СКБ201 Тур Тимофей

Теория вероятности, долгосрочные домашние задания. Вариант 68: дискретное - 3, непрерывное - 5.

3 - Дискретное равномерное 1: $P(x) = \theta^{-1}, x \in \{1, \dots, \theta\}, \theta = 29$

Обозначим дискретное распределение в дальнейшем за ξ

5 - Треугольное: $f(x) = \begin{cases} \dfrac{2x}{\theta}, & \text{если } x \in [0, \theta] \\ \dfrac{2(1-x)}{1-\theta}, & \text{если } x \in (\theta, 1] \\ 0, & \text{иначе} \end{cases}, \theta = 0.45$

Обозначим абсолютно непрерывное распределение в дальнейшем за η

Навигация

Домашнее задание 1

Дискретное

Задание 1

Функция распределения

$$F(n) \stackrel{\text{def}}{=} P(\xi \leq n) = \sum_{k=1}^{n} P(\xi = k) = \sum_{k=1}^{n} \theta^{-1} = \underline{n\theta^{-1}}$$

Математическое ожидание

$$M\xi \stackrel{\text{def}}{=} \sum_{x=1}^{\theta} xP(\xi = x) = \sum_{x=1}^{\theta} x\theta^{-1} = \theta^{-1} \sum_{x=1}^{\theta} x = \theta^{-1} \frac{1+\theta}{2}\theta = \underline{\frac{\theta+1}{2}}$$

Дисперсия

$$D\xi \stackrel{\text{def}}{=} M(\xi - M\xi)^2 = M\xi^2 - (M\xi)^2$$

$$M\xi^2 \stackrel{\text{def}}{=} \sum_{x=1}^{\theta} x^2 P(\xi = x) = \theta^{-1} \sum_{x=1}^{\theta} x^2 = \theta^{-1} \frac{\theta(1-\theta)(1+2\theta)}{6} = \frac{(1+\theta)(1+2\theta)}{6}$$

$$\Rightarrow D\xi = M\xi^2 - (M\xi)^2 = \frac{(1+\theta)(1+2\theta)}{6} - (\frac{\theta+1}{2})^2 = \frac{2(1+3\theta+2\theta^2) - 3(1+2\theta+\theta^2)}{12} = \underline{\frac{\theta^2-1}{12}}$$

Квантиль уровня γ

$$P(\xi \leq x_\gamma) \geq \gamma \Leftrightarrow \sum_{k=1}^{x_\gamma} P(\xi = k) \geq \gamma \Leftrightarrow \sum_{k=1}^{x_\gamma} \theta^{-1} \geq \gamma \Leftrightarrow x_\gamma \theta^{-1} \geq \gamma \Leftrightarrow x_\gamma \geq \gamma\theta \Rightarrow \underline{x_\gamma = \gamma\theta}$$

Задание 2

Примером события с дискретным равномерным распределением может быть игра "Bingo". Но не вся она, а лишь ее часть. В ней, подобно лото, участникам выдаются цветные листки с числами и маркерами, а ведущий стоит у аппарата, поторый по нажатию кнопки выдает случайный шарик, крутящийся в нем. Шарик имеет цвет и номер, и участники выделяют соответсвующие ячейки на своем листе, пока у них не получатся какая-нибудь соответствующая последовательность. (Лично я увидел эту игру в сериале "Лучше звоните Солу" в первом сезоне). Чтобы эта модель была применима к нашему распределению, игру следует упростить: На листке всего 1 номер и мячики не имеют цвета. Тогда шанс появления какого-то мячика будет равен $\frac{1}{\text{количество мячиков} = \theta} = \theta^{-1}$, и, соответственно шанс выигрыша какого-то игрока тоже равен θ^{-1}

Задание 3

Поделим отрезок $[0, 1]$ на сегменты равные θ^{-1}. Их будет в точности θ штук, а выборка определяется вхождением в какой из последовательных отрезков получилось у случайной величины: $\sqsupset u$ - сгенерированная равномерно распределенная величина на отрезке $[0, 1]$, тогда x определяется по формуле $(x-1)\theta^{-1} \leq u < x\theta^{-1}$

In [1]:
```python
import numpy as np

def generate_xi(theta=29):
    rng = np.random.default_rng()
    u = rng.uniform()
    for k in range(1, theta + 1):
        if (k - 1) / theta <= u < k / theta:
            return k
```

Абсолютно непрерывное

Задание 1

Функция распределения

$$F(x) \overset{\text{def}}{=} \int_{\mathbb{R}} f(x) = \begin{cases} 0, & \text{если } x < 0 \\ \int_0^x \frac{2t}{\theta} dt, & \text{если } x \in [0, \theta] \\ \int_0^\theta \frac{2t}{\theta} dt + \int_\theta^x \frac{2(1-t)}{1-\theta} dt, & \text{если } x \in (\theta, 1] \\ \int_0^\theta \frac{2t}{\theta} dt + \int_\theta^1 \frac{2(1-t)}{1-\theta}, & \text{если } x > 1 \end{cases}$$

$$1) \int_0^x \frac{2t}{\theta} dt = \frac{1}{\theta} \int_0^x 2t dt = \frac{1}{\theta} t^2 \Big|_0^x = \frac{1}{\theta} x^2$$

$$2) \int_0^\theta \frac{2t}{\theta} dt + \int_\theta^x \frac{2(1-t)}{1-\theta} dt = \theta + \frac{2}{1-\theta} \int_\theta^x (1-t) dt = \theta + \frac{2}{1-\theta} (t - \frac{1}{2} t^2) \Big|_\theta^x =$$

$$= \theta + \frac{2}{1-\theta} (x - \frac{1}{2} x^2 - \theta + \frac{1}{2} \theta^2) = \theta + \frac{1}{1-\theta} (2x - x^2 - 2\theta + \theta^2) =$$

$$= \frac{1}{1-\theta} (2x - x^2 - \theta)$$

$$3) \int_0^\theta \frac{2t}{\theta} dt + \int_\theta^1 \frac{2(1-t)}{1-\theta} = \frac{1}{1-\theta} (2 - 1 - \theta) = \frac{1-\theta}{1-\theta} = 1$$

$$\Rightarrow F(x) = \begin{cases} 0, & \text{если } x < 0 \\ \frac{1}{\theta} x^2, & \text{если } x \in [0, \theta] \\ \frac{1}{1-\theta} (2x - x^2 - \theta), & \text{если } x \in (\theta, 1] \\ 1, & \text{если } x > 1 \end{cases}$$

Математическое ожидание

$$M\eta \stackrel{\text{def}}{=} \int_{\mathbb{R}} f(x)xdx = \int_0^\theta \frac{2x}{\theta}xdx + \int_\theta^1 \frac{2(1-x)}{1-\theta}xdx = \frac{2}{\theta}\int_0^\theta x^2dx + \frac{2}{1-\theta}\int_\theta^1 (x-x^2)dx =$$
$$= \frac{2}{3\theta}x^3\Big|_0^\theta + \frac{2}{1-\theta}(\frac{1}{2}x^2 - \frac{1}{3}x^3)\Big|_\theta^1 = \frac{2}{3\theta}\theta^3 + \frac{2}{1-\theta}(\frac{1}{2} - \frac{1}{3} - \frac{1}{2}\theta^2 + \frac{1}{3}\theta^3) =$$
$$= \frac{2}{3}\theta^2 + \frac{2}{1-\theta}(\frac{1}{6} + \frac{2\theta^3 - 3\theta^2}{6}) = \frac{2\theta^2}{3} + \frac{2\theta^3 - 3\theta^2 + 1}{3(1-\theta)} = \frac{2\theta^2 - 2\theta^3 + 2\theta^3 - 3\theta^2 + 1}{3(1-\theta)} =$$
$$= \frac{1-\theta^2}{3(1-\theta)} = \underline{\frac{1+\theta}{3}}$$

Дисперсия

$$D\eta \stackrel{\text{def}}{=} M(\eta - M\eta)^2 = M\eta^2 - (M\eta)^2$$

$$M\eta^2 \stackrel{\text{def}}{=} \int_{\mathbb{R}} f(x)x^2dx = \int_0^\theta \frac{2x}{\theta}x^2dx + \int_\theta^1 \frac{2(1-x)}{1-\theta}x^2dx = \frac{2}{\theta}\int_0^\theta x^3dx + \frac{2}{1-\theta}\int_\theta^1 (x^2-x^3)dx =$$
$$= \frac{1}{2\theta}x^4\Big|_0^\theta + \frac{2}{1-\theta}(\frac{1}{3}x^3 - \frac{1}{4}x^4)\Big|_\theta^1 = \frac{1}{2\theta}\theta^4 + \frac{2}{1-\theta}(\frac{1}{3} - \frac{1}{4} - \frac{1}{3}\theta^3 + \frac{1}{4}\theta^4) =$$
$$= \frac{1}{2}\theta^3 + \frac{2}{1-\theta}(\frac{1}{12} + \frac{3\theta^4 - 4\theta^3}{12}) = \frac{1}{2}\theta^3 + \frac{1}{6(1-\theta)}(3\theta^4 - 4\theta^3 + 1) =$$
$$= \frac{1}{6(1-\theta)}(3\theta^4 - 4\theta^3 + 1 + 3\theta^3 - 3\theta^4) = \frac{1}{6(1-\theta)}(1-\theta^3) = \frac{1+\theta+\theta^2}{6}$$

$$\Rightarrow D\eta = M\eta^2 - (M\eta)^2 = \frac{1+\theta+\theta^2}{6} - (\frac{1+\theta}{3})^2 = \frac{3(1+\theta+\theta^2) - 2(1+2\theta+\theta^2)}{18} = \underline{\frac{1-\theta+\theta^2}{18}}$$

Квантиль уровня γ

$$F(x_\gamma) \geq \gamma \Rightarrow \begin{cases} x_\gamma = 0, & \text{если } \gamma < 0 \\ \dfrac{1}{\theta}x_\gamma^2 \geq \gamma, & \text{если } \gamma \in [0, \theta] \\ \dfrac{1}{1-\theta}(2x_\gamma - x_\gamma^2 - \theta) \geq \gamma, & \text{если } \gamma \in (\theta, 1] \\ x_\gamma = 1, & \text{если } \gamma > 1 \end{cases}$$

1) $\dfrac{1}{\theta}x_\gamma^2 \geq \gamma \Leftrightarrow x_\gamma \geq \sqrt{\theta\gamma} \Rightarrow x_\gamma = \sqrt{\theta\gamma}$

2) $\dfrac{1}{1-\theta}(2x_\gamma - x_\gamma^2 - \theta) \geq \gamma \Rightarrow \dfrac{1}{1-\theta}(2x_\gamma - x_\gamma^2 - \theta) = \gamma \Leftrightarrow -x_\gamma^2 + 2x_\gamma - \theta = (1-\theta)\gamma \Leftrightarrow$

$$\Leftrightarrow -x_\gamma^2 + 2x_\gamma - \theta - \gamma + \theta\gamma = 0 \Rightarrow$$
$$\Rightarrow D = 4 - 4(\theta + \gamma - \theta\gamma) = 4(1 - \theta - \gamma + \theta\gamma) \Rightarrow$$
$$\Rightarrow x_\gamma = \frac{-2 \pm 2\sqrt{1-\theta-\gamma+\theta\gamma}}{-2} = 1 \pm \sqrt{1-\theta-\gamma+\theta\gamma}\,.$$
$$x_\gamma \in [\theta, 1] \Rightarrow x_\gamma = 1 - \sqrt{1-\theta-\gamma+\theta\gamma}$$

$$\Rightarrow x_\gamma = \begin{cases} \sqrt{\theta\gamma}, & \text{если } \gamma \in [0, \theta] \\ 1 - \sqrt{1-\theta-\gamma+\theta\gamma}, & \text{если } \gamma \in (\theta, 1] \end{cases}$$

Задание 2

Треугольное распределение на практике используется часто, потому что оно имеет минимум, максимум и пик, что делает его уже достаточным к реальности распределением, так еще и оно очень простое по своей математике и применению. Конкретно в приведенной формуле распределение ограничено 0 и 1 и имеет пик в θ, а в обычных случаях оно позволяет посчитать предполагаемую прибыль какого-то ресторана, просто делая предположение о минимуме, максимуме и наиболее вероятном значении при помощи анализа полученного распределения (например через математическое ожидание). Также, в силу простоты, оно может служить некоторой заменой к другим распределениям подобной структуры. Так, если мы, например, наблюдаем образование бактерий на влажной сахарной линии, то очевидно, что надо использовать нормальное распределение, потому что это почти именно то, что оно и отображает. Однако, чтобы использовать нормальное распределение также практическим методом потребуется вычислить дисперсию, что может быть трудной задачей, потому временной заменой может послужить простое треугольное распределение, чтобы пронаблюдать на нем отклонения.

Задание 3

Чтобы построить выборку от равномерного случайного распределения требуется найти $F^{-1}(u)$, что мы фактически искали, вычисляя квантиль уровня γ. Чем я и воспользуюсь, описав код ниже.

In [2]:
```
import numpy as np
def generate_eta(theta=0.45):
    rng = np.random.default_rng()
    u = rng.uniform()
    if u <= theta: return (theta*u)**0.5
    return 1-(1-theta-u+theta*u)**0.5
```

Домашнее задание 2

Дискретное

Задание 1

Демонстрировать выборки по 5 штук в 1000 элементов числами, это, конечно, интересно, и, наверняка, невероятно увлекательной задачей будет их оценивать на глаз. Поэтому решил лучше выборки продемонстрировать на графиках их эмперической функции и полигонах частот, которые будут приведены ниже, а здесь для демонстрации показать все значения первой выборки в 1000 элементов.

In [3]:
```
# Здесь допустимо использование функций генераторов, указанных ранее
# theta задана в каждой функции генератора параметром по умолчанию
# потому отдельное упоминание не требуется
n = [5, 10, 100, 200, 400, 600, 800, 1000]
sample_xi = [[np.sort(np.array([generate_xi() for i in range(j)]))
              for i in range(5)] for j in n]

demo_pxi = sample_xi[7][0].reshape((40, -1))
for i in demo_pxi:
    print(*i, sep = ' ')
```

```
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 3 3
3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
3 3 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
4 4 4 4 4 4 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6
6 6 6 6 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 8 8 8 8 8 8 8 8 8 8
8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8
8 8 8 8 8 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9
9 9 9 9 9 9 9 9 9 9 9 9 10 10 10 10 10 10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 10 11 11 11 11 11 11
11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11 11
11 11 11 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 12
12 12 12 12 12 12 12 12 12 12 12 12 12 12 12 13 13 13 13 13 13 13 13 13 13 13
13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13 13
13 13 13 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14
14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 14 15 15 15 15 15 15 15 15 15 15
15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15 15
15 15 15 15 15 15 15 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16 16
16 16 16 16 16 16 16 16 16 16 16 16 16 16 17 17 17 17 17 17 17 17 17 17 17 17
17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17 17
17 17 17 17 17 17 17 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18 18
18 18 18 18 18 18 18 18 18 18 18 18 18 18 19 19 19 19 19 19 19 19 19 19 19 19
19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19 19
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21 21
21 21 21 21 21 21 21 21 21 21 21 21 22 22 22 22 22 22 22 22 22 22 22 22 22 22
22 22 22 22 22 22 22 22 22 22 22 22 22 22 22 23 23 23 23 23 23 23 23 23 23 23
23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 23 24 24 24 24 24 24 24 24 24 24
24 24 24 24 24 24 24 24 24 24 24 24 24 24 24 24 24 24 24 24 25 25 25 25 25 25
25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25
25 25 25 25 25 25 25 25 25 25 25 25 25 26 26 26 26 26 26 26 26 26 26 26 26 26
26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 27 27 27 27 27 27
27 27 27 27 27 27 27 27 27 27 27 27 27 27 27 27 27 27 27 27 27 27 27 27 27 27
27 27 27 27 27 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28
28 28 28 28 28 28 28 28 28 28 28 28 28 29 29 29 29 29 29 29 29 29 29 29 29 29
29 29 29 29 29 29 29 29 29 29 29 29 29 29 29 29 29 29 29 29 29 29 29 29 29 29
```

Д32Д, Задание 2

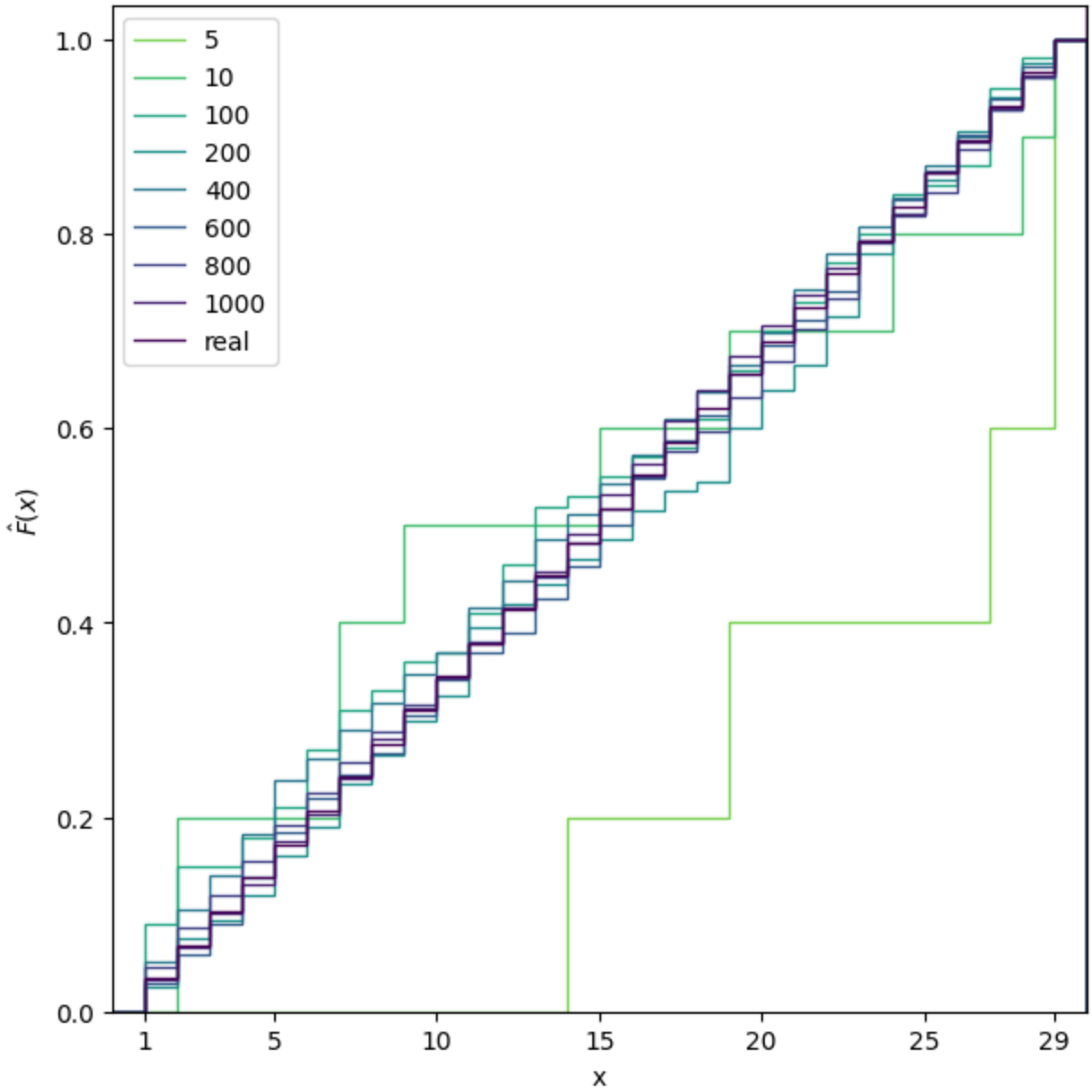
In [4]:

```
def xi_distr(sample, x):
    res = 0
    for i in sample:
        if i <= x:
            res += 1
    return res / len(sample)

def xi_distr_real(x: int, theta=29):
    return x / theta

X_realxi = np.arange(1-1, 29+1+1)
Y_realxi = xi_distr_real(X_realxi)

Yxi = np.array([[[xi_distr(sample_xi[k][j], x) for x in X_realxi]
                 for j in range(5)] for k in range(len(n))])

def xi_Dmn(Yn, Ym, n, m):
    res = 0
    for i in range(29):
        d = abs(Yn[i]-Ym[i])
        if d>res: res = d
    return (n*m/(n+m))**0.5*res

diffsxi = np.array([[[('%.2f' % xi_Dmn(Yxi[i][k], Yxi[j][k], n[i], n[i]))
                      if i>j else '-' for i in range(len(n))]
                     for j in range(len(n))]
                    for k in range(5)])
```

In [5]:
```
from matplotlib import pyplot as plt
colors = ['#7bd152', '#45be71', '#25a885', '#21908c',
          '#2b798e', '#355f8d', '#414486', '#482574']

# graph[0]
fig, ax = plt.subplots(2,1, figsize=(7,12), height_ratios = [2,1])
for i in range(8):
    ax[0].stairs(Yxi[i][0], np.append(X_realxi, 30), color = colors[i])
ax[0].stairs(Y_realxi, np.append(X_realxi, 30), color = '#440154')
ax[0].set(xticks = [1]+list(range(5,26,5))+[29], xmargin = 0, ymargin = 0,
          xlabel = 'x', ylabel = r'$\hat{F}(x)$',
          title = 'Дискретное равномерное, выборка 1 \n$\\xi , \\,\\theta$ = 29')
ax[0].legend([*n, 'real'], loc='upper left');

ax[1].table(cellText = diffsxi[0], rowLabels=n, colLabels=n,
            loc='center').scale(1, 1.5)
ax[1].set_axis_off()
ax[1].set_title(r'$D_{m,n}=\sqrt{\frac{nm}{n+m}}\sup_{x\in\mathbb{R}}$'+
                r'$|F_n(x)-F_m(x)|$');
```

Дискретное равномерное, выборка 1

ξ, $\theta = 29$



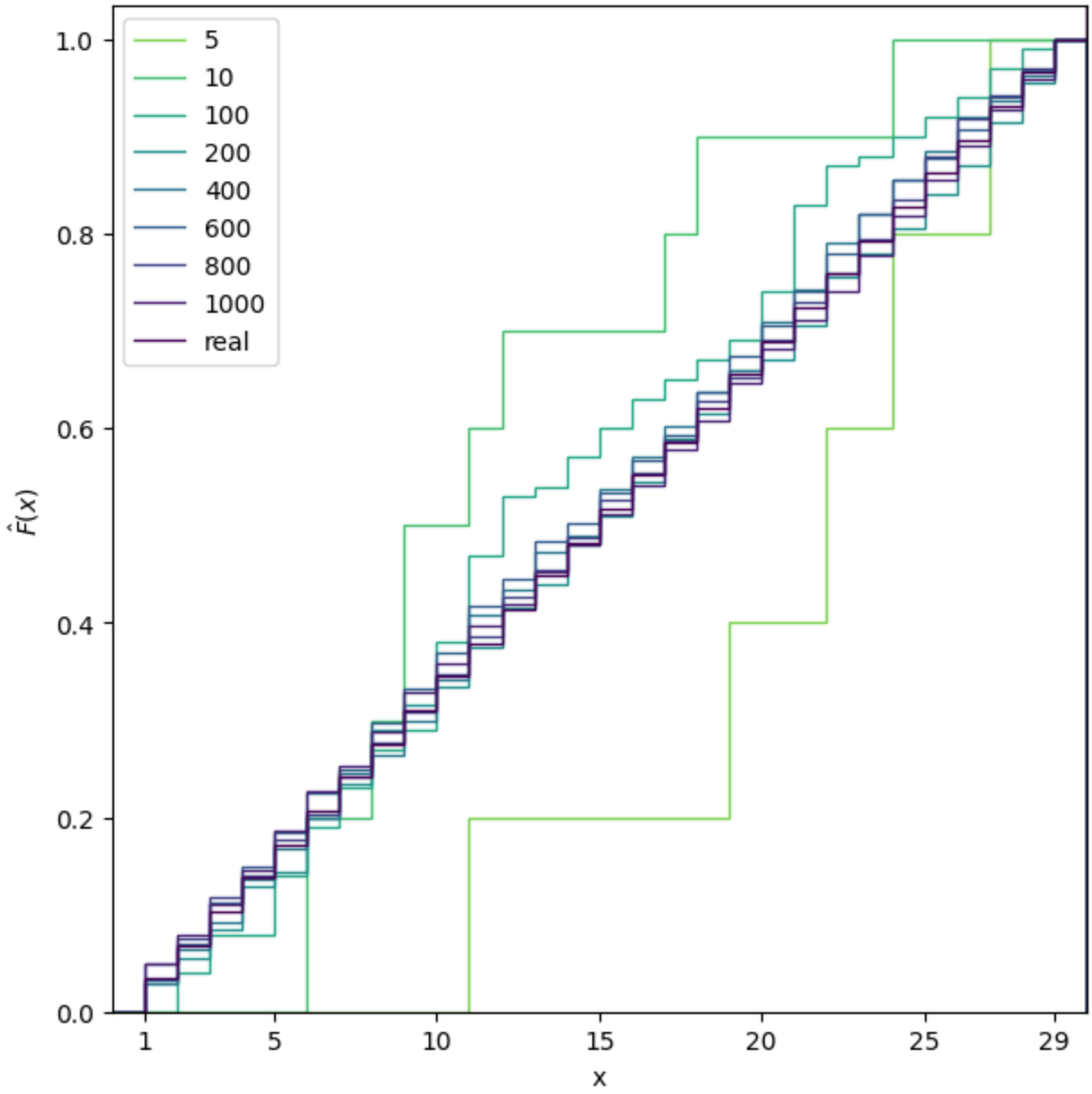
$$D_{m,n} = \sqrt{\frac{nm}{n+m}} \sup_{x \in \mathbb{R}} |F_n(x) - F_m(x)|$$

	5	10	100	200	400	600	800	1000
5	-	1.12	3.68	5.05	7.07	8.69	9.72	11.07
10	-	-	1.06	2.00	2.16	3.38	3.68	4.20
100	-	-	-	0.80	0.64	1.65	1.45	1.86
200	-	-	-	-	1.31	1.18	1.05	2.10
400	-	-	-	-	-	1.04	0.95	1.40
600	-	-	-	-	-	-	0.60	0.73
800	-	-	-	-	-	-	-	0.98
1000	-	-	-	-	-	-	-	-

In [6]:
```
# graph[1]
fig, ax = plt.subplots(2,1, figsize=(7,12), height_ratios = [2,1])
for i in range(8):
    ax[0].stairs(Yxi[i][1], np.append(X_realxi, 30), color = colors[i])
ax[0].stairs(Y_realxi, np.append(X_realxi, 30), color = '#440154')
ax[0].set(xticks = [1]+list(range(5,26,5))+[29], xmargin = 0, ymargin = 0,
          xlabel = 'x', ylabel = r'$\hat{F}(x)$',
          title = 'Дискретное равномерное, выборка 2 \n$\\xi , \\,\\theta$ = 29')
ax[0].legend([*n, 'real'], loc='upper left');

ax[1].table(cellText = diffsxi[1], rowLabels=n, colLabels=n,
            loc='center').scale(1, 1.5)
ax[1].set_axis_off()
ax[1].set_title(r'$D_{m,n}=\sqrt{\frac{nm}{n+m}}\sup_{x\in\mathbb{R}}$'+
                r'$|F_n(x)-F_m(x)|$');
```

Дискретное равномерное, выборка 2

ξ, $\theta = 29$



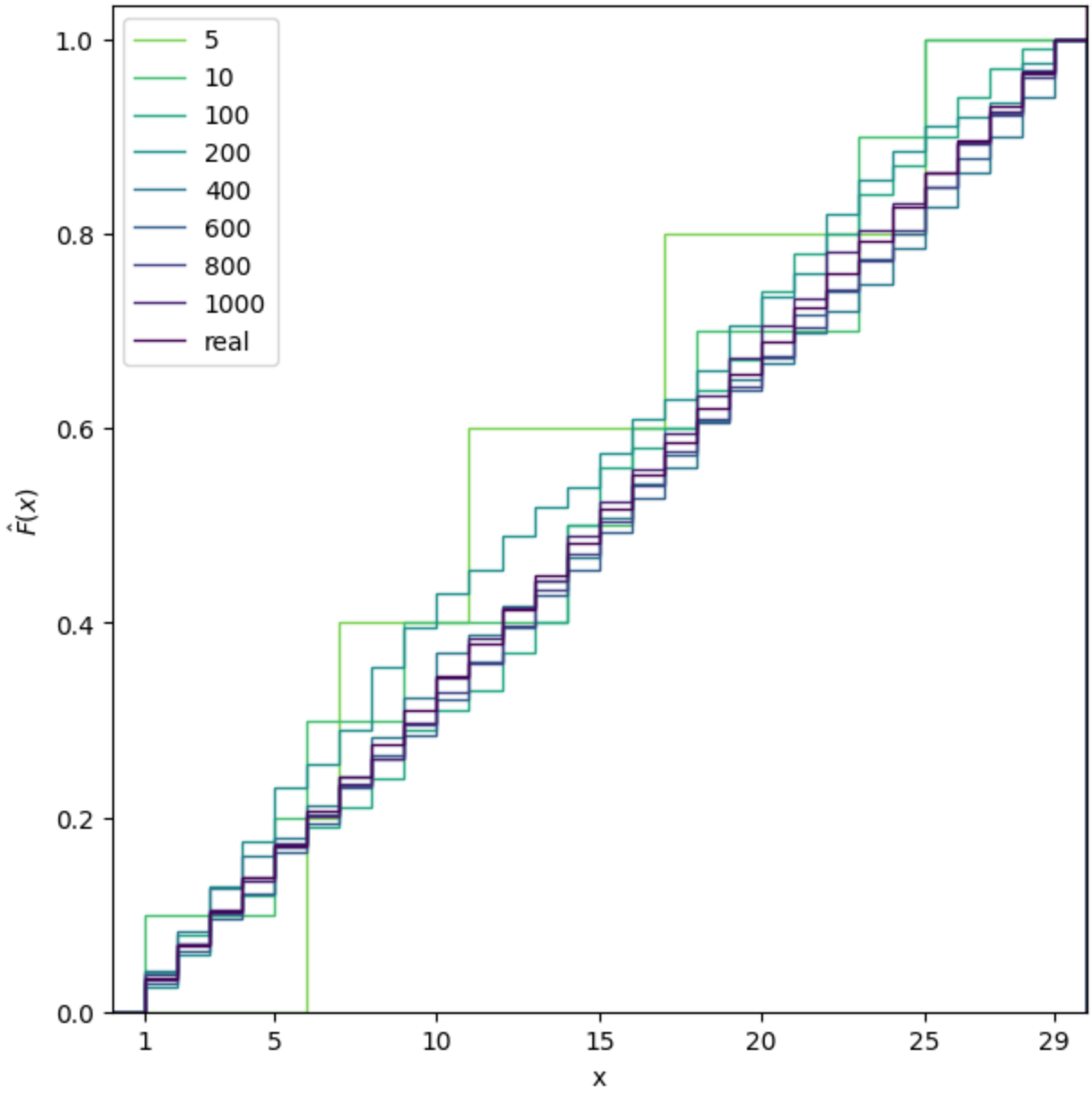
$$D_{m,n} = \sqrt{\frac{nm}{n+m}} \sup_{x \in \mathbb{R}} |F_n(x) - F_m(x)|$$

	5	10	100	200	400	600	800	1000
5	-	1.57	3.32	4.15	6.19	7.56	8.55	9.12
10	-	-	1.63	2.85	3.75	4.56	5.45	6.53
100	-	-	-	1.25	1.34	1.56	2.20	2.88
200	-	-	-	-	0.71	0.87	0.95	0.92
400	-	-	-	-	-	0.58	0.60	1.10
600	-	-	-	-	-	-	0.64	0.94
800	-	-	-	-	-	-	-	0.59
1000	-	-	-	-	-	-	-	-

In [7]:
```
# graph[2]
fig, ax = plt.subplots(2,1, figsize=(7,12), height_ratios = [2,1])
for i in range(8):
    ax[0].stairs(Yxi[i][2], np.append(X_realxi, 30), color = colors[i])
ax[0].stairs(Y_realxi, np.append(X_realxi, 30), color = '#440154')
ax[0].set(xticks = [1]+list(range(5,26,5))+[29], xmargin = 0, ymargin = 0,
          xlabel = 'x', ylabel = r'$\hat{F}(x)$',
          title = 'Дискретное равномерное, выборка 3 \n$\\xi , \\,\\theta$ = 29')
ax[0].legend([*n, 'real'], loc='upper left');

ax[1].table(cellText = diffsxi[2], rowLabels=n, colLabels=n,
            loc='center').scale(1, 1.5)
ax[1].set_axis_off()
ax[1].set_title(r'$D_{m,n}=\sqrt{\frac{nm}{n+m}}\sup_{x\in\mathbb{R}}$'+
                r'$|F_n(x)-F_m(x)|$');
```

Дискретное равномерное, выборка 3

ξ, $\theta = 29$



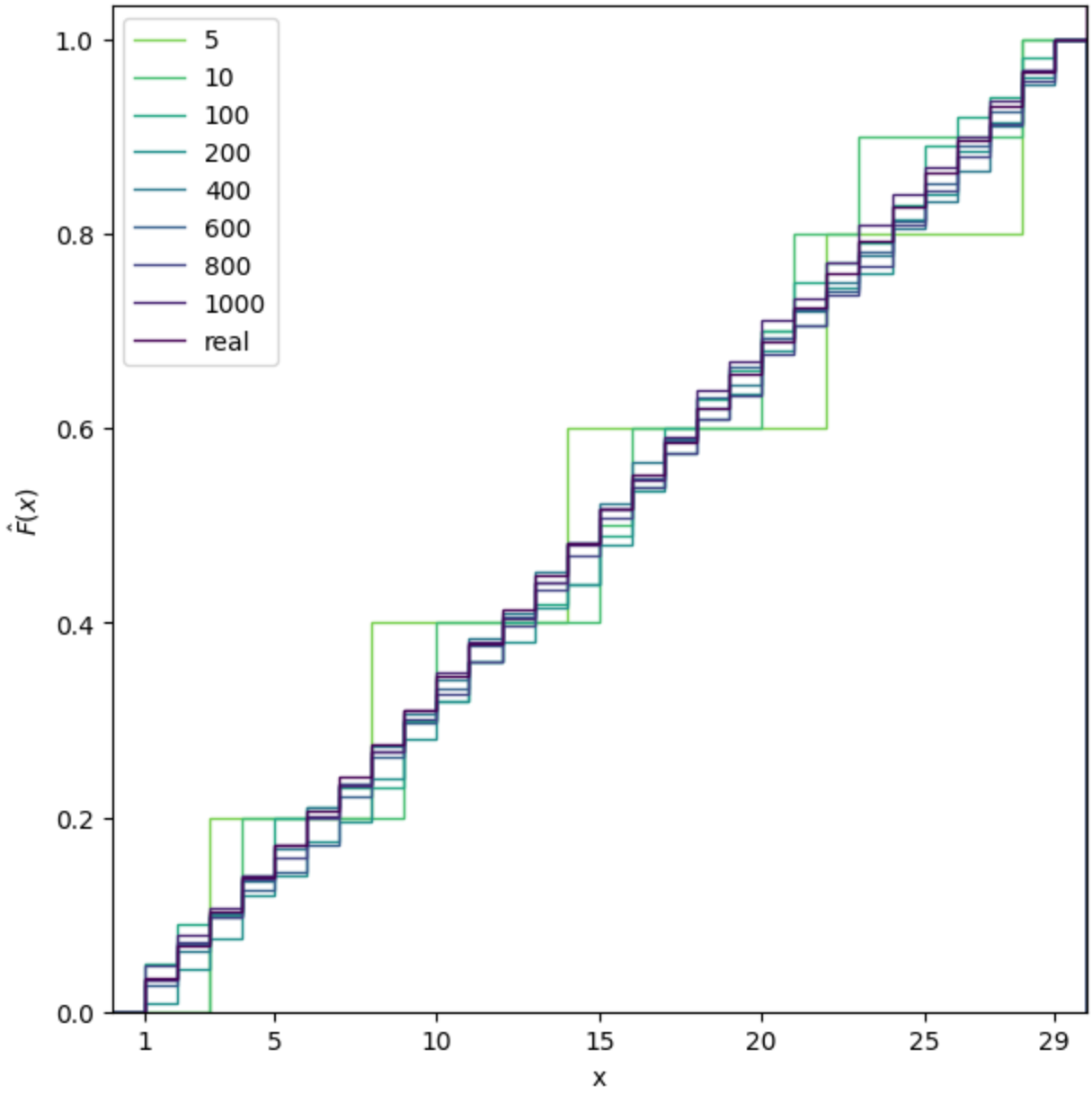
$$D_{m,n} = \sqrt{\frac{nm}{n+m}} \sup_{x \in \mathbb{R}} |F_n(x) - F_m(x)|$$

	5	10	100	200	400	600	800	1000
5	-	0.45	1.91	2.30	3.39	4.16	4.82	4.83
10	-	-	0.78	1.20	2.44	2.63	3.02	3.09
100	-	-	-	1.25	1.31	1.21	1.53	1.21
200	-	-	-	-	1.52	1.91	2.03	2.19
400	-	-	-	-	-	0.84	0.83	1.39
600	-	-	-	-	-	-	0.34	0.94
800	-	-	-	-	-	-	-	0.88
1000	-	-	-	-	-	-	-	-

In [8]:
```
# graph[3]
fig, ax = plt.subplots(2,1, figsize=(7,12), height_ratios = [2,1])
for i in range(8):
    ax[0].stairs(Yxi[i][3], np.append(X_realxi, 30), color = colors[i])
ax[0].stairs(Y_realxi, np.append(X_realxi, 30), color = '#440154')
ax[0].set(xticks = [1]+list(range(5,26,5))+[29], xmargin = 0, ymargin = 0,
          xlabel = 'x', ylabel = r'$\hat{F}(x)$',
          title = 'Дискретное равномерное, выборка 4 \n$\\xi , \\,\\theta$ = 29')
ax[0].legend([*n, 'real'], loc='upper left');

ax[1].table(cellText = diffsxi[3], rowLabels=n, colLabels=n,
            loc='center').scale(1, 1.5)
ax[1].set_axis_off()
ax[1].set_title(r'$D_{m,n}=\sqrt{\frac{nm}{n+m}}\sup_{x\in\mathbb{R}}$'+
                r'$|F_n(x)-F_m(x)|$');
```

Дискретное равномерное, выборка 4

ξ, $\theta = 29$



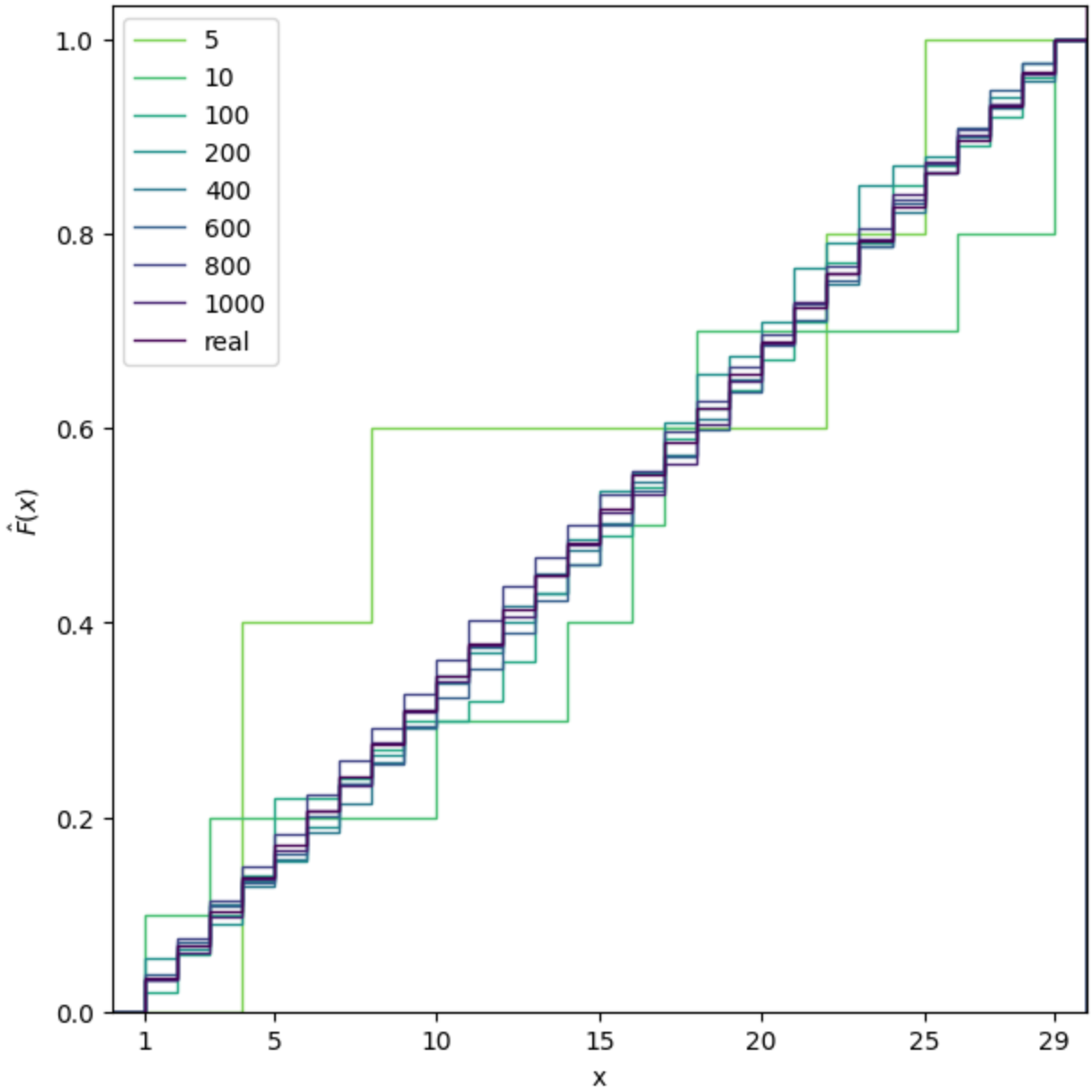
$$D_{m,n} = \sqrt{\frac{nm}{n+m}} \sup_{x \in \mathbb{R}} |F_n(x) - F_m(x)|$$

	5	10	100	200	400	600	800	1000
5	-	0.45	1.20	1.60	1.80	2.40	2.63	3.06
10	-	-	0.78	1.40	1.73	2.05	2.68	2.01
100	-	-	-	0.60	0.81	0.95	0.93	0.92
200	-	-	-	-	0.60	0.72	0.78	1.12
400	-	-	-	-	-	0.66	0.50	0.82
600	-	-	-	-	-	-	0.59	0.69
800	-	-	-	-	-	-	-	0.98
1000	-	-	-	-	-	-	-	-

In [9]:
```
# graph[4]
fig, ax = plt.subplots(2,1, figsize=(7,12), height_ratios = [2,1])
for i in range(8):
    ax[0].stairs(Yxi[i][4], np.append(X_realxi, 30), color = colors[i])
ax[0].stairs(Y_realxi, np.append(X_realxi, 30), color = '#440154')
ax[0].set(xticks = [1]+list(range(5,26,5))+[29], xmargin = 0, ymargin = 0,
          xlabel = 'x', ylabel = r'$\hat{F}(x)$',
          title = 'Дискретное равномерное, выборка 5 \n$\\xi , \\,\\theta$ = 29')
ax[0].legend([*n, 'real'], loc='upper left');

ax[1].table(cellText = diffsxi[4], rowLabels=n, colLabels=n,
            loc='center').scale(1, 1.5)
ax[1].set_axis_off()
ax[1].set_title(r'$D_{m,n}=\sqrt{\frac{nm}{n+m}}\sup_{x\in\mathbb{R}}$'+
                r'$|F_n(x)-F_m(x)|$');
```

Дискретное равномерное, выборка 5

ξ, $\theta = 29$



$$D_{m,n} = \sqrt{\frac{nm}{n+m}} \sup_{x \in \mathbb{R}} |F_n(x) - F_m(x)|$$

	5	10	100	200	400	600	800	1000
5	-	0.89	2.33	3.35	4.84	5.98	6.17	7.22
10	-	-	1.20	1.80	2.30	3.03	3.45	3.87
100	-	-	-	0.65	0.88	0.98	1.65	1.25
200	-	-	-	-	0.88	1.10	0.90	1.23
400	-	-	-	-	-	0.48	0.87	0.49
600	-	-	-	-	-	-	0.98	0.57
800	-	-	-	-	-	-	-	0.72
1000	-	-	-	-	-	-	-	-

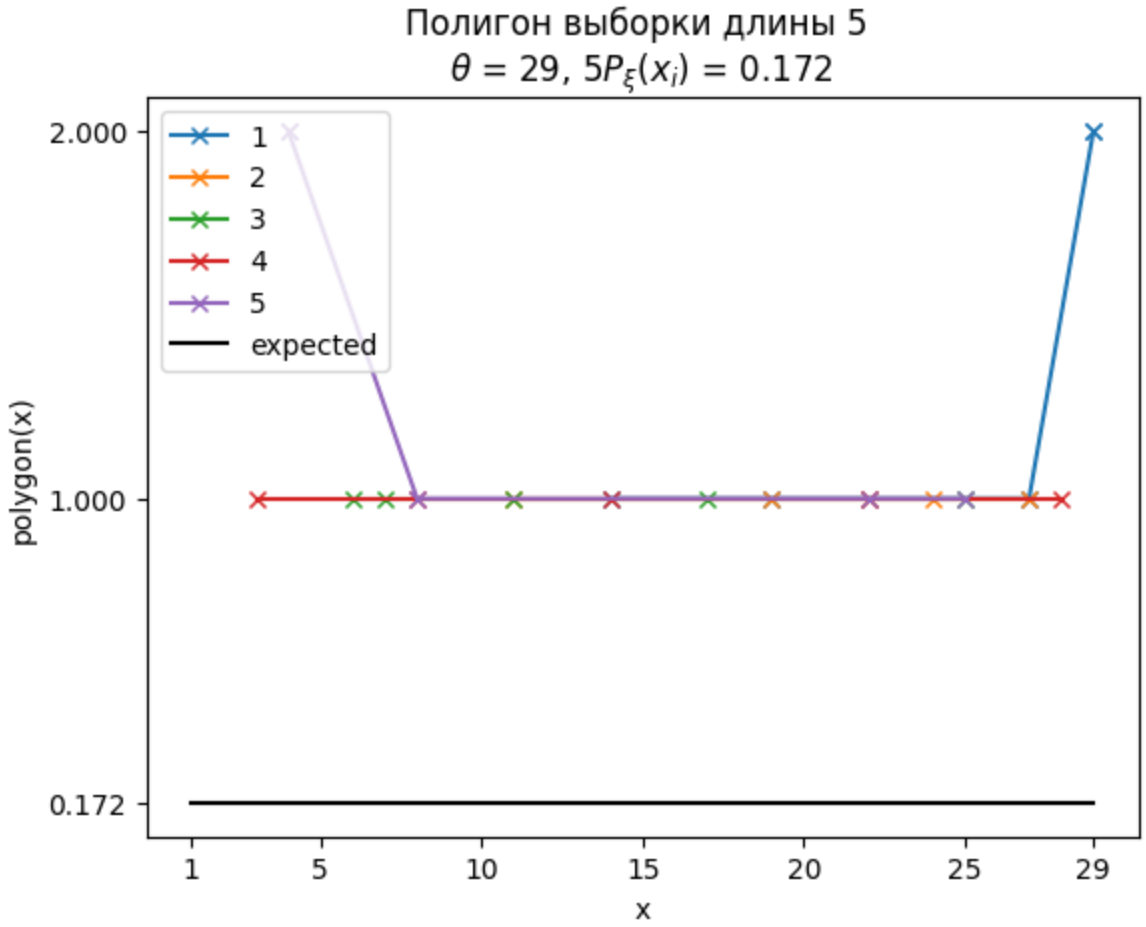
Д32Д, Задание 3

$\hat{F}(x) = \frac{1}{n}\sum_{i=1}^{n} I(x_i \le x) \Rightarrow P(x_i) = \hat{F}(x_i) - \hat{F}(x_{i-1}) = \frac{1}{n} k\, I(x_i = x_i) = \frac{k}{n}$, где k - частота встречаемости элемента в выборке.

$\Rightarrow$ при $n \to \infty$ $k \to nP(x_i)$, что значит, что график вероятности подводим к полигону частот через домножение на количество элементов в выборке.

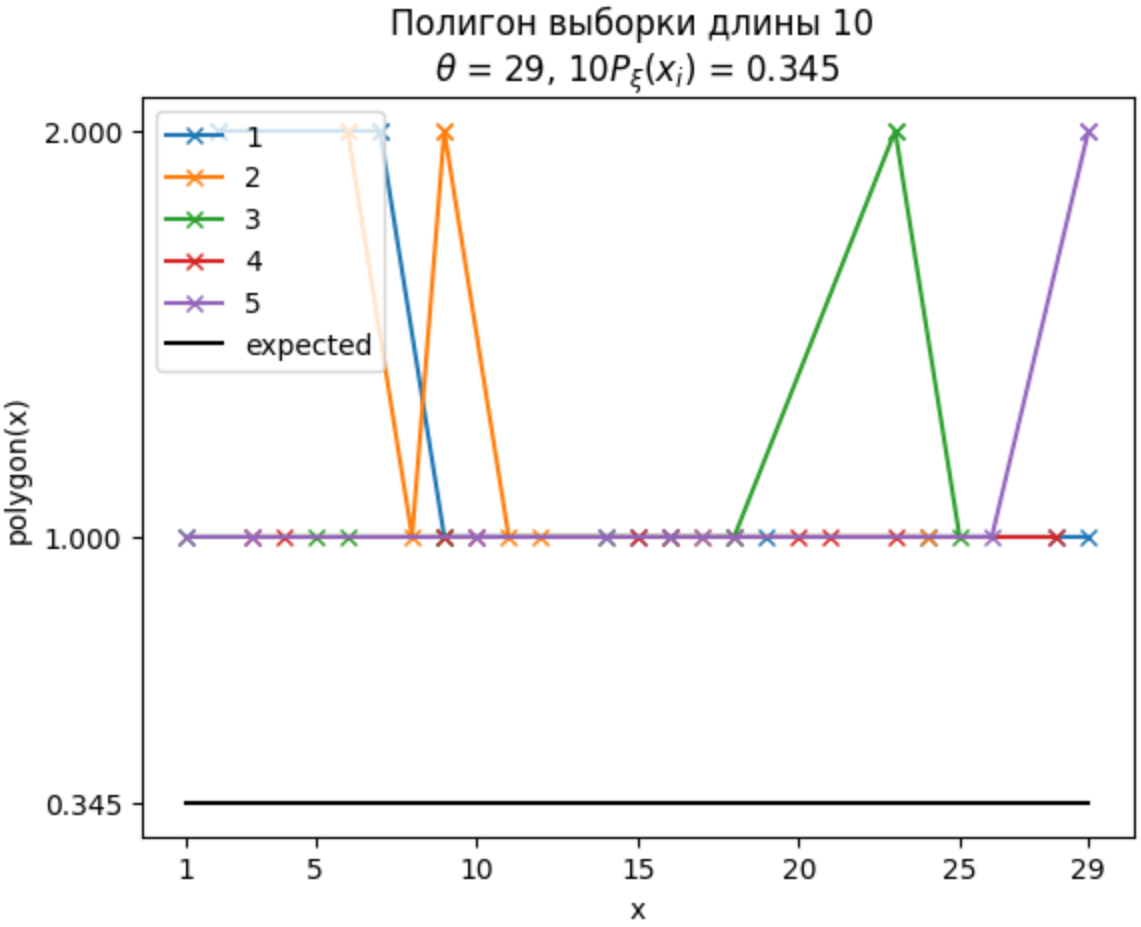
In [10]:
```
def xi_pilygon(sample, x):
    return np.count_nonzero(sample==x)

def min_t(samples):
    res = samples[0][0]
    for i in samples:
        t = min(i)
        if t < res: res = t
    return res

def max_t(samples):
    res = samples[0][0]
    for i in samples:
        t = max(i)
        if t > res: res = t
    return res

Y_polxi = [[np.array([xi_pilygon(sample_xi[k][j], sample_xi[k][j][i])
                      for i in range(n[k])])
            for j in range(5)] for k in range(len(n))]

y_limsxi = np.array([[min_t(j), max_t(j)] for j in Y_polxi])

posibilityxi = np.full((1000), 1/29)
```

In [11]:
```
# n=5
for i in range(5):
    plt.plot(sample_xi[0][i], Y_polxi[0][i], '-x')
plt.plot(np.linspace(1, 29, 1000), posibilityxi*5, 'k')
plt.xticks([1]+list(range(5,26,5))+[29])
plt.yticks(np.append(np.arange(y_limsxi[0,0], y_limsxi[0,1]+1), 5/29))
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 5 \n' +
          '$\\theta$ = 29, $5P_{\\xi}(x_i)$ = %.3f'% (5/29))
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```

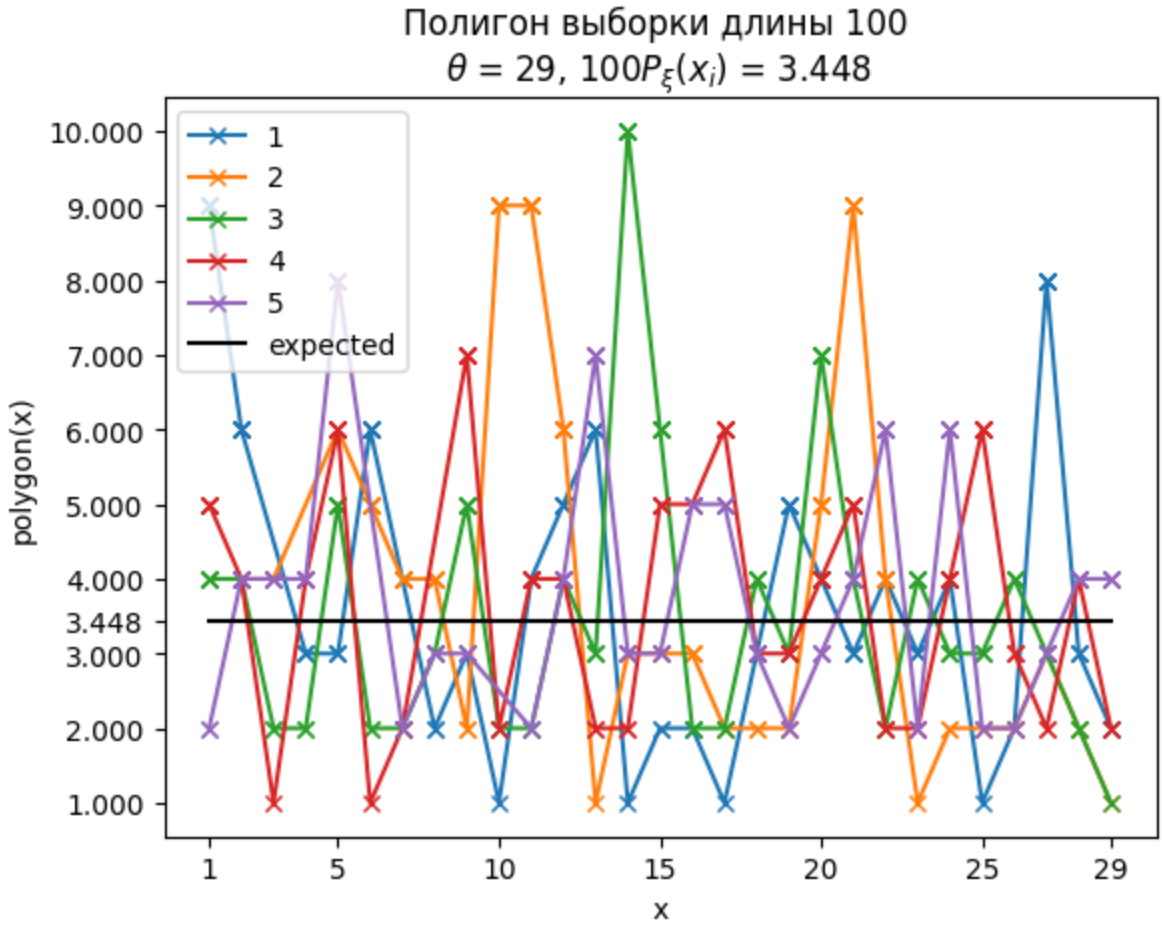


In [12]:

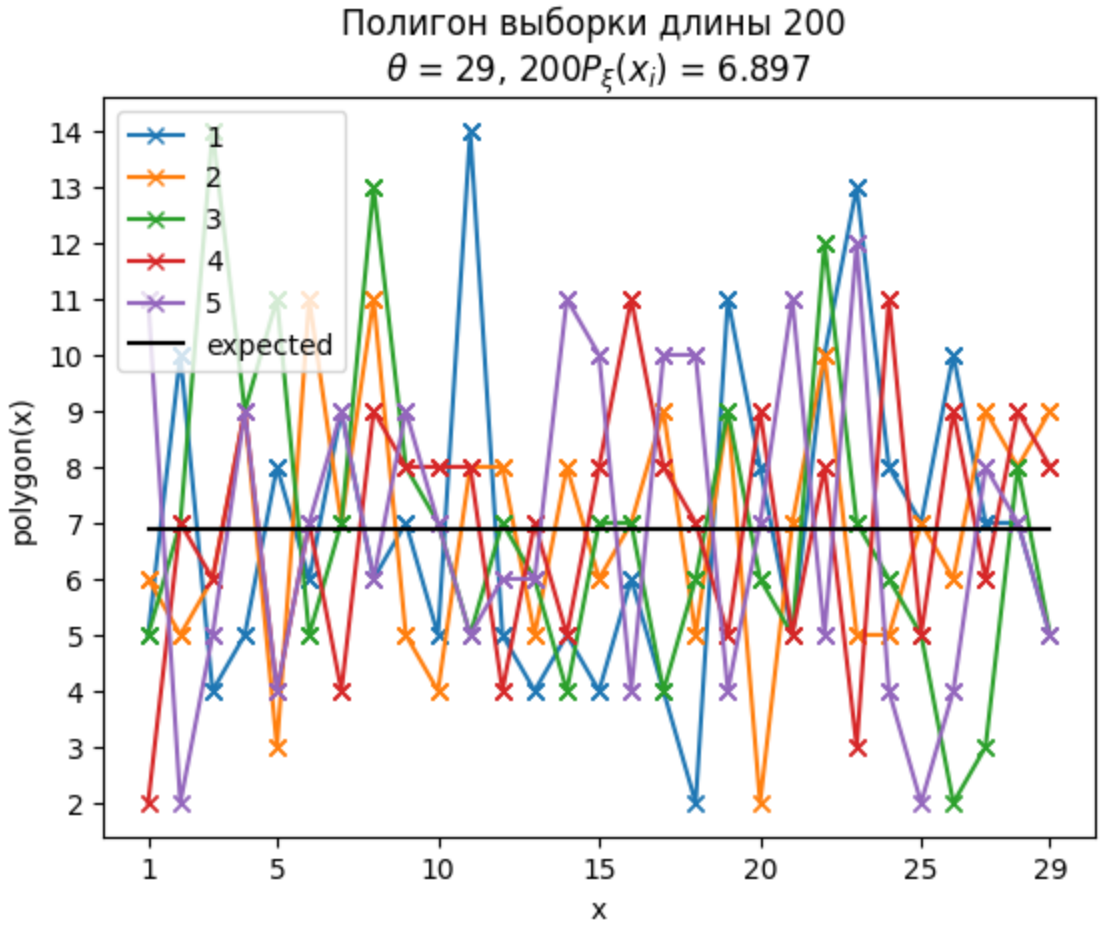
```
# n=10
for i in range(5):
    plt.plot(sample_xi[1][i], Y_polxi[1][i], '-x')
plt.plot(np.linspace(1, 29, 1000), posibilityxi*10, 'k')
plt.xticks([1]+list(range(5,26,5))+[29])
plt.yticks(np.append(np.arange(y_limsxi[1,0], y_limsxi[1,1]+1), 10/29))
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 10 \n' +
          '$\\theta$ = 29, $10P_{\\xi}(x_i)$ = %.3f'% (10/29))
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



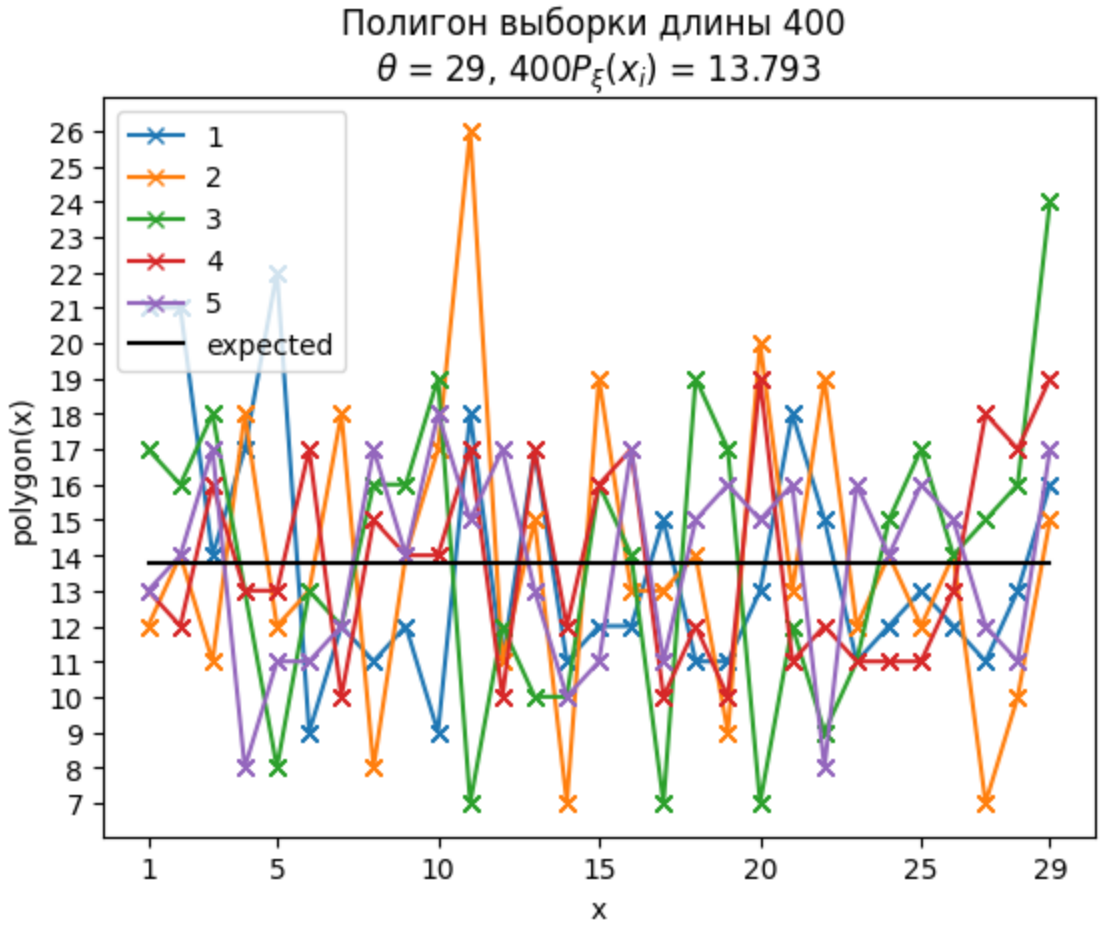
In [13]:
```
# n=100
for i in range(5):
    plt.plot(sample_xi[2][i], Y_polxi[2][i], '-x')
plt.plot(np.linspace(1, 29, 1000), posibilityxi*100, 'k')
plt.xticks([1]+list(range(5,26,5))+[29])
plt.yticks(np.append(np.arange(y_limsxi[2,0], y_limsxi[2,1]+1), 100/29))
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 100 \n' +
          '$\\theta$ = 29, $100P_{\\xi}(x_i)$ = %.3f'% (100/29))
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



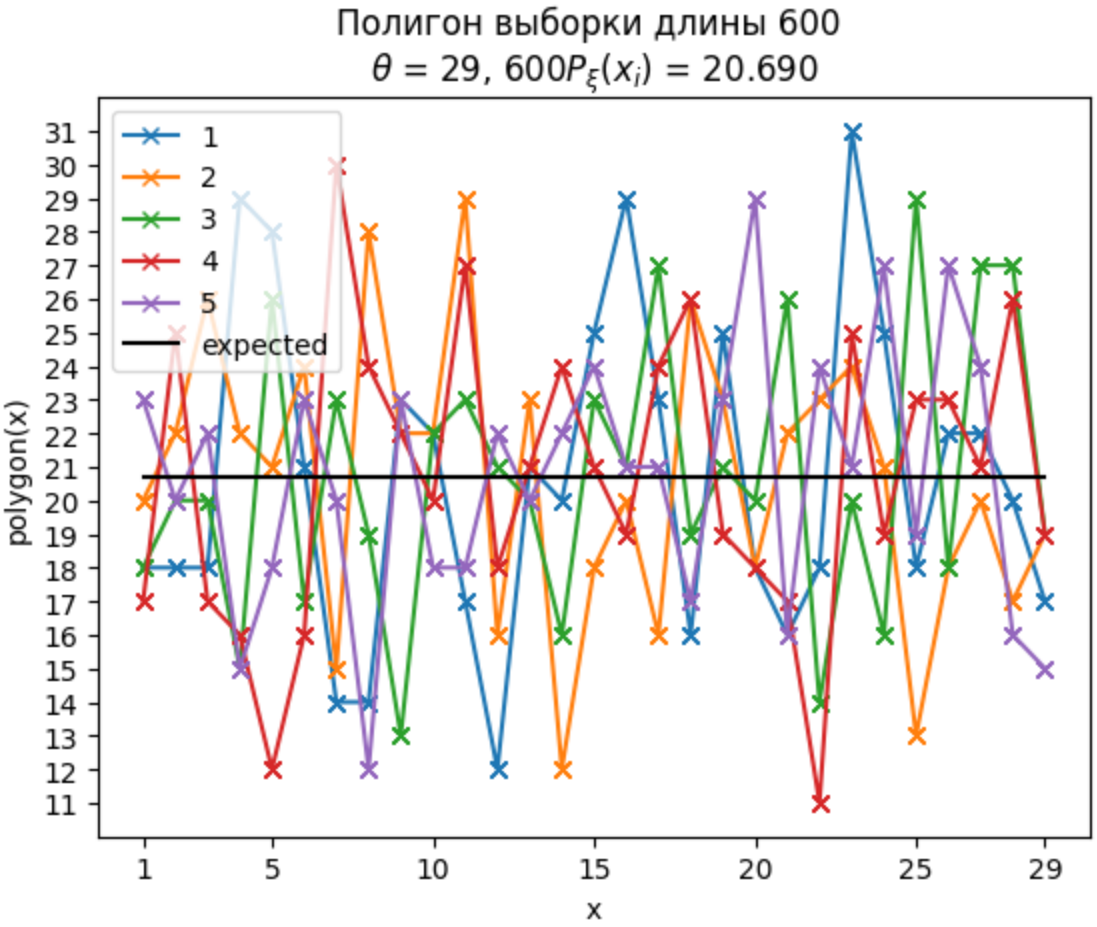
In [14]:
```
# n=200
for i in range(5):
    plt.plot(sample_xi[3][i], Y_polxi[3][i], '-x')
plt.plot(np.linspace(1, 29, 1000), posibilityxi*200, 'k')
plt.xticks([1]+list(range(5,26,5))+[29])
plt.yticks(np.arange(y_limsxi[3,0], y_limsxi[3,1]+1))
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 200 \n' +
          '$\\theta$ = 29, $200P_{\\xi}(x_i)$ = %.3f'% (200/29))
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



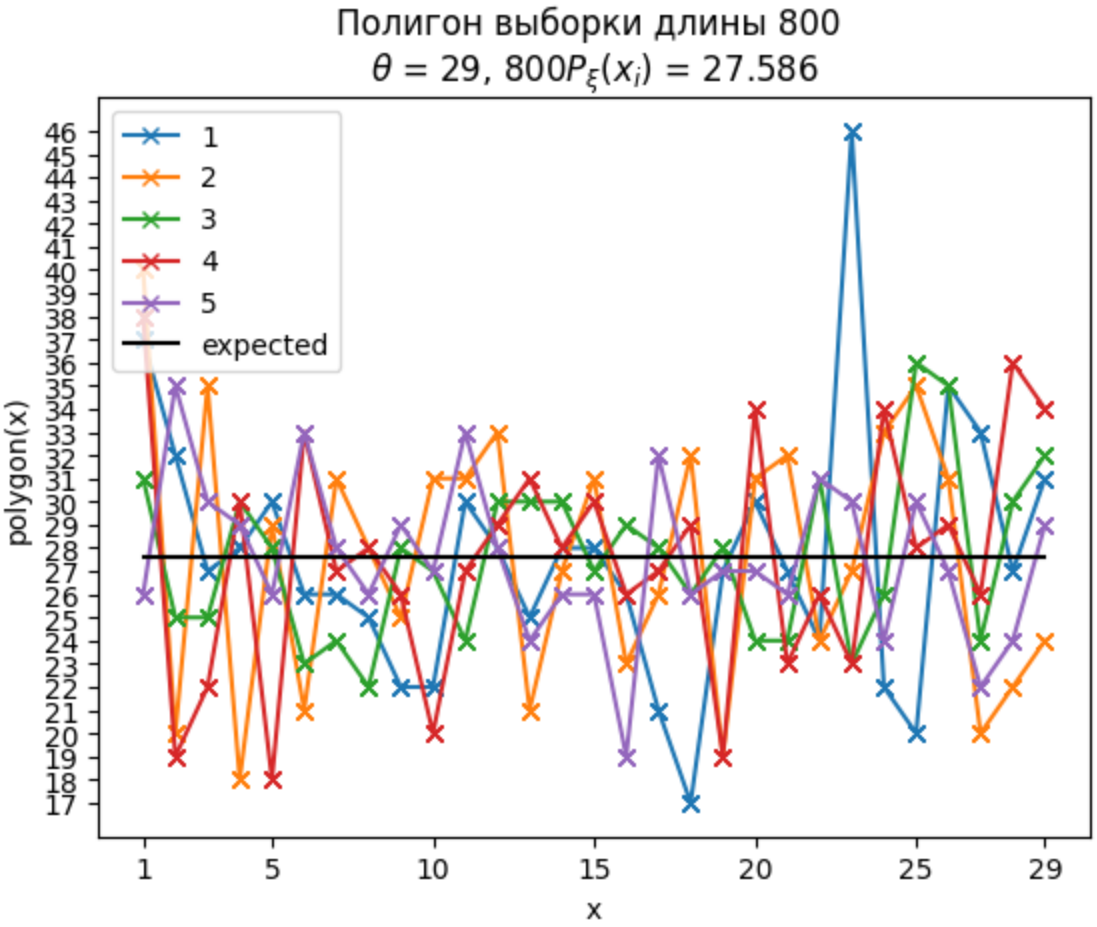
In [15]:
```
# n=400
for i in range(5):
    plt.plot(sample_xi[4][i], Y_polxi[4][i], '-x')
plt.plot(np.linspace(1, 29, 1000), posibilityxi*400, 'k')
plt.xticks([1]+list(range(5,26,5))+[29])
plt.yticks(np.arange(y_limsxi[4,0], y_limsxi[4,1]+1))
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 400 \n' +
          '$\\theta$ = 29, $400P_{\\xi}(x_i)$ = %.3f'% (400/29))
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



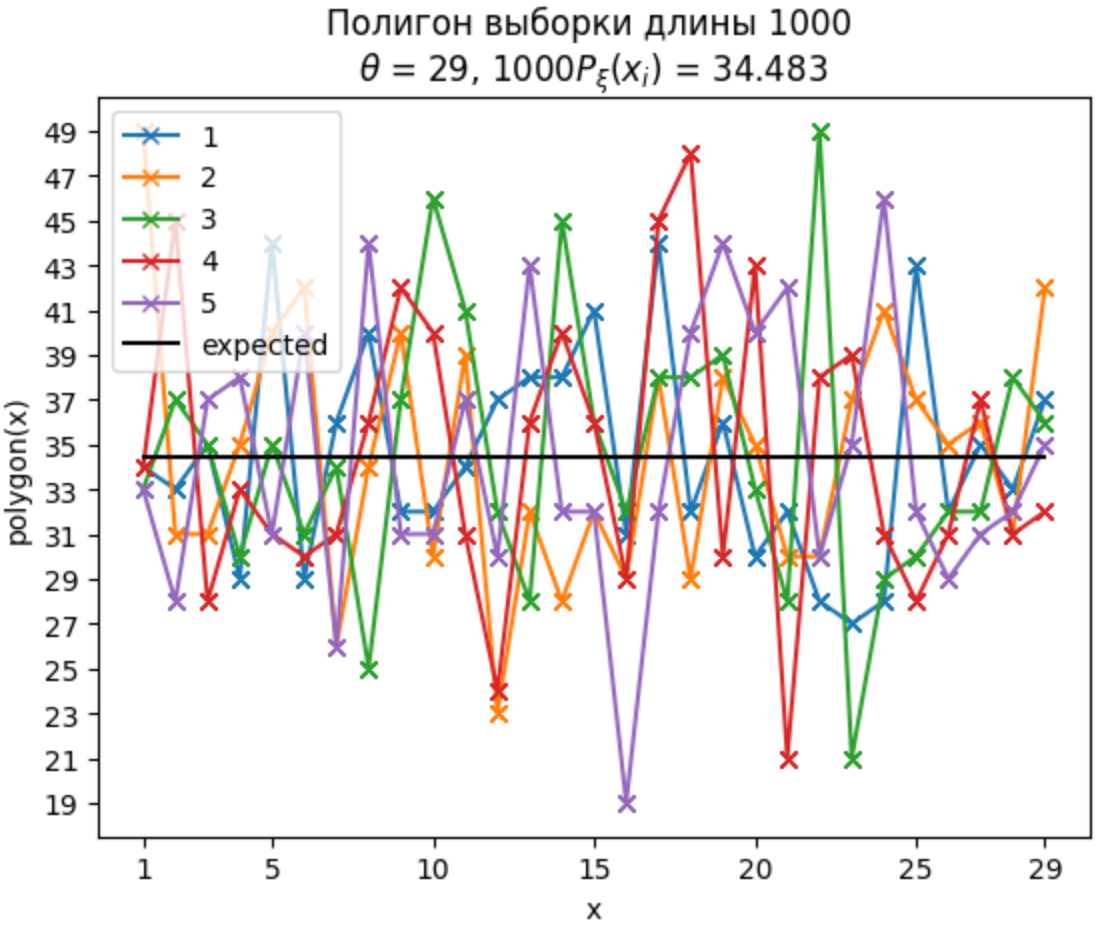
In [16]:
```
# n=600
for i in range(5):
    plt.plot(sample_xi[5][i], Y_polxi[5][i], '-x')
plt.plot(np.linspace(1, 29, 1000), posibilityxi*600, 'k')
plt.xticks([1]+list(range(5,26,5))+[29])
plt.yticks(np.arange(y_limsxi[5,0], y_limsxi[5,1]+1))
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 600 \n' +
          '$\\theta$ = 29, $600P_{\\xi}(x_i)$ = %.3f'% (600/29))
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



In [17]:
```
# n=800
for i in range(5):
    plt.plot(sample_xi[6][i], Y_polxi[6][i], '-x')
plt.plot(np.linspace(1, 29, 1000), posibilityxi*800, 'k')
plt.xticks([1]+list(range(5,26,5))+[29])
plt.yticks(np.arange(y_limsxi[6,0], y_limsxi[6,1]+1))
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 800 \n' +
          '$\\theta$ = 29, $800P_{\\xi}(x_i)$ = %.3f'% (800/29))
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



In [18]:
```
# n=1000
for i in range(5):
    plt.plot(sample_xi[7][i], Y_polxi[7][i], '-x')
plt.plot(np.linspace(1, 29, 1000), posibilityxi*1000, 'k')
plt.xticks([1]+list(range(5,26,5))+[29])
plt.yticks(np.arange(y_limsxi[7,0], y_limsxi[7,1]+1, 2))
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 1000 \n' +
          '$\\theta$ = 29, $1000P_{\\xi}(x_i)$ = %.3f'% (1000/29))
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



По полигону частот и графику домноженной вероятности можно заметить, что встречаемость возможных значений распределения держится вокруг вероятности каждого из них, и с увеличением числа элементов выборки эту закономерность наблюдать становится легче. Это наблюдение соответствует теореме о схождении эмперической вероятности к математической (в курсе лекций это теорема о функциях распределения, но вероятность из этого следует). Конечно, у нас мог произойти случай, когда все элементы выборки попали в 1, но шанс этого в силу построения равен 29^{-1000}, так что дополнительный разброс в виде 5 выборок на каждое указанное количество элементов создает общую картину, которая со стремлением n к бесконечности, устремит полигон частот к домноженному графику вероятности. Также следует заметить, что на графиках в 5 и 10 элементов выборки домноженная вероятность даже не близка к частотам. Это, во-первых, еще раз подтверждает теорему, а во-вторых, домноженная вероятность сильно меньше единицы, что еще больше мешает приближению.

Д32Д, Задание 4

In [19]:
```
def xi_sample_mean(sample):
    return sum(sample)/len(sample)

def xi_sample_variance(sample):
    return sum((sample-xi_sample_mean(sample))**2)/len(sample)

meansxi = np.array([[xi_sample_mean(sample_xi[k][j])for j in range(5)]
                    for k in range(len(n))])
variancesxi = np.array([[xi_sample_variance(sample_xi[k][j])for j in range(5)]
                        for k in range(len(n))])
means_pxi = np.array([['%.2f' % i for i in j] for j in meansxi])
variances_pxi = np.array([['%.2f' % i for i in j] for j in variancesxi])
expectationxi = (29+1)/2
variancexi = (29*29-1)/12
means_difxi = np.array([['%.2f' % i for i in j] for j in (meansxi-expectationxi)])
variances_difxi = np.array([['%.2f' % i for i in j]
                            for j in (variancesxi-variancexi)])
```

Для начала следует продемонстрировать выборочные средние и выборочные дисперсии, чтобы дать большее придставление о числах, с которыми идет работа. Благо их не так много.

In [20]:
```
fig, ax = plt.subplots(2,1, figsize=(7,6.5))
ax[0].table(cellText = means_pxi, rowLabels=n, colLabels=[1,2,3,4,5],
            loc='center').scale(1, 1.5)
ax[0].set_axis_off()
ax[0].set_title('Выборочные средние');
ax[1].table(cellText = variances_pxi, rowLabels=n, colLabels=[1,2,3,4,5],
            loc='center').scale(1, 1.5)
ax[1].set_axis_off()
ax[1].set_title('Выборочные дисперсии');
```

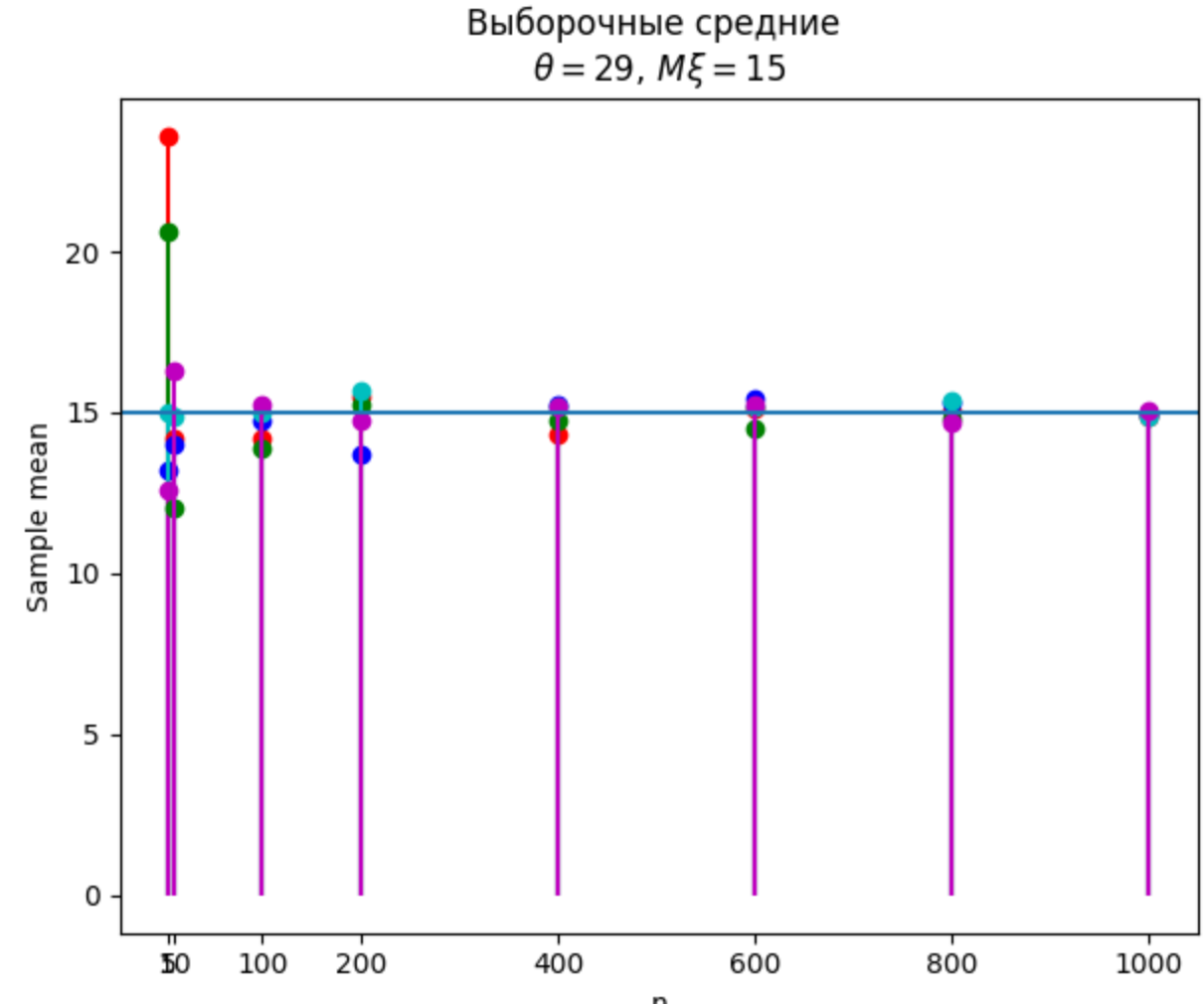
Выборочные средние

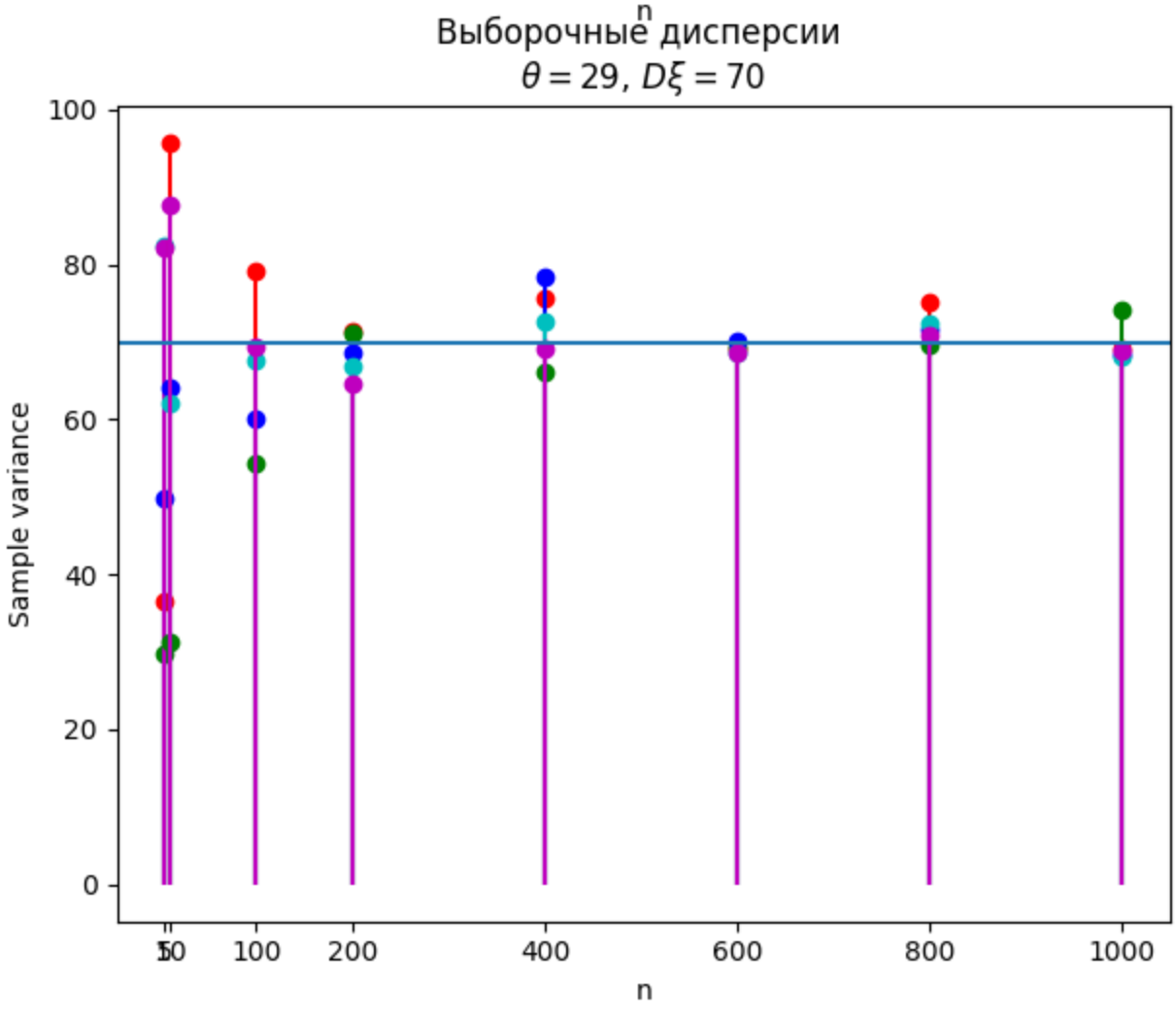
	1	2	3	4	5
5	23.60	20.60	13.20	15.00	12.60
10	14.20	12.00	14.00	14.90	16.30
100	14.20	13.91	14.77	15.02	15.25
200	15.52	15.26	13.68	15.66	14.77
400	14.29	14.72	15.25	15.15	15.20
600	15.13	14.53	15.43	15.21	15.25
800	15.04	14.84	15.30	15.34	14.69
1000	14.89	14.97	14.94	14.90	15.08

Выборочные дисперсии

	1	2	3	4	5
5	36.64	29.84	49.76	82.40	82.24
10	95.76	31.20	64.20	62.09	87.61
100	79.10	54.32	60.02	67.58	69.49
200	71.34	71.17	68.60	66.98	64.51
400	75.74	66.06	78.29	72.58	69.01
600	69.22	69.31	70.22	68.53	68.55
800	75.10	69.66	71.55	72.37	70.88
1000	69.19	74.20	68.42	68.18	68.84

In [21]:
```
fig, ax = plt.subplots(2,1, figsize=(7,12))
for k in range(len(n)):
    for j in range(5):
        ax[0].stem(n[k], meansxi[k][j], 'rgbcm'[j])
ax[0].axhline(expectationxi)
ax[0].set(xticks = n, xlabel = 'n', ylabel = 'Sample mean',
          title = 'Выборочные средние \n' +
                  '$\\theta = 29, \\,M\\xi = %d$'%expectationxi)

for k in range(len(n)):
    for j in range(5):
        ax[1].stem(n[k], variancesxi[k][j], 'rgbcm'[j])
ax[1].axhline(y = variancexi)
ax[1].set(xticks = n, xlabel = 'n', ylabel = 'Sample variance',
          title = 'Выборочные дисперсии \n' +
                  '$\\theta = 29, \\,D\\xi = %d$'%variancexi);
```


Выборочные средние
$\theta = 29$, $M\xi = 15$
Sample mean
0
5
10
15
20
1
50
100
200
400
600
800
1000
n


Выборочные дисперсии
$\theta = 29$, $D\xi = 70$
Sample variance
0
20
40
60
80
100
1
50
100
200
400
600
800
1000
n

По графикам можно заметить, что с увеличением количества элементов выборки и математическое ожидание, и дисперсия сходятся к предполагаемому значению. Что еще раз подтверждает теорему, упомянутую в предыдущей задаче.

In [22]:
```
fig, ax = plt.subplots(2,1, figsize=(7,7))
ax[0].table(cellText = means_difxi, rowLabels=n, colLabels=[1,2,3,4,5],
            loc='center').scale(1, 1.5)
ax[0].set_axis_off()
ax[0].set_title('Разница выборочного среднего и математического ожидания' +
                '\n $M\\xi = $'+str(expectationxi));

ax[1].table(cellText = variances_difxi, rowLabels=n, colLabels=[1,2,3,4,5],
            loc='center').scale(1, 1.5)
ax[1].set_axis_off()
ax[1].set_title('Разница выборочной дисперсии и дисперсии' +
                '\n $D\\xi = $'+str(variancexi));
```

Разница выборочного среднего и математического ожидания
$M\xi = 15.0$

	1	2	3	4	5
5	8.60	5.60	-1.80	0.00	-2.40
10	-0.80	-3.00	-1.00	-0.10	1.30
100	-0.80	-1.09	-0.23	0.02	0.25
200	0.52	0.26	-1.32	0.66	-0.23
400	-0.71	-0.28	0.25	0.15	0.20
600	0.13	-0.47	0.43	0.21	0.25
800	0.04	-0.16	0.30	0.34	-0.31
1000	-0.11	-0.03	-0.06	-0.10	0.08

Разница выборочной дисперсии и дисперсии
$D\xi = 70.0$

	1	2	3	4	5
5	-33.36	-40.16	-20.24	12.40	12.24
10	25.76	-38.80	-5.80	-7.91	17.61
100	9.10	-15.68	-9.98	-2.42	-0.51
200	1.34	1.17	-1.40	-3.02	-5.49
400	5.74	-3.94	8.29	2.58	-0.99
600	-0.78	-0.69	0.22	-1.47	-1.45
800	5.10	-0.34	1.55	2.37	0.88
1000	-0.81	4.20	-1.58	-1.82	-1.16

Эти таблицы (как и предыдущее много чего) еще раз демонстрируют справедливость теоремы.

Д32, Абсолютно непрерывное

Большая часть материала уже разобрана выше на примере дискретного случая. Поэтому здесь различных описаний будет меньше, так как они будут почти 1 в 1 повторяться. Это самое почти, при необходимости, и будет описываться.

Д32А, Задание 1

In [23]:
```
sample_eta = [[np.sort(np.array([generate_eta() for i in range(j)]))
               for i in range(5)] for j in n]

demo_peta = np.array(['%.3f'%i for i in sample_eta[7][0]]).reshape((-1, 20))
for i in demo_peta:
    print(*i, sep = ' ')
```

0.017 0.029 0.030 0.040 0.041 0.051 0.051 0.052 0.056 0.058 0.059 0.062 0.068 0.
068 0.069 0.071 0.074 0.074 0.077 0.079
0.084 0.086 0.086 0.088 0.088 0.095 0.097 0.099 0.100 0.105 0.105 0.105 0.109 0.
109 0.113 0.114 0.117 0.118 0.119 0.119
0.120 0.124 0.125 0.125 0.127 0.130 0.131 0.133 0.134 0.137 0.138 0.140 0.141 0.
141 0.141 0.143 0.143 0.144 0.145 0.145
0.149 0.150 0.151 0.151 0.151 0.151 0.152 0.156 0.157 0.157 0.159 0.159 0.160 0.
160 0.161 0.161 0.162 0.162 0.162 0.166
0.166 0.169 0.170 0.171 0.172 0.172 0.175 0.176 0.177 0.178 0.179 0.180 0.180 0.
181 0.182 0.182 0.183 0.183 0.183 0.183
0.184 0.188 0.195 0.196 0.196 0.196 0.198 0.199 0.200 0.201 0.203 0.204 0.206 0.
206 0.207 0.207 0.207 0.208 0.208 0.209
0.209 0.210 0.211 0.211 0.212 0.212 0.214 0.214 0.216 0.217 0.217 0.218 0.218 0.
221 0.221 0.221 0.222 0.223 0.223 0.226
0.227 0.227 0.227 0.230 0.230 0.231 0.233 0.234 0.235 0.235 0.236 0.236 0.236 0.
237 0.238 0.240 0.243 0.243 0.244 0.245
0.248 0.250 0.251 0.251 0.253 0.255 0.256 0.257 0.258 0.258 0.259 0.259 0.260 0.
260 0.261 0.261 0.261 0.262 0.263 0.263
0.265 0.265 0.266 0.266 0.267 0.267 0.267 0.267 0.268 0.268 0.269 0.269 0.271 0.
271 0.273 0.273 0.273 0.273 0.274 0.275
0.276 0.277 0.278 0.279 0.281 0.281 0.281 0.281 0.282 0.283 0.284 0.284 0.285 0.
286 0.286 0.286 0.288 0.288 0.289 0.289
0.289 0.289 0.290 0.292 0.292 0.292 0.293 0.294 0.295 0.295 0.295 0.296 0.298 0.
298 0.298 0.300 0.301 0.301 0.302 0.303
0.304 0.304 0.304 0.306 0.307 0.307 0.308 0.308 0.309 0.311 0.312 0.315 0.316 0.
316 0.316 0.316 0.316 0.317 0.317 0.317
0.320 0.321 0.321 0.322 0.322 0.322 0.322 0.323 0.324 0.325 0.325 0.326 0.326 0.
327 0.327 0.327 0.327 0.328 0.328 0.328
0.328 0.329 0.329 0.329 0.330 0.333 0.333 0.334 0.335 0.337 0.337 0.338 0.338 0.
339 0.339 0.339 0.339 0.339 0.341 0.341
0.342 0.342 0.344 0.344 0.344 0.346 0.347 0.347 0.348 0.350 0.351 0.352 0.353 0.
356 0.356 0.356 0.357 0.357 0.359 0.360
0.360 0.360 0.361 0.363 0.363 0.364 0.365 0.366 0.366 0.366 0.368 0.368 0.369 0.
369 0.369 0.370 0.371 0.372 0.372 0.373
0.373 0.375 0.377 0.377 0.378 0.378 0.378 0.379 0.380 0.380 0.380 0.381 0.383 0.
383 0.384 0.385 0.385 0.386 0.387 0.388
0.389 0.389 0.394 0.394 0.395 0.395 0.395 0.396 0.397 0.397 0.398 0.399 0.400 0.
400 0.400 0.401 0.403 0.403 0.403 0.404
0.404 0.405 0.407 0.408 0.408 0.408 0.408 0.409 0.409 0.410 0.410 0.410 0.411 0.
412 0.412 0.412 0.413 0.414 0.414 0.414
0.416 0.416 0.416 0.416 0.416 0.416 0.417 0.418 0.419 0.420 0.420 0.421 0.422 0.
422 0.423 0.423 0.423 0.423 0.424 0.426
0.426 0.426 0.426 0.427 0.429 0.430 0.432 0.433 0.434 0.435 0.435 0.435 0.436 0.
436 0.437 0.437 0.438 0.438 0.439 0.440
0.440 0.440 0.441 0.441 0.441 0.442 0.443 0.443 0.443 0.443 0.444 0.445 0.446 0.
447 0.447 0.447 0.448 0.448 0.449 0.449
0.449 0.449 0.451 0.452 0.453 0.453 0.454 0.455 0.455 0.457 0.458 0.458 0.458 0.
459 0.460 0.460 0.460 0.461 0.461 0.461
0.461 0.461 0.462 0.464 0.466 0.466 0.467 0.468 0.468 0.469 0.470 0.470 0.470 0.
471 0.471 0.471 0.471 0.472 0.472 0.472
0.473 0.473 0.473 0.474 0.474 0.474 0.475 0.475 0.475 0.475 0.475 0.476 0.476 0.
477 0.477 0.478 0.479 0.479 0.480 0.480
0.480 0.481 0.481 0.481 0.481 0.482 0.483 0.483 0.484 0.484 0.484 0.485 0.486 0.
487 0.487 0.488 0.488 0.489 0.490 0.491
0.493 0.494 0.494 0.494 0.495 0.495 0.495 0.495 0.496 0.497 0.497 0.498 0.498 0.
499 0.500 0.500 0.501 0.501 0.501 0.502
0.502 0.503 0.503 0.504 0.505 0.505 0.506 0.506 0.507 0.507 0.508 0.509 0.510 0.
510 0.510 0.510 0.512 0.513 0.513 0.513
0.513 0.514 0.514 0.515 0.515 0.516 0.518 0.518 0.519 0.520 0.522 0.523 0.524 0.
524 0.524 0.526 0.526 0.527 0.528 0.528
0.528 0.528 0.528 0.529 0.529 0.529 0.530 0.530 0.530 0.532 0.533 0.534 0.535 0.

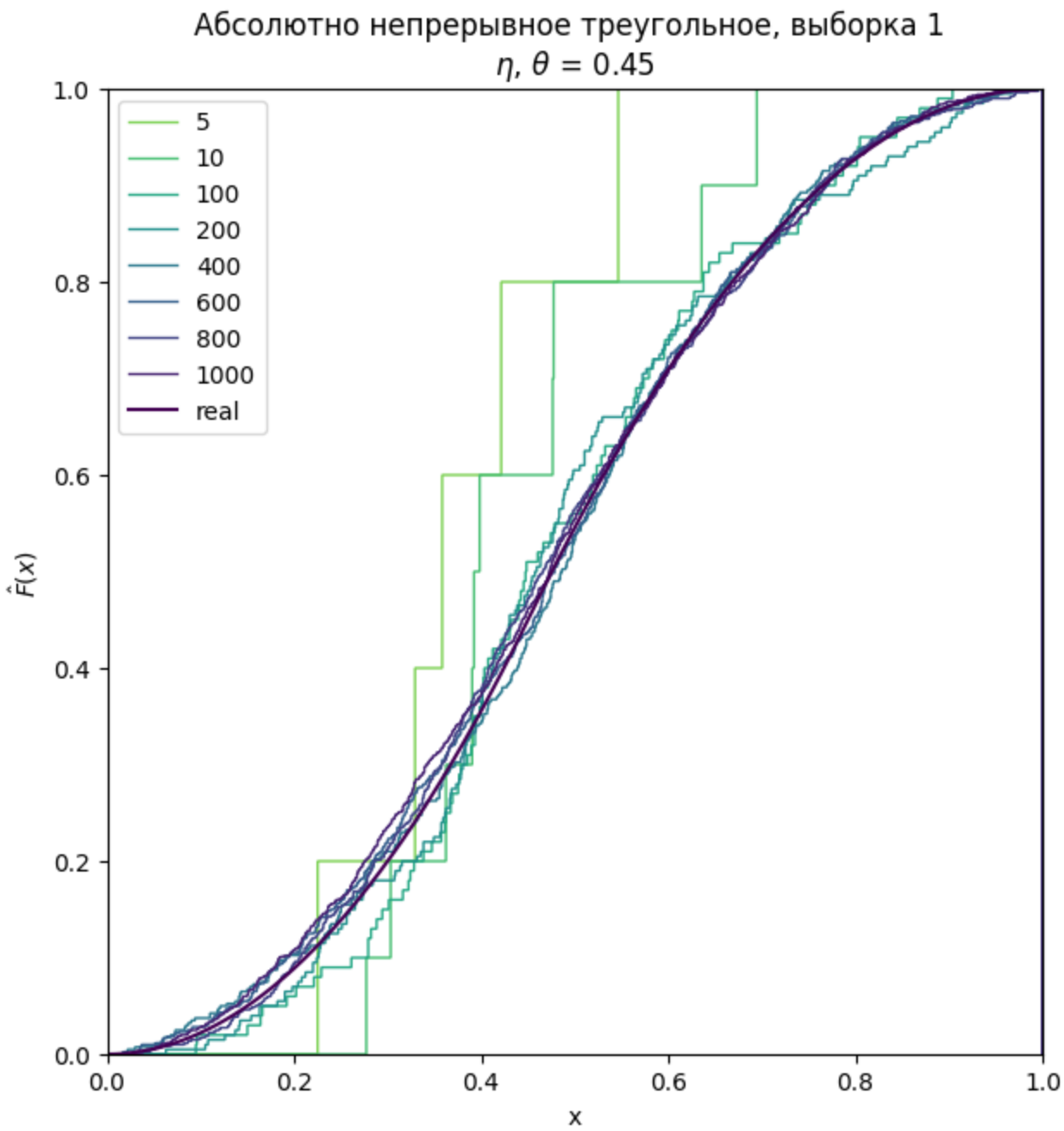
536 0.536 0.536 0.536 0.536 0.537 0.537
0.538 0.539 0.539 0.540 0.540 0.540 0.541 0.542 0.543 0.543 0.544 0.547 0.548 0.
548 0.548 0.548 0.549 0.549 0.549 0.550
0.550 0.551 0.552 0.552 0.552 0.552 0.555 0.555 0.556 0.558 0.559 0.559 0.560 0.
561 0.562 0.562 0.564 0.564 0.565 0.567
0.568 0.568 0.568 0.569 0.570 0.570 0.572 0.574 0.575 0.576 0.578 0.578 0.578 0.
579 0.580 0.580 0.581 0.581 0.581 0.583
0.584 0.584 0.586 0.586 0.586 0.587 0.588 0.588 0.588 0.589 0.590 0.591 0.592 0.
593 0.593 0.594 0.595 0.595 0.595 0.596
0.596 0.596 0.597 0.598 0.599 0.599 0.599 0.600 0.600 0.600 0.602 0.605 0.606 0.
606 0.606 0.607 0.608 0.608 0.608 0.608
0.609 0.611 0.612 0.612 0.612 0.613 0.615 0.615 0.615 0.616 0.616 0.617 0.617 0.
617 0.618 0.621 0.621 0.621 0.622 0.626
0.626 0.627 0.628 0.628 0.630 0.630 0.633 0.633 0.633 0.635 0.635 0.636 0.637 0.
638 0.639 0.639 0.639 0.640 0.641 0.642
0.643 0.644 0.644 0.644 0.647 0.648 0.648 0.649 0.649 0.649 0.649 0.650 0.650 0.
650 0.651 0.652 0.656 0.658 0.658 0.658
0.659 0.659 0.660 0.660 0.661 0.661 0.661 0.662 0.664 0.664 0.665 0.665 0.665 0.
666 0.666 0.670 0.670 0.670 0.671 0.672
0.672 0.675 0.676 0.676 0.676 0.679 0.680 0.682 0.683 0.684 0.685 0.686 0.687 0.
687 0.688 0.690 0.691 0.692 0.693 0.693
0.694 0.694 0.695 0.696 0.697 0.700 0.700 0.701 0.702 0.702 0.702 0.703 0.703 0.
704 0.705 0.706 0.707 0.709 0.709 0.709
0.710 0.712 0.712 0.713 0.713 0.713 0.714 0.716 0.717 0.717 0.718 0.719 0.719 0.
723 0.724 0.725 0.728 0.730 0.732 0.735
0.736 0.736 0.738 0.738 0.739 0.739 0.740 0.742 0.743 0.743 0.744 0.748 0.748 0.
749 0.749 0.750 0.750 0.751 0.752 0.752
0.752 0.754 0.754 0.755 0.755 0.757 0.757 0.757 0.758 0.759 0.760 0.761 0.763 0.
764 0.765 0.765 0.765 0.766 0.766 0.767
0.768 0.770 0.771 0.771 0.771 0.771 0.772 0.773 0.774 0.779 0.779 0.780 0.781 0.
781 0.782 0.782 0.784 0.785 0.786 0.788
0.788 0.789 0.789 0.790 0.791 0.791 0.795 0.800 0.801 0.801 0.803 0.805 0.809 0.
809 0.810 0.811 0.812 0.813 0.814 0.815
0.815 0.816 0.817 0.817 0.822 0.826 0.827 0.828 0.828 0.830 0.832 0.832 0.833 0.
834 0.837 0.840 0.841 0.842 0.843 0.852
0.852 0.855 0.856 0.856 0.859 0.859 0.862 0.868 0.870 0.870 0.870 0.870 0.871 0.
872 0.874 0.874 0.878 0.879 0.880 0.880
0.881 0.886 0.888 0.889 0.894 0.897 0.898 0.908 0.916 0.919 0.920 0.921 0.924 0.
929 0.939 0.945 0.949 0.959 0.992 0.993

Д32А, Задание 2

In [24]:

```
def eta_distr(sample, x):
    res = 0
    for i in sample:
        if i <= x:
            res += 1
    return res / len(sample)

def eta_distr_real(x, theta=(0.45)):
    if x<0: return 0
    if x<theta: return x*x/theta
    if x<1: return (2*x-x*x-theta)/(1-theta)
    return 1

X_realeta = np.linspace(0,1,1000)
Y_realeta = np.array([eta_distr_real(x) for x in X_realeta])

Yeta = np.array([[[eta_distr(sample_eta[k][j], x) for x in X_realeta]
                  for j in range(5)] for k in range(len(n))])

def eta_Dmn(Yn, Ym, n, m):
    res = 0
    for i in range(29):
        d = abs(Yn[i]-Ym[i])
        if d>res: res = d
    return (n*m/(n+m))**0.5*res

diffseta = np.array([[[('%.2f' % eta_Dmn(Yeta[i][k], Yeta[j][k], n[i], n[i]))
                       if i>j else '-' for i in range(len(n))]
                      for j in range(len(n))]
                     for k in range(5)])
```

In [25]:
```
colors = ['#7bd152', '#45be71', '#25a885', '#21908c',
          '#2b798e', '#355f8d', '#414486', '#482574']

# graph[0]
fig, ax = plt.subplots(2,1, figsize=(7,12), height_ratios = [2,1])
for i in range(8):
    ax[0].stairs(Yeta[i][0], np.append(X_realeta, 1), color = colors[i])
ax[0].plot(X_realeta, Y_realeta, color = '#440154')
ax[0].set(xmargin = 0, ymargin = 0, xlabel = 'x', ylabel = r'$\hat{F}(x)$',
          title = 'Абсолютно непрерывное треугольное, выборка 1 \n' +
                  '$\\eta , \\,\\theta$ = 0.45')
ax[0].legend([*n, 'real'], loc='upper left');

ax[1].table(cellText = diffsxi[0], rowLabels=n, colLabels=n,
            loc='center').scale(1, 1.5)
ax[1].set_axis_off()
ax[1].set_title(r'$D_{m,n}=\sqrt{\frac{nm}{n+m}}\sup_{x\in\mathbb{R}}$' +
                r'$|F_n(x)-F_m(x)|$');
```

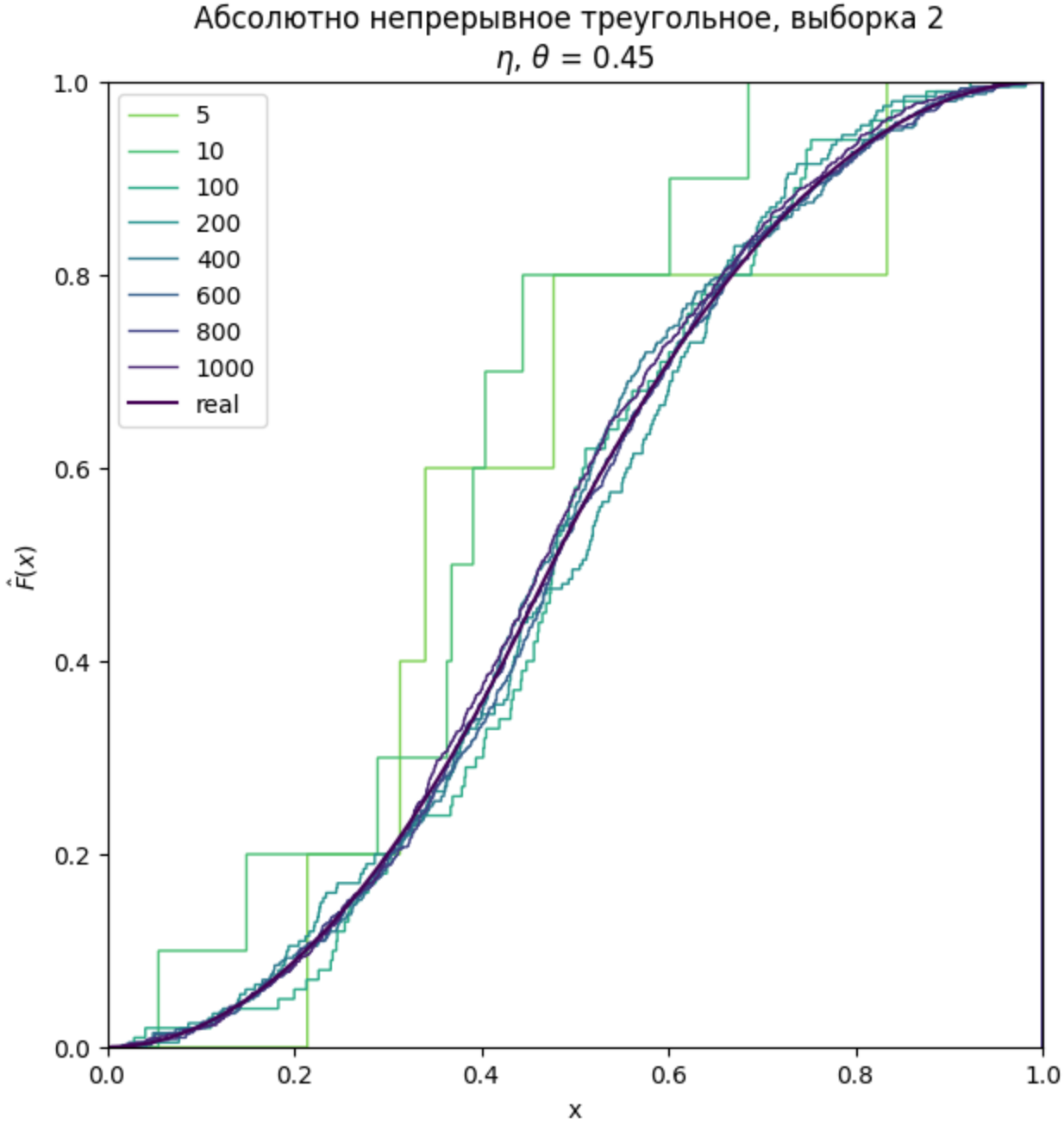


$$D_{m,n} = \sqrt{\frac{nm}{n+m}} \sup_{x \in \mathbb{R}} |F_n(x) - F_m(x)|$$

	5	10	100	200	400	600	800	1000
5	-	1.12	3.68	5.05	7.07	8.69	9.72	11.07
10	-	-	1.06	2.00	2.16	3.38	3.68	4.20
100	-	-	-	0.80	0.64	1.65	1.45	1.86
200	-	-	-	-	1.31	1.18	1.05	2.10
400	-	-	-	-	-	1.04	0.95	1.40
600	-	-	-	-	-	-	0.60	0.73
800	-	-	-	-	-	-	-	0.98
1000	-	-	-	-	-	-	-	-

In [26]:

```
# graph[1]
fig, ax = plt.subplots(2,1, figsize=(7,12), height_ratios = [2,1])
for i in range(8):
    ax[0].stairs(Yeta[i][1], np.append(X_realeta, 1), color = colors[i])
ax[0].plot(X_realeta, Y_realeta, color = '#440154')
ax[0].set(xmargin = 0, ymargin = 0, xlabel = 'x', ylabel = r'$\hat{F}(x)$',
          title = 'Абсолютно непрерывное треугольное, выборка 2 \n' +
                  '$\\eta , \\,\\theta$ = 0.45')
ax[0].legend([*n, 'real'], loc='upper left');

ax[1].table(cellText = diffsxi[1], rowLabels=n, colLabels=n,
            loc='center').scale(1, 1.5)
ax[1].set_axis_off()
ax[1].set_title(r'$D_{m,n}=\sqrt{\frac{nm}{n+m}}\sup_{x\in\mathbb{R}}$' +
                r'$|F_n(x)-F_m(x)|$');
```

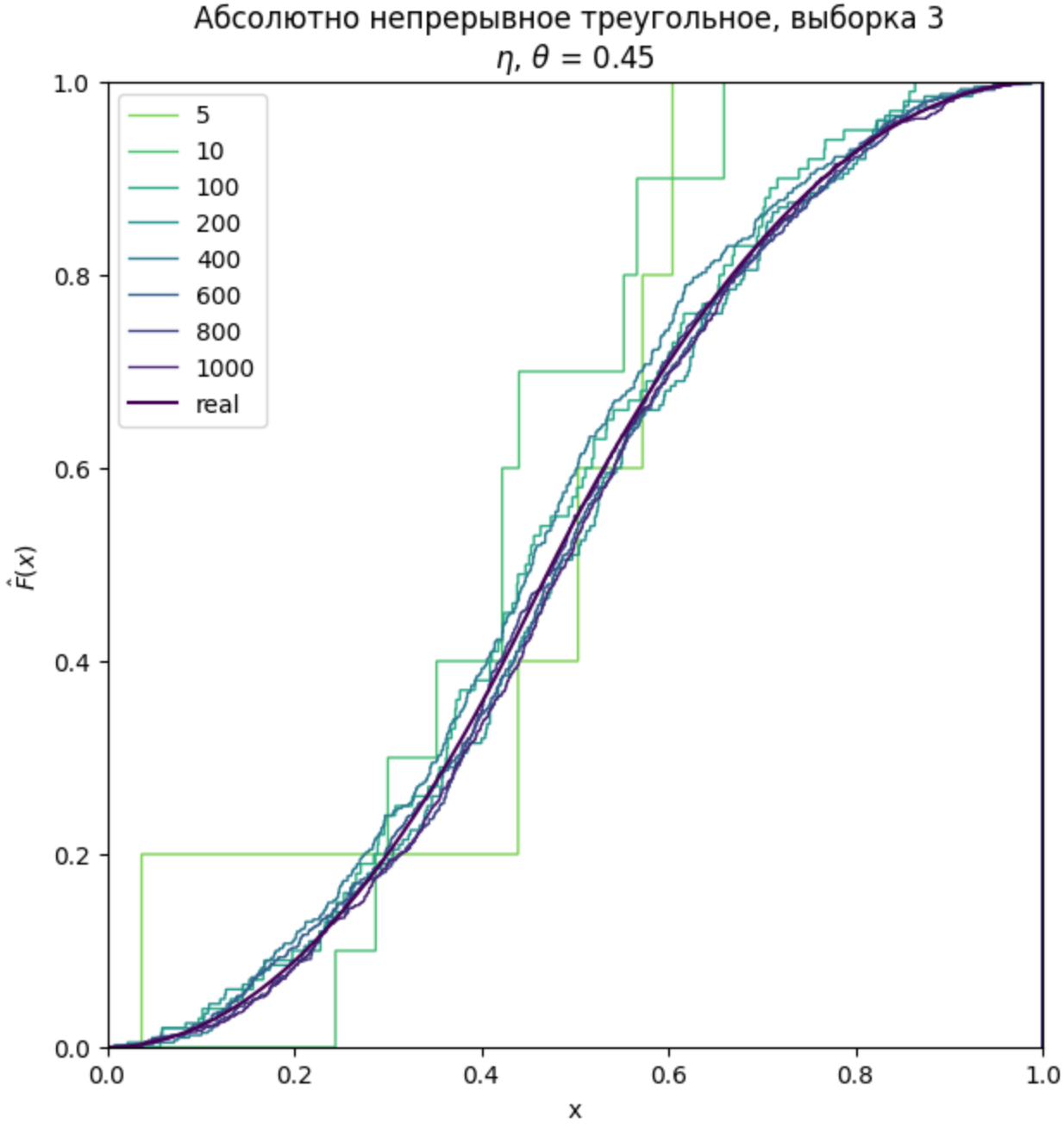


$$D_{m,n} = \sqrt{\frac{nm}{n+m}} \sup_{x \in \mathbb{R}} |F_n(x) - F_m(x)|$$

	5	10	100	200	400	600	800	1000
5	-	1.57	3.32	4.15	6.19	7.56	8.55	9.12
10	-	-	1.63	2.85	3.75	4.56	5.45	6.53
100	-	-	-	1.25	1.34	1.56	2.20	2.88
200	-	-	-	-	0.71	0.87	0.95	0.92
400	-	-	-	-	-	0.58	0.60	1.10
600	-	-	-	-	-	-	0.64	0.94
800	-	-	-	-	-	-	-	0.59
1000	-	-	-	-	-	-	-	-

In [27]:

```
# graph[2]
fig, ax = plt.subplots(2,1, figsize=(7,12), height_ratios = [2,1])
for i in range(8):
    ax[0].stairs(Yeta[i][2], np.append(X_realeta, 1), color = colors[i])
ax[0].plot(X_realeta, Y_realeta, color = '#440154')
ax[0].set(xmargin = 0, ymargin = 0, xlabel = 'x', ylabel = r'$\hat{F}(x)$',
          title = 'Абсолютно непрерывное треугольное, выборка 3 \n' +
                  '$\\eta , \\,\\theta$ = 0.45')
ax[0].legend([*n, 'real'], loc='upper left');

ax[1].table(cellText = diffsxi[2], rowLabels=n, colLabels=n,
            loc='center').scale(1, 1.5)
ax[1].set_axis_off()
ax[1].set_title(r'$D_{m,n}=\sqrt{\frac{nm}{n+m}}\sup_{x\in\mathbb{R}}$' +
                r'$|F_n(x)-F_m(x)|$');
```

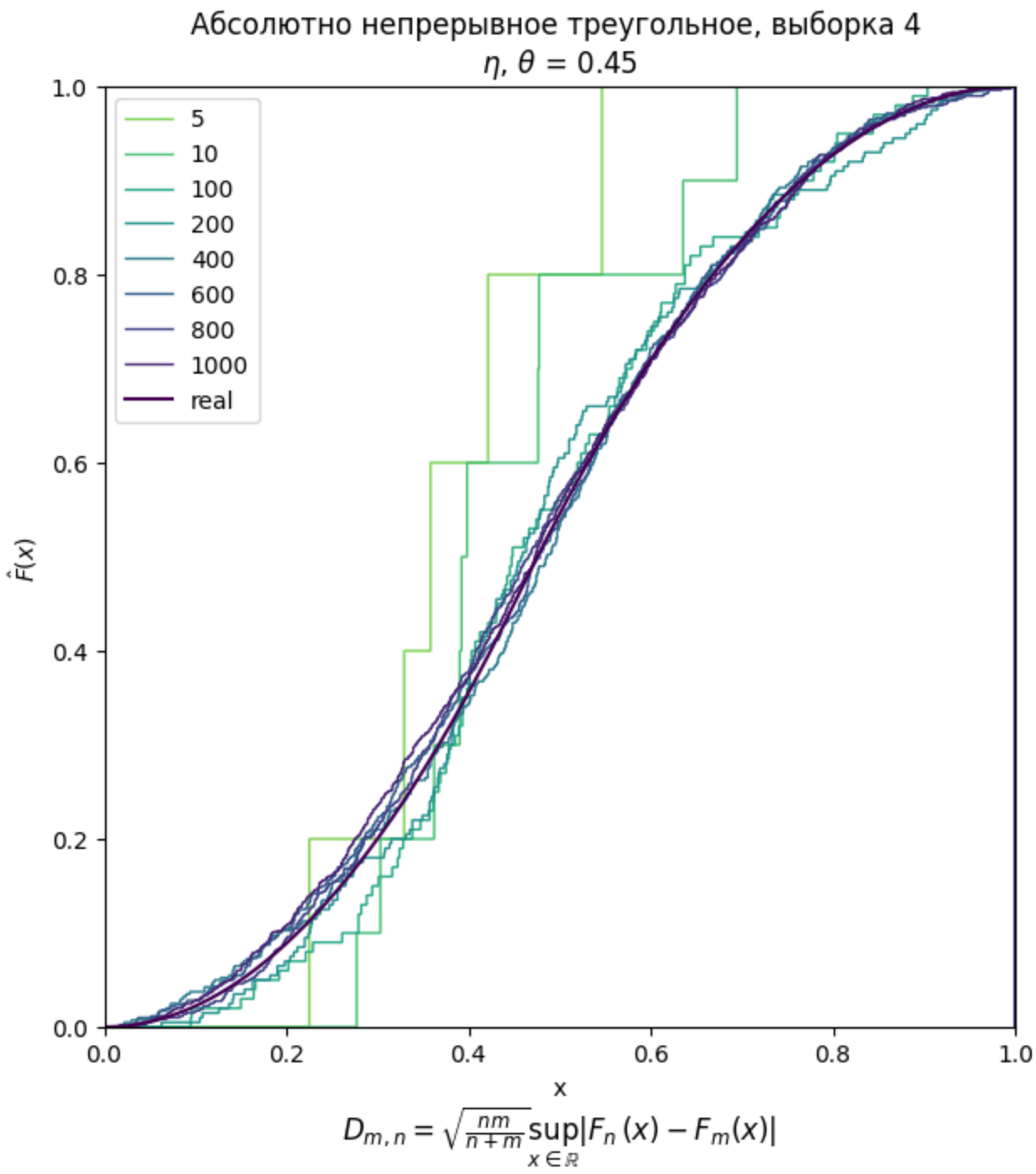


$$D_{m,n} = \sqrt{\frac{nm}{n+m}} \sup_{x \in \mathbb{R}} |F_n(x) - F_m(x)|$$

	5	10	100	200	400	600	800	1000
5	-	0.45	1.91	2.30	3.39	4.16	4.82	4.83
10	-	-	0.78	1.20	2.44	2.63	3.02	3.09
100	-	-	-	1.25	1.31	1.21	1.53	1.21
200	-	-	-	-	1.52	1.91	2.03	2.19
400	-	-	-	-	-	0.84	0.83	1.39
600	-	-	-	-	-	-	0.34	0.94
800	-	-	-	-	-	-	-	0.88
1000	-	-	-	-	-	-	-	-

In [28]:

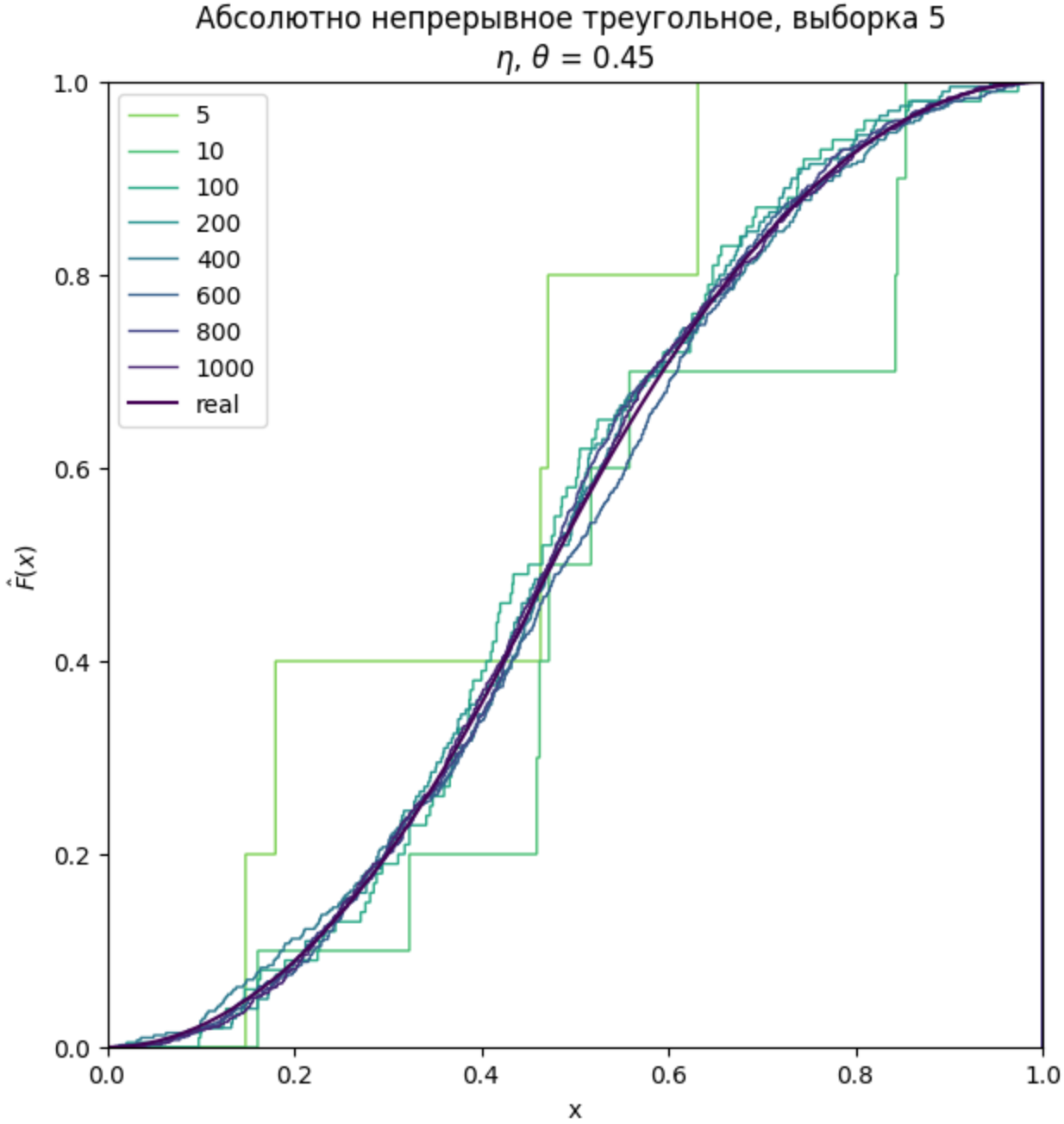
```
# graph[3]
fig, ax = plt.subplots(2,1, figsize=(7,12), height_ratios = [2,1])
for i in range(8):
    ax[0].stairs(Yeta[i][0], np.append(X_realeta, 1), color = colors[i])
ax[0].plot(X_realeta, Y_realeta, color = '#440154')
ax[0].set(xmargin = 0, ymargin = 0, xlabel = 'x', ylabel = r'$\hat{F}(x)$',
          title = 'Абсолютно непрерывное треугольное, выборка 4 \n' +
                  '$\\eta , \\,\\theta$ = 0.45')
ax[0].legend([*n, 'real'], loc='upper left');

ax[1].table(cellText = diffsxi[3], rowLabels=n, colLabels=n,
            loc='center').scale(1, 1.5)
ax[1].set_axis_off()
ax[1].set_title(r'$D_{m,n}=\sqrt{\frac{nm}{n+m}}\sup_{x\in\mathbb{R}}$' +
                r'$|F_n(x)-F_m(x)|$');
```



$$D_{m,n} = \sqrt{\frac{nm}{n+m}} \sup_{x \in \mathbb{R}} |F_n(x) - F_m(x)|$$

	5	10	100	200	400	600	800	1000
5	-	0.45	1.20	1.60	1.80	2.40	2.63	3.06
10	-	-	0.78	1.40	1.73	2.05	2.68	2.01
100	-	-	-	0.60	0.81	0.95	0.93	0.92
200	-	-	-	-	0.60	0.72	0.78	1.12
400	-	-	-	-	-	0.66	0.50	0.82
600	-	-	-	-	-	-	0.59	0.69
800	-	-	-	-	-	-	-	0.98
1000	-	-	-	-	-	-	-	-

In [29]:
```
# graph[4]
fig, ax = plt.subplots(2,1, figsize=(7,12), height_ratios = [2,1])
for i in range(8):
    ax[0].stairs(Yeta[i][4], np.append(X_realeta, 1), color = colors[i])
ax[0].plot(X_realeta, Y_realeta, color = '#440154')
ax[0].set(xmargin = 0, ymargin = 0, xlabel = 'x', ylabel = r'$\hat{F}(x)$',
          title = 'Абсолютно непрерывное треугольное, выборка 5 \n' +
                  '$\\eta , \\,\\theta$ = 0.45')
ax[0].legend([*n, 'real'], loc='upper left');

ax[1].table(cellText = diffsxi[4], rowLabels=n, colLabels=n,
            loc='center').scale(1, 1.5)
ax[1].set_axis_off()
ax[1].set_title(r'$D_{m,n}=\sqrt{\frac{nm}{n+m}}\sup_{x\in\mathbb{R}}$' +
                r'$|F_n(x)-F_m(x)|$');
```



$$D_{m,n} = \sqrt{\frac{nm}{n+m}} \sup_{x \in \mathbb{R}} |F_n(x) - F_m(x)|$$

	5	10	100	200	400	600	800	1000
5	-	0.89	2.33	3.35	4.84	5.98	6.17	7.22
10	-	-	1.20	1.80	2.30	3.03	3.45	3.87
100	-	-	-	0.65	0.88	0.98	1.65	1.25
200	-	-	-	-	0.88	1.10	0.90	1.23
400	-	-	-	-	-	0.48	0.87	0.49
600	-	-	-	-	-	-	0.98	0.57
800	-	-	-	-	-	-	-	0.72
1000	-	-	-	-	-	-	-	-

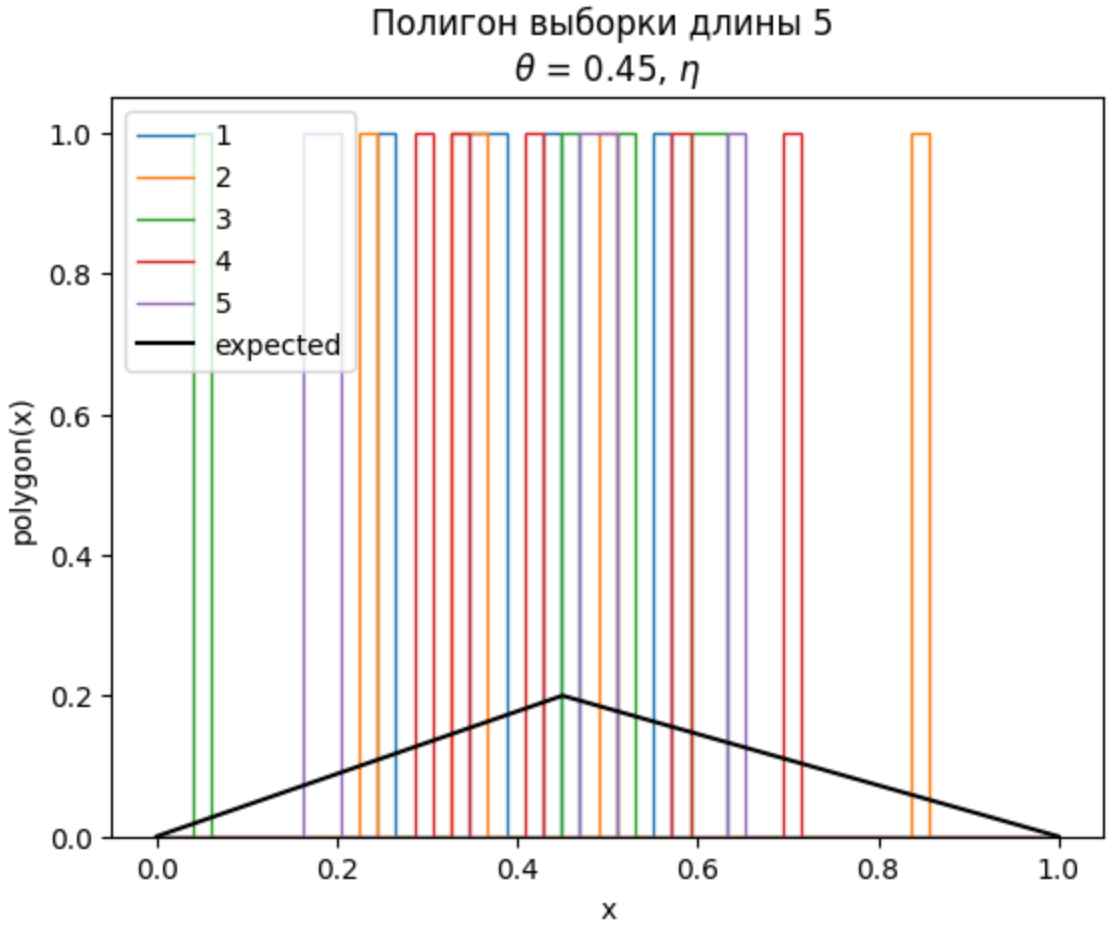
Д32А, Задание 3

Логика с домножением вероятности здесь также применима. Однако здесь есть другой аспект: полигон частот почти не будет иметь смысла, так как вероятность попадания в каждое значение стремится к нулю. Для этого и только для этого я разобью отрезок [0, 1] на оси oX на 50 равных отрезков. Да, это сведет все к подобию дискретного случая, но иначе никакого смысла из сравнения графиков не получить.

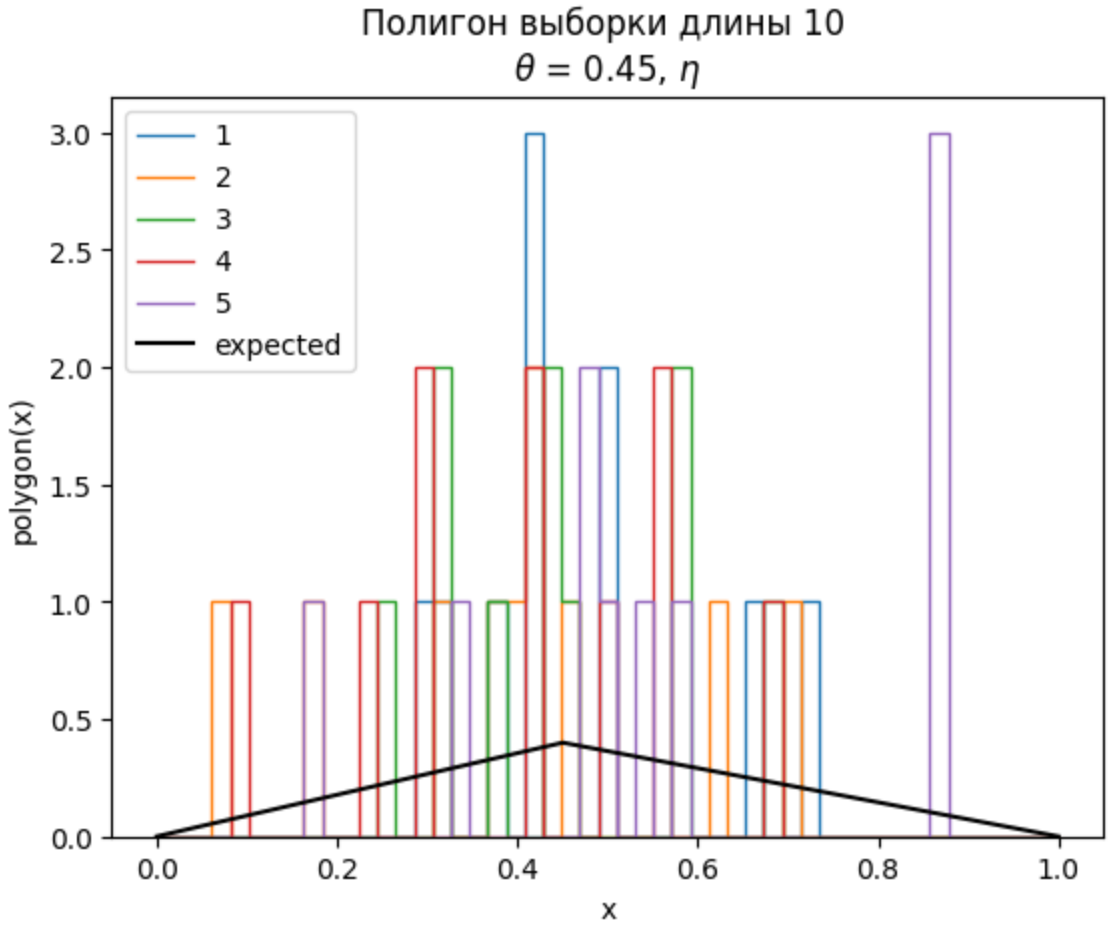
В связи с таким решением всплывет другая проблема - как теперь домножать вероятность? (этот вопрос я дописываю уже постфактумом, когда столкнулся с проблемой того что просто домножение больше не работает) Раньше вероятность домножалась на выборку, но эта выборка целиком умещалась в целых числах от 0 до θ. Теперь же выборка лежит в отрезке уже после факта обработки выборки, да и отрезок не в целых числах. Практикой было найдено, что если поделить домноженную выборку на количество отрезков, графики вероятности и частоты примерно сойдутся. Почему? Без понятия, но оно работает. Да и не задача это дз (хотя задача сравнить.. а для этого нужно подвести одно к другому..)

In [30]:
```
def eta_pilygon(sample, X):
    Y = np.zeros(X.shape)
    tick = 1
    for i in sample:
        while i > X[tick]: tick+=1
        Y[tick]+=1
    return Y

def eta_posib(x, theta = 0.45):
    if x<0: return 0
    if x<theta: return 2*x/theta
    if x<1: return 2*(1-x)/(1-theta)
    return 0

X_poleta = np.linspace(0,1,50)
Y_poleta = [[eta_pilygon(sample_eta[k][j], X_poleta) for j in range(5)]
            for k in range(len(n))]

posibilityeta = np.array([eta_posib(x) for x in X_realeta])
```

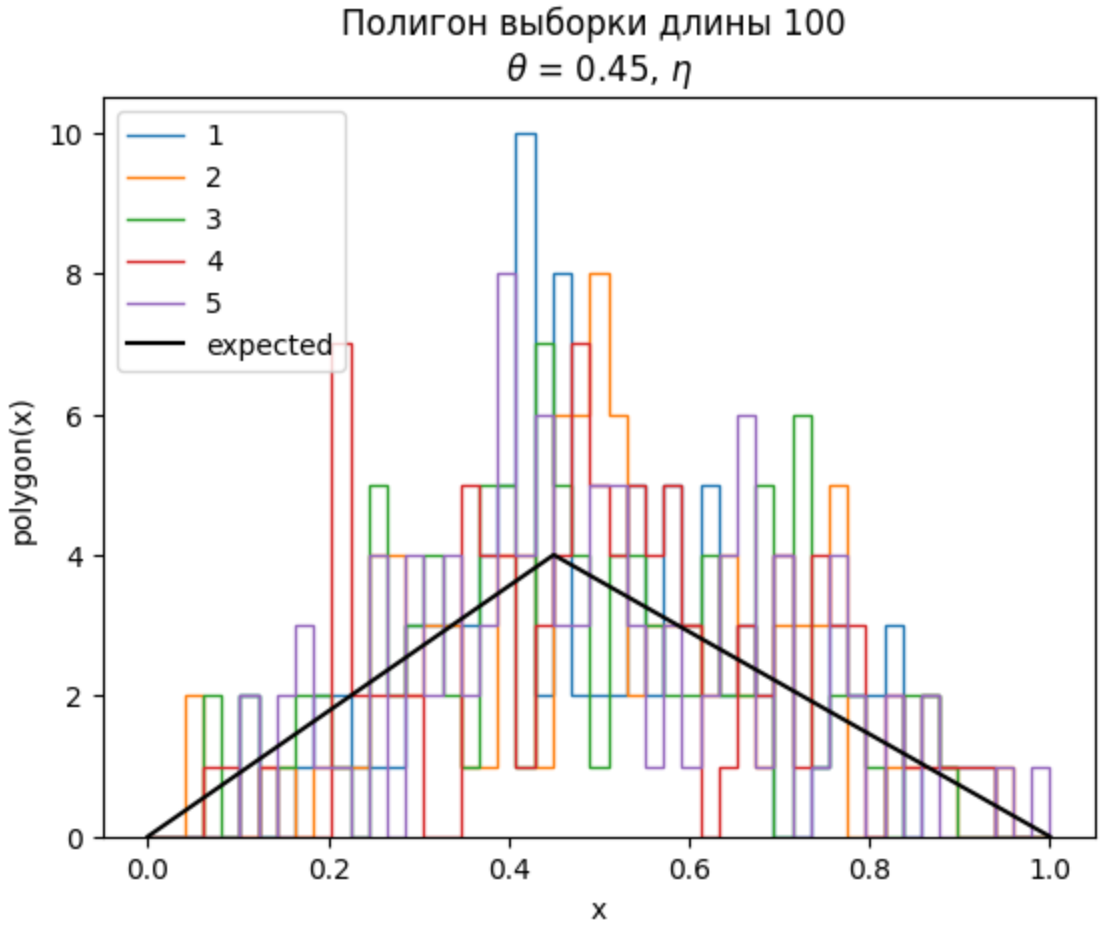
In [31]:
```
# n=5
for i in range(5):
    plt.stairs(Y_poleta[0][i], np.append(X_poleta,1))
plt.plot(X_realeta, posibilityeta*5/50, 'k')
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 5 \n$\\theta$ = 0.45, $\\eta$')
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



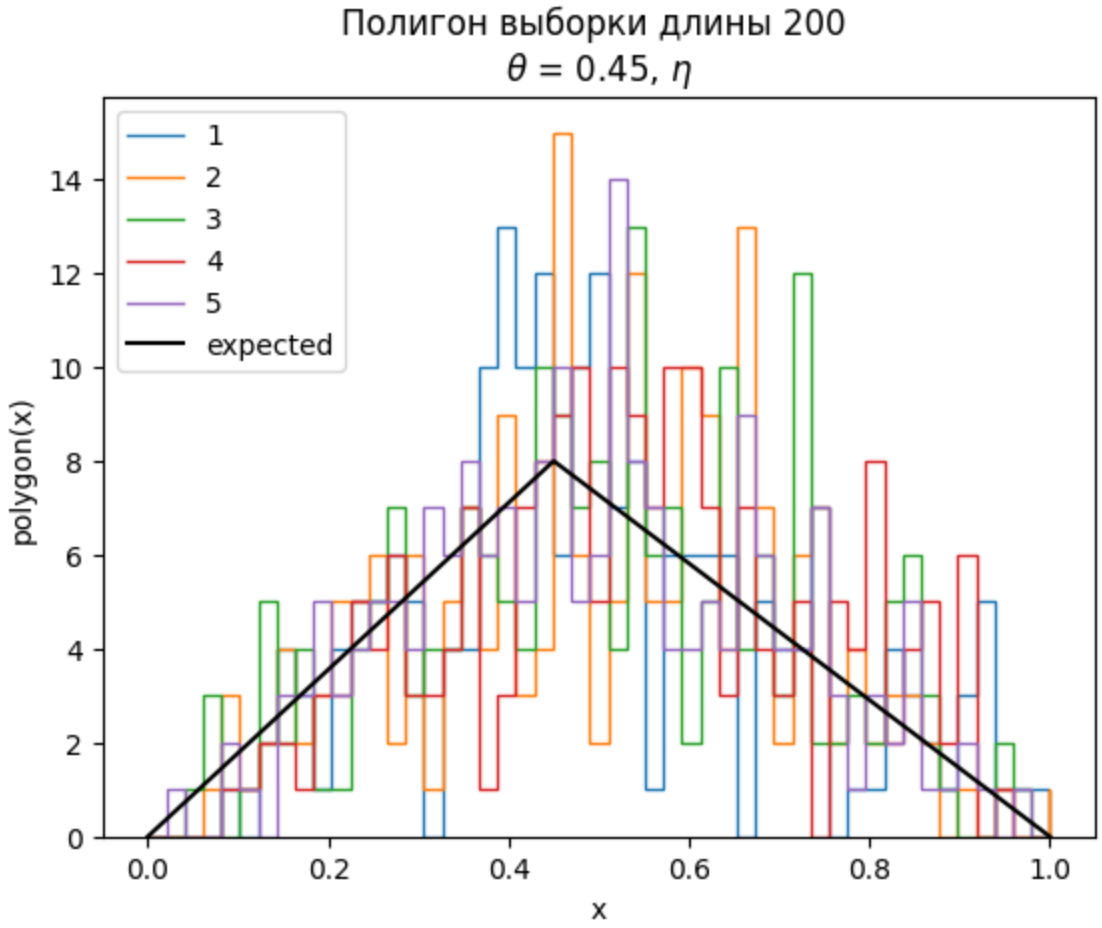
In [32]:
```
# n=10
for i in range(5):
    plt.stairs(Y_poleta[1][i], np.append(X_poleta, 1))
plt.plot(X_realeta, posibilityeta*10/50, 'k')
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 10 \n$\\theta$ = 0.45, $\\eta$')
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



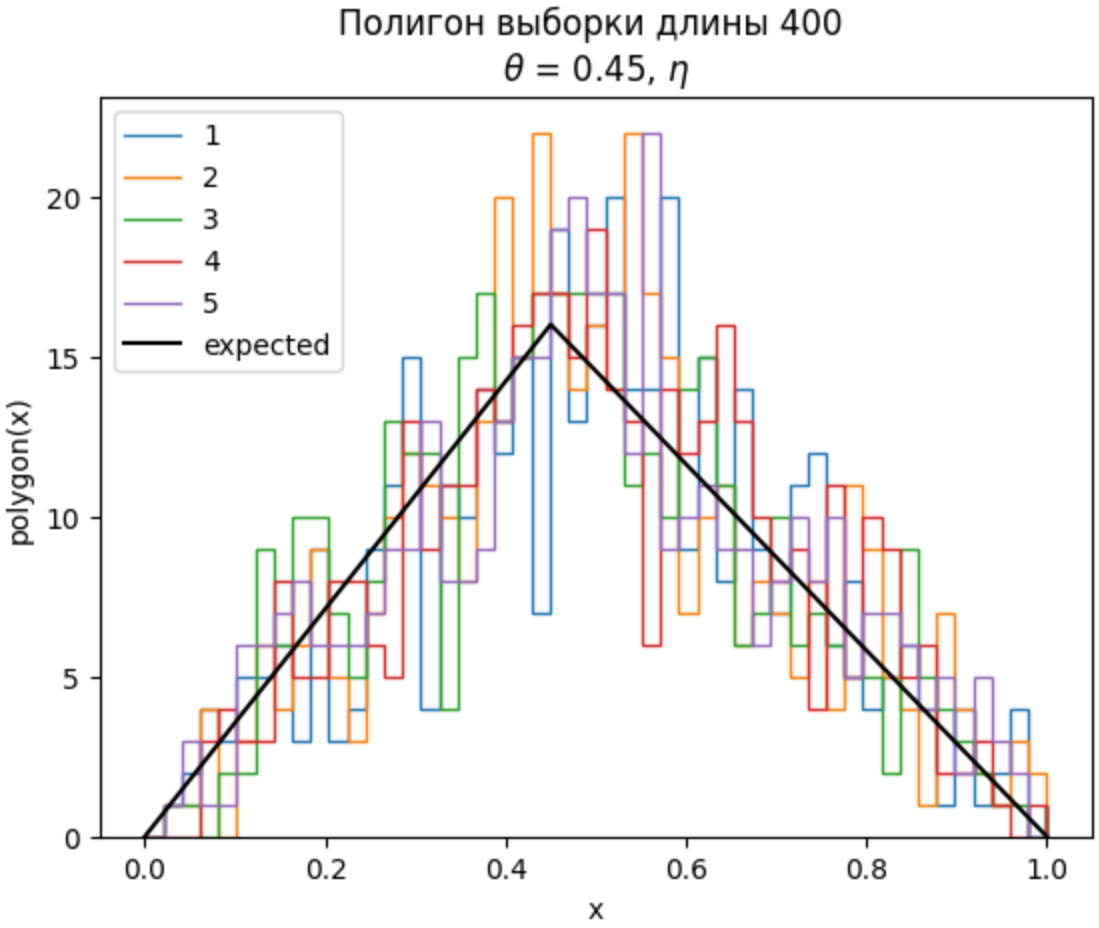
In [33]:
```
# n=100
for i in range(5):
    plt.stairs(Y_poleta[2][i], np.append(X_poleta, 1))
plt.plot(X_realeta, posibilityeta*100/50, 'k')
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 100 \n$\\theta$ = 0.45, $\\eta$')
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



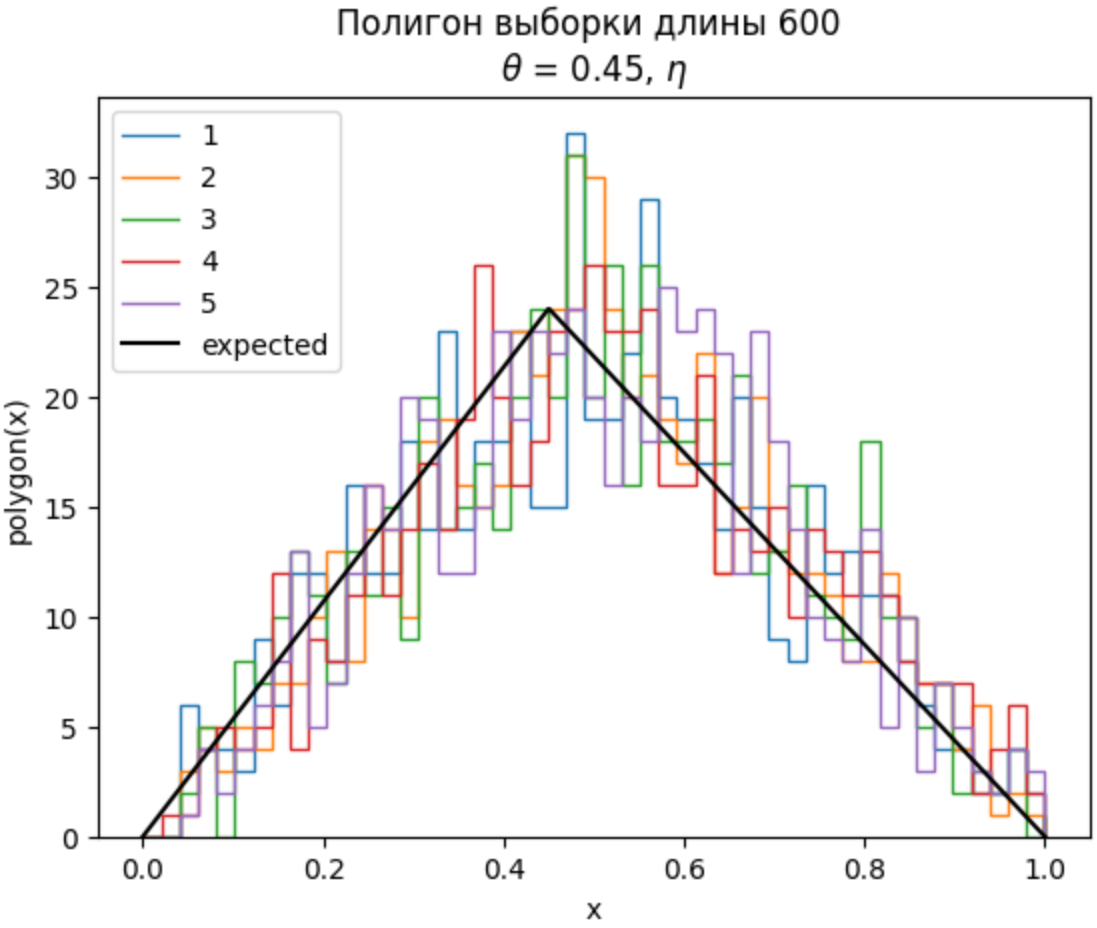
In [34]:
```
# n=200
for i in range(5):
    plt.stairs(Y_poleta[3][i], np.append(X_poleta, 1))
plt.plot(X_realeta, posibilityeta*200/50, 'k')
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 200 \n$\\theta$ = 0.45, $\\eta$')
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



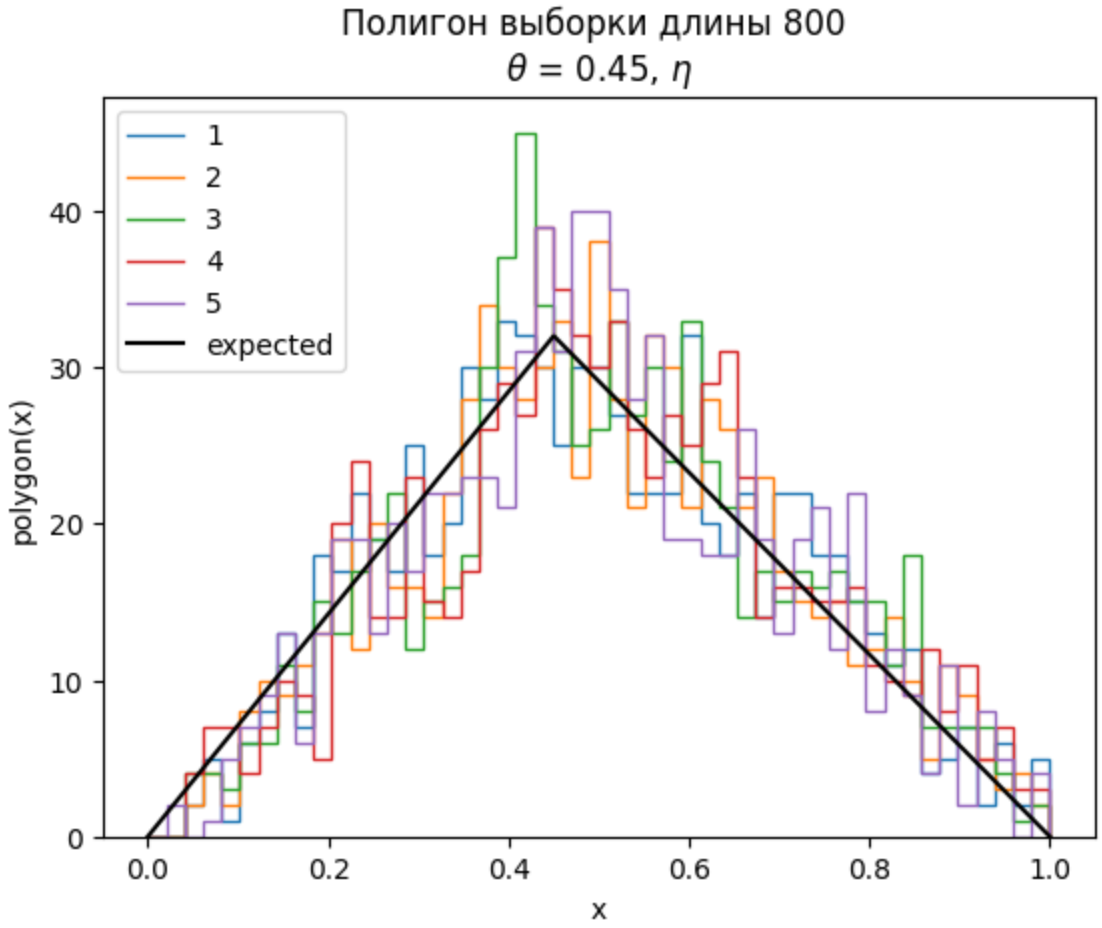
In [35]:
```
# n=400
for i in range(5):
    plt.stairs(Y_poleta[4][i], np.append(X_poleta, 1))
plt.plot(X_realeta, posibilityeta*400/50, 'k')
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 400 \n$\\theta$ = 0.45, $\\eta$')
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



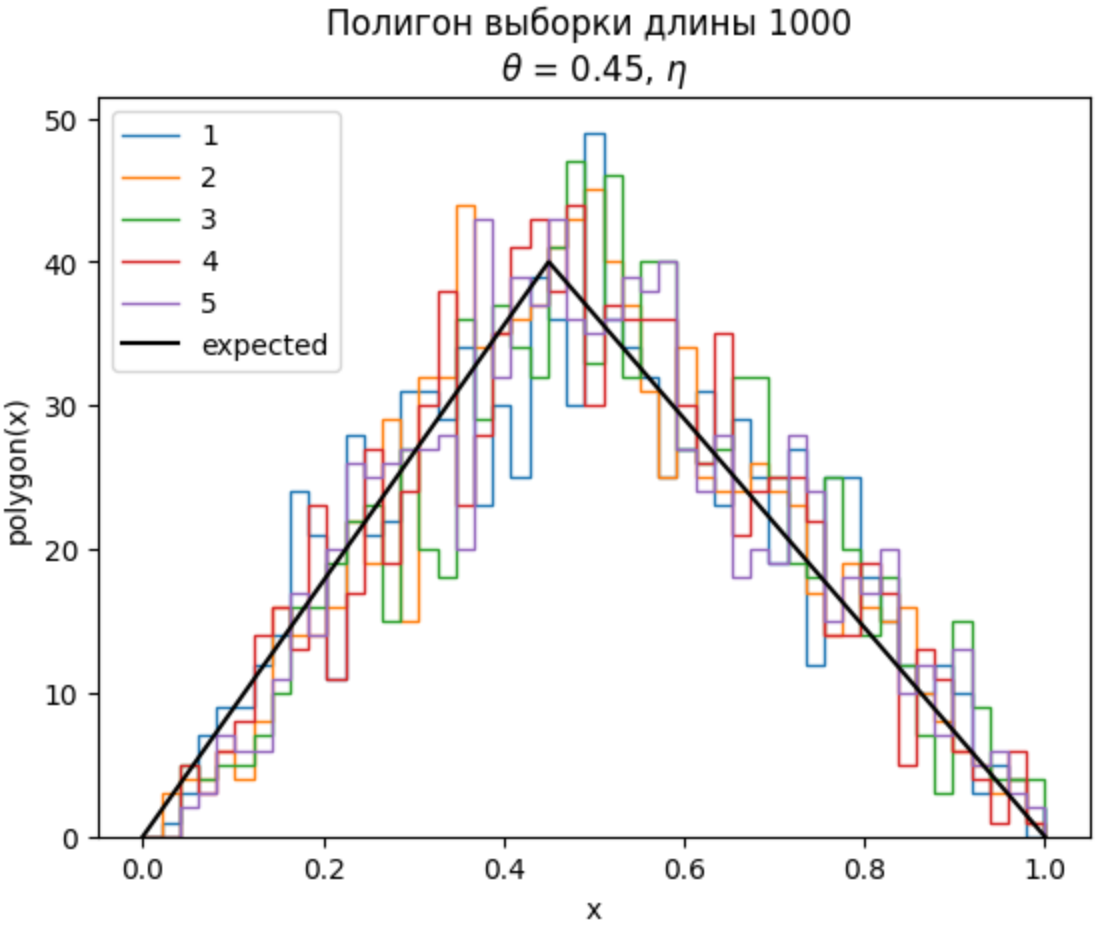
In [36]:
```
# n=600
for i in range(5):
    plt.stairs(Y_poleta[5][i], np.append(X_poleta, 1))
plt.plot(X_realeta, posibilityeta*600/50, 'k')
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 600 \n$\\theta$ = 0.45, $\\eta$')
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



In [37]:
```
# n=800
for i in range(5):
    plt.stairs(Y_poleta[6][i], np.append(X_poleta, 1))
plt.plot(X_realeta, posibilityeta*800/50, 'k')
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 800 \n$\\theta$ = 0.45, $\\eta$')
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



In [38]:
```
# n=1000
for i in range(5):
    plt.stairs(Y_poleta[7][i], np.append(X_poleta, 1))
plt.plot(X_realeta, posibilityeta*1000/50, 'k')
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 1000 \n$\\theta$ = 0.45, $\\eta$')
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



Чтобы отметить, что я все еще анализирую графики, а не просто переписываю код под непрерывный случай: графики и результаты двух предыдущих заданий опять демонстрируют справедливость теоремы.

ДЗ2А, Задание 4

In [39]:
```
def eta_sample_mean(sample):
    return sum(sample)/len(sample)

def eta_sample_variance(sample):
    return sum((sample-xi_sample_mean(sample))**2)/len(sample)

meanseta = np.array([[eta_sample_mean(sample_eta[k][j])for j in range(5)]
                     for k in range(len(n))])
varianceseta = np.array([[eta_sample_variance(sample_eta[k][j])for j in range(5)]
                         for k in range(len(n))])
means_peta = np.array([['%.4f' % i for i in j] for j in meanseta])
variances_peta = np.array([['%.4f' % i for i in j] for j in varianceseta])
expectationeta = (1+0.45)/3
varianceeta = (1-0.45+0.45**2)/18
means_difeta = np.array([['%.4f' % i for i in j]
                         for j in (meanseta-expectationeta)])
variances_difeta = np.array([['%.4f' % i for i in j]
                             for j in (varianceseta-varianceeta)])
```

In [40]:
```
fig, ax = plt.subplots(2,1, figsize=(7,6.5))
ax[0].table(cellText = means_peta, rowLabels=n, colLabels=[1,2,3,4,5],
            loc='center').scale(1, 1.5)
ax[0].set_axis_off()
ax[0].set_title('Выборочные средние');
ax[1].table(cellText = variances_peta, rowLabels=n, colLabels=[1,2,3,4,5],
            loc='center').scale(1, 1.5)
ax[1].set_axis_off()
ax[1].set_title('Выборочные дисперсии');
```

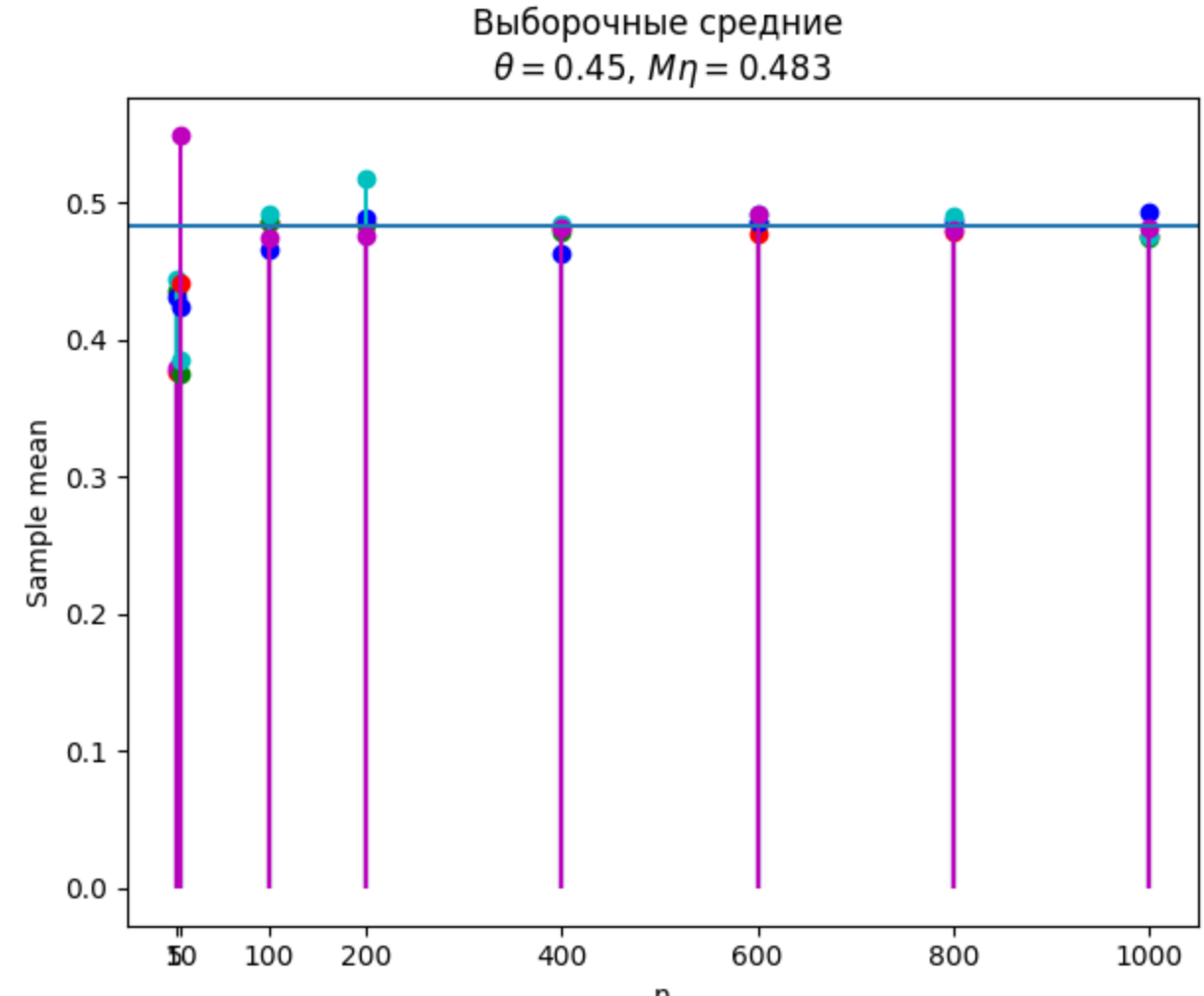
Выборочные средние

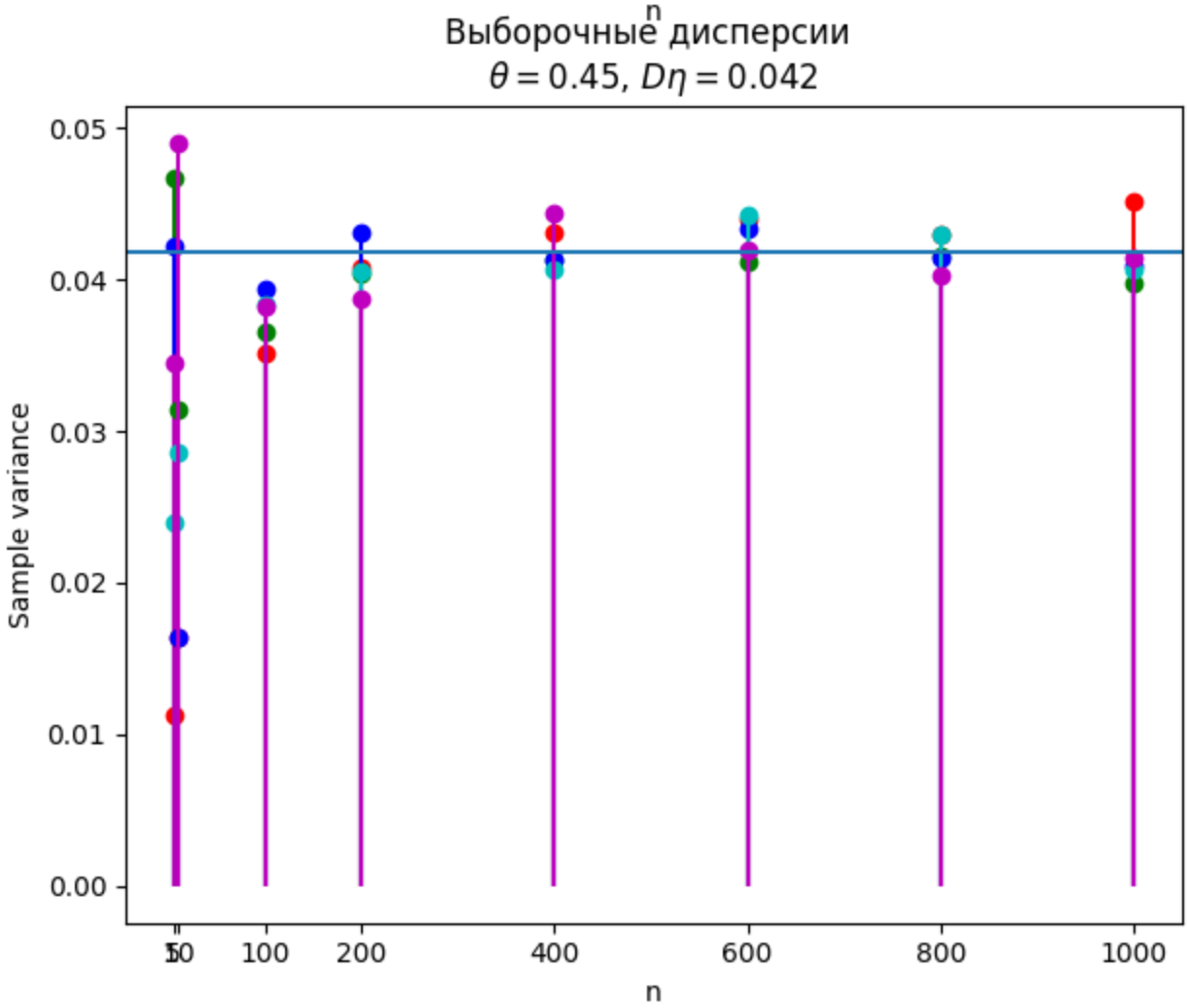
	1	2	3	4	5
5	0.3757	0.4352	0.4309	0.4435	0.3785
10	0.4405	0.3747	0.4243	0.3851	0.5492
100	0.4852	0.4860	0.4655	0.4912	0.4736
200	0.4837	0.4843	0.4886	0.5166	0.4754
400	0.4819	0.4784	0.4628	0.4846	0.4810
600	0.4771	0.4861	0.4853	0.4913	0.4907
800	0.4789	0.4846	0.4870	0.4893	0.4802
1000	0.4750	0.4746	0.4922	0.4750	0.4819

Выборочные дисперсии

	1	2	3	4	5
5	0.0113	0.0467	0.0422	0.0240	0.0345
10	0.0163	0.0315	0.0164	0.0285	0.0490
100	0.0352	0.0365	0.0394	0.0384	0.0382
200	0.0408	0.0404	0.0431	0.0406	0.0388
400	0.0431	0.0413	0.0414	0.0407	0.0444
600	0.0440	0.0412	0.0433	0.0443	0.0419
800	0.0430	0.0415	0.0414	0.0430	0.0403
1000	0.0452	0.0398	0.0410	0.0406	0.0414

In [44]:
```
fig, ax = plt.subplots(2,1, figsize=(7,12))
for k in range(len(n)):
    for j in range(5):
        ax[0].stem(n[k], meanseta[k][j], 'rgbcm'[j])
ax[0].axhline(expectationeta)
ax[0].set(xticks = n, xlabel = 'n', ylabel = 'Sample mean',
          title = 'Выборочные средние \n$\\theta = 0.45, ' +
                  '\\,M\\eta = %.3f$'%expectationeta)

for k in range(len(n)):
    for j in range(5):
        ax[1].stem(n[k], varianceseta[k][j], 'rgbcm'[j])
ax[1].axhline(y = varianceeta)
ax[1].set(xticks = n, xlabel = 'n', ylabel = 'Sample variance',
          title = 'Выборочные дисперсии \n$\\theta = 0.45, ' +
                  '\\,D\\eta = %.3f$'%varianceeta);
```


Выборочные средние
$\theta = 0.45$, $M\eta = 0.483$
Sample mean
0.0
0.1
0.2
0.3
0.4
0.5
50
100
200
400
600
800
1000
n


Выборочные дисперсии
$\theta = 0.45$, $D\eta = 0.042$
Sample variance
0.00
0.01
0.02
0.03
0.04
0.05
50
100
200
400
600
800
1000
n

In [43]:
```
fig, ax = plt.subplots(2,1, figsize=(7,7))
ax[0].table(cellText = means_difeta, rowLabels=n, colLabels=[1,2,3,4,5],
            loc='center').scale(1, 1.5)
ax[0].set_axis_off()
ax[0].set_title('Разница выборочного среднего и математического ожидания' +
                '\n $M\\eta = %.4f$'%expectationeta);

ax[1].table(cellText = variances_difeta, rowLabels=n, colLabels=[1,2,3,4,5],
            loc='center').scale(1, 1.5)
ax[1].set_axis_off()
ax[1].set_title('Разница выборочной дисперсии и дисперсии' +
                '\n $D\\eta = %.4f$'%varianceeta);
```

Разница выборочного среднего и математического ожидания
$M\eta = 0.4833$

	1	2	3	4	5
5	-0.1076	-0.0481	-0.0525	-0.0399	-0.1048
10	-0.0429	-0.1086	-0.0590	-0.0983	0.0658
100	0.0019	0.0027	-0.0179	0.0079	-0.0098
200	0.0004	0.0010	0.0053	0.0333	-0.0079
400	-0.0015	-0.0049	-0.0205	0.0012	-0.0023
600	-0.0063	0.0028	0.0020	0.0080	0.0074
800	-0.0045	0.0013	0.0037	0.0060	-0.0032
1000	-0.0084	-0.0088	0.0089	-0.0084	-0.0014

Разница выборочной дисперсии и дисперсии
$D\eta = 0.0418$

	1	2	3	4	5
5	-0.0305	0.0049	0.0004	-0.0178	-0.0073
10	-0.0255	-0.0103	-0.0254	-0.0133	0.0072
100	-0.0066	-0.0053	-0.0024	-0.0034	-0.0036
200	-0.0010	-0.0014	0.0013	-0.0012	-0.0030
400	0.0013	-0.0005	-0.0004	-0.0011	0.0026
600	0.0022	-0.0006	0.0015	0.0025	0.0001
800	0.0012	-0.0003	-0.0004	0.0012	-0.0015
1000	0.0034	-0.0020	-0.0008	-0.0012	-0.0004

... - что еще раз на практике подтверждает теорему - ...

In [ ]: