# Problem Set #6

## 2013-10963 Seha Lee

In [22]:
```
import numpy as np
import matplotlib.pyplot as plt
import scipy
```

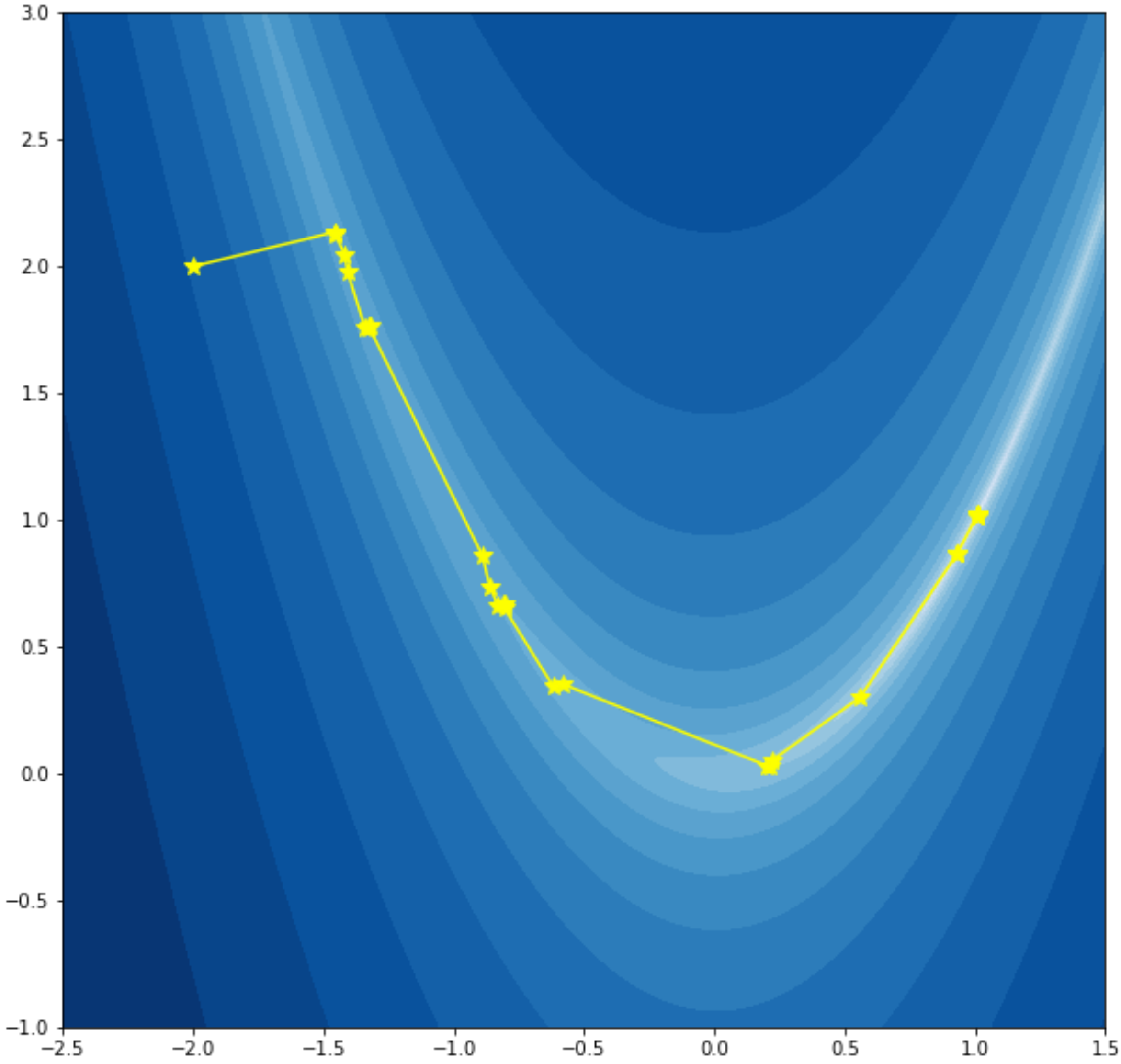
In [51]:
```
#1
def golden2Dm(x,dr,TOL):
    R,err,loop=0.61803399,10.,-1
    a= x-dr/np.sqrt(sum(dr**2))
    b= x+dr/np.sqrt(sum(dr**2))

    while(err > TOL):
        loop += 1
        x1= b-R*(b-a)
        x2= a+R*(b-a)
        f1=func(x1)
        f2=func(x2)
        if(f2>f1): b=x2
        else: a=x1
        err=sum((a-b)**2)
        err=np.sqrt(err)
    xmin=a
    return xmin,err,loop
```

In [123]:
```
def func(x):
    return 100*(x[1]-x[0]**2)**2 + (1-x[0])**2

def deriv(x):
    f1=-400*x[0]*(x[1]-x[0]**2)-2*(1-x[0])
    f2=200*(x[1]-x[0]**2)
    return np.array([f1,f2])

err,TOL=1.,1.e-5
x=np.array([-2.,2.])
xt,yt=[],[]
xt,yt=np.append(xt,x[0]),np.append(yt,x[1])

loop=0
while err>TOL:
    loop+=1
    if loop==1:
        g_now=-deriv(x)
        dr=g_now
        xmin,err_lp,loop_lp=golden2Dm(x,dr,1.e-5)
        err=np.sqrt(sum((xmin-x)**2))
        x=xmin
        xt,yt=np.append(xt,x[0]),np.append(yt,x[1])
        g_prev=g_now
    else:
        g_now=-deriv(x)
        l=np.sqrt(sum(g_now**2))/np.sqrt(sum(g_prev**2))
        dr=g_now+l*g_prev
        xmin,err_lp,loop_lp=golden2Dm(x,dr,1.e-5)
        err=np.sqrt(sum((x-xmin)**2))
        x=xmin
        xt=np.append(xt,x[0])
        yt=np.append(yt,x[1])
        g_prev=g_now

print ('minimum=',xmin,'number of iterations=',loop)

x=np.linspace(-2.5,1.5,400)
y=np.linspace(-1,3,400)
X, Y =np.meshgrid(x,y)
E=np.log10(100*(Y-X**2)**2 + (1.-X)**2)
dmax=np.max(E)
dmin=-3.
levels=(dmax-dmin)*np.arange(20)/19.+dmin
cmap=plt.cm.Blues

plt.figure(figsize=(10,10))
plt.plot(xt,yt,marker='*',c='#fdff00',mec='#fdff00',ms='11')
plt.contourf(x, y, E, levels,cmap=cmap)
plt.xlim(-2.5,1.5)
plt.ylim(-1.,3.)
```

```
minimum= [1.00880923 1.01774142] number of iterations= 34
```

