In [75]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import scipy.stats as sts
import math
%matplotlib inline
```

In [76]:
```
f = open('cauchy.txt', 'r')
data = np.array([])
for s in f:
    data = np.append(data, float(s))
f.close()
print data.shape
```

```
(1000,)
```

In [83]:
```
sample = data[:500]
distr = sts.cauchy()

import math

def Lnfp(theta):
    return sum([-1.*math.log(1. + (x-theta)**2.) for x in sample])

thetaline = np.arange(-1000.,1000.,.01)
res = np.argmax([Lnfp(theta) for theta in thetaline])
print 'N/2: ', thetaline[res]
```

```
N/2:  156.189999999
115619
```

In [84]:
```
sample = data
res = np.argmax([Lnfp(theta) for theta in thetaline])
print 'N: ', thetaline[res]
```

```
N:  156.199999999
```

In [89]:
```
#а можно было сделать по-нормальному:
def LnfpMinimize(x0):
    return -Lnfp(x0)

sample = data[:500]
from scipy.optimize import minimize
print 'N/2: \n',minimize(LnfpMinimize, x0=0, method='BFGS')

sample = data
print '\n\n\nN: \n', minimize(LnfpMinimize, x0=0, method='BFGS')
```

```
N/2:
   status: 0
  success: True
     njev: 14
     nfev: 42
 hess_inv: array([[ 0.00377878]])
      fun: 652.1853461842097
        x: array([ 156.18831671])
  message: 'Optimization terminated successfully.'
      jac: array([ 0.])
      nit: 3



N:
   status: 2
  success: False
     njev: 23
     nfev: 70
 hess_inv: array([[  8.01729038e-06]])
      fun: 1355.1710309624477
        x: array([ 156.20449001])
  message: 'Desired error not necessarily achieved due to precisio
n loss.'
      jac: array([  1.52587891e-05])
      nit: 5
```

a) 156.18831661

b) 156.20448992