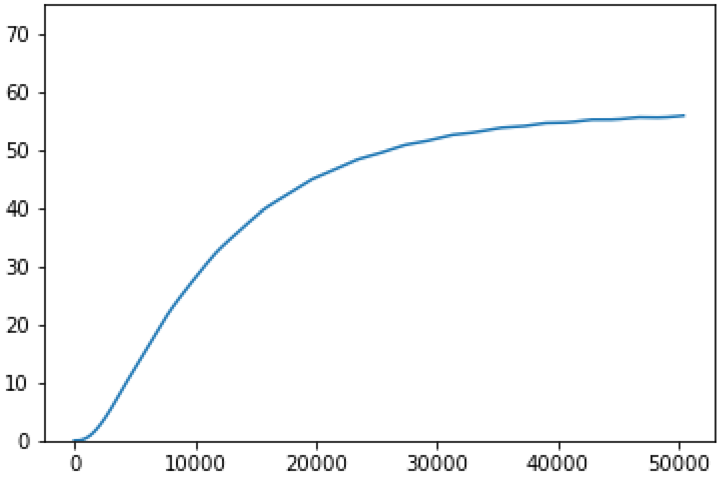
condition 95



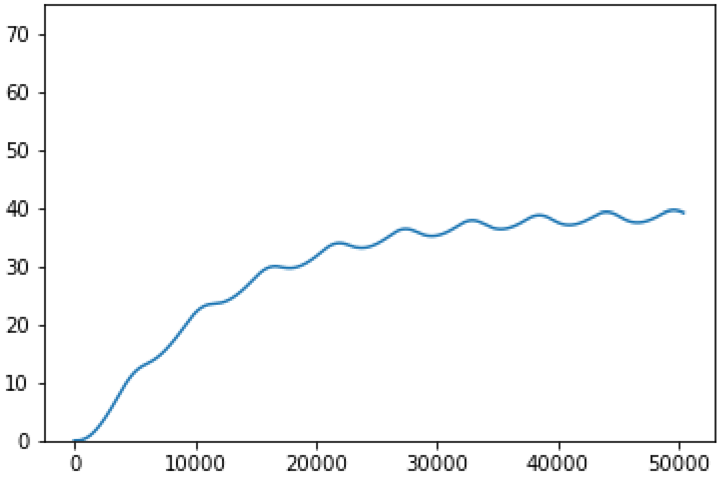
condition 96



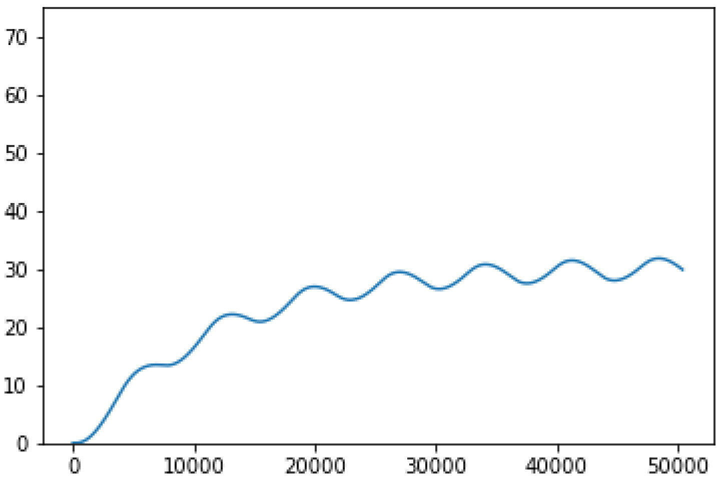
condition 97



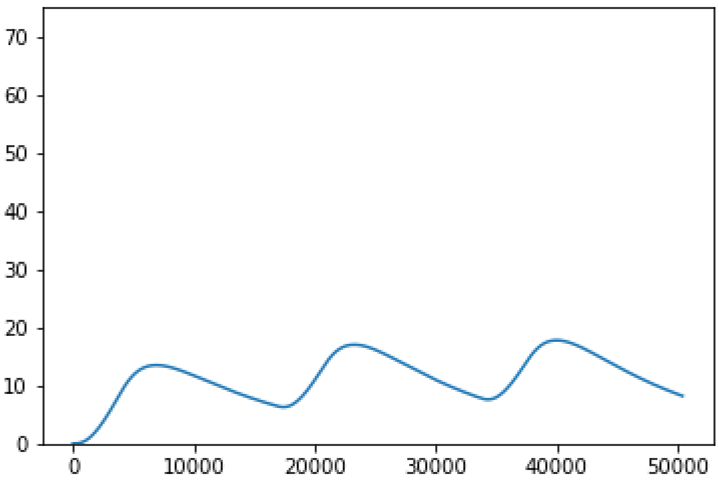
condition 98



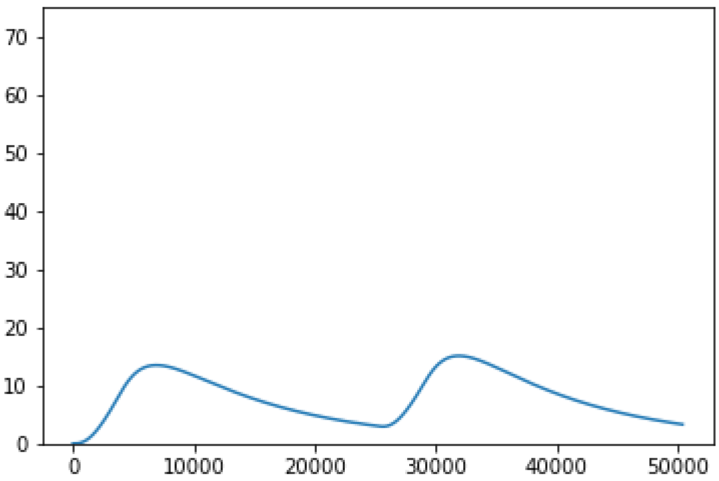
condition 99



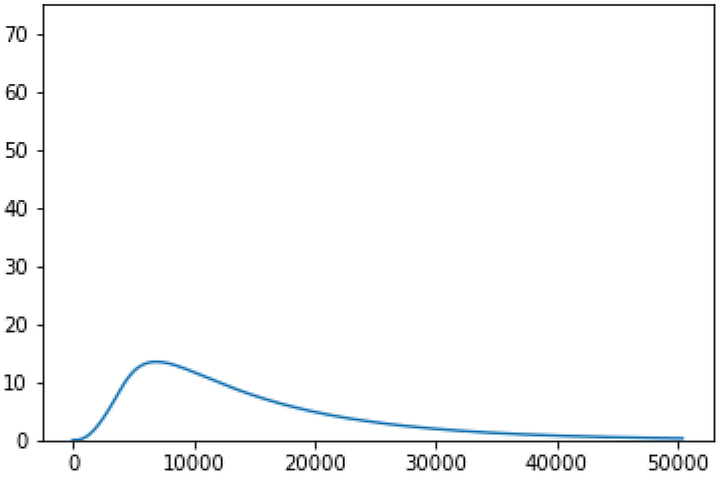
condition 100



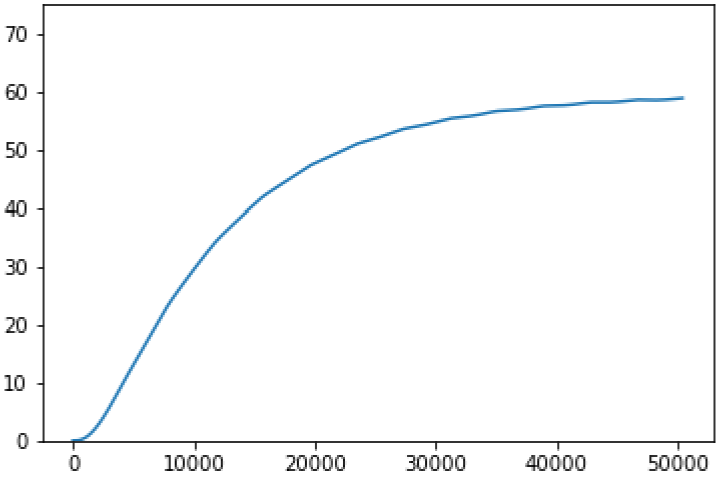
condition 101



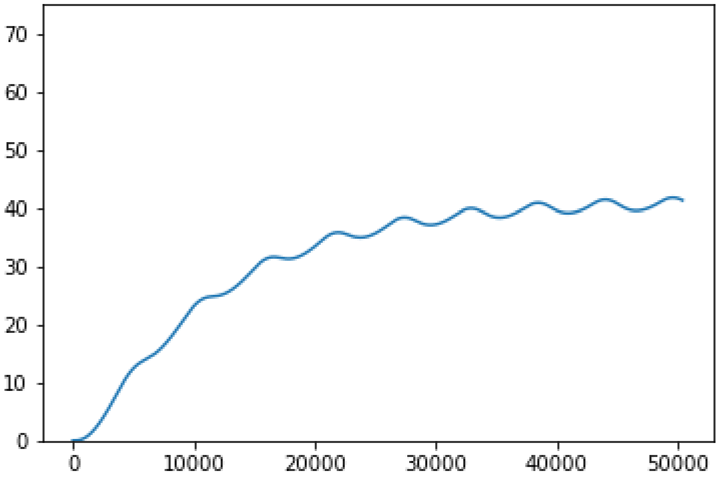
condition 102



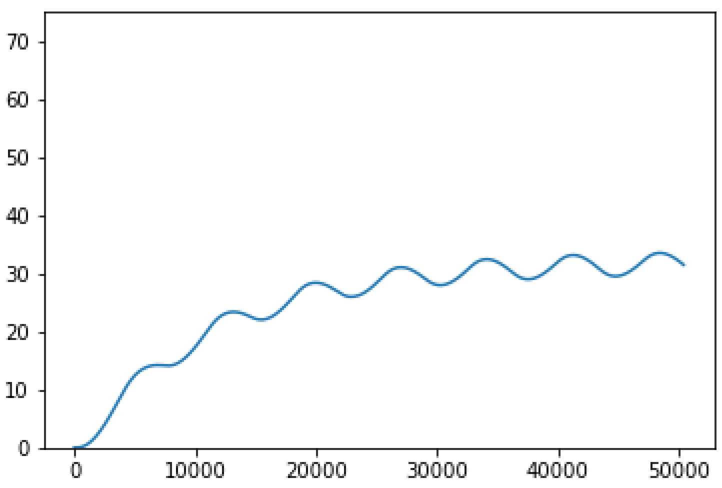
condition 103



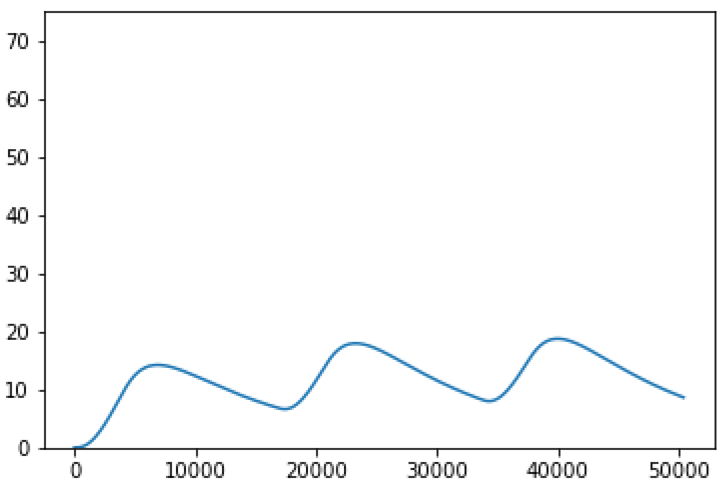
condition 104



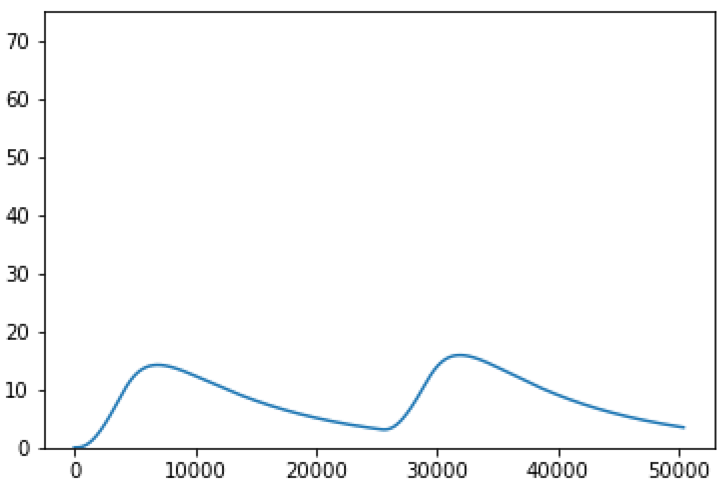
condition 105

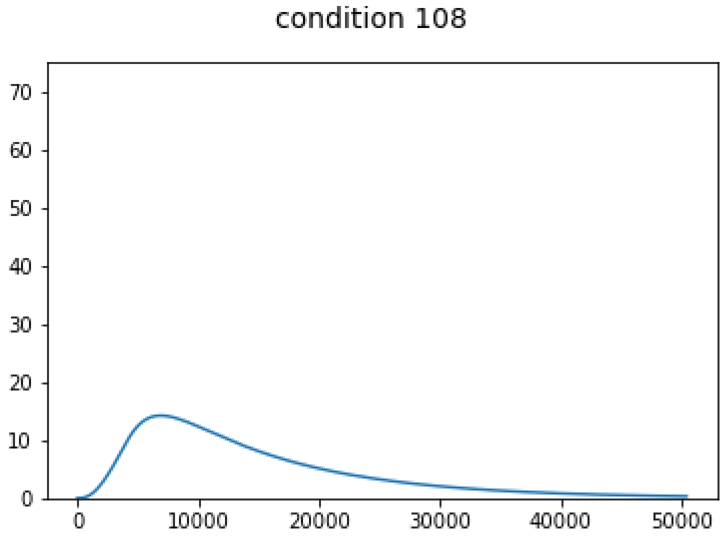