Out[123]:

```
(-1.0, 3.0)
```

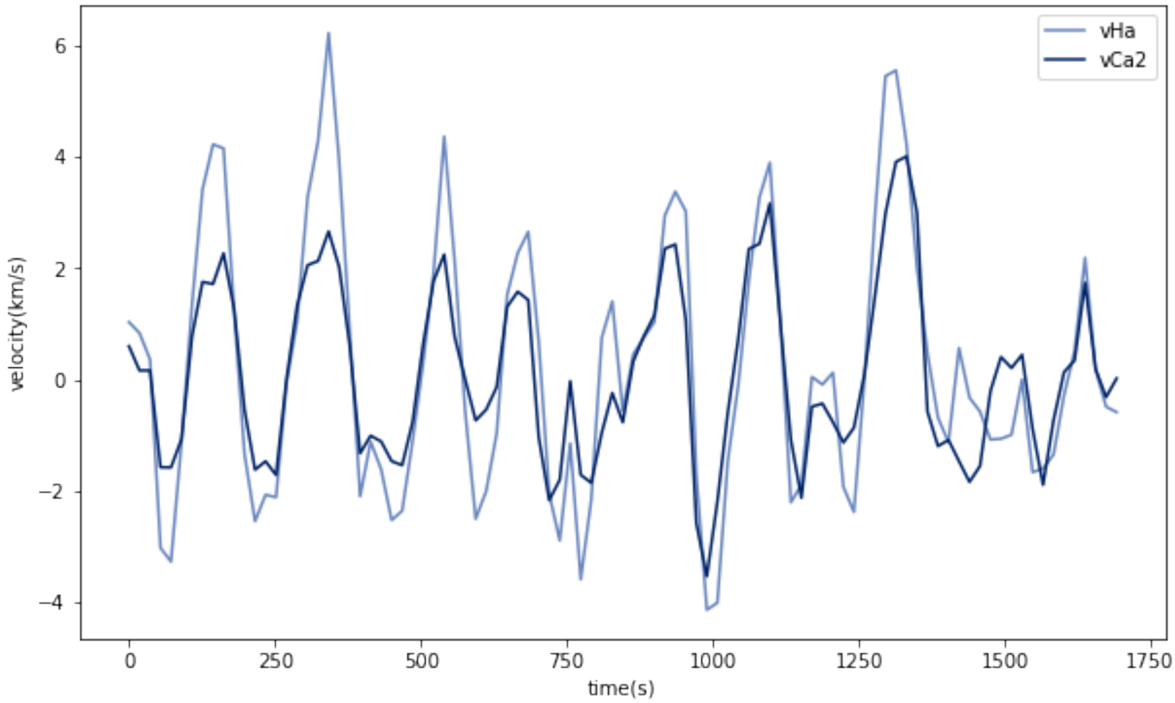


In [65]:
```
#2
from scipy.optimize import minimize

def func(x):
    return 100*(x[1]-x[0]**2)**2+(1-x[0])**2+100*(x[2]-x[1]**2)**2+
(1-x[2])**2
res = minimize(func, x0, method='CG', options={'xtol': 1e-5, 'disp'
: True})
print(res)
```

```
Optimization terminated successfully.
         Current function value: 0.000000
         Iterations: 49
         Function evaluations: 495
         Gradient evaluations: 99
     fun: 1.2399916943871707e-11
     jac: array([ 4.80267411e-06, -3.07791197e-06, -1.77681355e-07
])
 message: 'Optimization terminated successfully.'
    nfev: 495
     nit: 49
    njev: 99
  status: 0
 success: True
       x: array([0.99999914, 0.99999828, 0.99999659])

/Users/sehalee/anaconda3/lib/python3.7/site-packages/ipykernel_lau
ncher.py:5: OptimizeWarning: Unknown solver options: xtol
  """
```

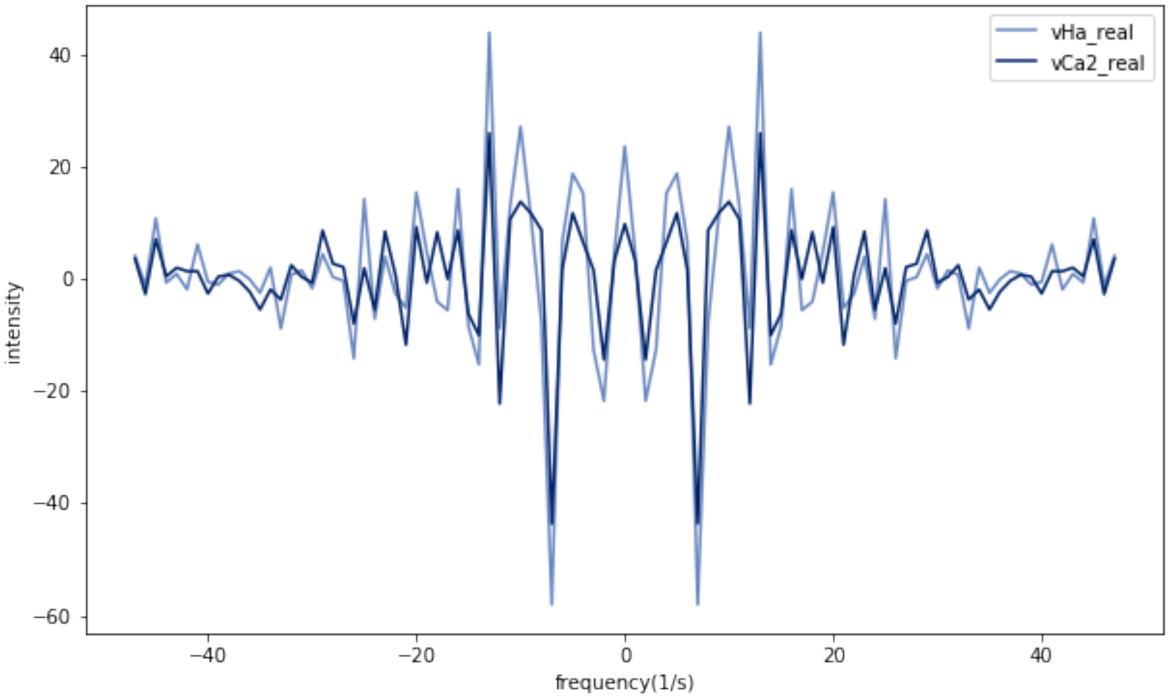
In [125]:
```
#3(a)
fn ='./Downloads/sol_vel.dat.txt'
t0,vHa,vCa2=np.loadtxt(fn,unpack=True,usecols=(0,1,2))
t=t0*60 #to convert from minutes to seconds
file=open('sol_vel.dat.txt','w')
plt.figure(figsize=(10,6))
plt.plot(t,vHa,label='vHa',c='#6a87bf')
plt.plot(t,vCa2,label='vCa2',c='#002366')
plt.ylabel('velocity(km/s)')
plt.xlabel('time(s)')
plt.legend()
```

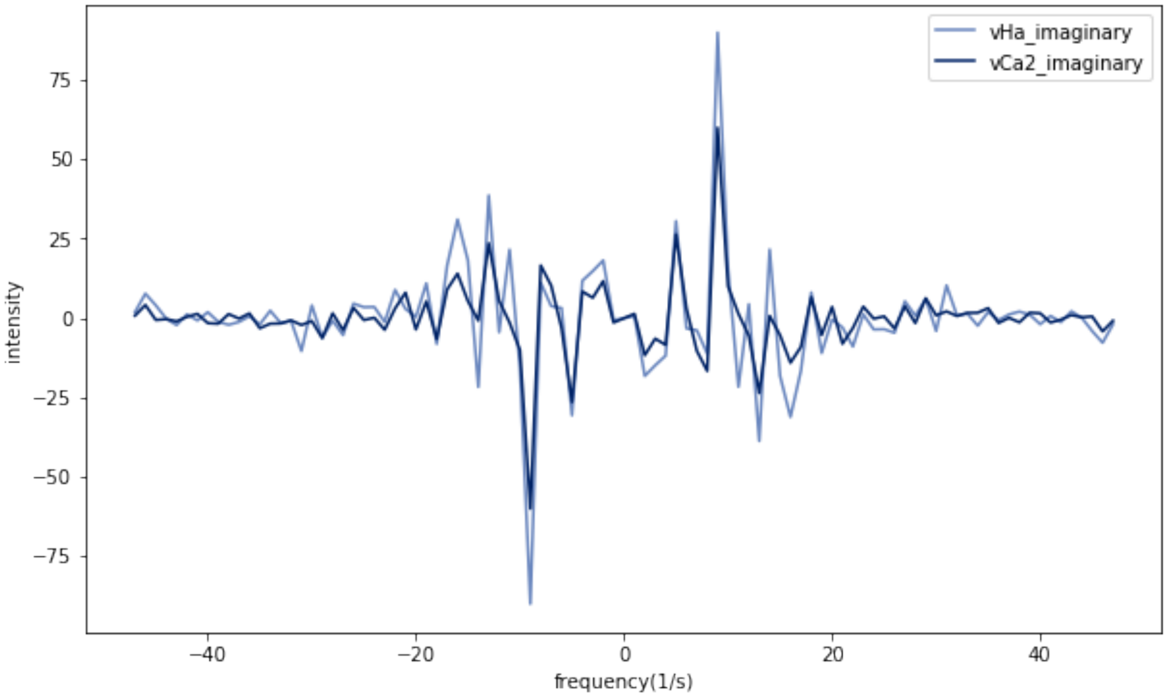
Out[125]: <matplotlib.legend.Legend at 0x1c271c5860>



In [124]:
```
#3(b)
F1=np.fft.fft(vHa)
F1=np.fft.fftshift(F1)

F2=np.fft.fft(vCa2)
F2=np.fft.fftshift(F2)

n=len(t)
k=np.fft.fftfreq(n,1./n)
k=np.fft.fftshift(k)

plt.figure(1,figsize=(10,6))
plt.plot(k,F1.real,label='vHa_real',c='#6a87bf')
plt.plot(k,F2.real,label='vCa2_real',c='#002366')
plt.xlabel('frequency(1/s)');plt.ylabel('intensity')
plt.legend()

plt.figure(2,figsize=(10,6))
plt.plot(k,F1.imag,label='vHa_imaginary',c='#6a87bf')
plt.plot(k,F2.imag,label='vCa2_imaginary',c='#002366')
plt.xlabel('frequency(1/s)');plt.ylabel('intensity')
plt.legend()
```

Out[124]: <matplotlib.legend.Legend at 0x1c26e2ac18>



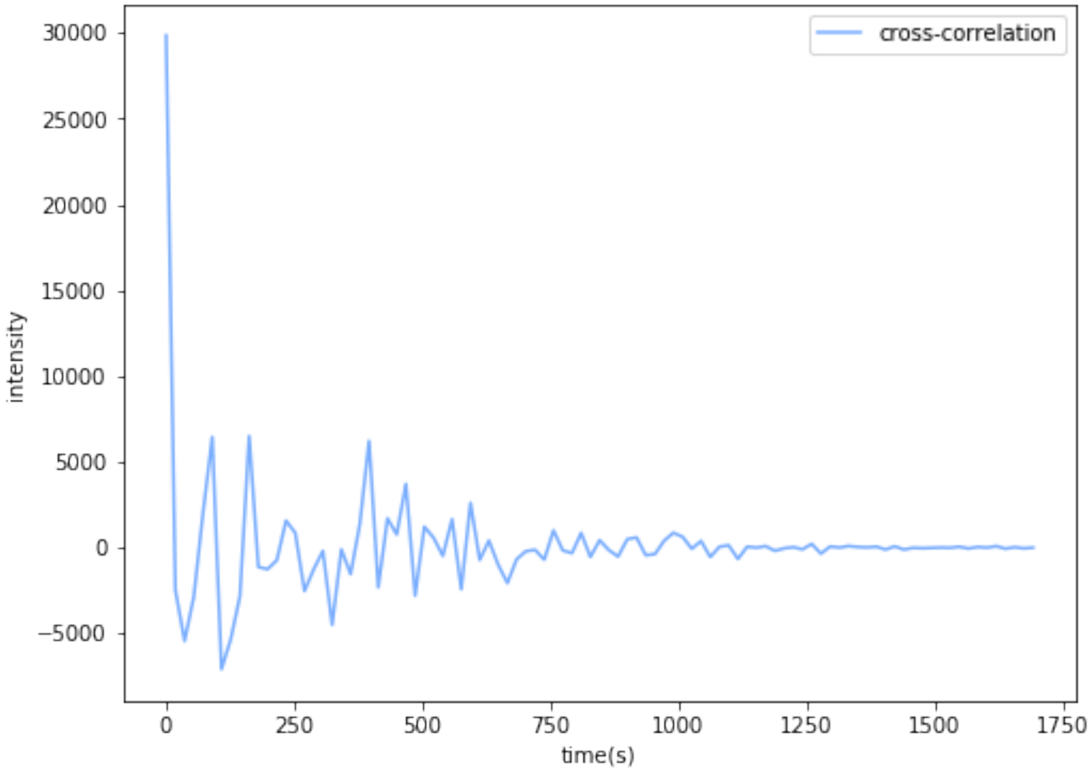



In [107]:
```
#3(c)
pfr1=(np.abs(F1.real)).tolist()
pfi1=(np.abs(F1.imag)).tolist()
print('the peak frequency of vHa\'s real part=',abs(k[pfr1.index(ma
x(pfr1))]))
print('the peak frequency of vHa\'s imaginary part=',abs(k[pfi1.ind
ex(max(pfi1))]))

pfr2=(np.abs(F2.real)).tolist()
pfi2=(np.abs(F2.imag)).tolist()
print('the peak frequency of vCa2\'s real part=',abs(k[pfr2.index(m
ax(pfr2))]))
print('the peak frequency of vCa2\'s imaginary part=',abs(k[pfi2.in
dex(max(pfi2))]))
p1=np.amax(t)/abs(k[pfr1.index(max(pfr1))])
p2=np.amax(t)/abs(k[pfr2.index(max(pfr2))])
print('the oscillation period of vHa=',p1,'seconds')
print('the oscillation period of vCa2=',p2,'seconds')
```

```
the peak frequency of vHa's real part= 7.0
the peak frequency of vHa's imaginary part= 9.0
the peak frequency of vCa2's real part= 7.0
the peak frequency of vCa2's imaginary part= 9.0
the oscillation period of vHa= 241.71428571428572 seconds
the oscillation period of vCa2= 241.71428571428572 seconds
```

In [126]:
```
#3(d)
from scipy import signal
Corr=scipy.signal.correlate(F1,F2,method='fft')
plt.figure(1,figsize=(8,6))
plt.xlabel('time(s)');plt.ylabel('intensity')
plt.plot(t,Corr[94:190],label='cross-correlation',c='#7aaeff')
plt.legend()
```

Out[126]: <matplotlib.legend.Legend at 0x1c26e20b00>




In [ ]: