Last Updated: 11/01/17

**Recreating Figures in:**

**“Understanding Variation in Macrophage Cell Death Following *M. tuberculosis* Infection: A Systems Biology Approach (Crutcher, 2017).**

**Figures 1-6 are not plots, but rather equations and diagrams.**

**Figure 7(a): generated using model\_Syt7.py**

from run\_model\_Syt7at import \*

chg\_val(m.PKA\_0,0) #figure 7a compares the relationship between mitochondrial calcium loading and MPT by scanning abundances of ER calcium stores, so I just turn off any inhibitory effect of PKA here

W,X,Y,Z = create\_phase() #you'll notice this function is really just designed for this figure and 7c

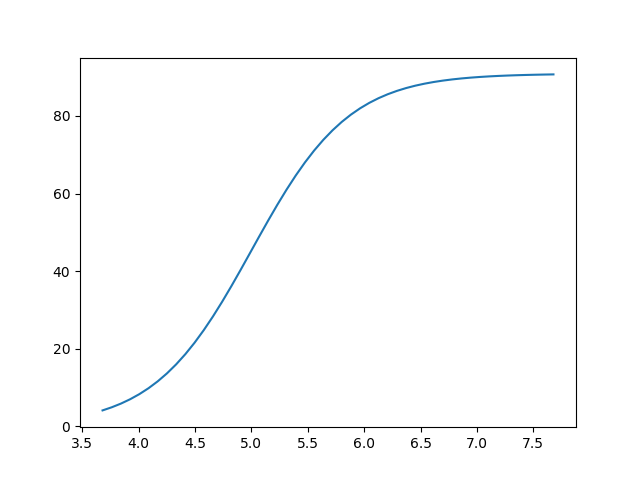
pl.ion()

fig = pl.figure()

pl.plot(W,X)

plt.xlabel('log Mito Ca++')

plt.ylabel('% MPT')



**Figure 7(b): generated using model\_Syt7.py**

chg\_val(m.eCa\_0,7e6) #if you've just done 7a and scanned ER calcium store abundances up to 1e8, this resets to default value

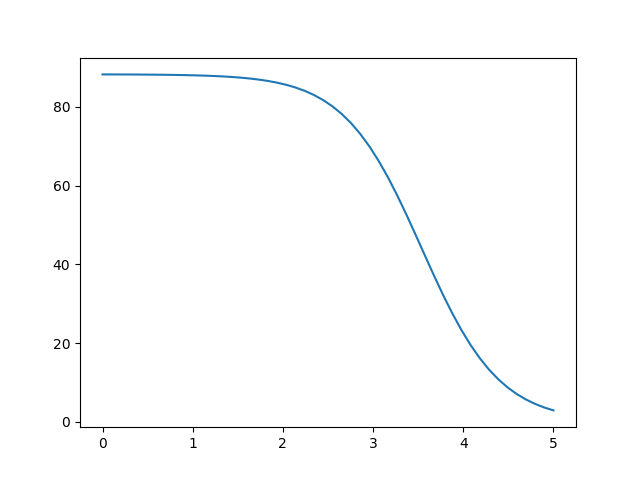
chg\_val(m.PKA\_MIM\_b\_kf,2e-5) #oops! I noted a discrepancy between this initial condition as coded in the model\_Syt7 file and the reaction list in my report. This is the correct value and renders the correct figure.

a,X,Y,Z = create\_phase2(m.PKA\_0,[0,5]) #this function is a generalization of create\_phase() in that it allows you to choose the parameter you want to scan, but the outputs are mostly the same.

pl.plot(a,X)

plt.xlabel('log PKA (Copies)')

plt.ylabel('% MPT')



**Figure 7(c): generated using model\_Syt7.py**

chg\_val(m.synth\_ROS\_k,0.1)

W,X,Y,Z = create\_phase()

pl.ion()

pl.figure()

pl.plot(X,Z)

chg\_val(m.synth\_ROS\_k,0)

W,X,Y,Z = create\_phase()

pl.plot(X,Z)

chg\_val(m.synth\_ROS\_k,0.05)

W,X,Y,Z = create\_phase()

pl.plot(X,Z)

chg\_val(m.synth\_ROS\_k,0.01)

W,X,Y,Z = create\_phase()

