# 1 Построение доверительных интервалов стандартных распределений

Построим доверительные интервалы параметров распределений:

- выборка $\mathbb{X} = (X_1, X_2, ..., X_n) \sim \mathbb{R}\left[0, \theta\right]$
- выборка $\mathbb{X} = (X_1, X_2, ..., X_n) \sim \mathbb{N}\left[\theta, \sigma^2\right]$
- выборка $\mathbb{X} = (X_1, X_2, ..., X_n) \sim \mathbb{N}\left[\mu, \theta^2\right]$

И асимптотический доверительный интервал параметра распределения выборки $\mathbb{X} = (X_1, X_2, ..., X_n) \sim Be(\theta)$

## 1.1 Равномерное распределение $\mathbb{R}[0, \theta]$

Для того чтобы получить доверительный интервал используем преобразование Смирнова. В качестве статистики рассмотрим $T(\mathbb{X}) = \frac{X_{(n)}}{\theta}$, тогда справедливо, что для выборки $\mathbb{X} = (X_1, ..., X_n) \sim \mathbb{R}[0, \theta]$: $T(\mathbb{X}) \overset{\text{п.н}}{\leq} 1$. Так как $F_{T(\mathbb{X})}(x) = x^n,$ для $x \in [0, 1]$, то $\mathbb{P}\left[T(\mathbb{X}) \leq t\right] = t =>$

$=> \left| \mathbb{P}\left[X_{(n)} \leq \theta\right] \overset{\text{п.н}}{=} 1, F(a) = 1 - \gamma = a^n => a = (1-\gamma)^{\frac{1}{n}} \right| =>$

$=> \mathbb{P}\left[X_{(n)} \leq \theta \leq \frac{X_{(n)}}{(1-\gamma)^{\frac{1}{n}}}\right] = \gamma.$

Таким образом, $\theta \in \left[X_{(n)}, \frac{X_{(n)}}{(1-\gamma)^{\frac{1}{n}}}\right]$.

Рассмотрим программную реализацию такого оценивания на языке Python:

Напишем функцию подсчета доверительного интервала по приведенной выше формуле с заданным по умолчанию параметром $\gamma = 0.95$:

```
def R_conf(X, gamma = 0.95):
  div = 1-gamma
  div = div**(1/X.size)
  ans = [X.max(), X.max()/div]

  return ans
```

Так же напишем функцию создания выборки заданной длины $n$,используя модуль numpy(импортированный под именем np), которая будет выводить полученные оценки:

```
def R(n = 100, theta = 5.0):
  print("Estimating of theta parameter=" + str(theta))
  R = np.random.uniform(high=theta, size = (n,))
  print("Sampling length " + str(n) + ' ' + str(R_conf(R)))
```

Получаем следующий вывод работы программы:

- Оценивание параметра theta=5.0
- Выборка длины 100 [4.868916901311929, 5.016983380704351]

- Оценивание параметра theta=5.0
- Выборка длины 100000 [4.999933992878683, 5.000083779758559]
- Оценивание параметра theta=5.0
- Выборка длины 100000000 [4.99999993481671, 5.000000084603324]

### 1.2 Нормальное распределение(параметр - математическое ожидание) $\mathbb{N}\left[\theta, \sigma^2\right]$

Используя следующие утверждения, получим доверительный интервал параметра - математического ожидания:

$$\mathbb{X} = (X_1, ..., X_n) \sim \mathbb{N}[\theta, \sigma^2] => X_k - \theta \sim \mathbb{N}[0, \sigma^2]; \frac{1}{n}\sum_{k=1}^{n}(X_k - \theta) \sim \mathbb{N}\left[0, \frac{\sigma^2}{n}\right]$$

$$\frac{\sqrt{n}}{\sigma n}\left(n\overline{X} - n\theta\right) = \frac{\sqrt{n}}{\sigma}\left(\overline{X} - \theta\right) \sim \mathbb{N}[0, 1]$$

$$\mathbb{P}\left[\Phi^{-1}_{\frac{1+\gamma}{2}} \leq \frac{\sqrt{n}}{\sigma}\left(\overline{X} - \theta\right) \leq -\Phi^{-1}_{\frac{1+\gamma}{2}}\right] \geq \gamma$$

$$\theta \in \left[\overline{X} + \frac{\sigma}{\sqrt{n}}\Phi^{-1}_{\frac{1+\gamma}{2}}, \overline{X} - \frac{\sigma}{\sqrt{n}}\Phi^{-1}_{\frac{1+\gamma}{2}}\right]$$

Запрограммируем этот интервал, как и в случае равномерного распределения:

```
import scipy.stats as st

def quant(X, val):
  return st.scoreatpercentile(X, val)

def N_e_conf(X, sigma=1, gamma=0.95):
  mult = quant(X, (1+gamma)/2)
  ans = [X.mean() + sigma/np.sqrt(X.size)*mult, X.mean() - sigma/np
                                    .sqrt(X.size)*mult]

  return ans

def N(n = 100, sigma = 1, theta = 0):
  print("Estimating of theta parameter=" + str(theta))
  N = np.random.normal(loc=theta, scale=sigma, size=(n,))
  print("Sampling length " + str(n) + ' ' +
    str(N_e_conf(N, sigma)))
```

Получаем следующий результат:

- Оценивание параметра theta=0
- Выборка длины 100 [-0.06927932188214225, 0.3479153829436803]
- Оценивание параметра theta=0

- Выборка длины 100000 [-0.004291419587833058, 0.010501507127366142]
- Оценивание параметра theta=0
- Выборка длины 100000000 [-0.0002462784310329848, 0.00022087784089780692]

### 1.3 Нормальное распределение(параметр - дисперсия) $\mathbb{N}\left[\mu, \theta^2\right]$

Для оценивания параметра дисперсии воспользуемся двумя способами и справним результаты:

- Приведение к $\mathbb{N}[0,1]$
- Использование распределения $\chi^2_n$

1)

$$X_k - \mu \sim \mathbb{N}[0, \theta^2] => \frac{\sqrt{n}}{\theta n}\left(n\overline{X} - n\mu\right) \sim \mathbb{N}[0,1]$$

$$\mathbb{P}\left[\Phi^{-1}_{\frac{1+\gamma}{2}} \leq \frac{\sqrt{n}}{\theta}\left(\overline{X} - \mu\right) \leq -\Phi^{-1}_{\frac{1+\gamma}{2}}\right] \geq \gamma$$

Предположим положительность разности выборочного среднего и мат.ожидания, тогда получим: $\theta \in \left[\frac{\sqrt{n}(\overline{X}-\mu)}{-\Phi^{-1}_{\frac{1+\gamma}{2}}}, \frac{\sqrt{n}(\overline{X}-\mu)}{\Phi^{-1}_{\frac{1+\gamma}{2}}}\right]$

Запрограммируем этот способ и оценим параметры:

```
def N_sigma_conf_error(X, mu=0, gamma=0.95):
  mult = quant(X, (1+gamma)/2)
  ans = [-np.sqrt(n)*(X.mean()-mu)/mult, np.sqrt(n)*(X.mean()-mu)/
                                  mult]

  return ans

def N_sigma_err(n = 100, mu = 0, theta = 1):
  print("Estimating of theta parameter=" + str(theta))
  N = np.random.normal(loc=mu, scale=theta, size=(n,))
  print("Sampling length " + str(n) + ' ' +
    str(N_sigma_conf_error(N, mu)))
```

- Оценивание параметра theta=1
- Выборка длины 100 [-0.9528333751739824, 0.9528333751739824]
- Оценивание параметра theta=1
- Выборка длины 100000 [-0.0075724578824839205, 0.0075724578824839205]
- Оценивание параметра theta=1
- Выборка длины 100000000 [-0.00029265994355404335, 0.00029265994355404335]

Как видим, результат очень плохой и с ростом длины выборки точность падает.

2)

$$S_{n,2} = \frac{1}{n}\sum_{k=1}^{n}(X_k - \mu)^2 => \frac{nS_{n,2}}{\theta^2} \sim \chi^2_n$$

Введем квантиль распределения $\chi^2_n$: $\chi^2_n(\frac{1+\gamma}{2})$, тогда

$$\theta^2 \in \left[\sqrt{\frac{nS_{n,2}}{\chi^2_n\left(\frac{1+\gamma}{2}\right)}}, \sqrt{\frac{nS_{n,2}}{\chi^2_n\left(\frac{1-\gamma}{2}\right)}}\right]$$

Запрограммируем этот способ и проанализируем вывод:

```
def sigm_quant(n, gamma):
  Chi = np.random.chisquare(n)
  return quant(Chi, gamma)

def N_sigma_conf(X, mu=0, gamma=0.95):
  mult1 = sigm_quant(X.size, (1+gamma)/2)
  mult2 = sigm_quant(X.size, (1-gamma)/2)
  S_2 = (X*X).mean()
  ans = [np.sqrt(X.size*S_2/mult1), np.sqrt(X.size*S_2/mult2)]

  return ans

def N_sigma(n = 100, mu = 0, theta = 1):
  print("Estimating of theta parametr=" + str(theta))
  N = np.random.normal(loc=mu, scale=theta, size=(n,))
  print("Sampling length " + str(n) + ' ' +
    str(N_sigma_conf(N, mu)))
```

- Оценивание параметра theta=1
- Выборка длины 100 [0.9811807087976958, 0.9711996898457366]
- Оценивание параметра theta=1
- Выборка длины 100000 [0.9990226155803235, 0.99826862747577]
- Оценивание параметра theta=1
- Выборка длины 100000000 [1.000112416519521, 1.000033297927437]

Данное оценивание лучше предыдущего, и с ростом длины выборки точность тоже растет.

## 1.4 Распределение Бернулли $Be(\theta)$

Рассмотрим статистику $T_n(\mathbb{X}) = \overline{X}$:

$$\mathbb{E}\overline{X} = \theta, \mathbb{D}\overline{X} = \frac{\theta(1-\theta)}{n}$$

По центральной предельной теореме:

$$\mathbb{L}_\theta\left(\frac{\overline{X}-\theta}{\sqrt{\frac{\theta(1-\theta)}{n}}}\right) \to \mathbb{N}[0,1]$$

Тогда $\theta \in \left[\overline{X} - \frac{c_\gamma\sqrt{\theta(1-\theta)}}{\sqrt{n}}, \overline{X} + \frac{c_\gamma\sqrt{\theta(1-\theta)}}{\sqrt{n}}\right]$, $c_\gamma$ - квантиль стандартного нормального распределения.

Запрограммируем эту оценку и оценим результат:

```
from scipy.stats import bernoulli
def Be_conf(X, gamma=0.95):
  mult = quant(X, (1+gamma)/2)
  st_dev = (X*X).mean()
  st_dev = np.sqrt(st_dev)
  ans = [X.mean()-mult*st_dev/np.sqrt(X.size), X.mean()+mult*st_dev
                                  /np.sqrt(X.size)]

  return ans

def Be(n=100, theta=0.5):
  print("Estimating of theta parametr=" + str(theta))
  B = bernoulli.rvs(size=(n,), p=theta)
  print("Sampling length " + str(n) + ' ' + str(Be_conf(B)))
```

Результат:

- Оценивание параметра theta=0.5
- Выборка длины 100 [0.4, 0.4]
- Оценивание параметра theta=0.5
- Выборка длины 100000 [0.49658, 0.49658]
- Оценивание параметра theta=0.5
- Выборка длины 100000000 [0.49992138, 0.49992138]

С ростои длины выборки точность оценки растет.