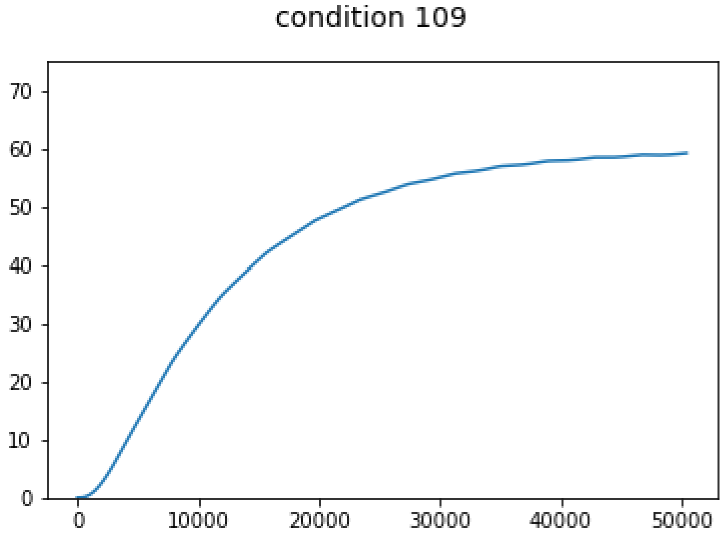
condition 106

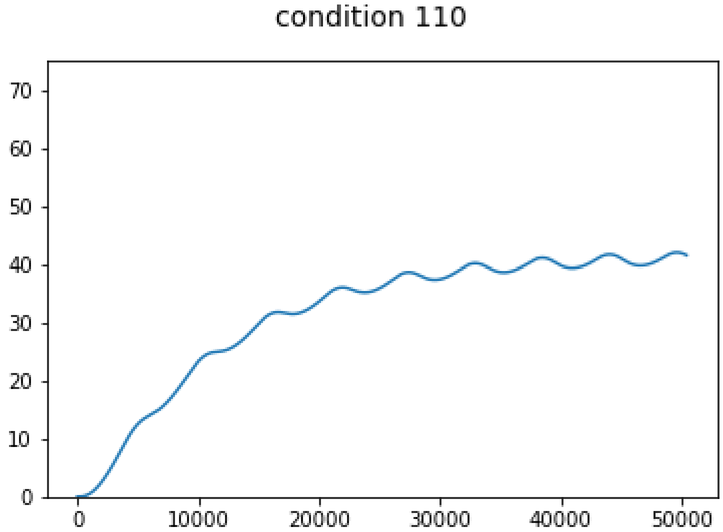


condition 107

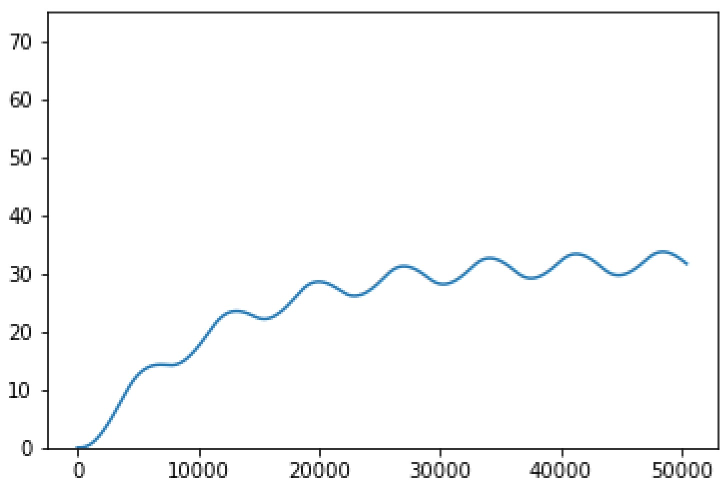



condition 108

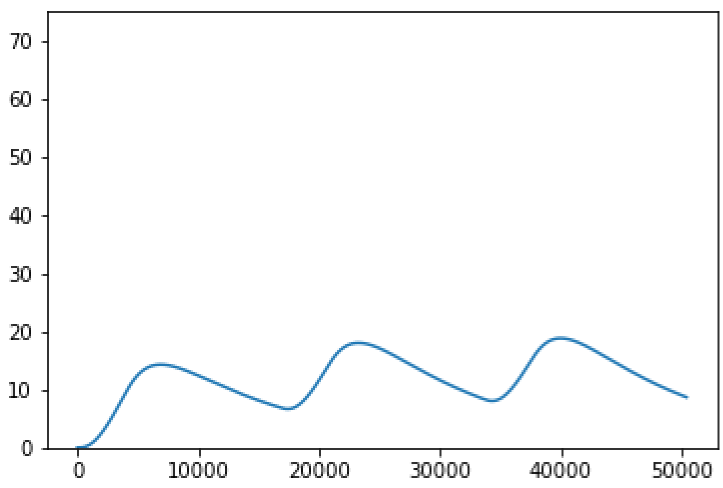

condition 109


condition 110

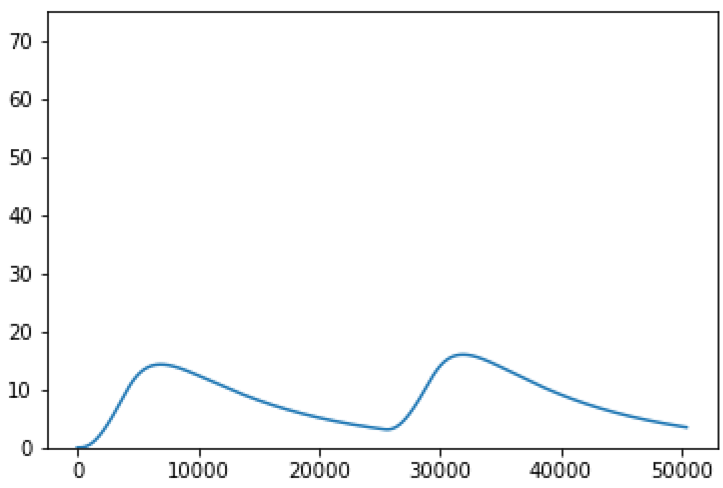
condition 111



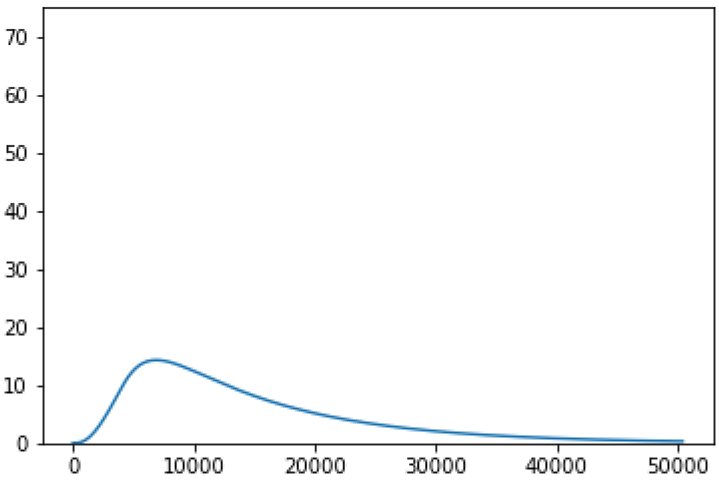
condition 112



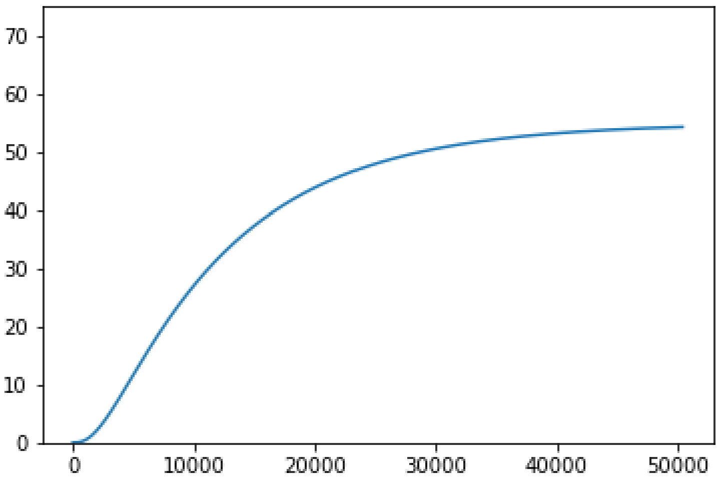
condition 113



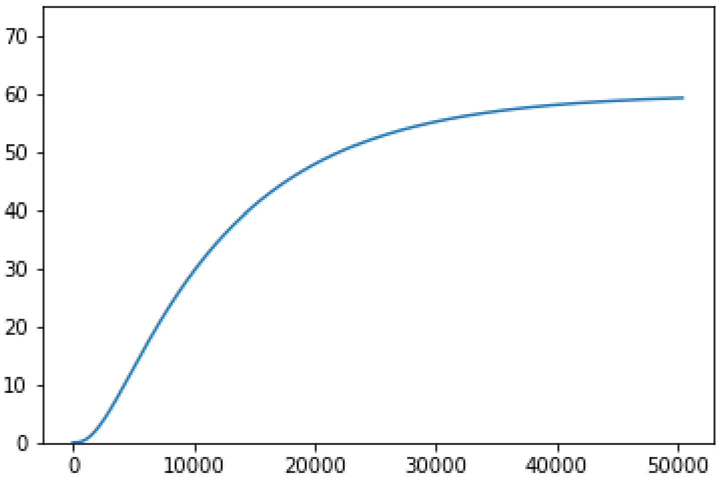
condition 114

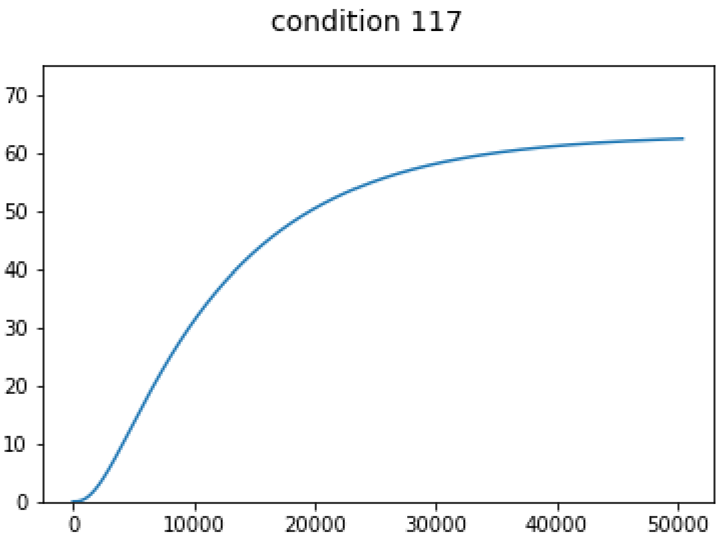


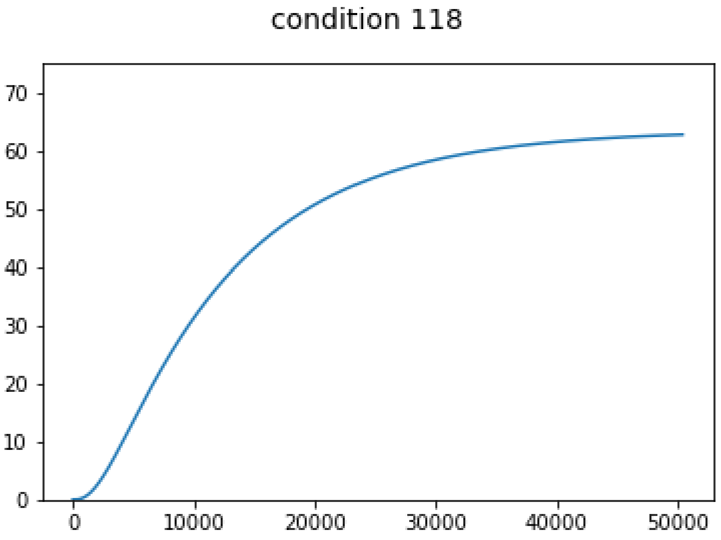
condition 115



condition 116




condition 117


condition 118

```
In [ ]:
```