

```
In [1]: 1 #Import Libraries
        2 import numpy as np
        3 import scipy.optimize as so
        4 import matplotlib.pyplot as plt
        5 import os
```

## Model Definitions

```
In [2]: 1 from modelfn import *
```

```
In [3]: 1 helpnotes
```

```
Out[3]: 'List of Model Functions: \n actkinr() actkin() d2DTOT() DTOT2d() DTOT2AK() DTOT2AKnor
m()\n List of Other Functions:\n gendat() checkpos() solrange() parsepar()\n Some oth
er functions accept a variable fitflag with value of 0,1 or2.\n fitflag=0 allows all p
arameters to vary.\n fitflag=1 sets Kdim to a value of 0.1\n fitflag=2 sets Kdim=0.1 a
nd RAF=0.04'
```

```
In [4]: 1 paramsbase={'f':0.01,'g':100,'KA':10,'rafr':0.4,'RAF':0.04,'Kdim':0.1,'Kd':0.1}
        2 dr=np.sqrt(2)/2 # irrational dr for validation will enhance any mismatches and avoi
        3 d1=dr*paramsbase['Kd']
        4 d1,actkinr(dr,paramsbase),actkin(d1,paramsbase), d2DTOT(d1,paramsbase),DTOT2d(0.218
```

```
Out[4]: (0.07071067811865477,
         0.1762055823754643,
         0.17620558237546433,
         0.07927729536203741,
         0.20579081054605694,
         0.24620257675531507)
```

```
In [5]: 1 paramst0=dict(paramsbase)
        2 def fKAScanFC(fin,ka):
        3     paramst0['f']=fin
        4     paramst0['KA']=ka
        5     return solranger(paramst0)
        6 # how low does rafr need to be to get to 5x FC with KA~0 and f,g at limiting magnit
        7 solrange({'f':10**-6,'g':10000., 'KA':0.0001, 'rafr':0.05, 'RAF':0.005, 'Kdim':0.1, 'Kd'
```

```
Out[5]: (0.09828136178419603, 5.932169015656421, 7024.616784336002)
```

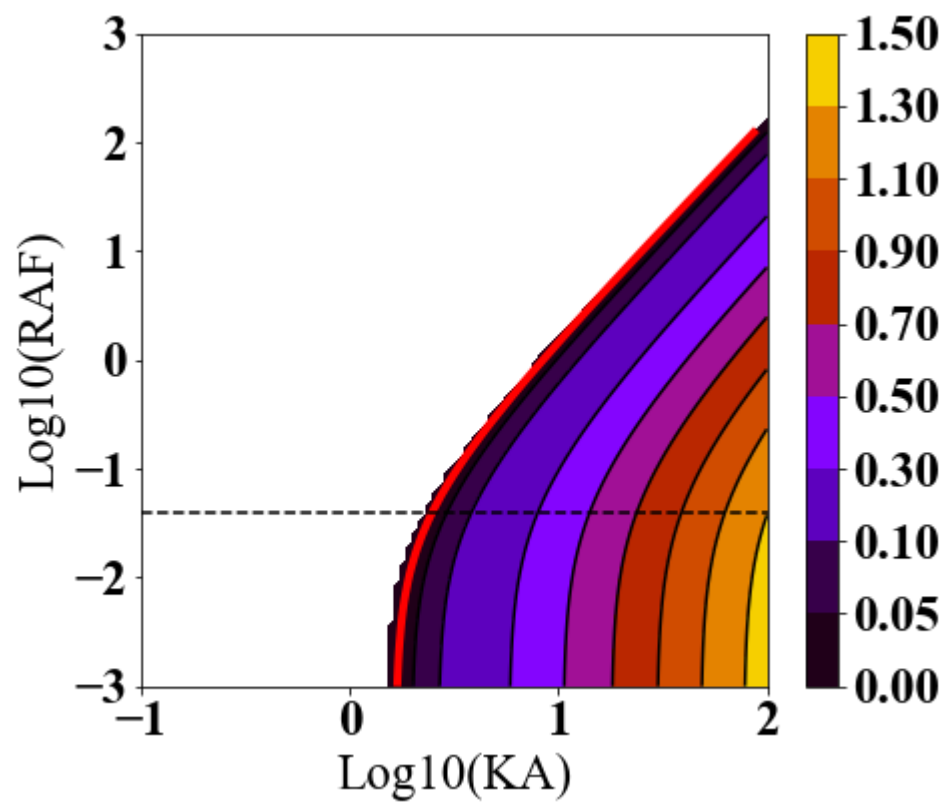
**Figure 2 : Conformal Autoinhibition model**

In [6]:

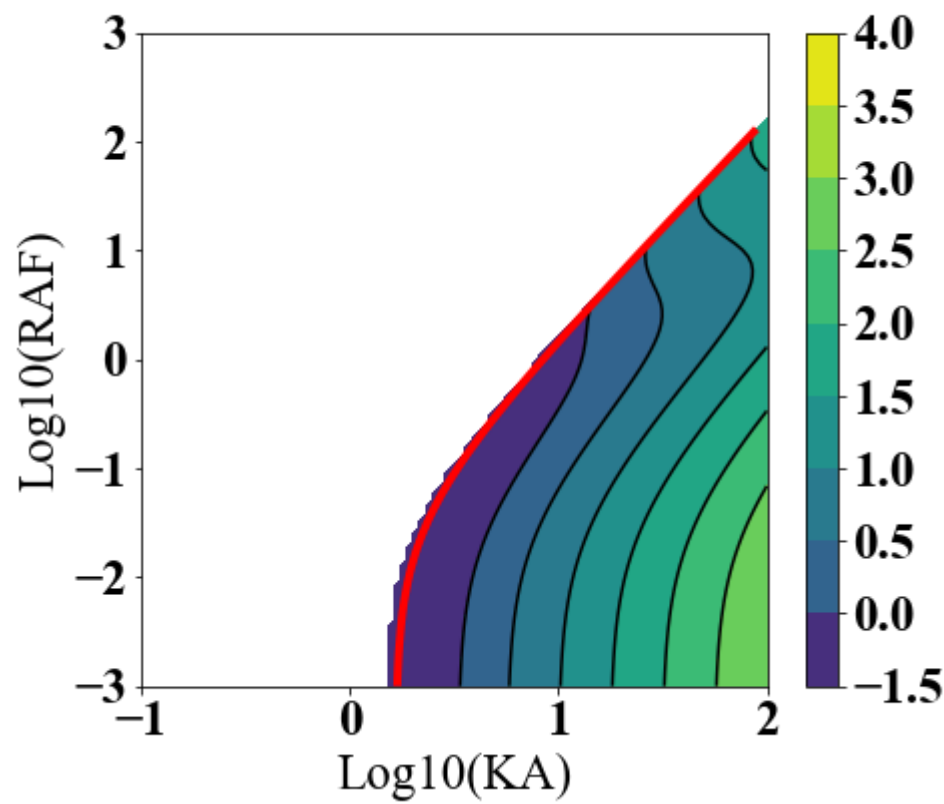
```
1 paramsbase={'KA':10., 'rafr':0.4, 'Kd':0.1, 'Kdim':0.1, 'RAF':0.04, 'f':1.0, 'g':1.0} #un
2
3 def gen_contourpts(xpar_label,xlog10range,ypar_label,ylog10range):
4     npts=10000 # total number of points to plot
5     npts=int(np.sqrt(npts)) # square root of the number of points to put on a squar
6     xlist = np.linspace(xlog10range[0],xlog10range[1],npts)
7     ylist = np.linspace(ylog10range[0],ylog10range[1],npts)
8     X, Y = np.meshgrid(xlist, ylist)
9     X, Y = np.meshgrid(xlist, ylist)
10    def callfnr(x1,y1):
11        params1=dict(paramsbase)
12        params1[ypar_label]=10**y1
13        params1[xpar_label]=10**x1
14        # print(params1)
15        try:
16            resarr=solrange(params1)
17            # print(resarr)
18            if resarr[0]>10**-5:
19                return [np.log10(resarr[1]),np.log10(resarr[2])]
20            else:
21                return [float('nan'),float('nan')]
22        except:
23            return [float('nan'),float('nan')]
24    zfc=[]
25    zr=[]
26    for itr in range(len(X)):
27        zfc=zfc+[[[]]]
28        zr=zr+[[[]]]
29        for jtr in range(len(X[itr])):
30            res=callfnr(X[itr][jtr],Y[itr][jtr])
31            zfc[itr]=zfc[itr]+[res[0]]
32            zr[itr]=zr[itr]+[res[1]]
33    return X,Y,zr,zfc
```

In [7]:

```
1 %%time
2 xpar_label='KA'
3 ypar_label='RAF'
4 xlog10range=[-1,2]
5 ylog10range=[-3,3]
6 X,Y,zr,zfc=gen_contourpts(xpar_label,xlog10range,ypar_label,ylog10range)
7 # npts=20
8 # xlist=np.linspace(-1,0.5,npts)
9 # ylist=np.linspace(-1,0,npts)
10 # Xin,Yin=np.meshgrid(xlist,ylist)
11 # Zin=[[callfmr(Xin[itr][jtr],Yin[itr][jtr]) for jtr in range(len(Xin[itr]))] for i
12 font = {'family' : 'Times New Roman',
13         'weight' : 'bold',
14         'size' : 25}
15 plt.rc('font', **font)
16 plt.figure(figsize=(7,6))
17 colormaptype='gnuplot'
18 fileid="foldchange"
19 levels=[0.0,0.05,0.1]+[i/100 for i in range(30,151,20)]
20 cp0 = plt.contourf(X,Y,zfc,cmap=colormaptype,levels=levels)
21 plt.contour(cp0,colors='k',linewidths=1.5)
22 plt.colorbar(cp0)
23 plt.plot([-1,2],[np.log10(0.04),np.log10(0.04)],'k--')
24 minlevel=0.0278
25 plt.contour(cp0,colors='r',linewidths=4.,levels=[minlevel],linestyles='solid')
26 plt.xlabel('Log10({})'.format(xpar_label))
27 plt.ylabel('Log10({})'.format(ypar_label))
28 # figname="Unified-fgKA-model_"+fileid+"_fvsg_"+str(paramsbaser['rafr'])+"
29 # plt.savefig(figname,dpi=300)
30 # print(figname)
31 plt.show()
32
33
34 plt.figure(figsize=(7,6))
35 colormaptype='viridis'
36 fileid="AR"
37 levels=[-1.5]+[i/10 for i in range(0,45,5)]
38 cp = plt.contourf(X,Y,zr,cmap=colormaptype,levels=levels)
39 plt.contour(cp,colors='k',linewidths=1.5)
40 plt.colorbar(cp)
41 plt.contour(cp0,colors='r',linewidths=4.,levels=[minlevel],linestyles='solid')
42 # plt.plot([-0.4,-0.4],[-5,0], '--k')
43 plt.xlabel('Log10({})'.format(xpar_label))
44 plt.ylabel('Log10({})'.format(ypar_label))
45 figname="CA-model_"+fileid+str(paramsbase)
46 plt.title(figname,size=10,y=1.1)
47 plt.show()
```



CA-model\_AR({'KA': 10.0, 'rafr': 0.4, 'Kd': 0.1, 'Kdim': 0.1, 'RAF': 0.04, 'P': 1.0, 'g': 1.0})



Wall time: 21.8 s

```
In [8]: 1 paramsCA=paramsbase.copy()
2 paramsCA['KA']=10.
3 paramsCA['RAF']=0.04
4 print('maximum fold change in CA model:',round(10**max([col for row in zfc for col
5 paramsCA['KA']=100.
6 '\nMax Fold change at RAF=0.04uM and KA=100 (max): ',round(solrange(paramsCA)[1],2)
```

maximum fold change in CA model: 25.25  
 Max Fold change at RAF=0.04uM and KA=10: 2.37

Out[8]: ('\nMax Fold change at RAF=0.04uM and KA=100 (max): ', 19.71)

### ***Contour plot base model for Kd vs KA and Kdim vs KA***

```
In [9]: 1 bdyglobal
```

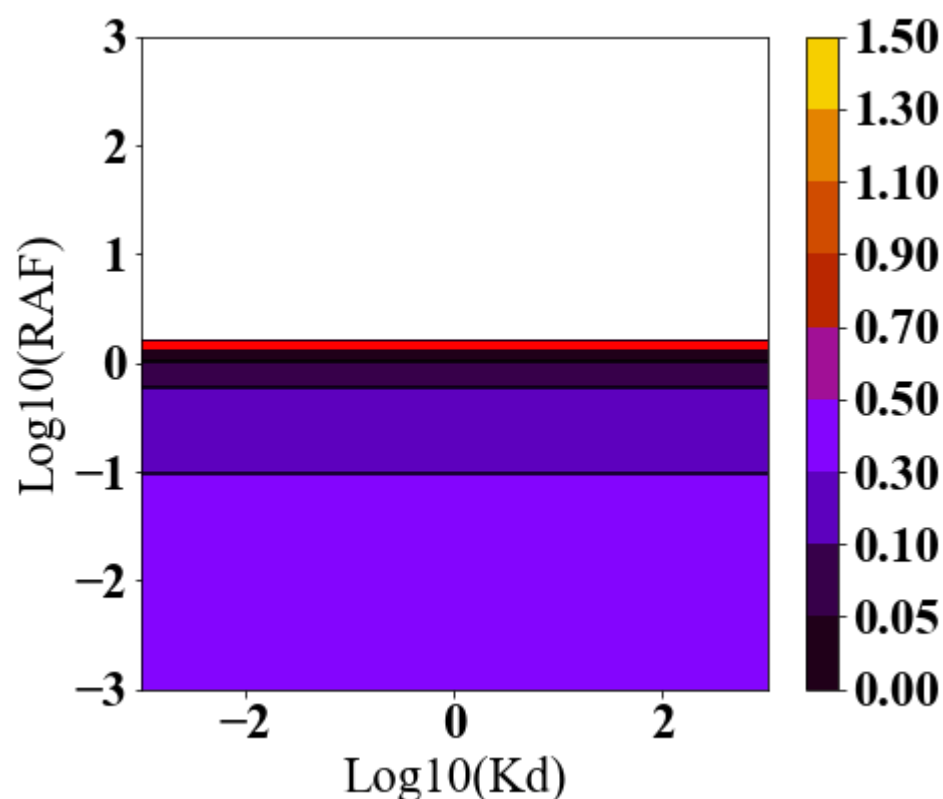
Out[9]: {'f': [1e-05, 100],  
 'g': [1, 10000],  
 'KA': [0.1, 100],  
 'Kd': [0.0001, 10000],  
 'Kdim': [0.0001, 10000],  
 'RAF': [0.0001, 1000]}

```
In [21]: 1 %%time
2 xpar_label='Kd'
3 ypar_label='RAF'
4 xlog10range=[-3,3]
5 ylog10range=[-3,3]
6 X,Y,zr,zfc=gen_contourpts(xpar_label,xlog10range,ypar_label,ylog10range)
```

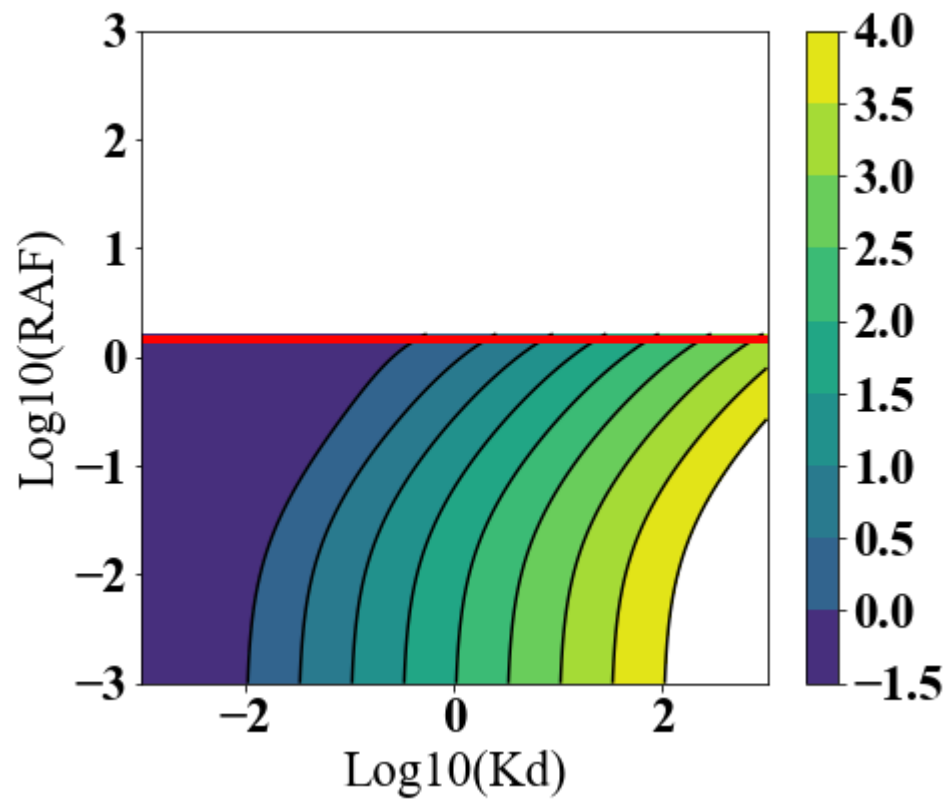
Wall time: 36.4 s

In [22]:

```
1 font = {'family' : 'Times New Roman',
2         'weight' : 'bold',
3         'size'   : 25}
4 plt.rc('font', **font)
5 plt.figure(figsize=(7,6))
6 colormaptype='gnuplot'
7 fileid="foldchange"
8 levels=[0.0,0.05,0.1]+[i/100 for i in range(30,151,20)]
9 cp0 = plt.contourf(X,Y,zfc,cmap=colormaptype,levels=levels)
10 plt.contour(cp0,colors='k',linewidths=1.5)
11 plt.colorbar(cp0)
12 # plt.plot([-1,2],[np.log10(0.04),np.log10(0.04)],'k--')
13 minlevel=0.0278
14 plt.contour(cp0,colors='r',linewidths=4.,levels=[minlevel],linestyles='solid')
15 plt.xlabel('Log10({})'.format(xpar_label))
16 plt.ylabel('Log10({})'.format(ypar_label))
17 # figname="Unified-fgKA-model_"+fileid+"_fvsg_"+"RAFrel"+str(paramsbaser['rafr'])+"
18 # plt.savefig(figname,dpi=300)
19 # print(figname)
20 plt.show()
21
22
23 plt.figure(figsize=(7,6))
24 colormaptype='viridis'
25 fileid="AR"
26 levels=[-1.5]+[i/10 for i in range(0,45,5)]
27 cp = plt.contourf(X,Y,zr,cmap=colormaptype,levels=levels)
28 plt.contour(cp,colors='k',linewidths=1.5)
29 plt.colorbar(cp)
30 plt.contour(cp0,colors='r',linewidths=4.,levels=[minlevel],linestyles='solid')
31 # plt.plot([-0.4,-0.4],[-5,0], '--k')
32 plt.xlabel('Log10({})'.format(xpar_label))
33 plt.ylabel('Log10({})'.format(ypar_label))
34 figname="CA-model_"+fileid+str(paramsbase)
35 plt.title(figname,size=10,y=1.1)
36 plt.show()
```



CA-model\_AR{'KA': 10.0, 'rafr': 0.4, 'Kd': 0.1, 'Kdim': 0.1, 'RAF': 0.04, 't': 1.0, 'g': 1.0}



In [17]:

```
1 %%time
2 xpar_label='Kdim'
3 ypar_label='RAF'
4 xlog10range=[-3,3]
5 ylog10range=[-3,3]
6 X,Y,zr,zfc=gen_contourpts(xpar_label,xlog10range,ypar_label,ylog10range)
```

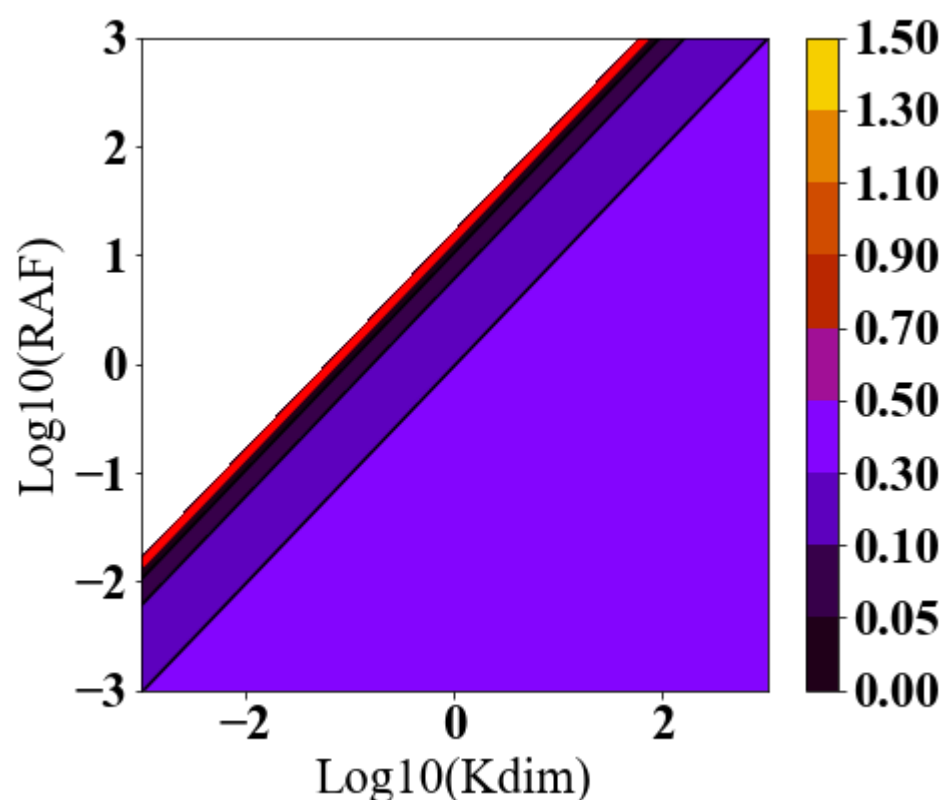
Wall time: 44.3 s

In [20]:

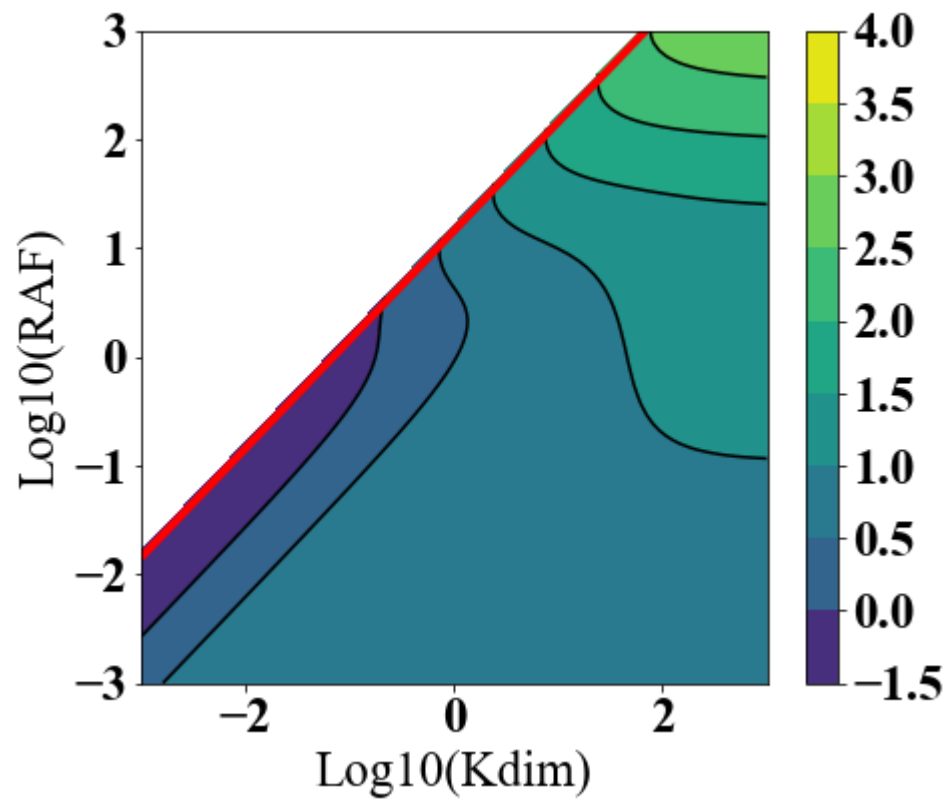
```

1 font = {'family' : 'Times New Roman',
2         'weight' : 'bold',
3         'size'   : 25}
4 plt.rc('font', **font)
5 plt.figure(figsize=(7,6))
6 colormaptype='gnuplot'
7 fileid="foldchange"
8 levels=[0.0,0.05,0.1]+[i/100 for i in range(30,151,20)]
9 cp0 = plt.contourf(X,Y,zfc,cmap=colormaptype,levels=levels)
10 plt.contour(cp0,colors='k',linewidths=1.5)
11 plt.colorbar(cp0)
12 # plt.plot([-1,2],[np.log10(0.04),np.log10(0.04)],'k--')
13 minlevel=0.0278
14 plt.contour(cp0,colors='r',linewidths=4.,levels=[minlevel],linestyles='solid')
15 plt.xlabel('Log10({})'.format(xpar_label))
16 plt.ylabel('Log10({})'.format(ypar_label))
17 # figname="Unified-fgKA-model_"+fileid+"_fvsg_"+str(RAFrel)+"RAFrel"+str(paramsbaser['rafr'])+"
18 # plt.savefig(figname,dpi=300)
19 # print(figname)
20 plt.show()
21
22
23 plt.figure(figsize=(7,6))
24 colormaptype='viridis'
25 fileid="AR"
26 levels=[-1.5]+[i/10 for i in range(0,45,5)]
27 cp = plt.contourf(X,Y,zr,cmap=colormaptype,levels=levels)
28 plt.contour(cp,colors='k',linewidths=1.5)
29 plt.colorbar(cp)
30 plt.contour(cp0,colors='r',linewidths=4.,levels=[minlevel],linestyles='solid')
31 # plt.plot([-0.4,-0.4],[-5,0], '--k')
32 plt.xlabel('Log10({})'.format(xpar_label))
33 plt.ylabel('Log10({})'.format(ypar_label))
34 figname="CA-model_"+fileid+str(paramsbase)
35 plt.title(figname,size=10,y=1.1)
36 plt.show()

```







**Impact of RAF concentration on model predictions**

**Supplementary Figure 2 A**

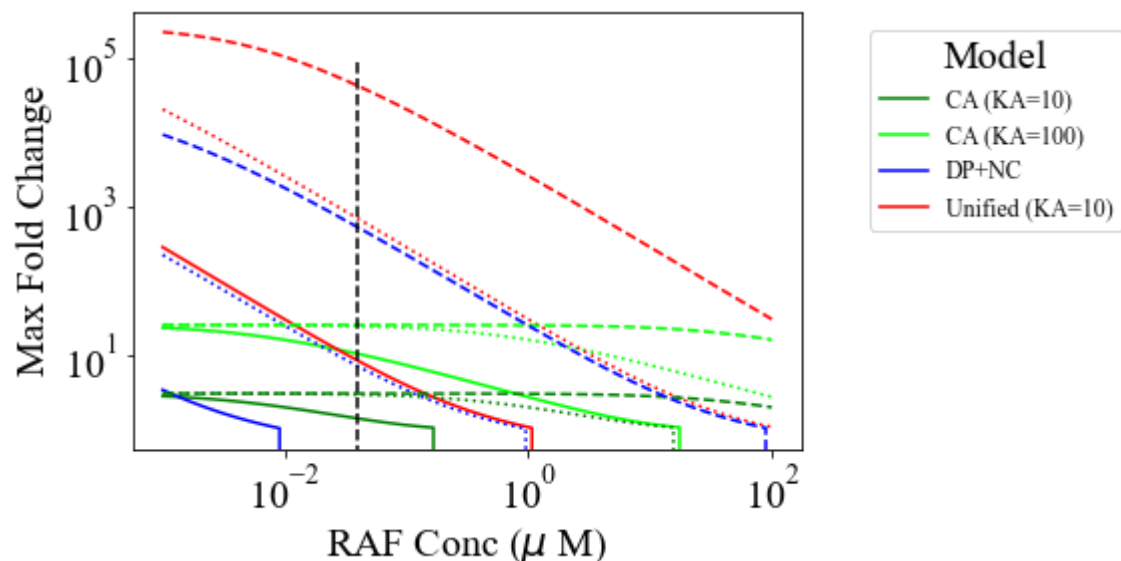
In [6]:

```
1 # make an example plot in the fg model. Note that setting Kdim to 0.1 gives much Lo
2
3 font = {'family' : 'Times New Roman',
4         'weight' : 'normal',
5         'size'   : 20}
6 plt.rc('font', **font)
7 xdat=list(10**np.linspace(-3,2,100)) # RAF concentration range
8
9 paramsplt0={'f':10**-5,'KA':10.,'g':100.,'rafr':0.4,'Kd':0.1,'Kdim':0.1} #units: mi
10 kdimvals=[0.01,1,100]
11 ltlist=['solid','dotted','dashed','dashdot',(0,(3,5,1,5,1,5))]
12 for it1 in range(len(kdimvals)):
13     paramsplt0['Kdim']=kdimvals[it1]
14     # plt.title(str(paramsplt0),size=10)
15     paramsplt0['g']=1.
16     paramsplt0['f']=1.
17     paramsplt0['KA']=10.
18     ydat= []
19     for raf in xdat:
20         paramsplt0['RAF']=raf
21         try:
22             solfc=solrange(paramsplt0)[1]
23         except:
24             solfc=1.
25         ydat=ydat+[solfc]
26     plt.plot(xdat,ydat,label='CA (KA=10)',color='g',linestyle=ltlist[it1])
27
28     paramsplt0['g']=1.
29     paramsplt0['f']=1.
30     paramsplt0['KA']=100.
31     ydat= []
32     for raf in xdat:
33         paramsplt0['RAF']=raf
34         try:
35             solfc=solrange(paramsplt0)[1]
36         except:
37             solfc=1.
38         ydat=ydat+[solfc]
39     plt.plot(xdat,ydat,label='CA (KA=100)',color='lime',linestyle=ltlist[it1])
40
41     paramsplt0['g']=100.
42     paramsplt0['f']=10**-5.
43     paramsplt0['KA']=0.0001
44     ydat= []
45     for raf in xdat:
46         paramsplt0['RAF']=raf
47         try:
48             solfc=solrange(paramsplt0)[1]
49         except:
50             solfc=1.
51         ydat=ydat+[solfc]
52     plt.plot(xdat,ydat,label='DP+NC',color='b',linestyle=ltlist[it1])
53
54     paramsplt0['g']=100.
55     paramsplt0['f']=10**-5.
56     paramsplt0['KA']=10.
57     ydat= []
58     for raf in xdat:
59         paramsplt0['RAF']=raf
60         try:
61             solfc=solrange(paramsplt0)[1]
62         except:
```

```

63         solfc=1.
64         ydat=ydat+[solfc]
65         plt.plot(xdat,ydat,label='Unified (KA=10)',color='r',linestyle=ltlist[it1])
66
67     if it1==0:
68         plt.legend(loc=(1.1,0.5),title='Model',fontsize=12)
69     plt.plot([0.04,0.04],[0.0,10**5],'k--') # typical concentration of RAF
70     plt.xscale('log')
71     plt.yscale('log')
72     plt.xlabel('RAF Conc ( $\mu$  M)')
73     plt.ylabel('Max Fold Change')
74     plt.show()
75     #-- Begin: custom legend - uncomment to redraw legend for the Kdim values.
76     # from matplotlib.lines import Line2D
77
78     # Legend_elements = [Line2D([0], [0], color='k', lw=4, label=kdimvals[i],linestyle
79     # plt.legend(handles=Legend_elements,loc='upper center',bbox_to_anchor=(0.5,1.4),ti
80     # #-- End: custom legend
81     # plt.show()

```



## Fold-chane in all models are a function of RAF/Kdim ratio and not RAF or Kdim separately

- As expected , all curves in a given model, with different Kdim, collapse into one curve with RAF/Kdim on the x-axis.

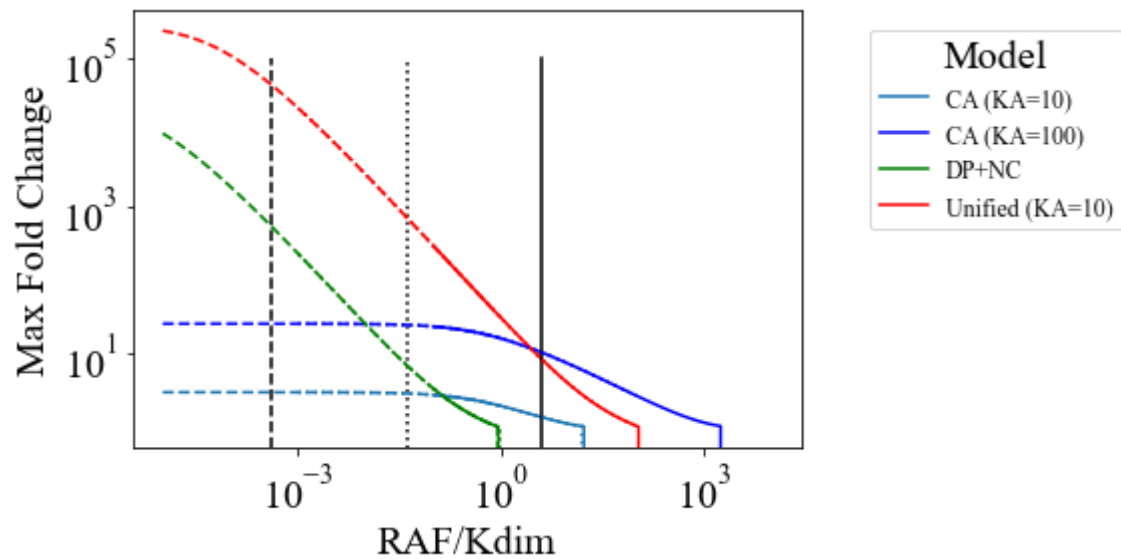
In [7]:

```
1 # make an example plot in the fg model. Note that setting Kdim to 0.1 gives much lo
2 font = {'family' : 'Times New Roman',
3         'weight' : 'normal',
4         'size'   : 20}
5 plt.rc('font', **font)
6 xdat=list(10**np.linspace(-3,2,100)) # RAF concentration range
7
8 paramsplt0={'f':10**-5,'KA':10.,'g':100.,'rafr':0.4,'Kd':0.1,'Kdim':0.1} #units: mi
9 kdimvals=[0.01,1,100]
10 ltlist=['solid','dotted','dashed','dashdot',(0,(3,5,1,5,1,5))]
11 for it1 in range(len(kdimvals)):
12     paramsplt0['Kdim']=kdimvals[it1]
13     # plt.title(str(paramsplt0),size=10)
14     paramsplt0['g']=1.
15     paramsplt0['f']=1.
16     paramsplt0['KA']=10.
17     ydat= []
18     for raf in xdat:
19         paramsplt0['RAF']=raf
20         try:
21             solfc=solrange(paramsplt0)[1]
22         except:
23             solfc=1.
24         ydat=ydat+[solfc]
25     xdat1=[xdati/paramsplt0['Kdim'] for xdati in xdat]
26     plt.plot(xdat1,ydat,label='CA (KA=10)',color='tab:blue',linestyle=ltlist[it1])
27
28     paramsplt0['g']=1.
29     paramsplt0['f']=1.
30     paramsplt0['KA']=100.
31     ydat= []
32     for raf in xdat:
33         paramsplt0['RAF']=raf
34         try:
35             solfc=solrange(paramsplt0)[1]
36         except:
37             solfc=1.
38         ydat=ydat+[solfc]
39     plt.plot(xdat1,ydat,label='CA (KA=100)',color='b',linestyle=ltlist[it1])
40
41     paramsplt0['g']=100.
42     paramsplt0['f']=10**-5.
43     paramsplt0['KA']=0.0001
44     ydat= []
45     for raf in xdat:
46         paramsplt0['RAF']=raf
47         try:
48             solfc=solrange(paramsplt0)[1]
49         except:
50             solfc=1.
51         ydat=ydat+[solfc]
52     plt.plot(xdat1,ydat,label='DP+NC',color='g',linestyle=ltlist[it1])
53
54     paramsplt0['g']=100.
55     paramsplt0['f']=10**-5.
56     paramsplt0['KA']=10.
57     ydat= []
58     for raf in xdat:
59         paramsplt0['RAF']=raf
60         try:
61             solfc=solrange(paramsplt0)[1]
62         except:
```

```

63         solfc=1.
64         ydat=ydat+[solfc]
65         plt.plot(xdat1,ydat,label='Unified (KA=10)',color='r',linestyle=ltlist[it1])
66
67     if it1==0:
68         plt.legend(loc=(1.1,0.5),title='Model',fontsize=12)
69         plt.plot([0.04/paramsplt0['Kdim'],0.04/paramsplt0['Kdim']], [0.0,10**5],color='k',
70 plt.xscale('log')
71 plt.yscale('log')
72 plt.xlabel('RAF/Kdim')
73 plt.ylabel('Max Fold Change')
74 plt.show()
75 #-- Begin: custom legend
76 # from matplotlib.lines import Line2D
77
78 # Legend_elements = [Line2D([0], [0], color='k', lw=4, label=kdimvals[i],linestyle=ltlist[it1]) for i in range(len(kdimvals))]
79 # plt.legend(handles=Legend_elements,loc='upper center',bbox_to_anchor=(0.5,1.4),title='Model')
80 # #-- End: custom legend
81 # plt.show()

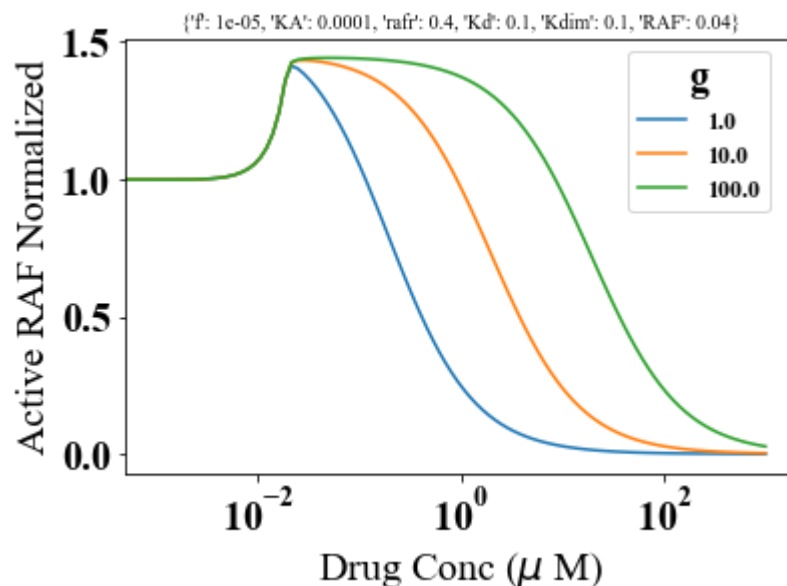
```



**Supplementary Figure 2C**

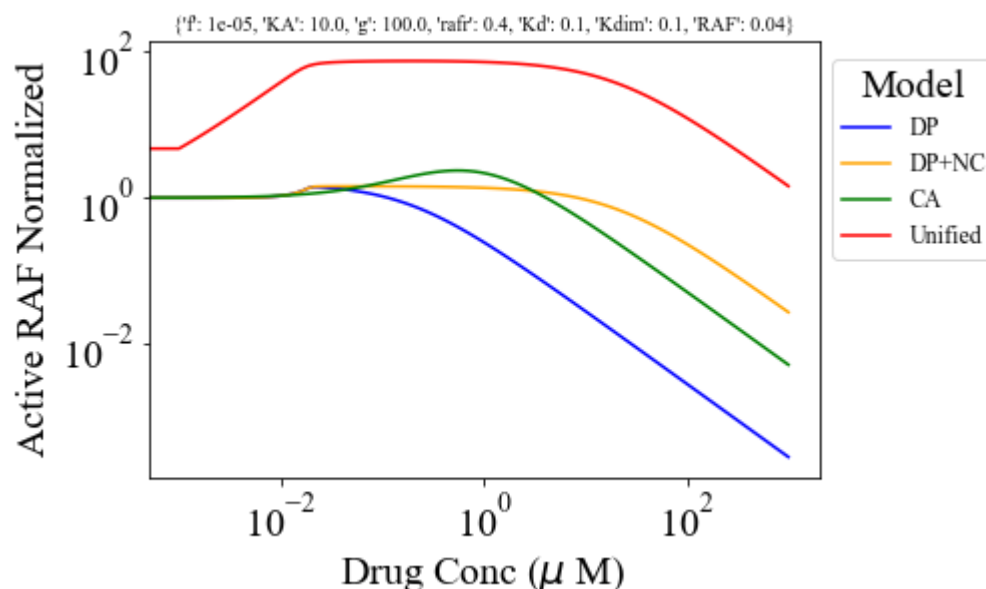
In [12]:

```
1 # make an example plot in the fg model. Note that setting Kdim to 0.1 gives much Lo
2
3 font = {'family' : 'Times New Roman',
4         'weight' : 'bold',
5         'size'    : 20}
6 plt.rc('font', **font)
7 xdat=[0.]+list(10**np.linspace(-3,3,100))
8
9 paramsplt0={'f':10**-5, 'KA':0.0001, 'rafr':0.4, 'Kd':0.1, 'Kdim':0.1, 'RAF':0.04} #unit
10 plt.title(str(paramsplt0),size=10)
11
12 paramsplt0['g']=1.
13 ydat= [DTOT2AK(xval,paramsplt0)/DTOT2AK(0,paramsplt0) for xval in xdat]
14 plt.plot(xdat,ydat,label=str(paramsplt0['g']))
15
16 paramsplt0['g']=10.
17 ydat= [DTOT2AK(xval,paramsplt0)/DTOT2AK(0,paramsplt0) for xval in xdat]
18 plt.plot(xdat,ydat,label=str(paramsplt0['g']))
19
20 paramsplt0['g']=100.
21 ydat= [DTOT2AK(xval,paramsplt0)/DTOT2AK(0,paramsplt0) for xval in xdat]
22 plt.plot(xdat,ydat,label=str(paramsplt0['g']))
23
24 plt.legend(loc=(0.75,0.6),title='g',fontsize=12)
25 plt.xscale('log')
26 plt.xlabel('Drug Conc ( $\mu$  M)')
27 plt.ylabel('Active RAF Normalized')
28
29 plt.show()
```



**Additional representative figure showing dose response curves with different mechanisms**

```
In [8]: 1 # make an example plot in the fg model. Note that setting Kdim to 0.1 gives much Lo
2
3 font = {'family' : 'Times New Roman',
4         'weight' : 'normal',
5         'size' : 20}
6 plt.rc('font', **font)
7 xdat=[0.]+list(10**np.linspace(-3,3,100))
8
9 paramsplt0={'f':10**-5,'KA':10.,'g':100.,'rafr':0.4,'Kd':0.1,'Kdim':0.1,'RAF':0.04}
10 plt.title(str(paramsplt0),size=10)
11
12 paramsplt0['g']=1.
13 paramsplt0['KA']=0.00001
14 ydat= [DTOT2AK(xval,paramsplt0)/DTOT2AK(0,paramsplt0) for xval in xdat]
15 plt.plot(xdat,ydat,label='DP',color='b')
16
17 paramsplt0['g']=100.
18 ydat= [DTOT2AK(xval,paramsplt0)/DTOT2AK(0,paramsplt0) for xval in xdat]
19 plt.plot(xdat,ydat,label='DP+NC',color='orange')
20
21 paramsplt0['g']=1.
22 paramsplt0['f']=1.
23 paramsplt0['KA']=10.
24 ydat= [DTOT2AK(xval,paramsplt0)/DTOT2AK(0,paramsplt0) for xval in xdat]
25 plt.plot(xdat,ydat,label='CA',color='g')
26
27 paramsplt0['g']=100.
28 paramsplt0['f']=1.*10**-5
29 paramsplt0['KA']=10.
30 ydat= [DTOT2AK(xval,paramsplt0)/DTOT2AK(0,paramsplt0) for xval in xdat]
31 plt.plot(xdat,ydat,label='Unified',color='r')
32
33 plt.legend(loc=(1.02,0.5),title='Model',fontsize=12)
34 plt.xscale('log')
35 plt.yscale('log')
36 plt.xlabel('Drug Conc ( $\mu$  M)')
37 plt.ylabel('Active RAF Normalized')
38
39 plt.show()
```



**Unified model**

## Additional Contour plots in Unified Model

In [20]:

```
1 %%time
2 paramsbase={'KA':10., 'rafr':0.4, 'Kd':0.1, 'Kdim':0.1, 'RAF':0.04, 'f':1.0, 'g':1.0} #un
3 # paramsbaser={'KA':0.0001, 'rafr':0.04, 'Kd':0.1}
4 npts=10000 # total number of points to plot
5 npts=int(np.sqrt(npts)) # square root of the number of points to put on a square gr
6 xlist = np.linspace(-1.0, 6.0,npts)
7 ylist = np.linspace(-5.0, 1.0,npts)
8 X, Y = np.meshgrid(xlist, ylist)
9 X, Y = np.meshgrid(xlist, ylist)
10 def callfnr(x1,y1):
11     params1=dict(paramsbase)
12     params1['f']=10**y1
13     params1['g']=10**x1
14     # print(params1)
15     try:
16         resarr=solrange(params1)
17         # print(resarr)
18         if resarr[0]>10**-5:
19             return [np.log10(resarr[1]),np.log10(resarr[2])]
20         else:
21             return [float('nan'),float('nan')]
22     except:
23         return [float('nan'),float('nan')]
24 zfc=[]
25 zr=[]
26 for itr in range(len(X)):
27     zfc=zfc+[[[]]]
28     zr=zr+[[[]]]
29     for jtr in range(len(X[itr])):
30         res=callfnr(X[itr][jtr],Y[itr][jtr])
31         zfc[itr]=zfc[itr]+[res[0]]
32         zr[itr]=zr[itr]+[res[1]]
33 # npts=20
34 # xlist=np.linspace(-1,0.5,npts)
35 # ylist=np.linspace(-1,0,npts)
36 # Xin,Yin=np.meshgrid(xlist,ylist)
37 # Zin=[[callfnr(Xin[itr][jtr],Yin[itr][jtr]) for jtr in range(len(Xin[itr]))] for i
```

Wall time: 26.3 s

In [21]:

```
1 'maximum fold change in Unified model:',round(10**max([col for row in zfc for col i
```

Out[21]: ('maximum fold change in Unified model:', 75.36)



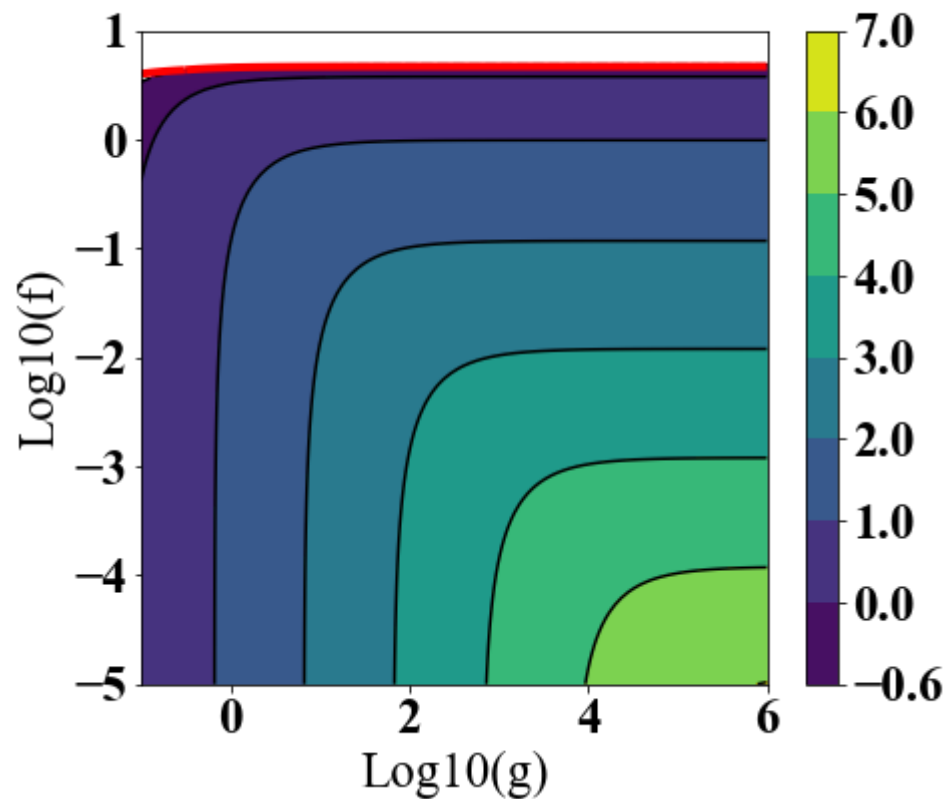
In [22]:

```
1 font = {'family' : 'Times New Roman',
2         'weight' : 'bold',
3         'size'   : 25}
4 plt.rc('font', **font)
5 plt.figure(figsize=(7,6))
6 colormaptype='gnuplot'
7 fileid="FC"
8 levels=[0]+[0.05,0.1]+[i/100 for i in range(20,201,20)]
9 cp = plt.contourf(X,Y,zfc,cmap=colormaptype,levels=levels)
10 plt.contour(cp,colors='k',linewidths=1.5)
11 plt.colorbar(cp)
12 plt.contour(cp,colors='r',linewidths=4.,levels=[0.01],linestyles='solid')
13 plt.xlabel('Log10(g)')
14 plt.ylabel('Log10(f)')
15 figname="Unified-fgKA-model_"+fileid+str(paramsbase)
16 plt.title(figname,size=10,y=1.1)
17 plt.close() # replace with plt.show() to view the PA fold change plot
```

In [23]:

```
1 font = {'family' : 'Times New Roman',
2         'weight' : 'bold',
3         'size'   : 25}
4 plt.rc('font', **font)
5 plt.figure(figsize=(7,6))
6 colormaptype='viridis'
7 fileid="AR"
8 minlevel=-.6
9 levels=[minlevel]+[i/10 for i in range(0,75,10)]
10 cp0 = plt.contourf(X,Y,zr,cmap=colormaptype,levels=levels)
11 plt.contour(cp0,colors='k',linewidths=1.5)
12 plt.colorbar(cp0)
13 plt.contour(cp,colors='r',linewidths=4.,levels=[0.01],linestyles='solid')
14 # plt.plot([-0.4,-0.4],[-5,0], '--k')
15 plt.xlabel('Log10(g)')
16 plt.ylabel('Log10(f)')
17 figname="Unified-model_"+fileid+str(paramsbase)
18 plt.title(figname,size=10,y=1.1)
19 plt.show()
```

Unified-model\_AR{'KA': 10.0, 'rafr': 0.4, 'Kd': 0.1, 'Kdim': 0.1, 'RAF': 0.04, 'P': 1.0, 'g': 1.0}



KA vs f

In [17]:

```
1 %%time
2 paramsbase={'KA':10., 'rafr':0.4, 'Kd':0.1, 'Kdim':0.1, 'RAF':0.04, 'f':1.0, 'g':10.0} #u
3 # paramsbaser={'KA':0.0001, 'rafr':0.04, 'Kd':0.1}
4 npts=10000 # total number of points to plot
5 npts=int(np.sqrt(npts)) # square root of the number of points to put on a square gr
6 xlist = np.linspace(-2.0, 1.0,npts)
7 ylist = np.linspace(-5.0, 1.0,npts)
8 X, Y = np.meshgrid(xlist, ylist)
9 X, Y = np.meshgrid(xlist, ylist)
10 def callfnr(x1,y1):
11     params1=dict(paramsbase)
12     params1['f']=10**y1
13     params1['KA']=10**x1
14     # print(params1)
15     try:
16         resarr=solrange(params1)
17         # print(resarr)
18         if resarr[0]>10**-5:
19             return [np.log10(resarr[1]),np.log10(resarr[2])]
20         else:
21             return [float('nan'),float('nan')]
22     except:
23         return [float('nan'),float('nan')]
24 zfc=[]
25 zr=[]
26 for itr in range(len(X)):
27     zfc=zfc+[[[]]]
28     zr=zr+[[[]]]
29     for jtr in range(len(X[itr])):
30         res=callfnr(X[itr][jtr],Y[itr][jtr])
31         zfc[itr]=zfc[itr]+[res[0]]
32         zr[itr]=zr[itr]+[res[1]]
33 # npts=20
34 # xlist=np.linspace(-1,0.5,npts)
35 # ylist=np.linspace(-1,0,npts)
36 # Xin,Yin=np.meshgrid(xlist,ylist)
37 # Zin=[[callfnr(Xin[itr][jtr],Yin[itr][jtr]) for jtr in range(len(Xin[itr]))] for i
```

Wall time: 24.5 s

In [18]:

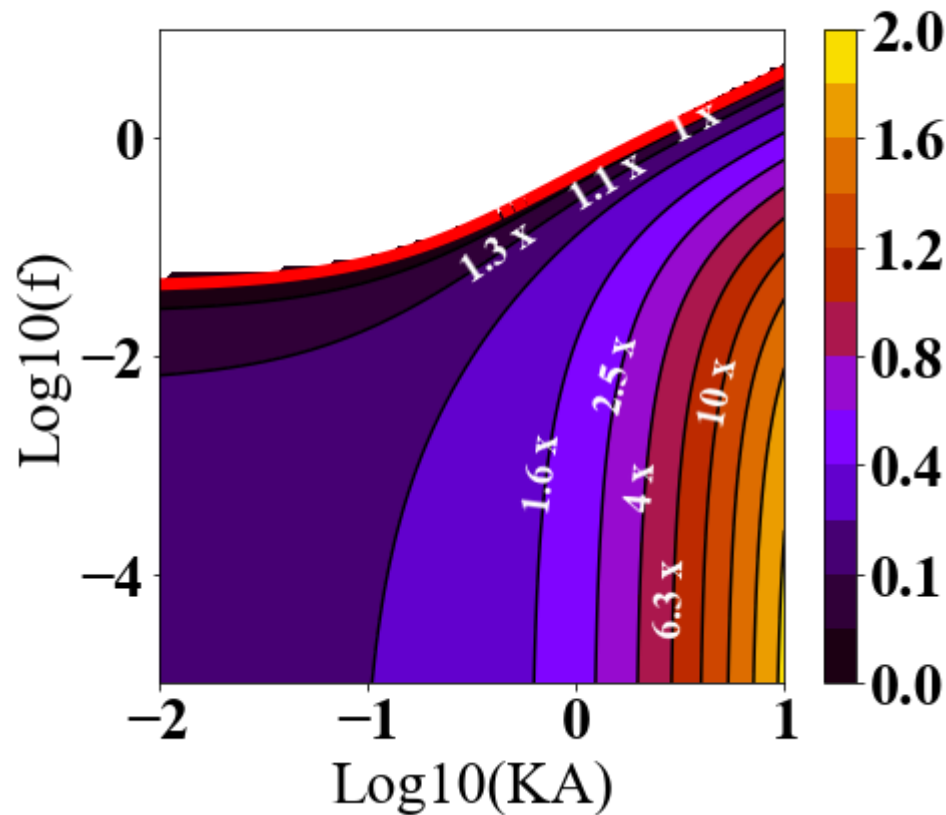
```
1 'maximum fold change in Unified model:',round(10**max([col for row in zfc for col i
```

Out[18]: ('maximum fold change in Unified model:', 71.85)

In [19]:

```
1 font = {'family' : 'Times New Roman',
2         'weight' : 'bold',
3         'size'    : 30}
4 plt.rc('font', **font)
5
6 def fmt(x):
7     x=10**x
8     s = f"{x:.1f}"
9     if s.endswith("0"):
10         s = f"{x:.0f}"
11     return rf"{s} x" if plt.rcParams["text.usetex"] else f"{s} x"
12
13 plt.figure(figsize=(7,6))
14 colormaptype='gnuplot'
15 fileid="FC"
16 levels=[0,0.05,0.1]+[i/100 for i in range(20,201,20)]
17 zfc1=[[10**icol for icol in irow] for irow in zfc]
18 cp = plt.contourf(X,Y,zfc,cmap=colormaptype,levels=levels)
19 plt.contour(cp,colors='k',linewidths=1.5)
20 plt.colorbar(cp)
21 plt.clabel(cp,levels=[0,0.05,0.1,0.2,0.4,0.6,0.8,1.],inline=4,fmt=fmt, fontsize=20,
22 plt.contour(cp,colors='r',linewidths=6.,levels=[0.0245],linestyles='solid')
23 plt.xlabel('Log10(KA)')
24 plt.ylabel('Log10(f)')
25 filename="Unified-fgKA-model_"+fileid+str(paramsbase)
26 plt.title(filename,size=10,y=1.1)
27 plt.show()
```

Unified-fgKA-model\_FC{'KA': 10.0, 'rafr': 0.4, 'Kd': 0.1, 'Kdim': 0.1, 'RAF': 0.04, 'f': 1.0, 'g': 10.0}



```

1 font = {'family' : 'Times New Roman',
2         'weight' : 'bold',
3         'size'   : 25}
4 plt.rc('font', **font)
5 plt.figure(figsize=(7,6))
6 colormaptype='viridis'
7 fileid="AR"
8 minlevel=-.6
9 levels=[minlevel]+[i/10 for i in range(0,75,10)]
10 cp0 = plt.contourf(X,Y,zr,cmap=colormaptype)
11 plt.contour(cp0,colors='k',linewidths=1.5)
12 plt.colorbar(cp0)
13 plt.contour(cp,colors='r',linewidths=6.,levels=[0.0245],linestyles='solid')
14 # plt.plot([-0.4,-0.4],[-5,0], '--k')
15 plt.xlabel('Log10(KA)')
16 plt.ylabel('Log10(f)')
17 figname="Unified-model_"+fileid+str(paramsbase)
18 plt.title(figname,size=10,y=1.1)
19 plt.show()

```