Задача 3.2

In [1]:

```
import numpy as np
import math as mth
import scipy.stats as sps
import matplotlib.pyplot as plt
from statsmodels.distributions.empirical_distribution import ECDF
from __future__ import division
```

Излучатель

In [2]:

```
sample = np.loadtxt('Cauchy.txt')
n = len(sample)
```

В данном случае функцией правдоподобия является величина

$$\prod_{i=1}^{n} \frac{1}{\pi(1 + (x_i - x_0)^2)}$$

где n - размер выборки. Т.к. с суммой величин работать гораздо приятнее, чем с произведением, я найду argmax не фукции правдоподобия, а ее логарифма (в силу монотонности логарифма максимум достигается в одной точке), равного

$$\sum_{i=1}^{n} - \ln \pi - \ln(1 + (x_i - x_0)^2)$$

но так как первое $n \ln \pi = const$, значит достаточно будет найти

$$argmax \sum_{i=1}^{n} - \ln(1 + (x_i - x_0)^2)$$

или argmin соответствующей суммы без минуса.

In [3]:

```
def count(x, minind, maxind):
    return sum([mth.log(1 + (sample[i] - x)**2) for i in np.arange(minind, max
ind)])
```

In [4]:

```
grid = np.linspace(-1000, 1000, 200001)
halfmin = -1000
wholemin = halfmin
#min1 - по половине, min2 - по всей
min1 = min2 = np.inf
for x in grid:
    y = count(x, 0, mth.floor(n/2))
    z = y + count(x, mth.floor(n/2), n)
    if (y < min1):
        min1 = y
        halfmin = x
    if (y < min2):
        min2 = z
        wholemin = x
```

Итак, оценивая параметр сдвига методом максимального правдоподобия, получим следующие результаты

In [5]:

```
print('По половине выборки -', halfmin)
print('По всей выборке -', wholemin)
```

```
По половине выборки - 345.58
По всей выборке - 1000.0
```

Банк

In [6]:

```
sample = np.loadtxt('Weibull.txt')
n=len(sample)
```

Прологарифмировав функцию правдоподобия, получим:

$$p_\gamma(\vec{x}) = \sum_{i=1}^{n} \ln(1 - e^{-x_i^\gamma})$$

(Индикатор только дожавляет ко всем значениям суммы одинаковую константу, здесь он везде = 1, поэтому я его даже не рассматриваю)

In [7]:

```
def weibull(gamma, minind, maxind):

    return sum([mth.log(1 - mth.e**(-sample[i]**gamma)) for i in np.arange(min
ind, maxind)])
```

Ищем её argmax:

In [8]:

```
max1 = max2 = -np.inf
max4 = max10 = -3 #тут будут ответы
#x = log_10 (gamma)
for x in np.linspace(-2, 2, 20001):
    y = weibull(10**x, 0, 1461)
    z = weibull(10**x, 1461, n) + y
    if(y > max1):
        max1 = y
        max4 = x
    if(z > max2):
        max2 = z;
        max10 = x
```

Оценив параметр формы по меттоду максимального правдоподобия, получаем следующие результаты:

In [9]:

```
print('По 4 годам - 10^', max4)
print('По 10 годам - 10^', max10)
```

```
По 4 годам - 10^ -2.0
По 10 годам - 10^ -2.0
```