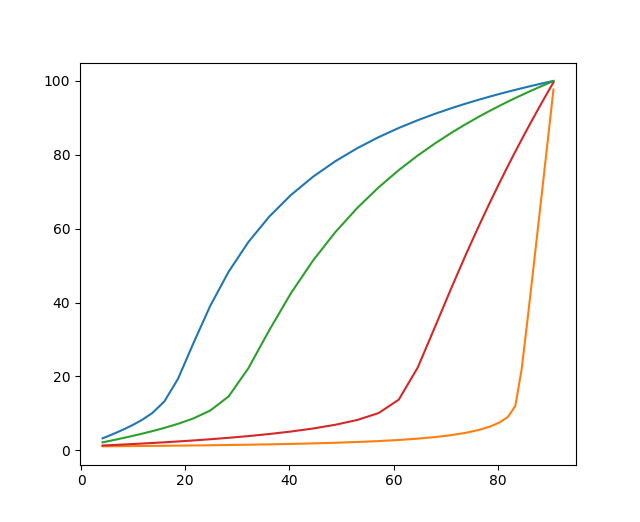
import matplotlib.pyplot as plt

pl.plot(X,Z)

plt.xlabel('% MPT')

plt.ylabel('Osmotic Stress (fold [Na] increase from baseline)')

Plt.legend(['Mem Damage (ROS\_ks = 0.1)','No Mem Damage (ROS\_ks = 0)','ROS\_ks = 0.05','ROS\_ks = 0.01'])



**Figure 7(d): generated using model\_Necrosis.py**

from run\_model\_Necrosisat import \*

chg\_val(m.Syt7\_0,0)

a,X,Y,Z = create\_phase2(m.PKA\_0,[0,5])

pl.figure()

pl.plot(a,Z)

chg\_val(m.Syt7\_0,2500)

a,X,Y,Z = create\_phase2(m.PKA\_0,[0,5])

pl.plot(a,Z)

chg\_val (m.Syt7\_0,5000)

a,X,Y,Z = create\_phase2(m.PKA\_0,[0,5])

pl.plot(a,Z)

chg\_val (m.Syt7\_0,7500)

a,X,Y,Z = create\_phase2(m.PKA\_0,[0,5])

pl.plot(a,Z)

chg\_val(m.Syt7\_0,10000)

a,X,Y,Z = create\_phase2(m.PKA\_0,[0,5])

pl.plot(a,Z)

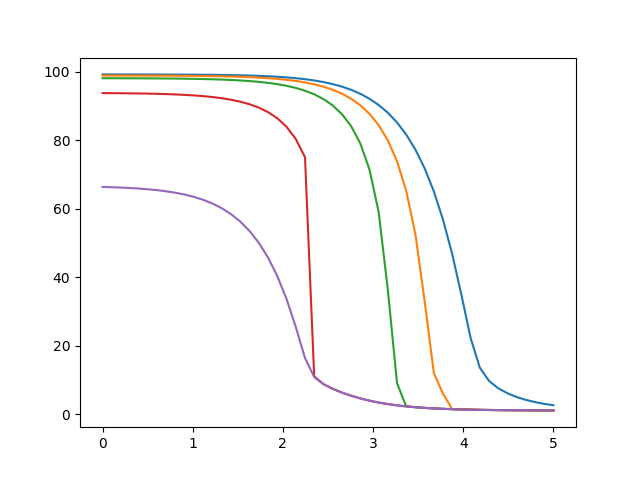
import matplotlib.pyplot as plt

plt.scatter(a,Z,s=None,c='k',marker='X')

plt.legend(['Syt7 = 0','Syt7 = 2500','Syt7 =5000','Syt7 =7500','Syt7 =10000','No Mem Damage'])

plt.xlabel('Log PKA (Copies)')

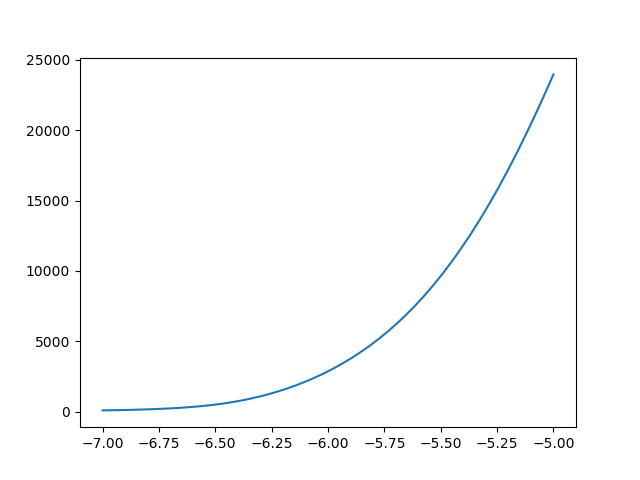
plt.ylabel('Osmotic Stress (Fold [Na] Increase from Baseline)')



**Figure 9(a): generated using Frontside.py**

Chg\_val(m.LO\_AA\_b\_kf, 1e-08)

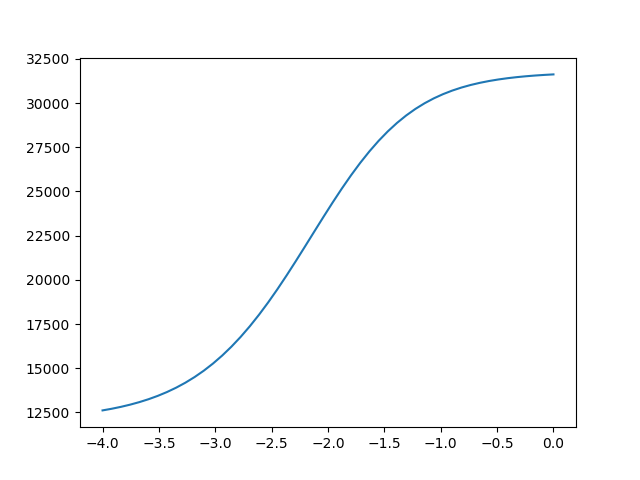
scan\_custom(m.p38p\_HSP27\_b\_kf,[-7,-5],'o\_Cox')

****

**Figure 9(b): generated using Frontside.py**

chg\_val(p38p\_HSP27\_b\_kf,3e-6)

scan\_custom(m.synth\_PIP3\_k,[-4,0],'o\_Cox')

****

**Figure 10(a): generated using Frontside.py**

chg\_val(m.LO\_AA\_b\_kf,1e-5)

from run\_Frontside import \*

chg\_val(m.LO\_0,1e5)

graph('o\_LXA4')

chg\_val(m.LO\_0,1e4)

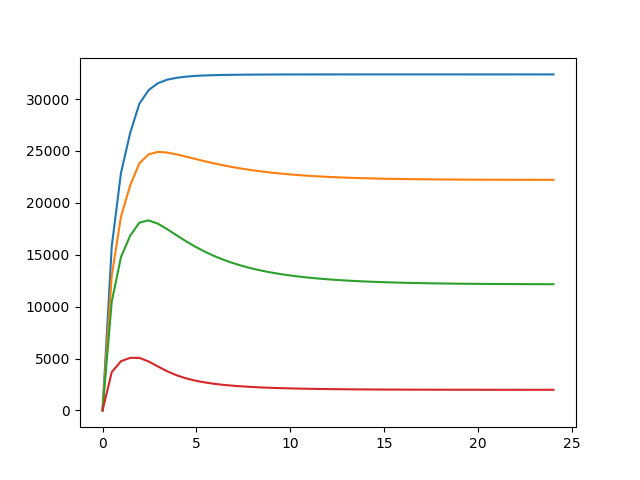
graph('o\_LXA4',prep\_fig=False)

chg\_val(m.LO\_0,5e3)

graph('o\_LXA4',prep\_fig=False)

chg\_val(m.LO\_0,1e3)

graph('o\_LXA4',prep\_fig=False)



**Figure 10(b): generated using Frontside.py**

chg\_val(m.LO\_0,1e3)

pl.figure()

s.run()

output = s.yobs['o\_PGE2'] + s.yobs['o\_EP2'] + s.yobs['o\_EP4']

pl.plot(t/3600,output)

chg\_val(m.LO\_0,5e3)

s.run()

output = s.yobs['o\_PGE2'] + s.yobs['o\_EP2'] + s.yobs['o\_EP4']

pl.plot(t/3600,output)

chg\_val(m.LO\_0,1e4)

s.run()

output = s.yobs['o\_PGE2'] + s.yobs['o\_EP2'] + s.yobs['o\_EP4']

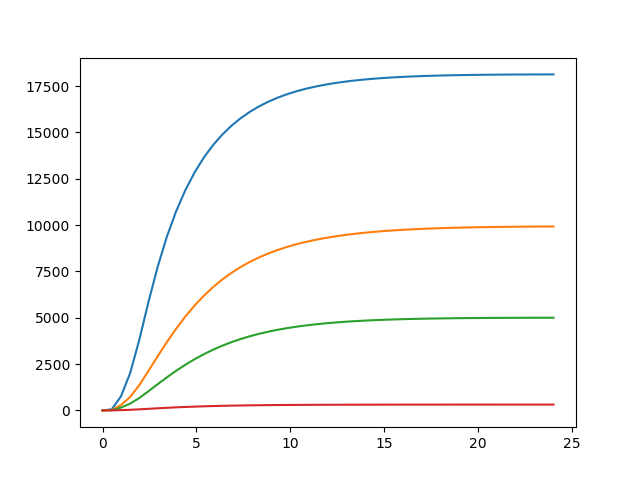
pl.plot(t/3600,output)

chg\_val(m.LO\_0,1e5)

s.run()

output = s.yobs['o\_PGE2'] + s.yobs['o\_EP2'] + s.yobs['o\_EP4']

pl.plot(t/3600,output)



**Figure 10(c): generated using Frontside.py**

chg\_val(m.LO\_0,1e3)

graph('o\_NFkB')

chg\_val(m.LO\_0,5e3)

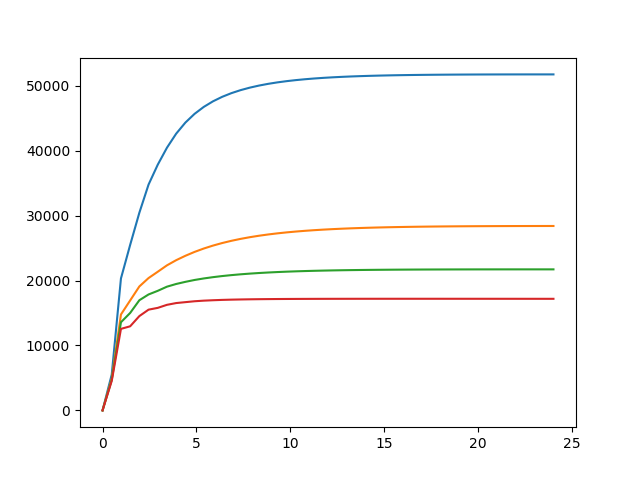
graph('o\_NFkB',prep\_fig=False)

chg\_val(m.LO\_0,1e4)

graph('o\_NFkB',prep\_fig=False)

chg\_val(m.LO\_0,1e5)

graph('o\_NFkB',prep\_fig=False)



**Figure 11: generated using Frontside.py**

\*\*\*Runs very slow.

LO\_wrap([3,5],'o\_NFkB',norm=True)

comp\_LO\_TNF([3,5],norm=True)

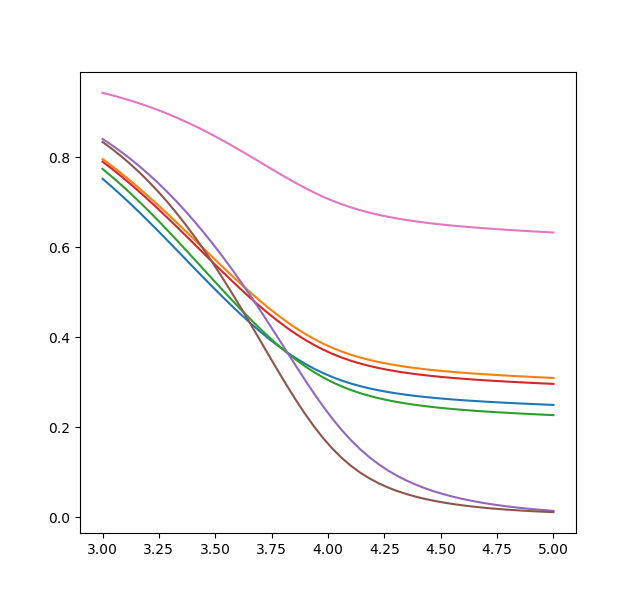
LO\_wrap([3,5],'o\_Cox',norm=True)

LO\_wrap([3,5],'o\_Syt7',norm=True)

comp\_LO\_PGE2([3,5],norm=True)

LO\_wrap([3,5],'o\_PKA',norm=True)

LO\_wrap([3,5],'o\_DISC',norm=True)



**Figure 12(a): generated using Frontside.py**

chg\_val(m.synth\_AA\_k, 0.05)

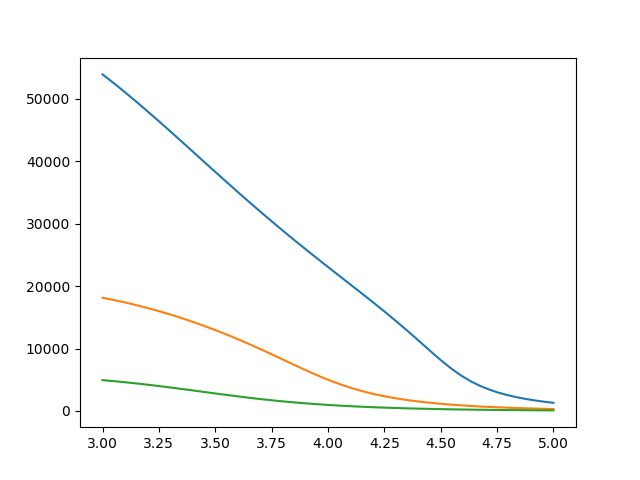
comp\_LO\_PGE2([3,5])

chg\_val(m.synth\_AA\_k, 0.01)

comp\_LO\_PGE2([3,5])

chg\_val(m.synth\_AA\_k, 0.002)

comp\_LO\_PGE2([3,5])



**Figure 12(b): generated using Frontside.py**

chg\_val(m.synth\_AA\_k, 0.05)

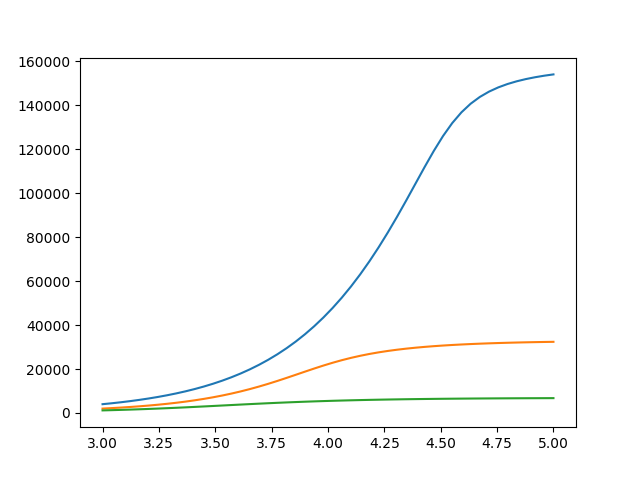
LO\_wrap([3,5],'o\_LXA4')

chg\_val(m.synth\_AA\_k, 0.01)

LO\_wrap([3,5],'o\_LXA4')

chg\_val(m.synth\_AA\_k, 0.002)

LO\_wrap([3,5],'o\_LXA4')



**Figure 12(c): generated using Frontside.py**

chg\_val(m.synth\_AA\_k, 0.05)

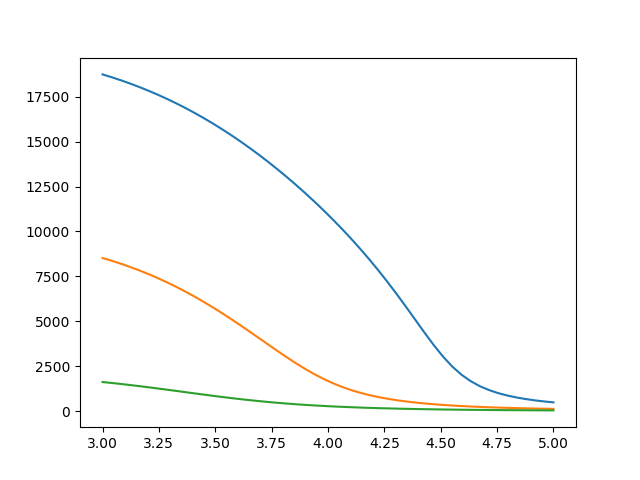
LO\_wrap([3,5],'o\_PKA')

chg\_val(m.synth\_AA\_k, 0.01)

LO\_wrap([3,5],'o\_PKA')

chg\_val(m.synth\_AA\_k, 0.002)

LO\_wrap([3,5],'o\_PKA')



**Figure 12(d): generated using Frontside.py**

chg\_val(m.synth\_AA\_k, 0.05)

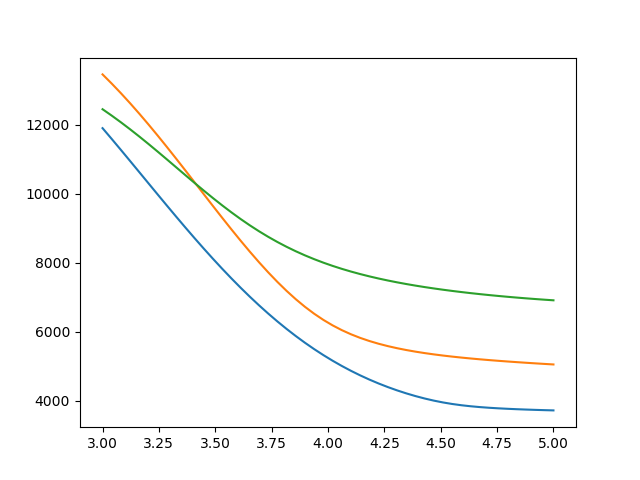
LO\_wrap([3,5],'o\_Syt7')

chg\_val(m.synth\_AA\_k, 0.01)

LO\_wrap([3,5],'o\_Syt7')

chg\_val(m.synth\_AA\_k, 0.002)

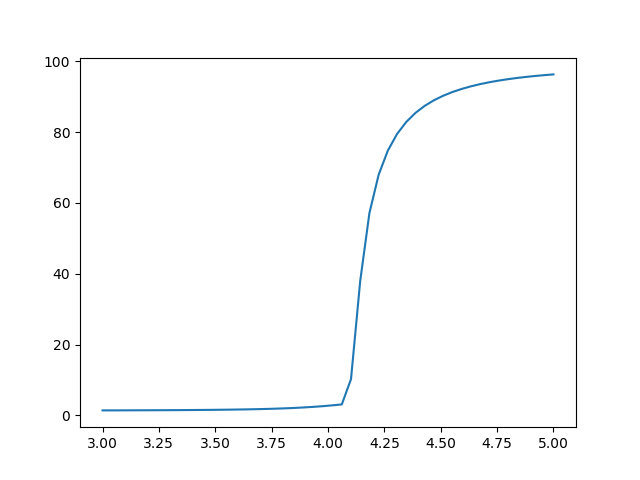
LO\_wrap([3,5],'o\_Syt7')



**Figure 13: generated using FullModel.py**

from run\_FullModelat import \*

Osm\_LO([3,5])



**Figure 14: generated using FullModel.py**

chg\_val(m.Pol\_dBcl2\_p\_kf,1e-8)

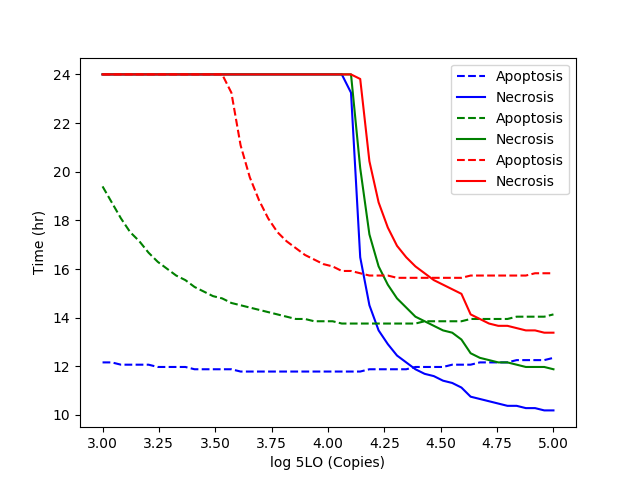
time\_to\_nec([3.0,5.0],'b')

chg\_val(m.Pol\_dBcl2\_p\_kf,3e-8)

time\_to\_nec([3.0,5.0],'g')

chg\_val(m.Pol\_dBcl2\_p\_kf,5e-8)

time\_to\_nec([3.0,5.0],'r')



**Figure 15(a): generated using Frontside.py**

chg\_val(m.LO\_AA\_b\_kf,1e-5)

LO\_wrap([3,5],'o\_PKA')

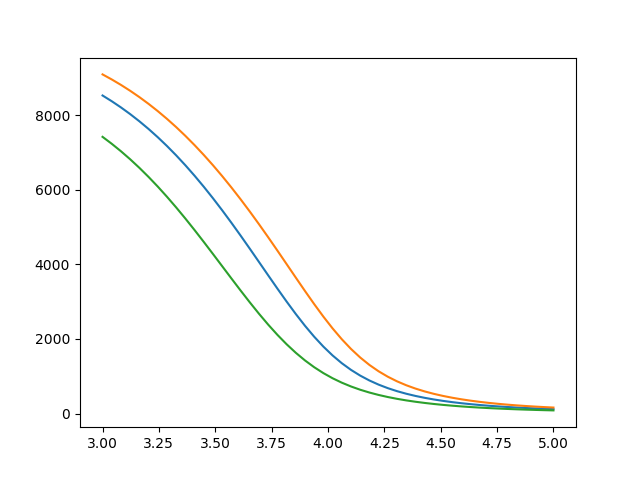
chg\_val(m.synth\_TNF\_k,60)

LO\_wrap([3,5],'o\_PKA')

chg\_val(m.synth\_TNF\_k,0)

chg\_val(m.synth\_TNFR2\_k, 60)

LO\_wrap([3,5],'o\_PKA')



**Figure 15(b): generated using Frontside.py**

chg\_val(m.LO\_AA\_b\_kf,1e-5)

LO\_wrap([3,5],'o\_Syt7')

chg\_val(m.synth\_TNF\_k,60)

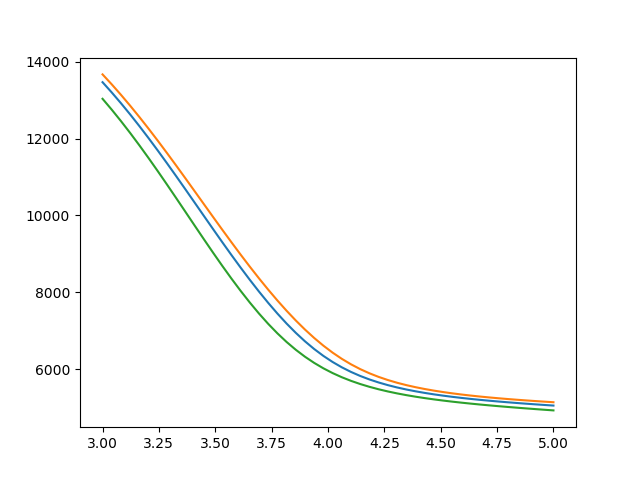
LO\_wrap([3,5],'o\_Syt7')

chg\_val(m.synth\_TNF\_k,0)

LO\_wrap([3,5],'o\_Syt7')

chg\_val(m.synth\_TNFR2\_k, 60)

LO\_wrap([3,5],'o\_Syt7')



**Figure 15(c): generated using FullModel.py**

chg\_val(m.LO\_AA\_b\_kf,1e-5)

time\_to\_nec([3.0,5.0],'b')

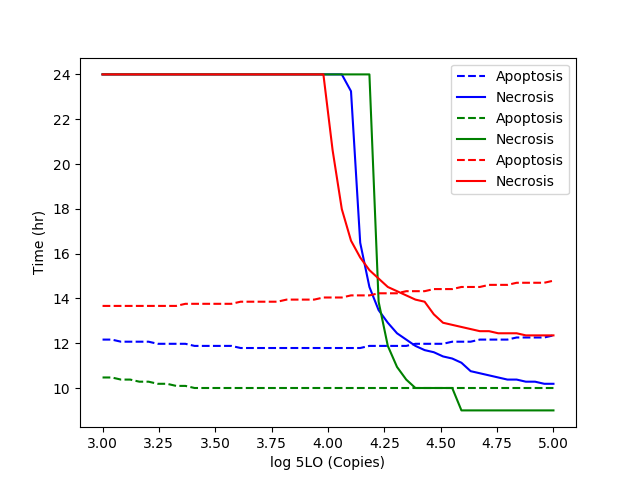
chg\_val(m.synth\_TNF\_k,60)

time\_to\_nec([3.0,5.0],'r')

chg\_val(m.synth\_TNF\_k,0)

chg\_val(m.synth\_TNFR2\_k, 60)

time\_to\_nec([3.0,5.0],'g')

****

**Figure 16(a): generated using Frontside.py**

chg\_val(m.LO\_AA\_b\_kf,1e-5)

chg\_val(m.synth\_p50\_k,45)

LO\_wrap([3,5],'o\_PKA')

chg\_val(m.synth\_p50\_k,75)

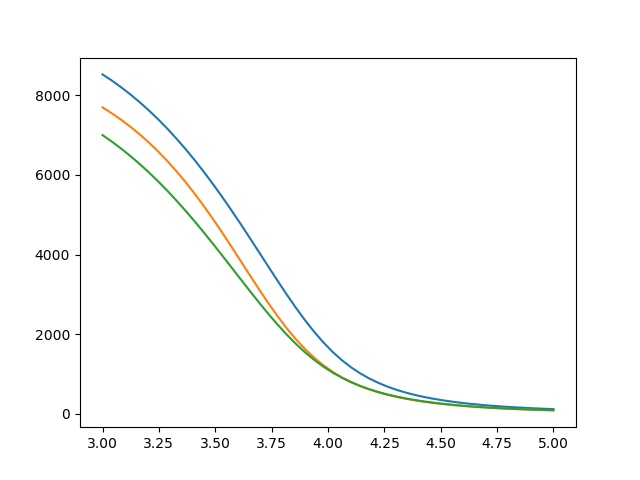
stable\_kB()

LO\_wrap([3,5],'o\_PKA')

chg\_val(m.synth\_p50\_k,30)

stable\_kB()

LO\_wrap([3,5], 'o\_PKA')



**Figure 16(b): generated using Frontside.py**

chg\_val(m.LO\_AA\_b\_kf,1e-5)

chg\_val(m.synth\_p50\_k,45)

LO\_wrap([3,5],'o\_Syt7')

chg\_val(m.synth\_p50\_k,75)

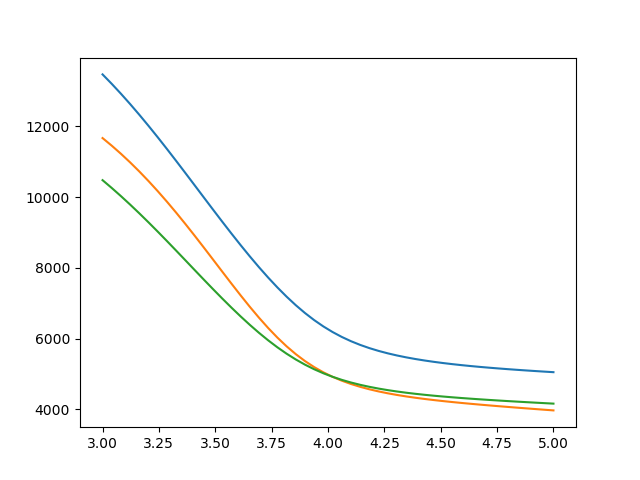
stable\_kB()

LO\_wrap([3,5],'o\_Syt7')

chg\_val(m.synth\_p50\_k,30)

stable\_kB()

LO\_wrap([3,5],'o\_Syt7')



**Figure 16(c): generated using FullModel.py**

chg\_val(m.LO\_AA\_b\_kf,1e-5)

chg\_val(m.synth\_p50\_k,45)

time\_to\_nec([3.0,5.0],'b')

chg\_val(m.synth\_p50\_k,75)

stable\_kB()

time\_to\_nec([3.0,5.0],'g')

chg\_val(m.synth\_p50\_k,30)

stable\_kB()

time\_to\_nec([3.0,5.0],'r')