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: $f(x) = \begin{cases} \dfrac{2x}{\theta}, & \text{если } x \in [0, \theta] \\ \dfrac{2(1-x)}{1-\theta}, & \text{если } x \in (\theta, 1] \\ 0, & \text{иначе} \end{cases}, \theta = 4.5$

Обозначим абсолютно непрерывное распределение в дальнейшем за η

Домашнее задание 1

ДЗ1, Дискретное

ДЗ1Д, Задание 1

Функция распределения

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

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

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

ДЗ1Д, 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}

ДЗ1Д, 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 $(x-1)\theta^{-1} \le u < x\theta^{-1}$

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

# np.array([[generate_discunif(theta) for i in range(26)]
#           for theta in range(57, 123, 13)])
```

ДЗ1, Абсолютно непрерывное

ДЗ1А, Задание 1

Функция распределения

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

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

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

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

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

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

ДЗ1А, 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.

Д31А, 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 под ниже.

In [2]:
```
import numpy as np
def generate_eta(theta=(1/4.5)):
    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

#np.array([[generate_triang(0.03+j*0.13) for i in range(25)] for j in range(6)])
```

Домашнее задание 2

Д32, Дискретное

Д32Д, 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 [4]:
```
# 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 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 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 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 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 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 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 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 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 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 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 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 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 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 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 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 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 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 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 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 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 [6]:

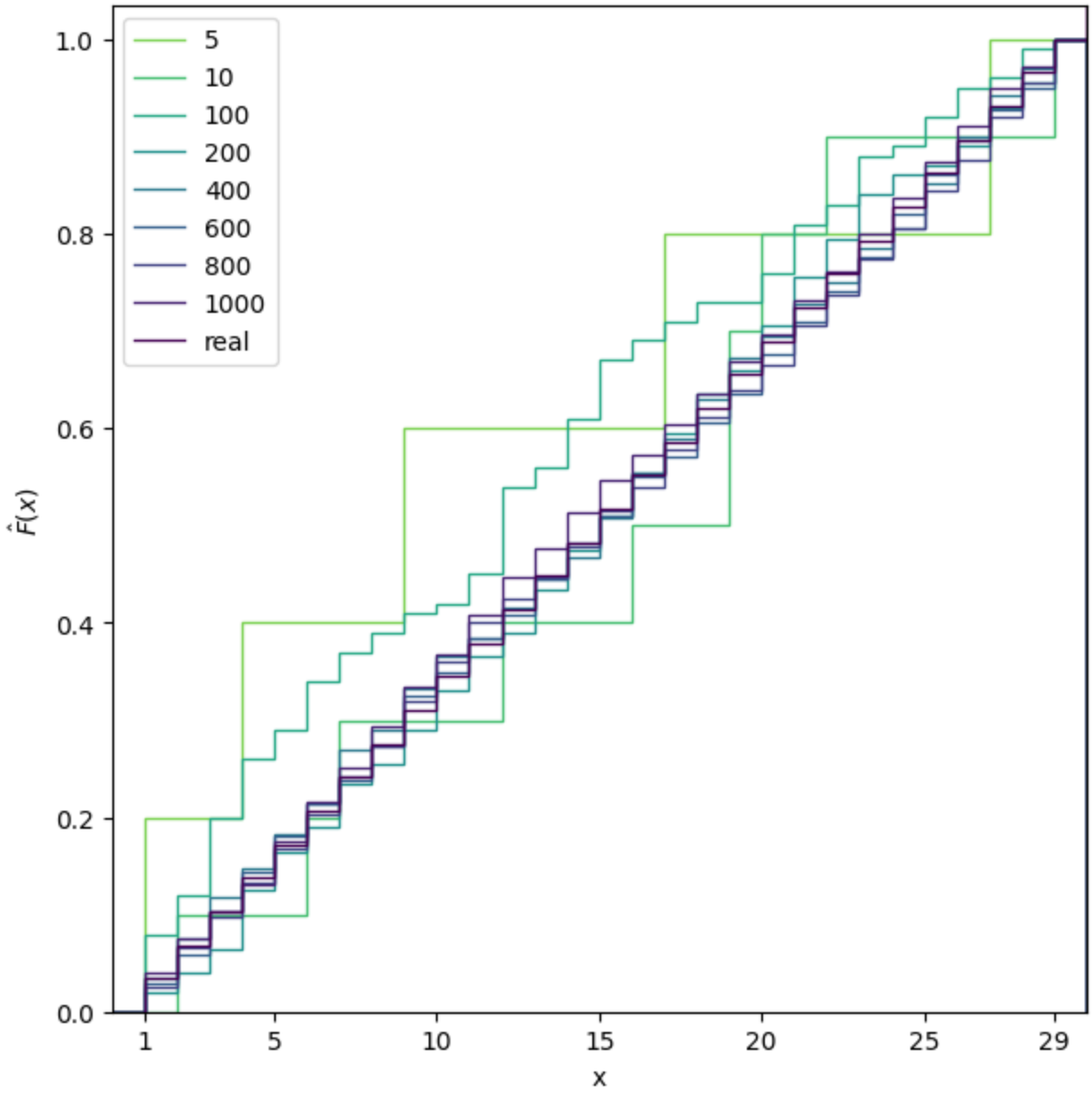
```
def xi_distr(sample, x):
    res = 0
    for i in sample:
        if i <= x:
            res += 1
    return res / len(sample)

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 [9]:

```
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, $\theta = 29$



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

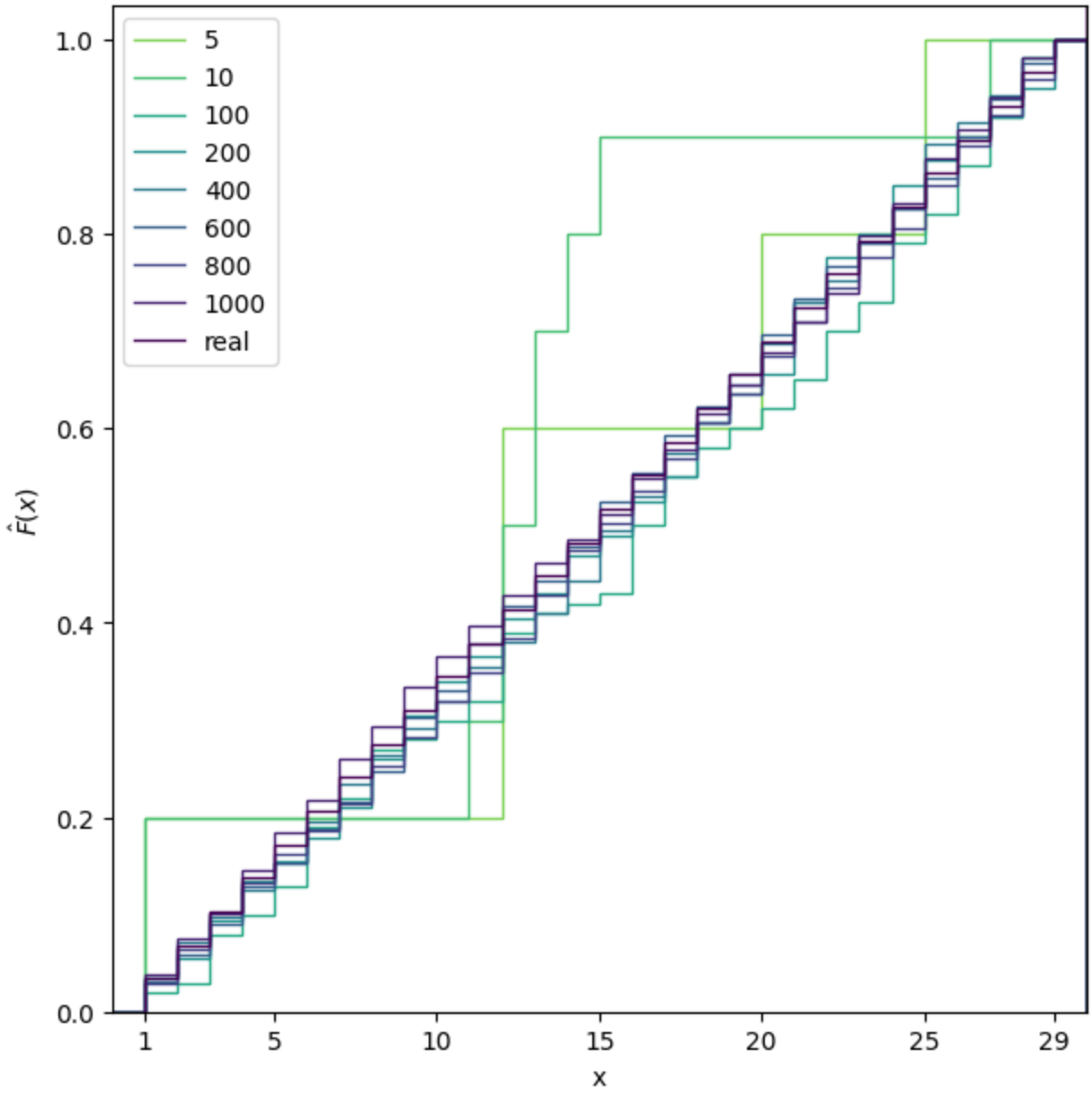
	5	10	100	200	400	600	800	1000
5	-	0.67	1.34	3.10	3.78	4.76	5.62	5.99
10	-	-	1.91	1.35	2.12	2.74	3.25	3.26
100	-	-	-	1.60	2.30	2.77	3.10	2.86
200	-	-	-	-	0.78	1.13	1.32	1.27
400	-	-	-	-	-	0.65	0.67	1.02
600	-	-	-	-	-	-	0.41	0.86
800	-	-	-	-	-	-	-	0.77
1000	-	-	-	-	-	-	-	-

In [12]:

```
# 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, $\theta = 29$



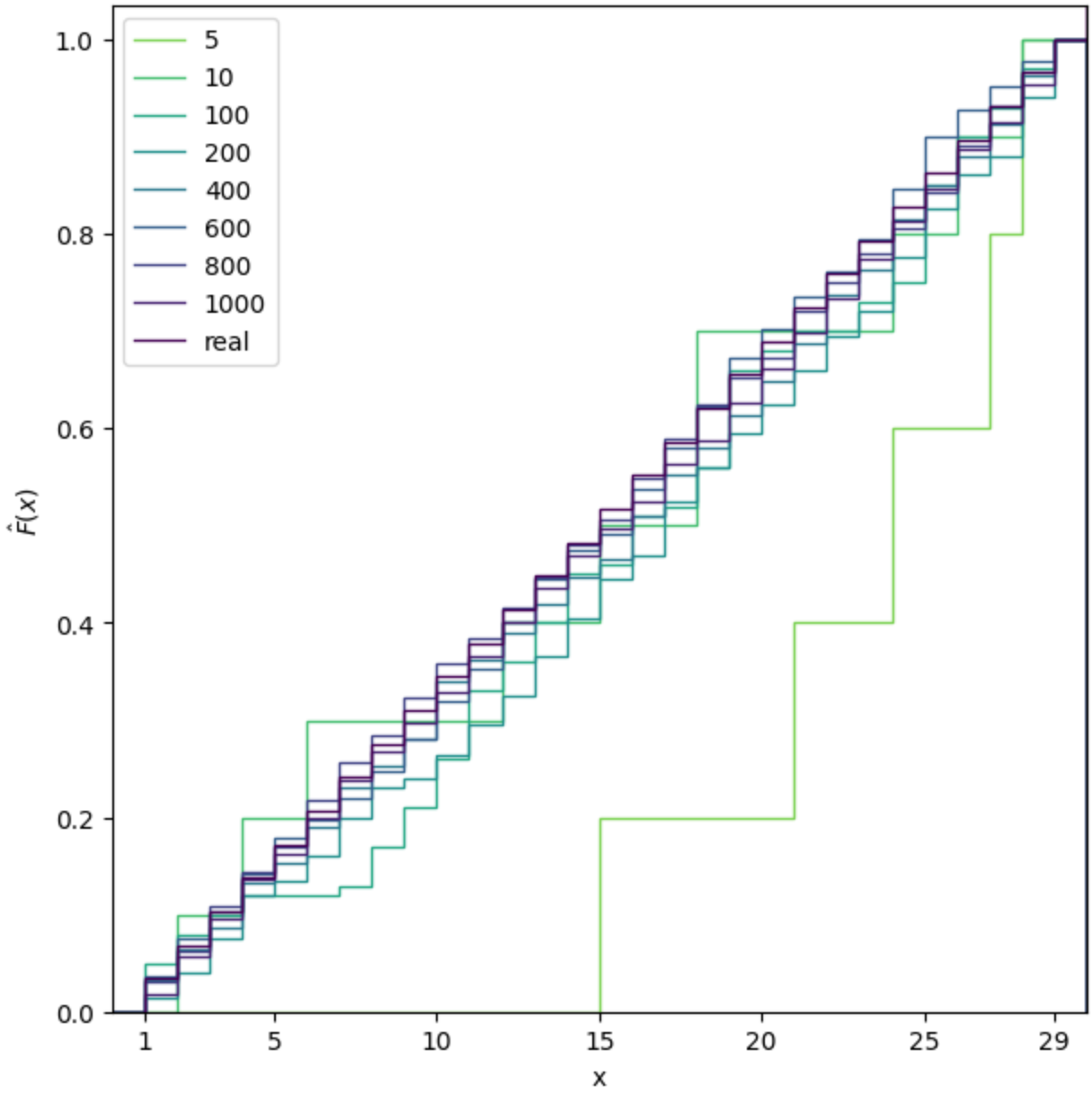
$$D_{m,n} = \sqrt{\frac{nm}{n+m}} \sup_{x \in \mathbb{R}} |F_n(x) - F_m(x)|$$

	5	10	100	200	400	600	800	1000
5	-	0.67	1.48	1.95	3.11	3.18	4.33	4.43
10	-	-	3.32	4.10	5.73	6.50	7.95	8.68
100	-	-	-	0.80	1.13	1.65	1.45	1.83
200	-	-	-	-	0.46	0.75	0.87	1.14
400	-	-	-	-	-	0.64	0.85	1.16
600	-	-	-	-	-	-	0.66	1.04
800	-	-	-	-	-	-	-	1.15
1000	-	-	-	-	-	-	-	-

In [13]:
```
# 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, $\theta = 29$



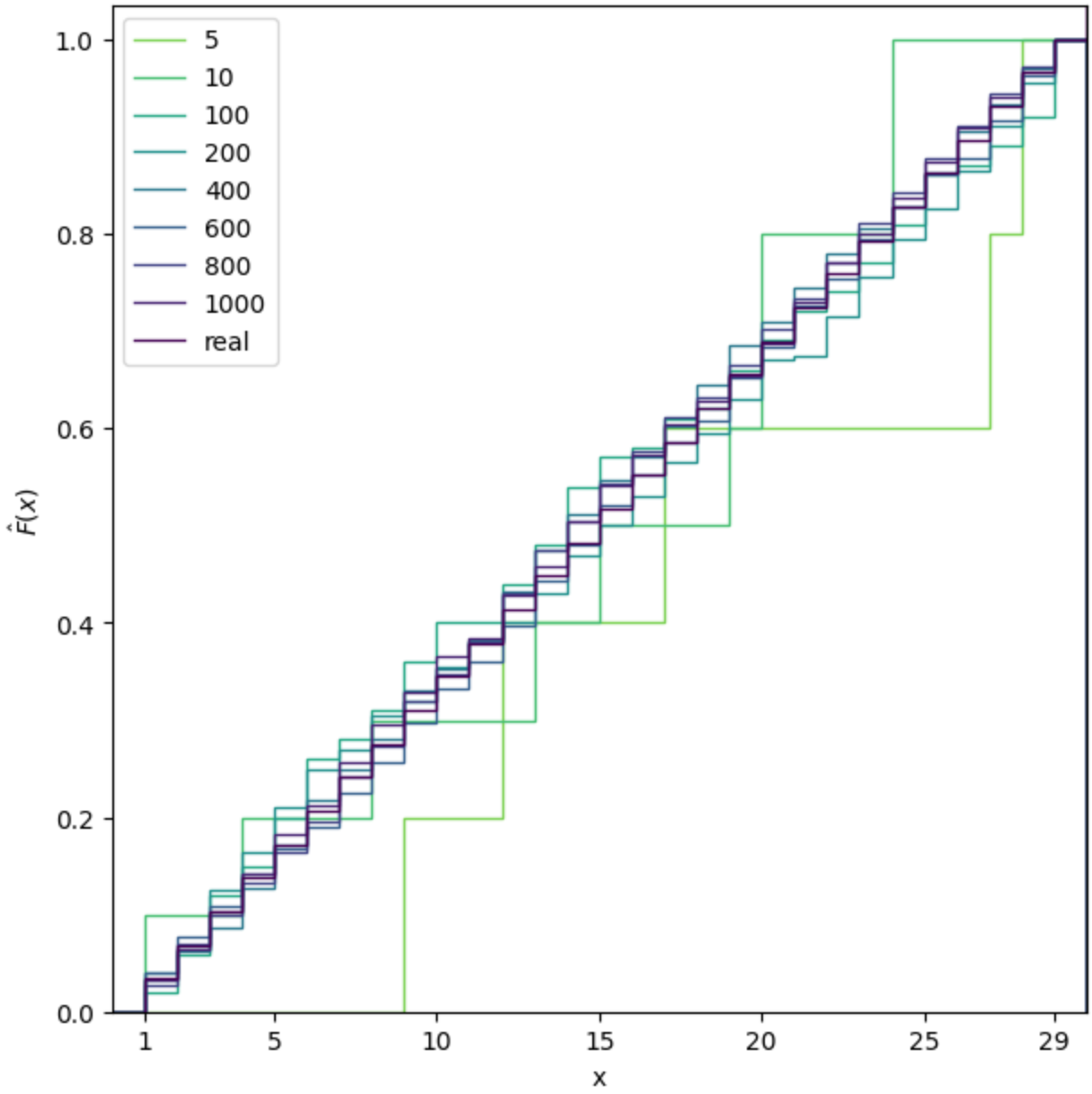
$$D_{m,n} = \sqrt{\frac{nm}{n+m}} \sup_{x \in \mathbb{R}} |F_n(x) - F_m(x)|$$

	5	10	100	200	400	600	800	1000
5	-	1.12	3.39	4.25	6.33	8.69	9.60	10.51
10	-	-	1.27	1.40	1.70	1.76	1.80	2.50
100	-	-	-	0.70	1.41	1.67	2.55	2.44
200	-	-	-	-	1.06	1.39	1.88	1.68
400	-	-	-	-	-	1.02	0.88	0.72
600	-	-	-	-	-	-	1.15	1.21
800	-	-	-	-	-	-	-	0.80
1000	-	-	-	-	-	-	-	-

In [14]:
```
# 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, $\theta = 29$



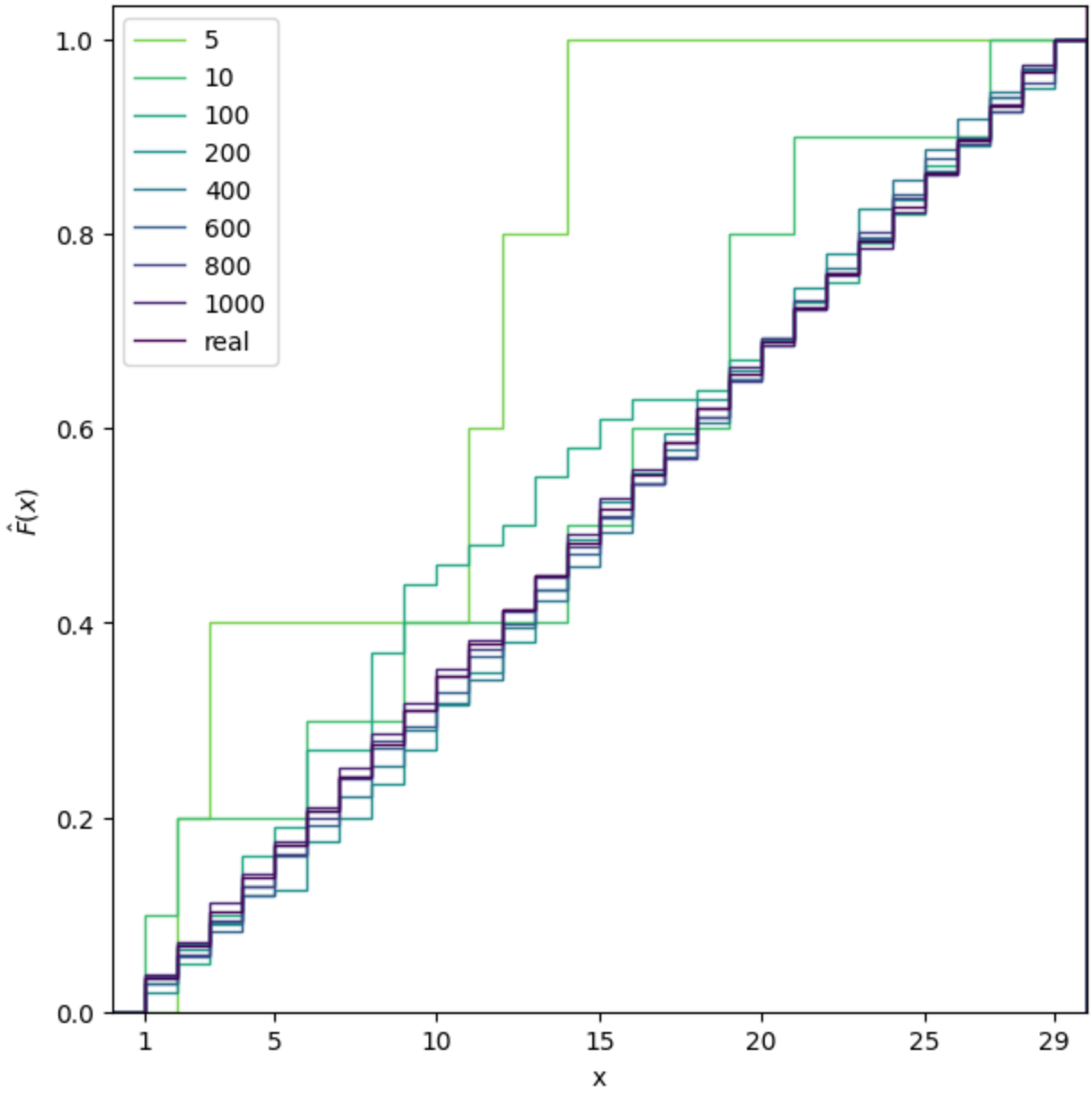
$$D_{m,n} = \sqrt{\frac{nm}{n+m}} \sup_{x \in \mathbb{R}} |F_n(x) - F_m(x)|$$

	5	10	100	200	400	600	800	1000
5	-	0.89	2.19	3.05	4.31	4.82	6.23	6.89
10	-	-	1.34	2.05	2.44	3.00	3.15	3.64
100	-	-	-	0.70	0.71	1.21	1.27	1.14
200	-	-	-	-	0.99	1.04	1.15	1.23
400	-	-	-	-	-	0.64	0.42	0.69
600	-	-	-	-	-	-	0.67	0.86
800	-	-	-	-	-	-	-	0.48
1000	-	-	-	-	-	-	-	-

In [15]:
```
# 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	-	1.12	2.97	5.15	7.67	9.15	10.43	11.36
10	-	-	1.20	1.55	2.47	3.09	3.38	3.98
100	-	-	-	1.70	2.12	2.54	2.57	2.73
200	-	-	-	-	0.67	0.72	0.88	1.14
400	-	-	-	-	-	0.39	0.60	0.91
600	-	-	-	-	-	-	0.36	0.66
800	-	-	-	-	-	-	-	0.43
1000	-	-	-	-	-	-	-	-

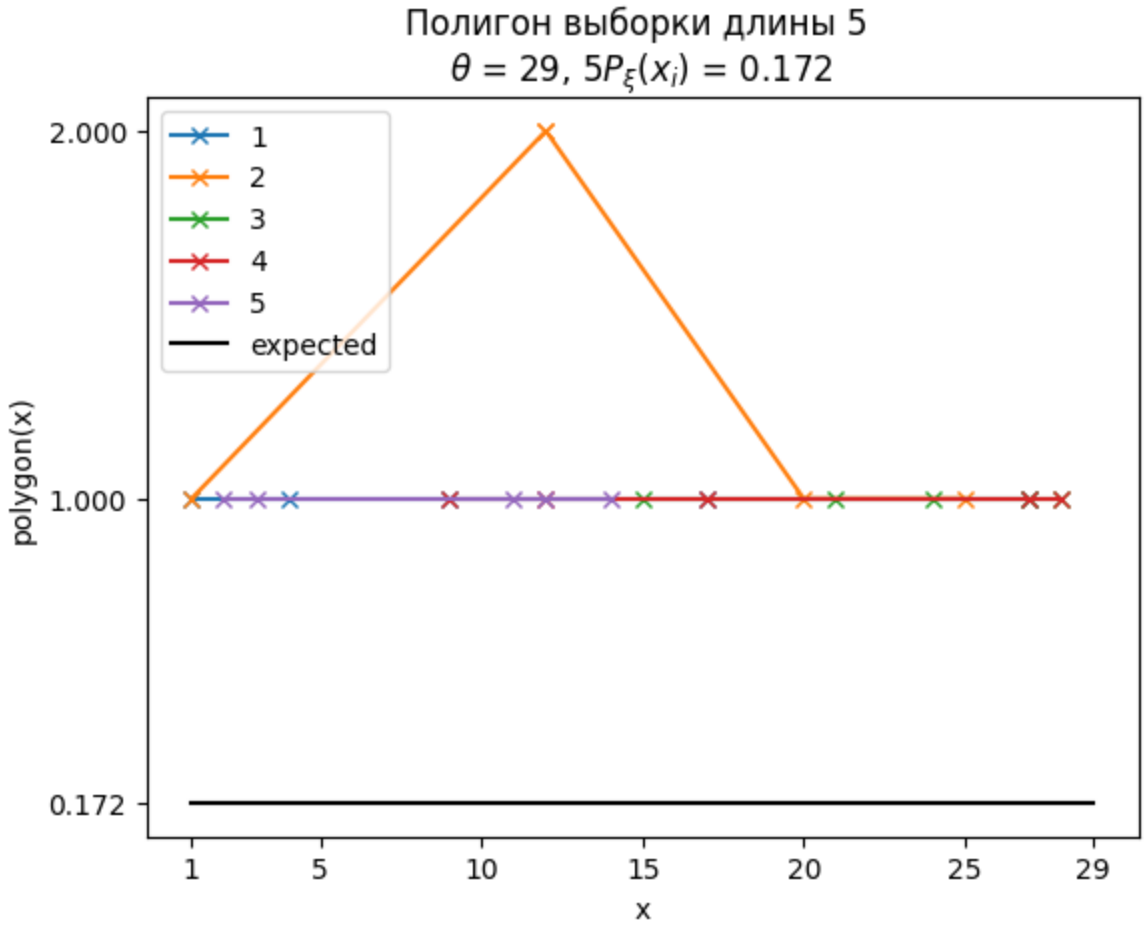
Д32Д, Задание 3

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

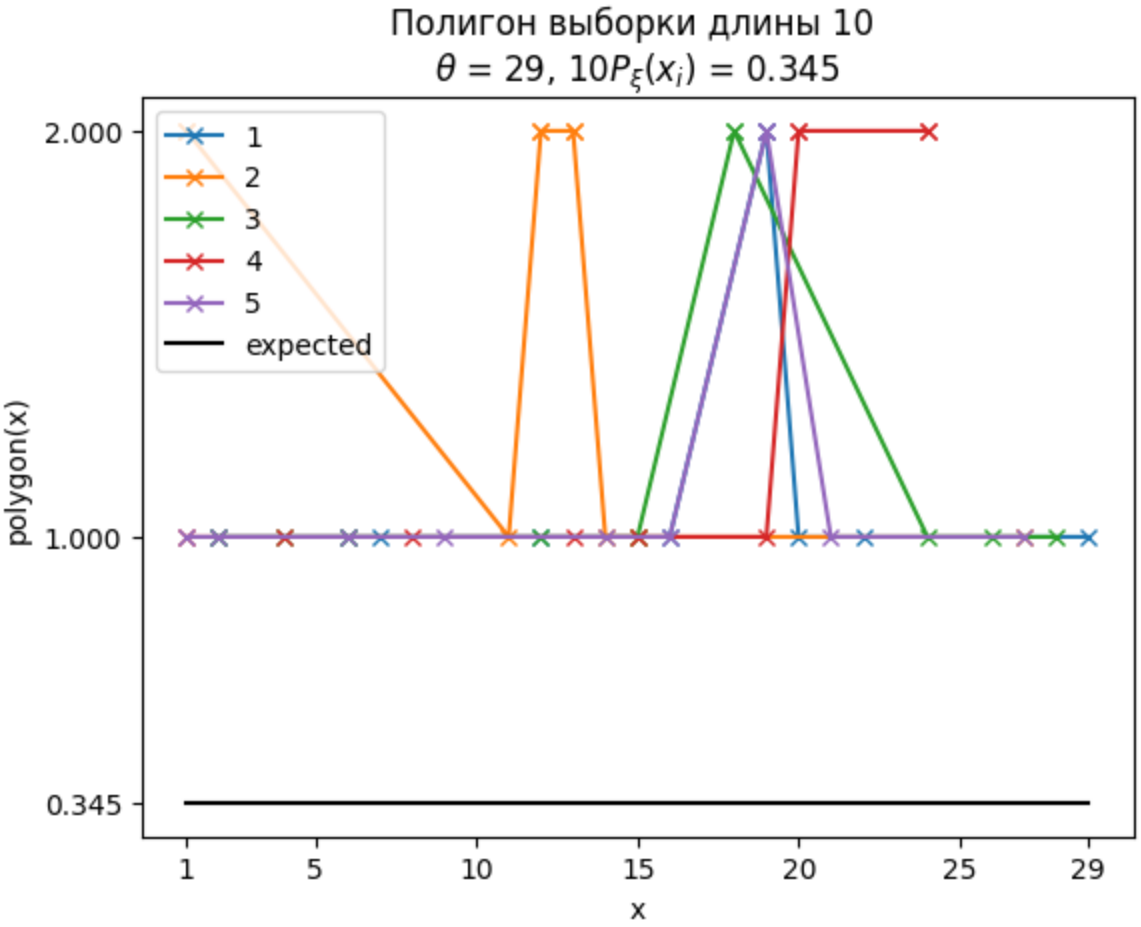
$\Rightarrow$ при $n \to \infty$ $k \to nP(x_i)$, что значит, что график вероятности подводим к полигону частот через домножение на количество элементов в выборке.

In [17]:
```
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)
```

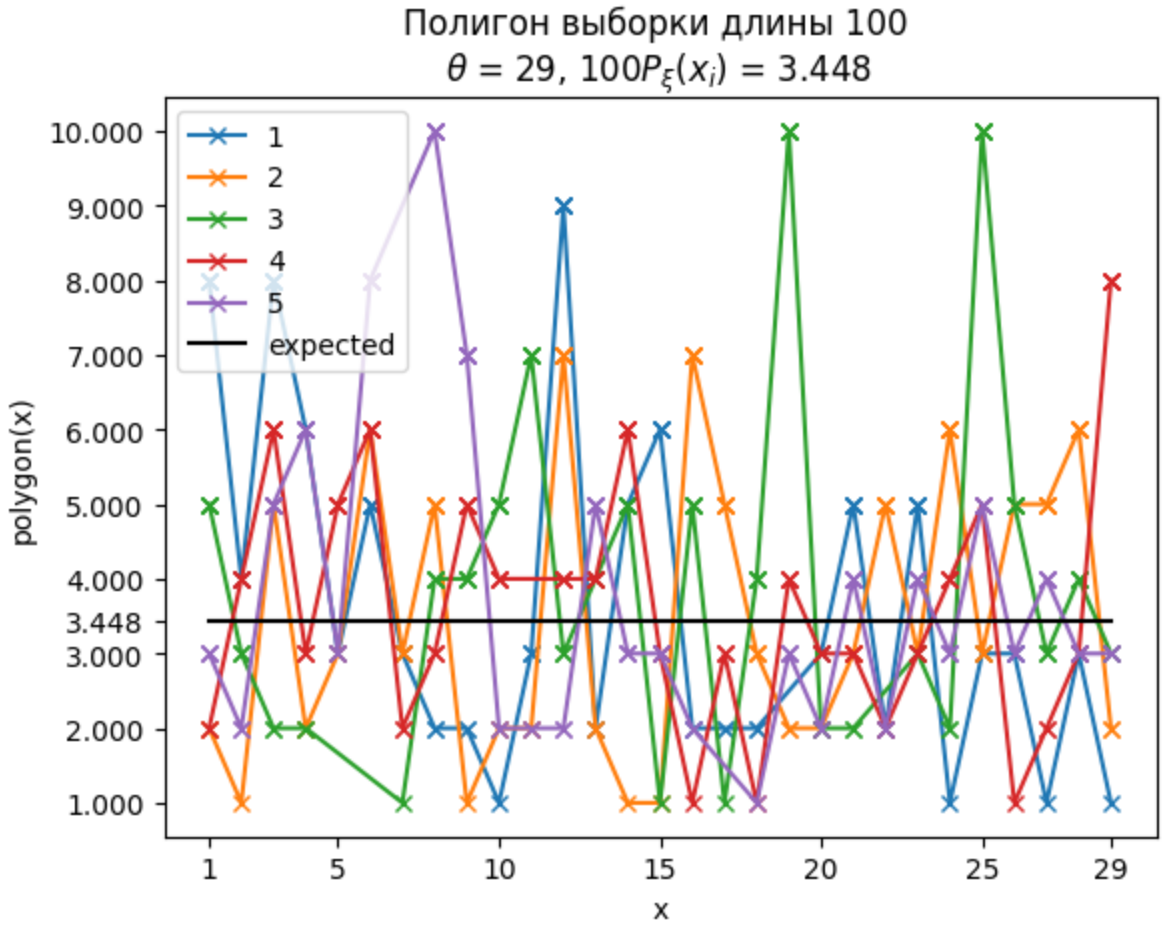
In [18]:
```
# n=5
for i in range(5):
    plt.plot(sample_xi[0][i], Y_polxi[0][i], '-x')
plt.plot(np.linspace(1, 29, 1000), posibilityxi*5, 'k')
plt.xticks([1]+list(range(5,26,5))+[29])
plt.yticks(np.append(np.arange(y_limsxi[0,0], y_limsxi[0,1]+1), 5/29))
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон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



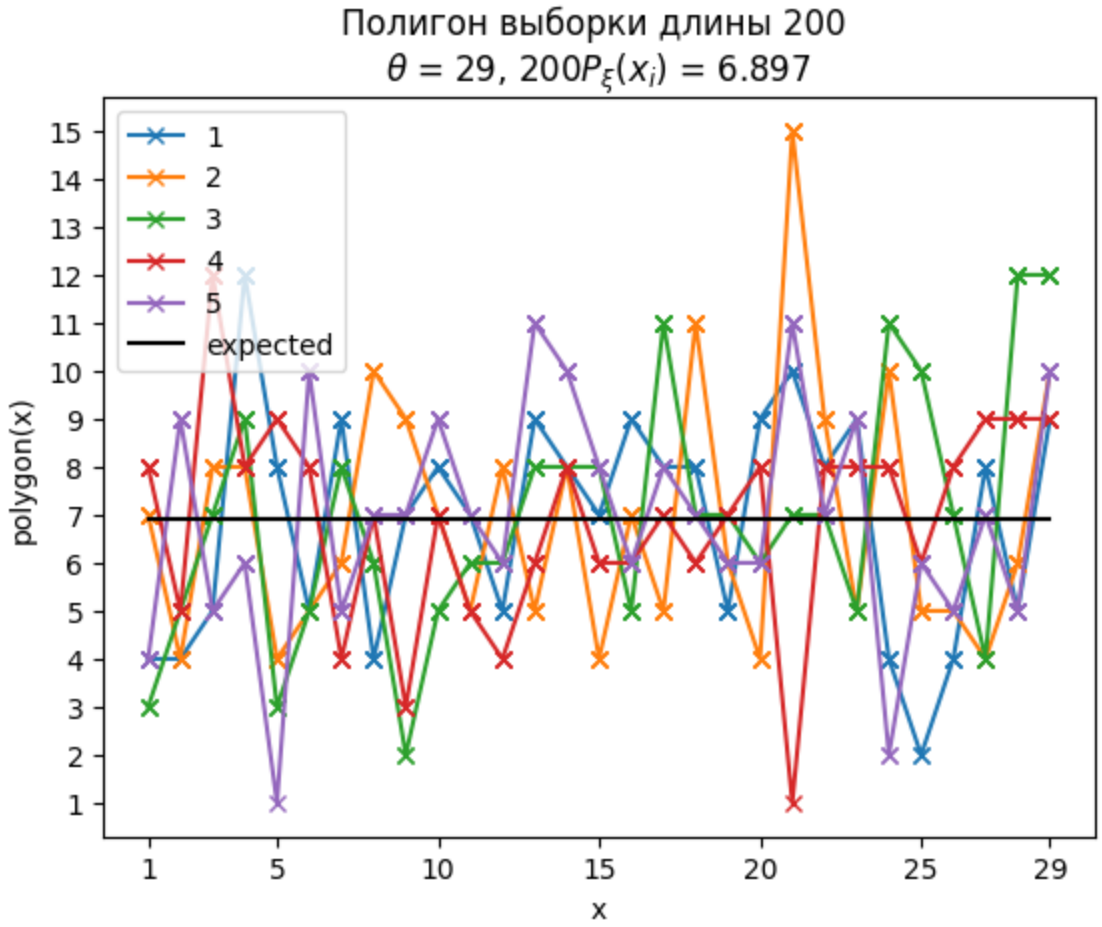
In [19]:
```
# n=10
for i in range(5):
    plt.plot(sample_xi[1][i], Y_polxi[1][i], '-x')
plt.plot(np.linspace(1, 29, 1000), posibilityxi*10, 'k')
plt.xticks([1]+list(range(5,26,5))+[29])
plt.yticks(np.append(np.arange(y_limsxi[1,0], y_limsxi[1,1]+1), 10/29))
plt.xlabel('x')
plt.ylabel('polygon(x)')
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');
```



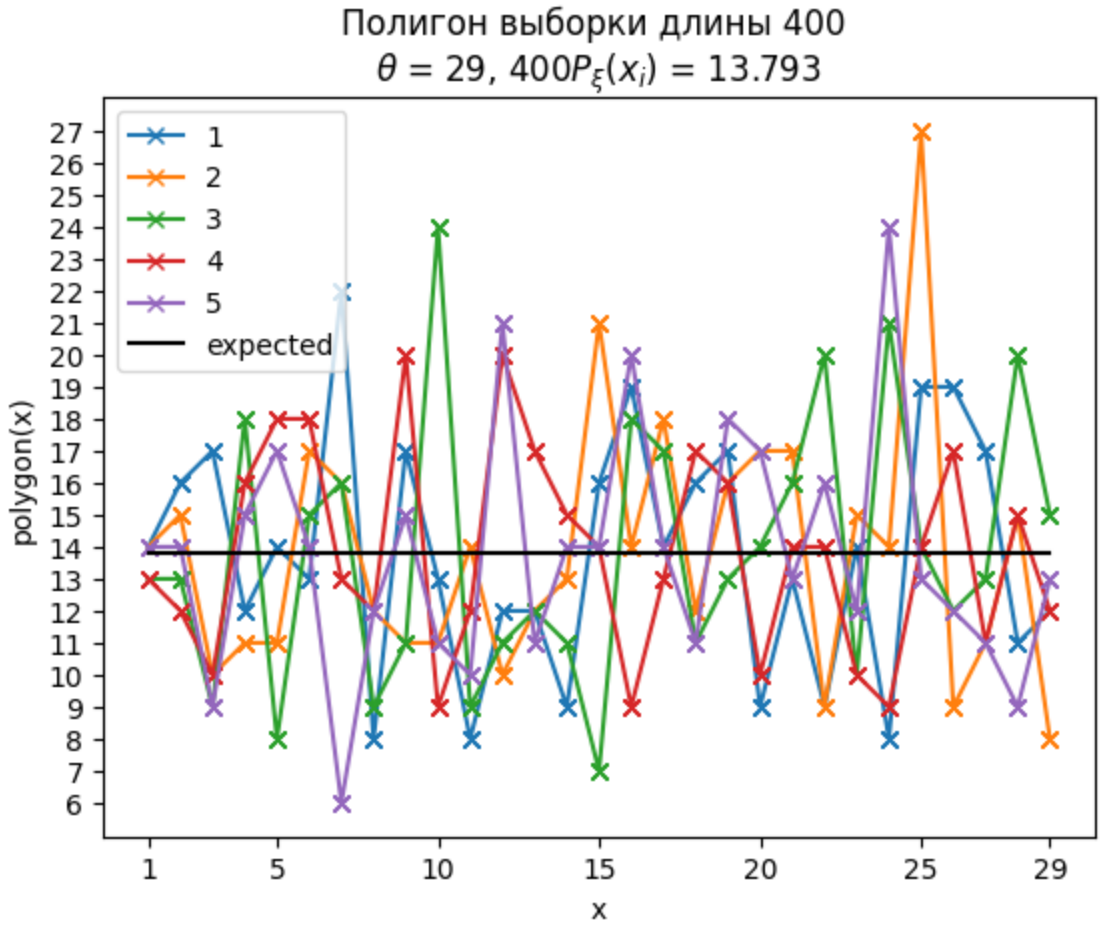
In [20]:
```
# n=100
for i in range(5):
    plt.plot(sample_xi[2][i], Y_polxi[2][i], '-x')
plt.plot(np.linspace(1, 29, 1000), posibilityxi*100, 'k')
plt.xticks([1]+list(range(5,26,5))+[29])
plt.yticks(np.append(np.arange(y_limsxi[2,0], y_limsxi[2,1]+1), 100/29))
plt.xlabel('x')
plt.ylabel('polygon(x)')
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');
```



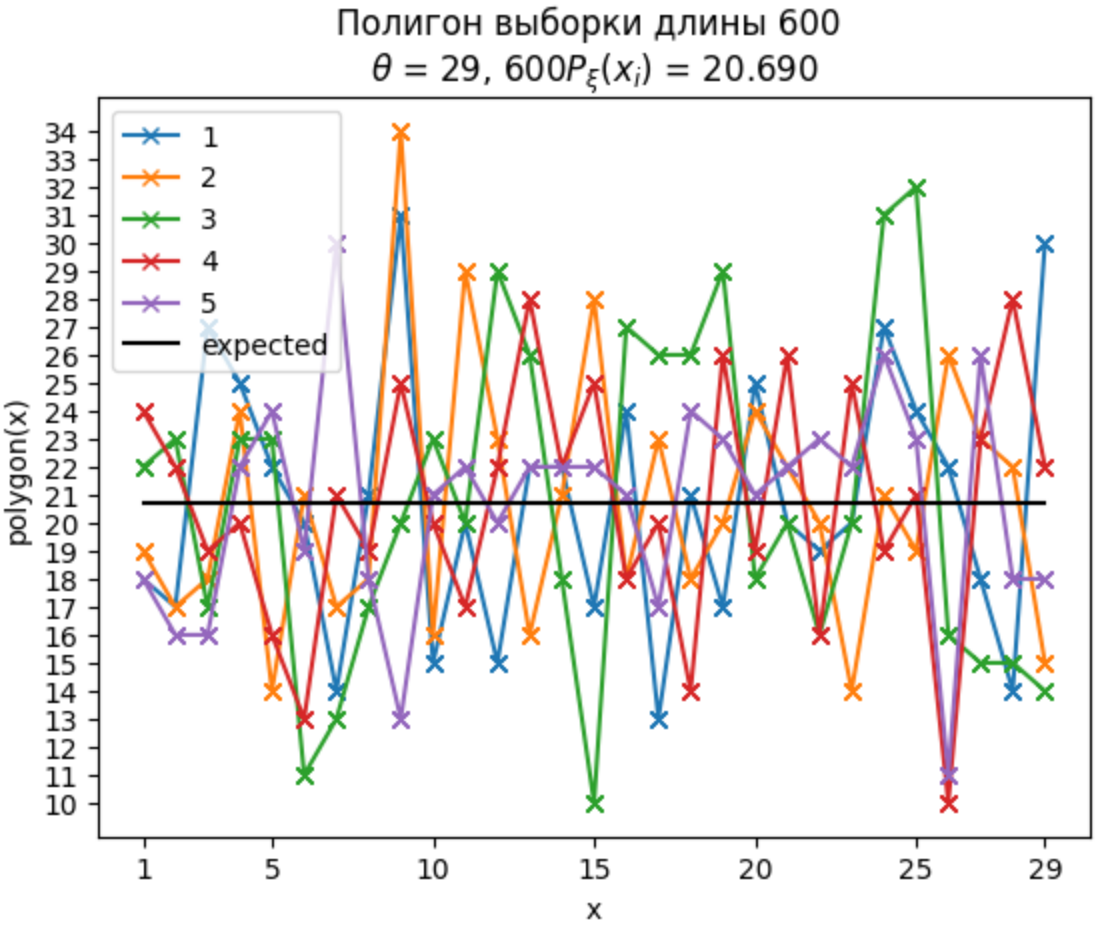
In [21]:
```
# n=200
for i in range(5):
    plt.plot(sample_xi[3][i], Y_polxi[3][i], '-x')
plt.plot(np.linspace(1, 29, 1000), posibilityxi*200, 'k')
plt.xticks([1]+list(range(5,26,5))+[29])
plt.yticks(np.arange(y_limsxi[3,0], y_limsxi[3,1]+1))
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 200 \n$\\theta$ = 29, $200P_{\\xi}(x_i)$ = %.3f'%
 (200/29))
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



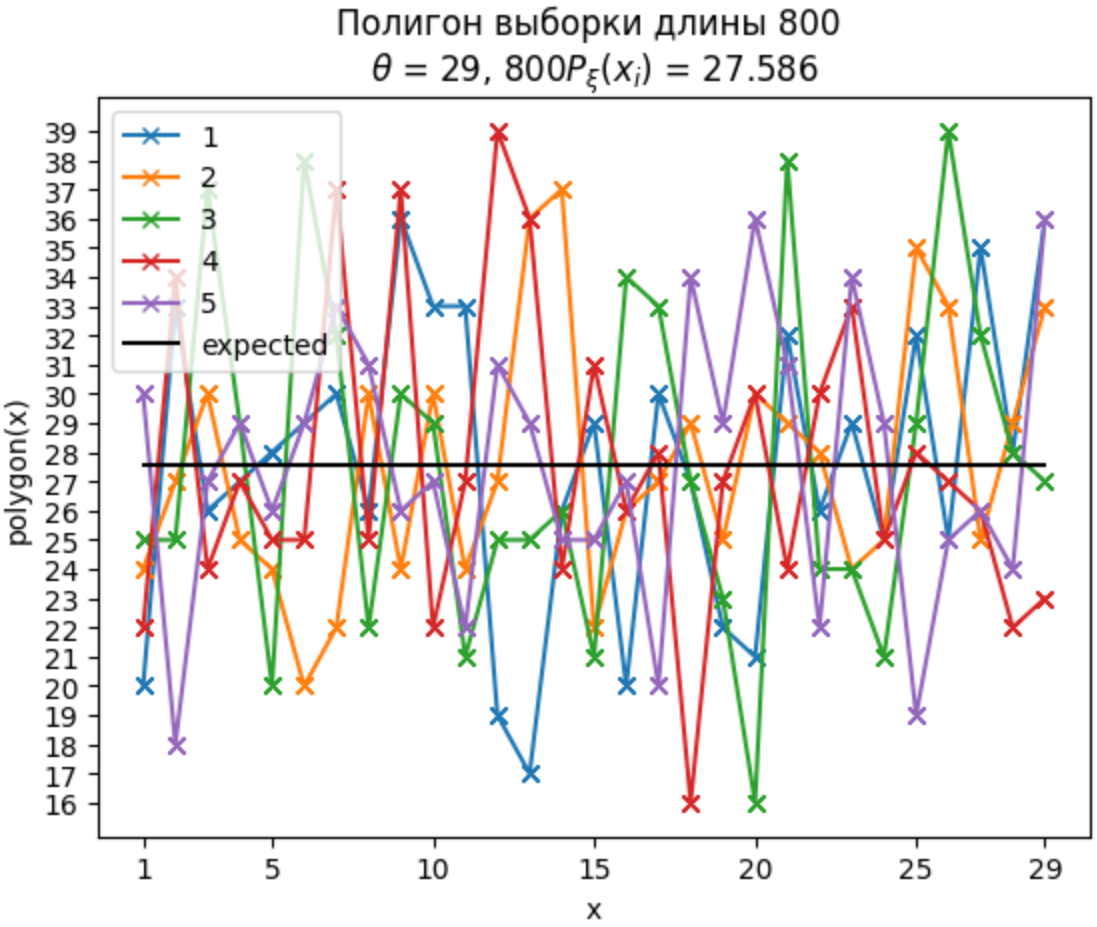
In [22]:
```
# n=400
for i in range(5):
    plt.plot(sample_xi[4][i], Y_polxi[4][i], '-x')
plt.plot(np.linspace(1, 29, 1000), posibilityxi*400, 'k')
plt.xticks([1]+list(range(5,26,5))+[29])
plt.yticks(np.arange(y_limsxi[4,0], y_limsxi[4,1]+1))
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 400 \n$\\theta$ = 29, $400P_{\\xi}(x_i)$ = %.3f'%
 (400/29))
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



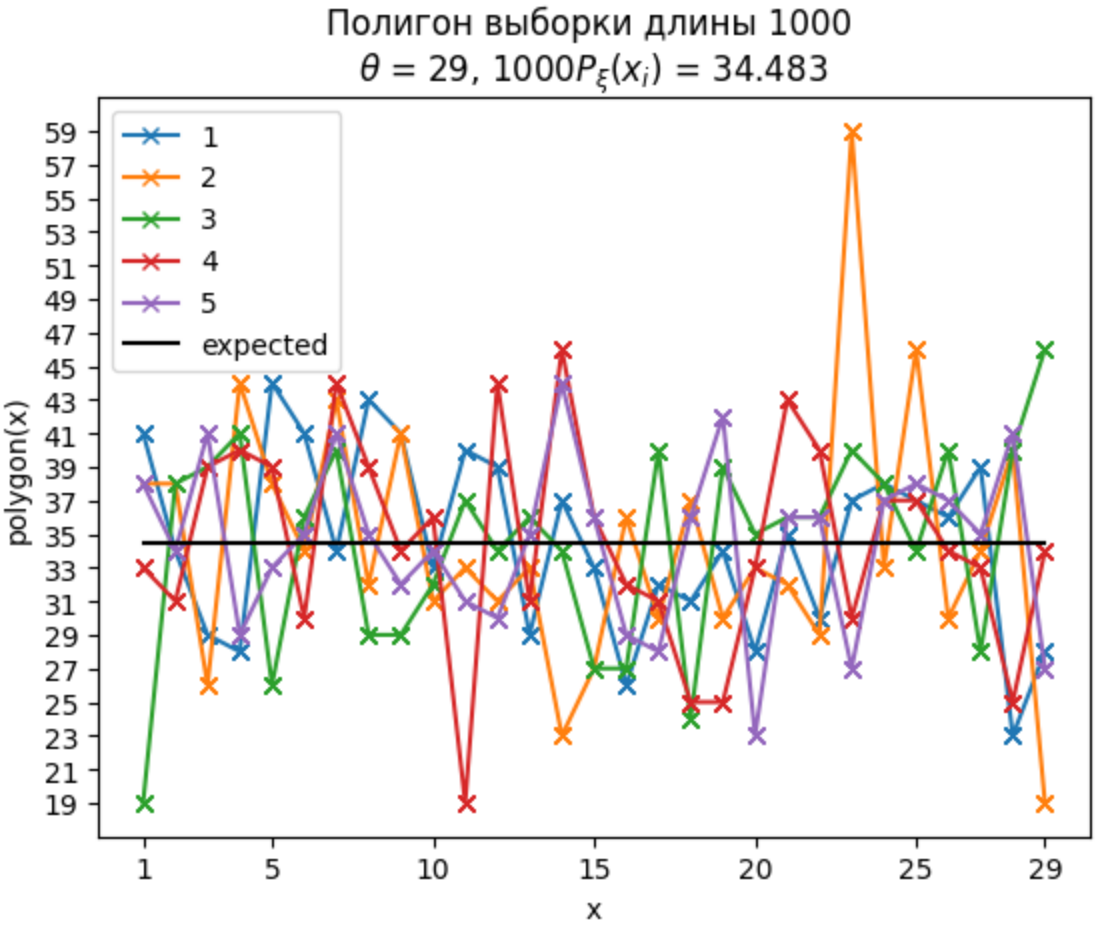
In [23]:
```
# n=600
for i in range(5):
    plt.plot(sample_xi[5][i], Y_polxi[5][i], '-x')
plt.plot(np.linspace(1, 29, 1000), posibilityxi*600, 'k')
plt.xticks([1]+list(range(5,26,5))+[29])
plt.yticks(np.arange(y_limsxi[5,0], y_limsxi[5,1]+1))
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон 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 [24]:
```
# 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 [25]:
```
# 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.

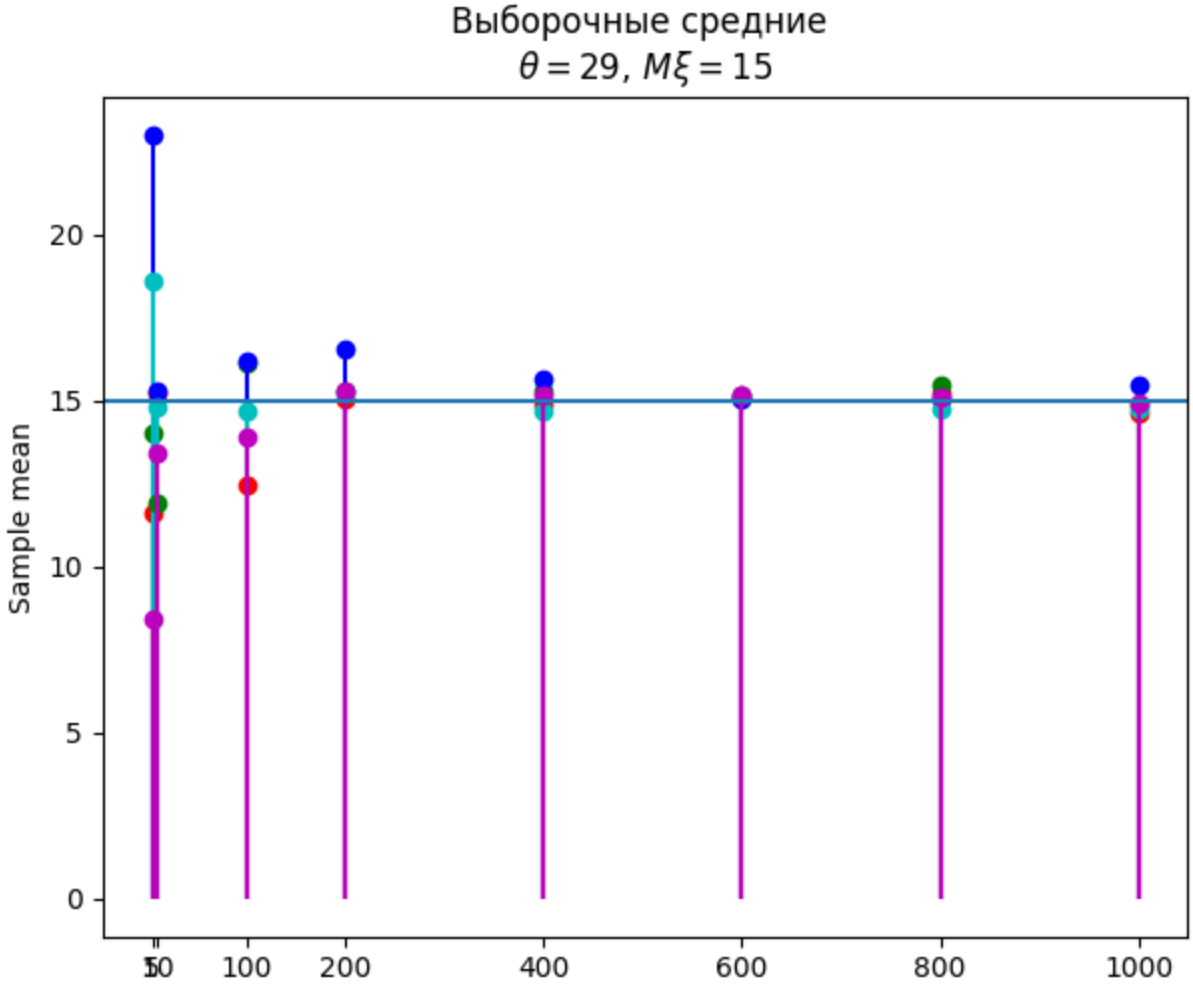
Д32Д, Задание 4

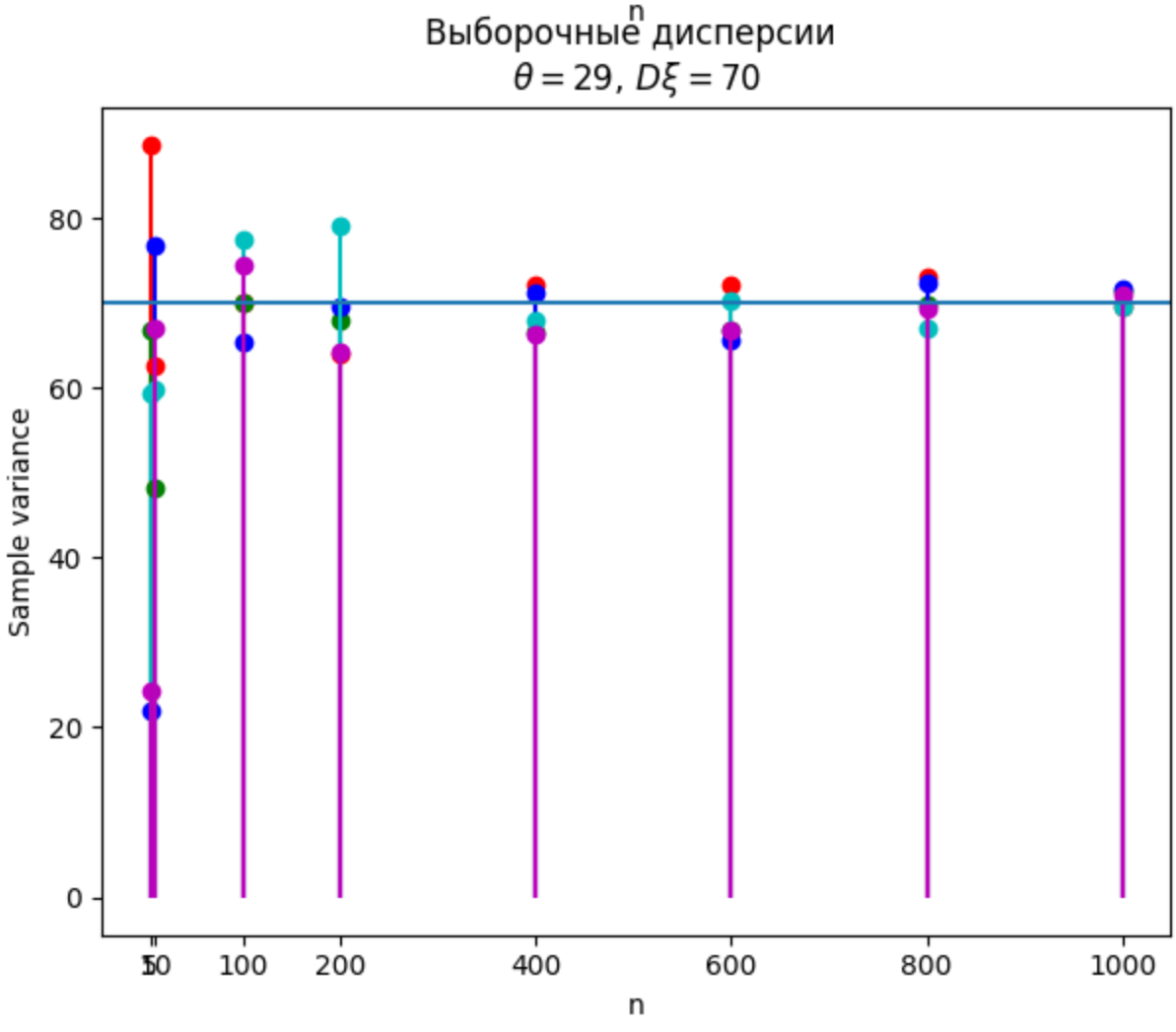
In [26]:
```
def xi_sample_mean(sample):
    return sum(sample)/len(sample)

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 [27]:
```
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');
```

In [110]:
```
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$\\theta = 29, \\,M\\xi = %d$'%expectatio
nxi)

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$\\theta = 29, \\,D\\xi = %d$'%variance
xi);
```


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


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

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 [30]:
```
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	-3.40	-1.00	8.00	3.60	-6.60
10	0.20	-3.10	0.30	-0.20	-1.60
100	-2.56	1.10	1.18	-0.33	-1.09
200	0.06	0.30	1.55	0.23	0.27
400	-0.12	0.26	0.64	-0.30	0.18
600	0.13	0.10	0.02	0.16	0.18
800	0.20	0.44	0.03	-0.26	0.10
1000	-0.39	-0.15	0.44	-0.27	-0.08

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

	1	2	3	4	5
5	18.64	-3.20	-48.00	-10.56	-45.76
10	-7.44	-21.71	6.81	-10.24	-2.96
100	-0.07	0.13	-4.51	7.50	4.50
200	-6.12	-2.02	-0.36	9.07	-5.76
400	2.13	-3.36	1.29	-2.00	-3.77
600	2.14	-3.14	-4.28	0.22	-3.12
800	2.94	-0.29	2.33	-2.94	-0.56
1000	-0.46	1.43	1.59	-0.37	1.06

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 [35]:
```
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 = ' ')
```

0.016 0.029 0.033 0.033 0.038 0.039 0.039 0.045 0.047 0.051 0.053 0.061 0.063 0.
065 0.067 0.067 0.067 0.068 0.068 0.073
0.075 0.076 0.077 0.077 0.081 0.081 0.083 0.083 0.083 0.084 0.087 0.087 0.088 0.
089 0.092 0.094 0.095 0.096 0.096 0.096
0.097 0.100 0.101 0.103 0.103 0.104 0.104 0.104 0.106 0.106 0.106 0.107 0.107 0.
108 0.108 0.108 0.108 0.109 0.111 0.112
0.112 0.112 0.112 0.112 0.113 0.113 0.113 0.116 0.118 0.118 0.119 0.119 0.121 0.
121 0.122 0.122 0.122 0.125 0.125 0.126
0.126 0.126 0.127 0.128 0.128 0.129 0.129 0.130 0.130 0.130 0.131 0.131 0.131 0.
132 0.132 0.133 0.133 0.133 0.133 0.133
0.134 0.135 0.136 0.136 0.136 0.137 0.137 0.139 0.140 0.142 0.142 0.144 0.144 0.
144 0.146 0.147 0.148 0.148 0.148 0.151
0.152 0.152 0.152 0.153 0.155 0.155 0.158 0.158 0.158 0.159 0.159 0.159 0.160 0.
160 0.160 0.162 0.162 0.163 0.163 0.163
0.165 0.166 0.166 0.167 0.167 0.167 0.168 0.168 0.169 0.170 0.171 0.171 0.171 0.
172 0.172 0.172 0.173 0.174 0.176 0.176
0.176 0.176 0.177 0.177 0.177 0.177 0.179 0.179 0.180 0.181 0.181 0.183 0.183 0.
183 0.184 0.184 0.184 0.185 0.188 0.189
0.189 0.190 0.191 0.192 0.193 0.193 0.194 0.194 0.194 0.196 0.196 0.197 0.198 0.
198 0.198 0.199 0.199 0.199 0.199 0.199
0.200 0.200 0.200 0.204 0.204 0.204 0.205 0.205 0.206 0.207 0.208 0.208 0.209 0.
209 0.210 0.210 0.210 0.211 0.211 0.211
0.212 0.212 0.212 0.212 0.213 0.213 0.214 0.214 0.216 0.217 0.217 0.217 0.218 0.
219 0.219 0.219 0.219 0.220 0.220 0.220
0.221 0.221 0.221 0.221 0.222 0.222 0.223 0.224 0.224 0.225 0.226 0.226 0.227 0.
227 0.227 0.227 0.227 0.227 0.228 0.228
0.228 0.230 0.230 0.232 0.233 0.233 0.233 0.233 0.233 0.236 0.238 0.238 0.238 0.
238 0.239 0.239 0.239 0.240 0.241 0.242
0.243 0.243 0.243 0.243 0.244 0.244 0.244 0.245 0.245 0.245 0.246 0.246 0.247 0.
247 0.247 0.248 0.249 0.249 0.250 0.251
0.251 0.252 0.252 0.252 0.253 0.253 0.254 0.255 0.255 0.255 0.256 0.256 0.256 0.
257 0.257 0.258 0.258 0.259 0.260 0.260
0.261 0.261 0.262 0.262 0.263 0.263 0.263 0.263 0.264 0.265 0.266 0.266 0.269 0.
270 0.270 0.270 0.271 0.272 0.272 0.272
0.277 0.277 0.277 0.277 0.278 0.279 0.280 0.280 0.281 0.284 0.285 0.285 0.285 0.
285 0.286 0.286 0.286 0.286 0.287 0.287
0.288 0.289 0.290 0.290 0.290 0.292 0.293 0.293 0.293 0.294 0.294 0.294 0.294 0.
294 0.295 0.295 0.295 0.295 0.295 0.296
0.296 0.296 0.299 0.299 0.299 0.299 0.300 0.300 0.301 0.301 0.303 0.303 0.304 0.
304 0.304 0.305 0.305 0.306 0.307 0.307
0.307 0.309 0.309 0.309 0.310 0.310 0.311 0.311 0.312 0.312 0.312 0.312 0.313 0.
317 0.317 0.317 0.317 0.318 0.319 0.319
0.320 0.321 0.322 0.323 0.324 0.324 0.325 0.327 0.328 0.328 0.329 0.330 0.330 0.
334 0.334 0.334 0.335 0.335 0.335 0.336
0.336 0.338 0.338 0.339 0.339 0.339 0.340 0.340 0.340 0.340 0.341 0.341 0.342 0.
344 0.344 0.344 0.345 0.345 0.345 0.345
0.346 0.346 0.347 0.349 0.349 0.350 0.351 0.351 0.352 0.352 0.355 0.355 0.355 0.
355 0.356 0.356 0.357 0.357 0.358 0.359
0.360 0.360 0.360 0.361 0.362 0.364 0.364 0.364 0.364 0.365 0.366 0.366 0.366 0.
366 0.366 0.367 0.368 0.368 0.368 0.369
0.370 0.370 0.370 0.370 0.370 0.371 0.372 0.372 0.372 0.373 0.374 0.374 0.374 0.
374 0.374 0.374 0.377 0.377 0.378 0.378
0.378 0.379 0.379 0.380 0.380 0.381 0.382 0.382 0.382 0.383 0.385 0.385 0.386 0.
386 0.386 0.387 0.387 0.388 0.388 0.388
0.388 0.388 0.389 0.390 0.390 0.391 0.392 0.394 0.394 0.394 0.395 0.395 0.395 0.
395 0.396 0.397 0.399 0.400 0.400 0.401
0.401 0.401 0.402 0.403 0.403 0.404 0.404 0.405 0.405 0.406 0.406 0.406 0.406 0.
407 0.409 0.413 0.413 0.414 0.416 0.416
0.416 0.418 0.419 0.419 0.422 0.422 0.423 0.423 0.425 0.425 0.426 0.426 0.427 0.
427 0.427 0.428 0.428 0.429 0.429 0.429
0.429 0.429 0.430 0.431 0.432 0.432 0.432 0.434 0.435 0.435 0.435 0.436 0.436 0.

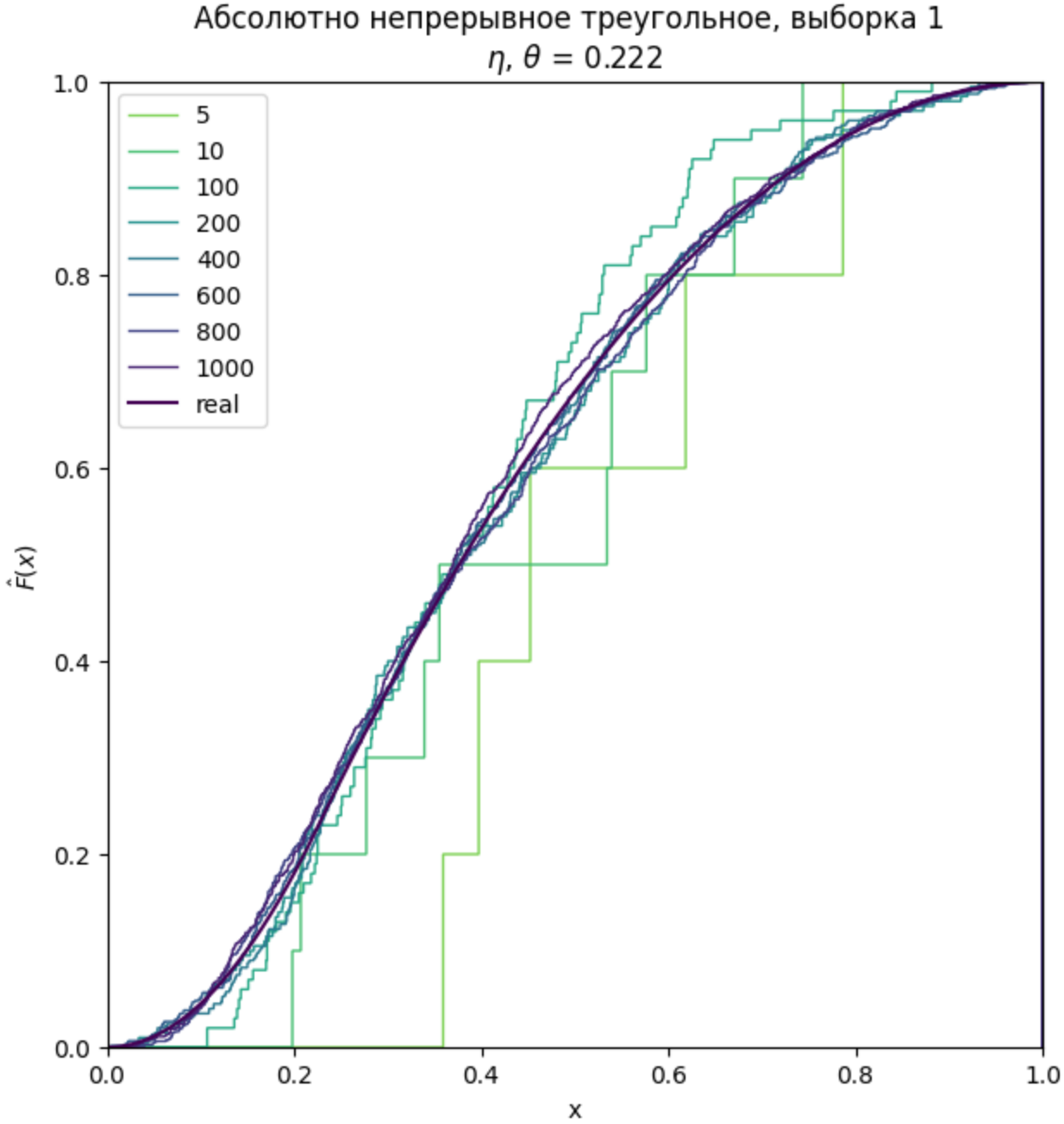
437 0.437 0.438 0.438 0.440 0.441 0.442
0.443 0.444 0.444 0.444 0.445 0.445 0.445 0.446 0.446 0.447 0.447 0.448 0.448 0.
448 0.450 0.450 0.452 0.453 0.453 0.454
0.454 0.454 0.454 0.456 0.456 0.457 0.457 0.458 0.459 0.459 0.459 0.460 0.460 0.
461 0.461 0.461 0.462 0.463 0.463 0.464
0.465 0.466 0.467 0.469 0.470 0.470 0.470 0.471 0.472 0.472 0.472 0.473 0.473 0.
475 0.475 0.475 0.476 0.481 0.482 0.482
0.482 0.483 0.484 0.484 0.484 0.486 0.487 0.487 0.489 0.490 0.492 0.492 0.495 0.
495 0.495 0.496 0.497 0.497 0.497 0.499
0.499 0.500 0.501 0.502 0.504 0.505 0.505 0.506 0.507 0.507 0.507 0.510 0.511 0.
511 0.515 0.515 0.515 0.515 0.515 0.516
0.517 0.517 0.518 0.518 0.519 0.520 0.520 0.521 0.523 0.523 0.524 0.526 0.526 0.
529 0.529 0.530 0.530 0.534 0.536 0.536
0.537 0.537 0.538 0.541 0.541 0.542 0.542 0.544 0.544 0.544 0.544 0.545 0.546 0.
547 0.547 0.547 0.548 0.548 0.549 0.551
0.551 0.552 0.553 0.557 0.558 0.560 0.560 0.562 0.562 0.563 0.565 0.565 0.566 0.
568 0.569 0.570 0.570 0.571 0.571 0.574
0.575 0.577 0.580 0.580 0.580 0.580 0.585 0.586 0.588 0.589 0.589 0.589 0.590 0.
591 0.592 0.592 0.593 0.593 0.595 0.597
0.597 0.599 0.600 0.601 0.601 0.602 0.602 0.604 0.605 0.605 0.607 0.609 0.610 0.
611 0.612 0.613 0.613 0.616 0.617 0.617
0.618 0.618 0.619 0.621 0.621 0.622 0.622 0.623 0.625 0.626 0.626 0.627 0.628 0.
630 0.633 0.633 0.634 0.634 0.634 0.636
0.637 0.637 0.638 0.639 0.640 0.644 0.646 0.646 0.646 0.647 0.650 0.650 0.651 0.
651 0.653 0.654 0.656 0.656 0.657 0.658
0.659 0.660 0.660 0.661 0.661 0.663 0.663 0.666 0.666 0.669 0.669 0.675 0.677 0.
678 0.679 0.681 0.681 0.685 0.686 0.688
0.688 0.691 0.693 0.694 0.695 0.696 0.698 0.698 0.700 0.700 0.700 0.703 0.703 0.
704 0.707 0.707 0.708 0.708 0.708 0.708
0.709 0.712 0.713 0.716 0.718 0.722 0.723 0.725 0.726 0.727 0.732 0.732 0.736 0.
739 0.741 0.742 0.743 0.743 0.747 0.747
0.749 0.750 0.753 0.754 0.755 0.759 0.759 0.763 0.763 0.766 0.767 0.769 0.770 0.
773 0.774 0.777 0.779 0.780 0.780 0.781
0.784 0.786 0.786 0.787 0.788 0.790 0.791 0.796 0.800 0.800 0.801 0.801 0.804 0.
805 0.805 0.806 0.807 0.810 0.812 0.813
0.815 0.817 0.819 0.825 0.832 0.832 0.833 0.835 0.835 0.836 0.842 0.846 0.851 0.
851 0.857 0.857 0.858 0.859 0.860 0.863
0.869 0.875 0.878 0.882 0.882 0.882 0.892 0.897 0.903 0.906 0.912 0.914 0.930 0.
939 0.941 0.946 0.955 0.960 0.962 0.980

Д32А, Задание 2

In [39]:

```
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=(1/4.5)):
    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 [48]:
```
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$\\eta , \\,\\th
eta$ = %.3f'%(1/4.5))
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'$|F_n(x)-F_m(x)|$');
```

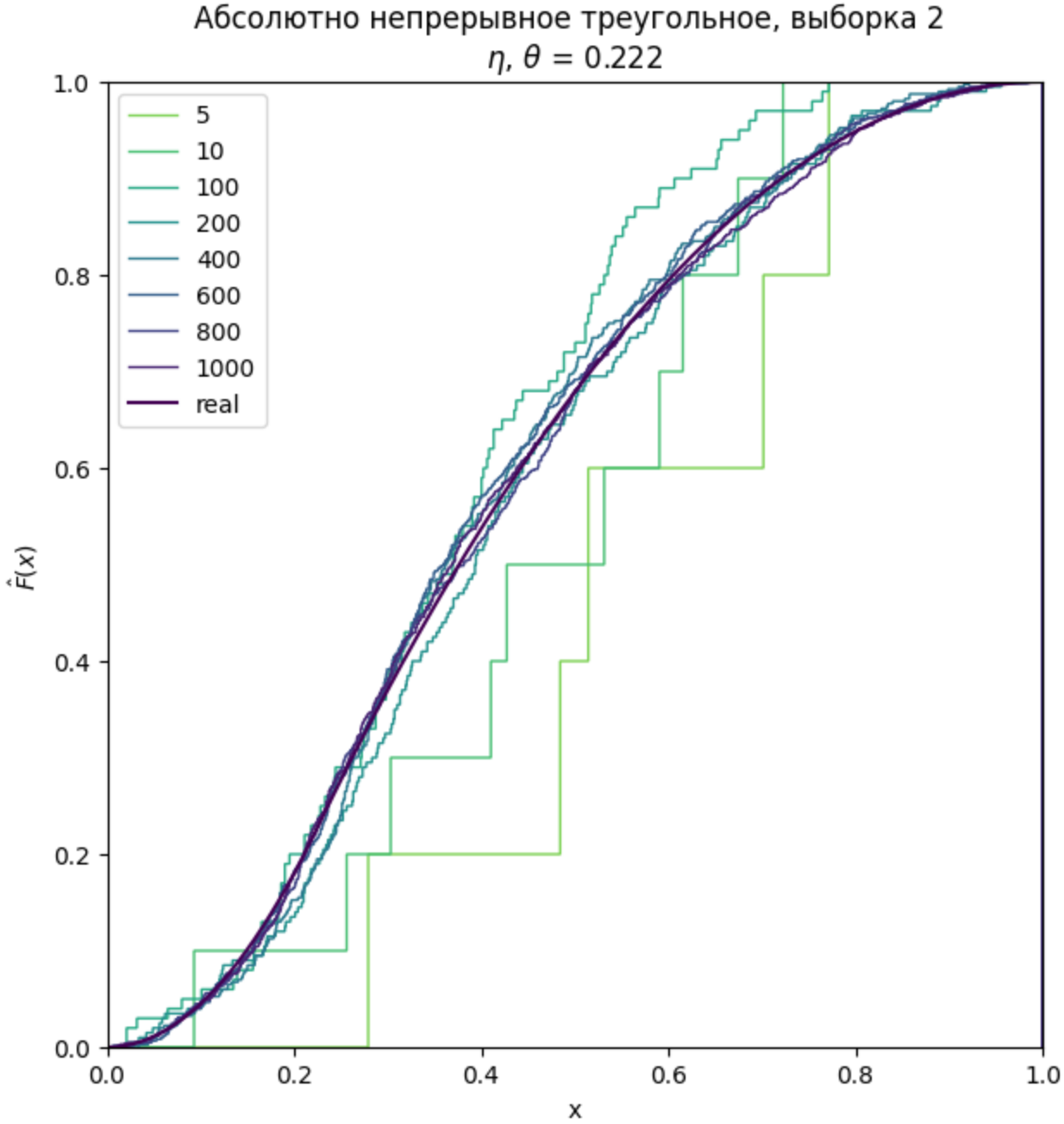


$$D_{m,n} = \sqrt{\frac{nm}{n+m}} \sup_{x \in \mathbb{R}} |F_n(x) - F_m(x)|$$

	5	10	100	200	400	600	800	1000
5	-	0.67	1.34	3.10	3.78	4.76	5.62	5.99
10	-	-	1.91	1.35	2.12	2.74	3.25	3.26
100	-	-	-	1.60	2.30	2.77	3.10	2.86
200	-	-	-	-	0.78	1.13	1.32	1.27
400	-	-	-	-	-	0.65	0.67	1.02
600	-	-	-	-	-	-	0.41	0.86
800	-	-	-	-	-	-	-	0.77
1000	-	-	-	-	-	-	-	-

In [49]:

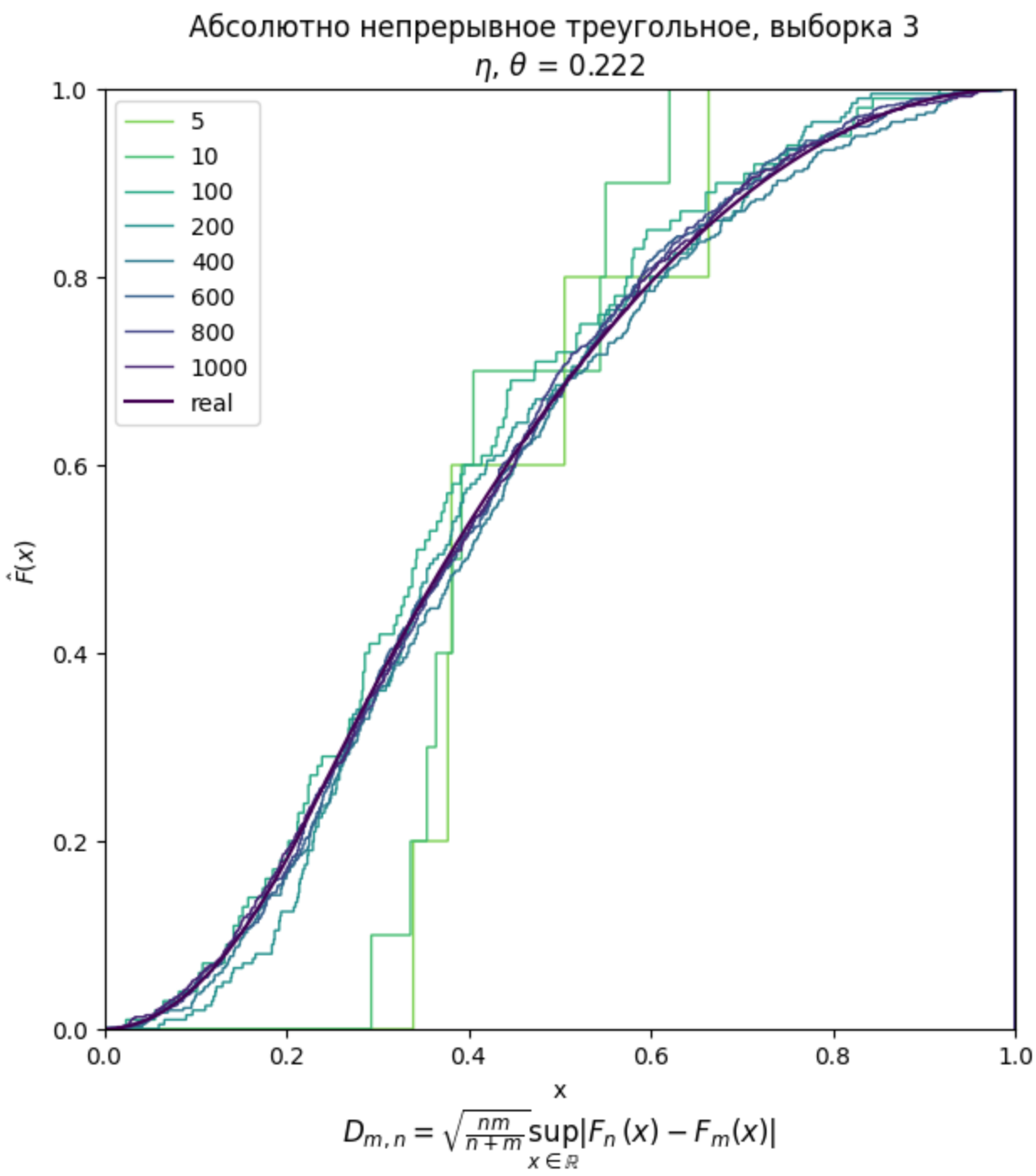
```
# graph[1]
fig, ax = plt.subplots(2,1, figsize=(7,12), height_ratios = [2,1])
for i in range(8):
    ax[0].stairs(Yeta[i][1], np.append(X_realeta, 1), color = colors[i])
ax[0].plot(X_realeta, Y_realeta, color = '#440154')
ax[0].set(xmargin = 0, ymargin = 0, xlabel = 'x', ylabel = r'$\hat{F}(x)$',
          title = 'Абсолютно непрерывное треугольное, выборка 2 \n$\\eta , \\,\\th
eta$ = %.3f'%(1/4.5))
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	-	0.67	1.48	1.95	3.11	3.18	4.33	4.43
10	-	-	3.32	4.10	5.73	6.50	7.95	8.68
100	-	-	-	0.80	1.13	1.65	1.45	1.83
200	-	-	-	-	0.46	0.75	0.87	1.14
400	-	-	-	-	-	0.64	0.85	1.16
600	-	-	-	-	-	-	0.66	1.04
800	-	-	-	-	-	-	-	1.15
1000	-	-	-	-	-	-	-	-

In [50]:
```
# 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$\\eta , \\,\\th
eta$ = %.3f'%(1/4.5))
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'$|F_n(x)-F_m(x)|$');
```

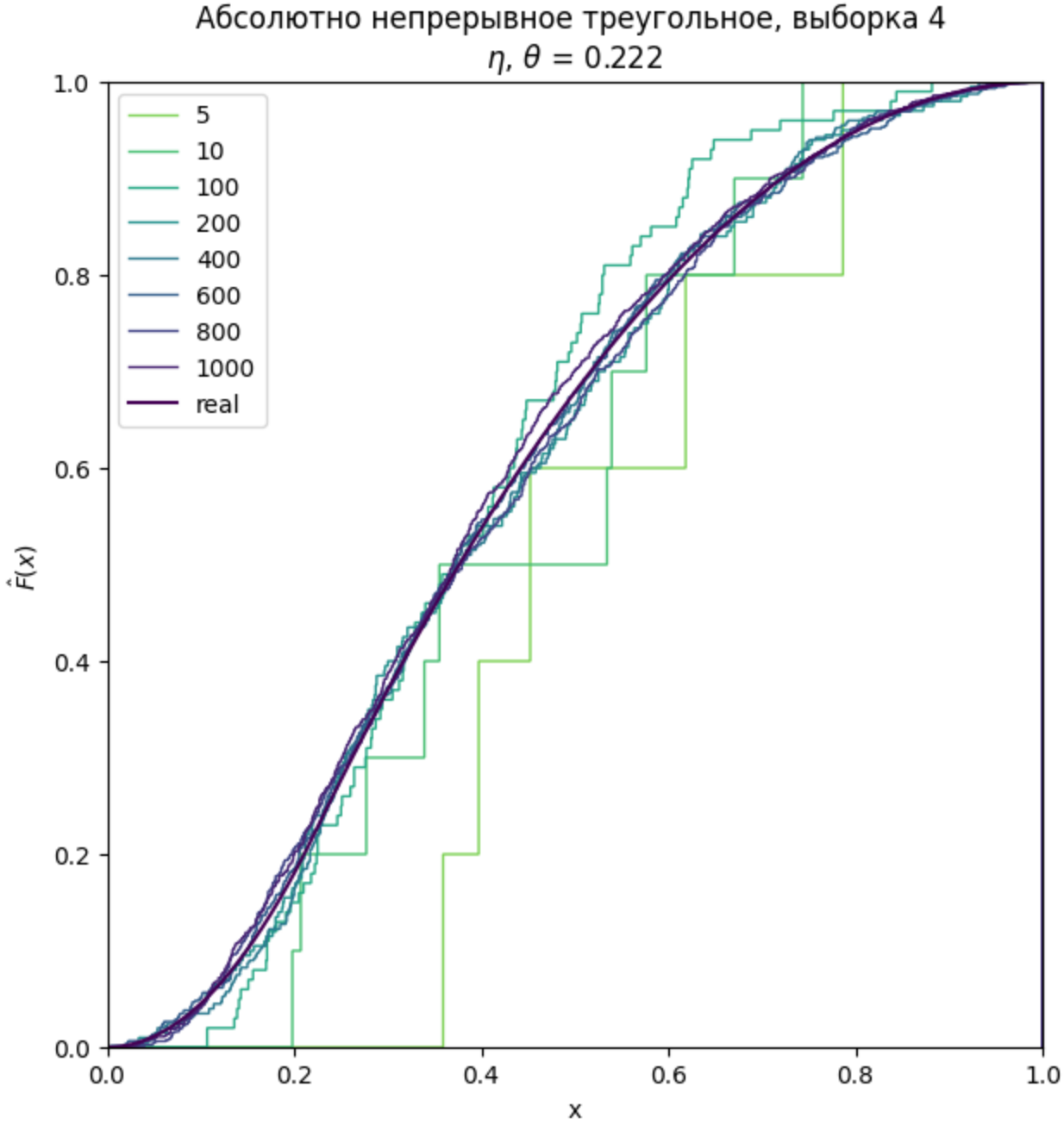


$$D_{m,n} = \sqrt{\frac{nm}{n+m}} \sup_{x \in \mathbb{R}} |F_n(x) - F_m(x)|$$

	5	10	100	200	400	600	800	1000
5	-	1.12	3.39	4.25	6.33	8.69	9.60	10.51
10	-	-	1.27	1.40	1.70	1.76	1.80	2.50
100	-	-	-	0.70	1.41	1.67	2.55	2.44
200	-	-	-	-	1.06	1.39	1.88	1.68
400	-	-	-	-	-	1.02	0.88	0.72
600	-	-	-	-	-	-	1.15	1.21
800	-	-	-	-	-	-	-	0.80
1000	-	-	-	-	-	-	-	-

In [51]:

```
# 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$\\eta , \\,\\th
eta$ = %.3f'%(1/4.5))
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'$|F_n(x)-F_m(x)|$');
```

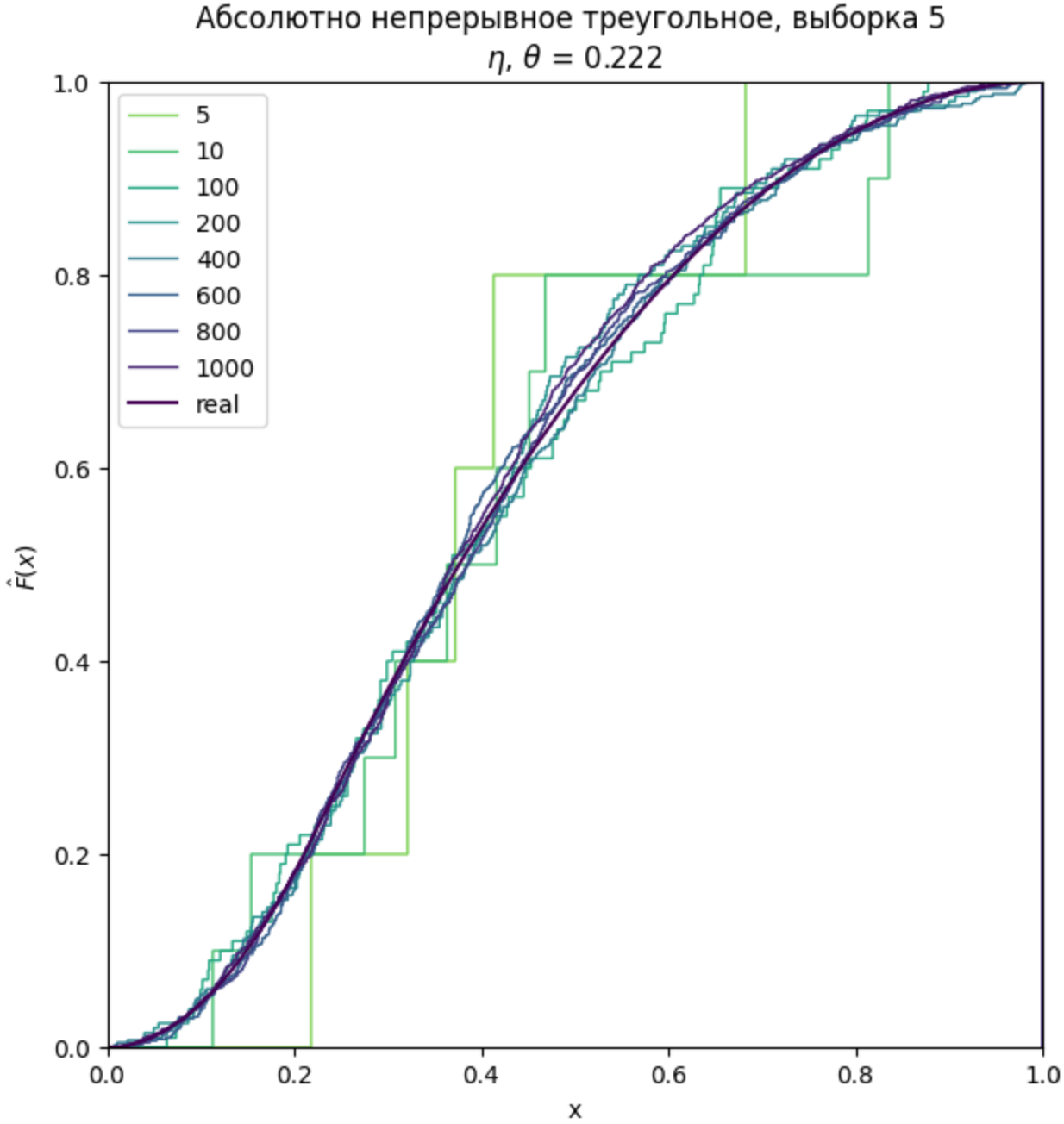


$$D_{m,n} = \sqrt{\frac{nm}{n+m}} \sup_{x \in \mathbb{R}} |F_n(x) - F_m(x)|$$

	5	10	100	200	400	600	800	1000
5	-	0.89	2.19	3.05	4.31	4.82	6.23	6.89
10	-	-	1.34	2.05	2.44	3.00	3.15	3.64
100	-	-	-	0.70	0.71	1.21	1.27	1.14
200	-	-	-	-	0.99	1.04	1.15	1.23
400	-	-	-	-	-	0.64	0.42	0.69
600	-	-	-	-	-	-	0.67	0.86
800	-	-	-	-	-	-	-	0.48
1000	-	-	-	-	-	-	-	-

In [52]:

```
# 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$\\eta , \\,\\th
eta$ = %.3f'%(1/4.5))
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	-	1.12	2.97	5.15	7.67	9.15	10.43	11.36
10	-	-	1.20	1.55	2.47	3.09	3.38	3.98
100	-	-	-	1.70	2.12	2.54	2.57	2.73
200	-	-	-	-	0.67	0.72	0.88	1.14
400	-	-	-	-	-	0.39	0.60	0.91
600	-	-	-	-	-	-	0.36	0.66
800	-	-	-	-	-	-	-	0.43
1000	-	-	-	-	-	-	-	-

Д32А, Задание 3

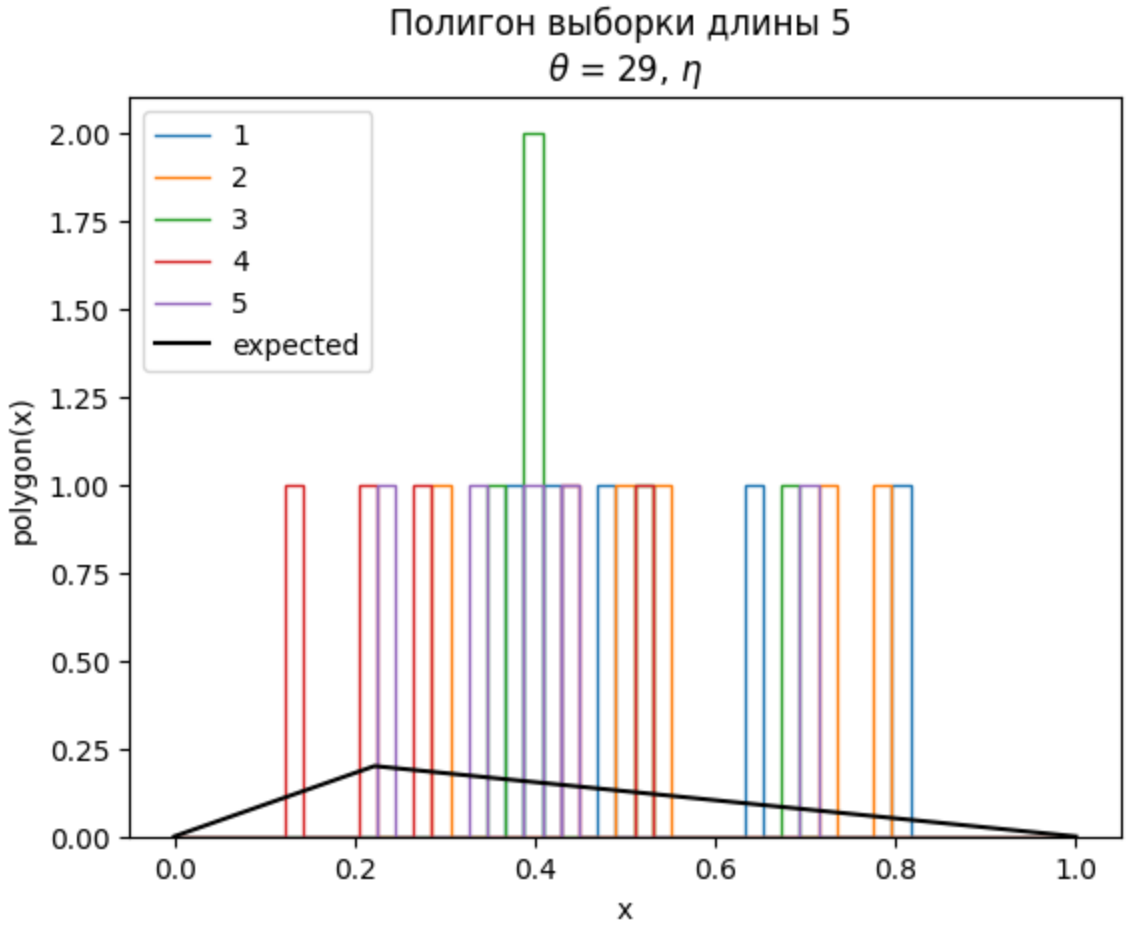
Логика с домножением вероятности здесь также применима. Однако здесь есть другой аспект: полигон частот почти не будет иметь смысла, так как вероятность попадания в каждое значение стремится к нулю. Для этого и только для этого я разобью отрезок [0, 1] на оси oX на 50 равных отрезков. Да, это сведет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

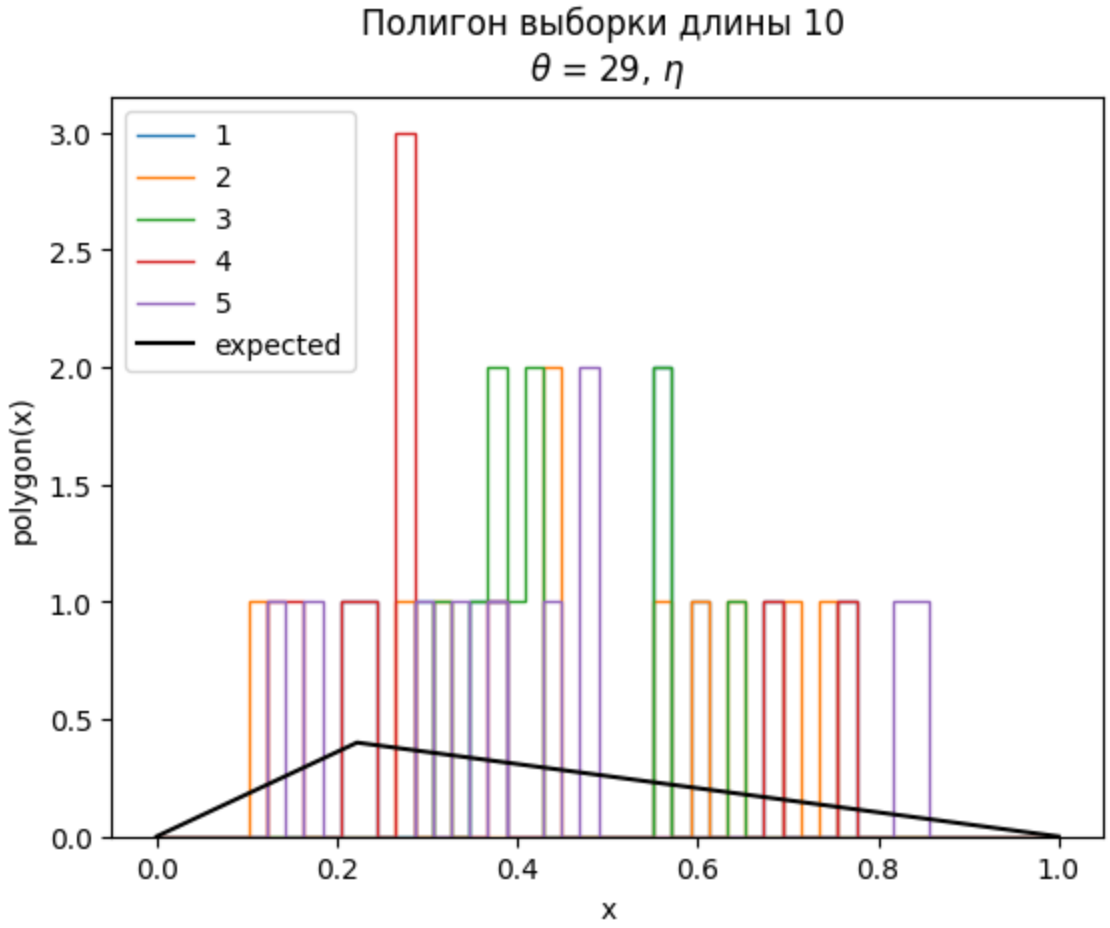
In [75]:
```
def eta_pilygon(sample, X):
    Y = np.zeros(X.shape)
    tick = 1
    for i in sample:
        while i > X[tick]: tick+=1
        Y[tick]+=1
    return Y

def eta_posib(x, theta = (1/4.5)):
    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 r
ange(len(n))]

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

In [111]:

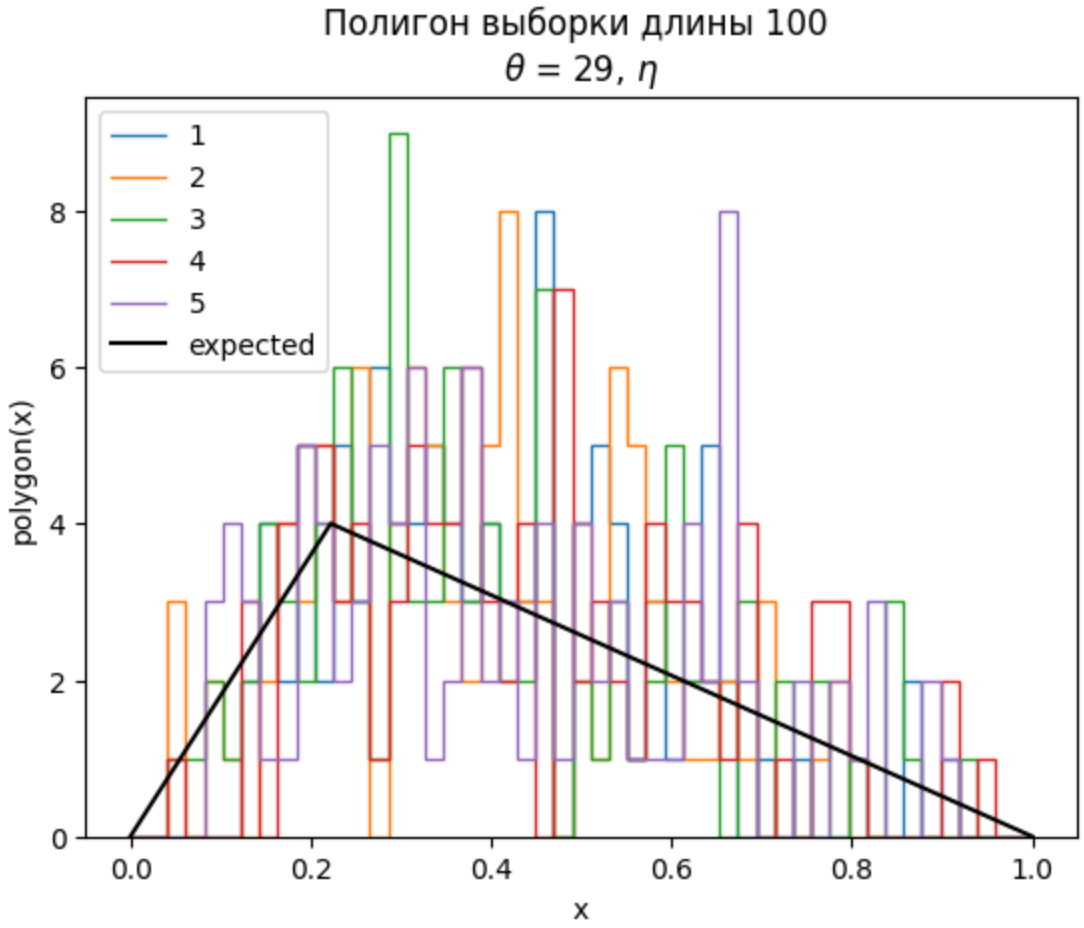
```
# n=5
for i in range(5):
    plt.stairs(Y_poleta[0][i], np.append(X_poleta,1))
plt.plot(X_realeta, posibilityeta*5/50, 'k')
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 5 \n$\\theta$ = 29, $\\eta$')
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



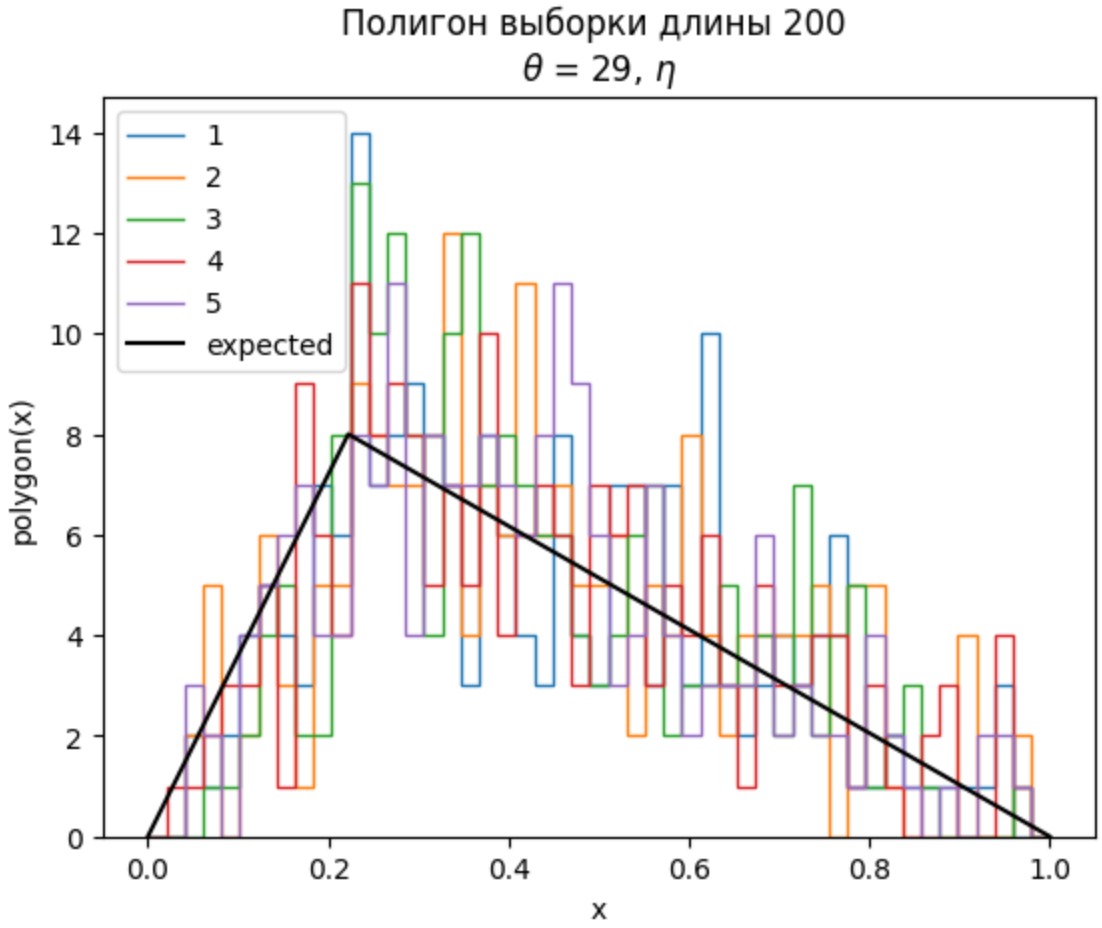
In [112]:
```
# n=10
for i in range(5):
    plt.stairs(Y_poleta[1][i], np.append(X_poleta, 1))
plt.plot(X_realeta, posibilityeta*10/50, 'k')
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 10 \n$\\theta$ = 29, $\\eta$')
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



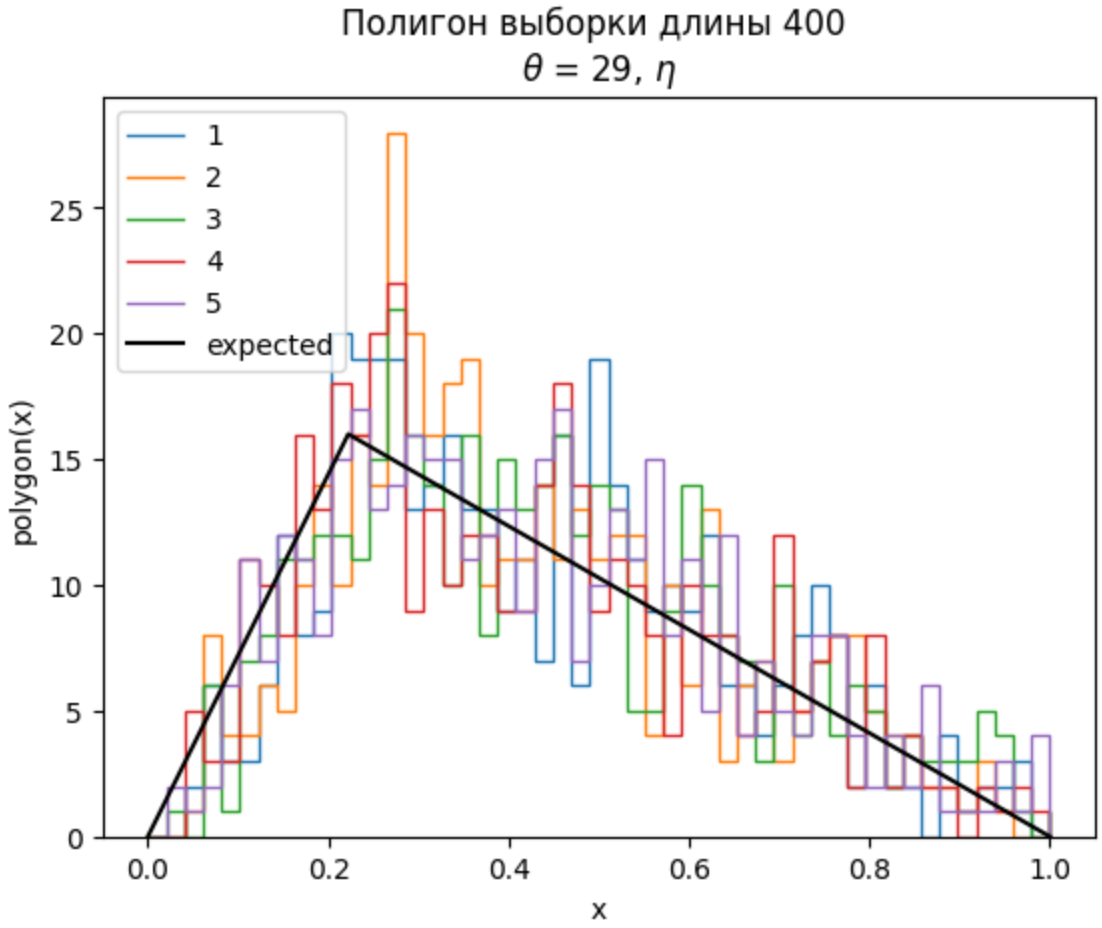
In [113]:
```
# n=100
for i in range(5):
    plt.stairs(Y_poleta[2][i], np.append(X_poleta, 1))
plt.plot(X_realeta, posibilityeta*100/50, 'k')
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 100 \n$\\theta$ = 29, $\\eta$')
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



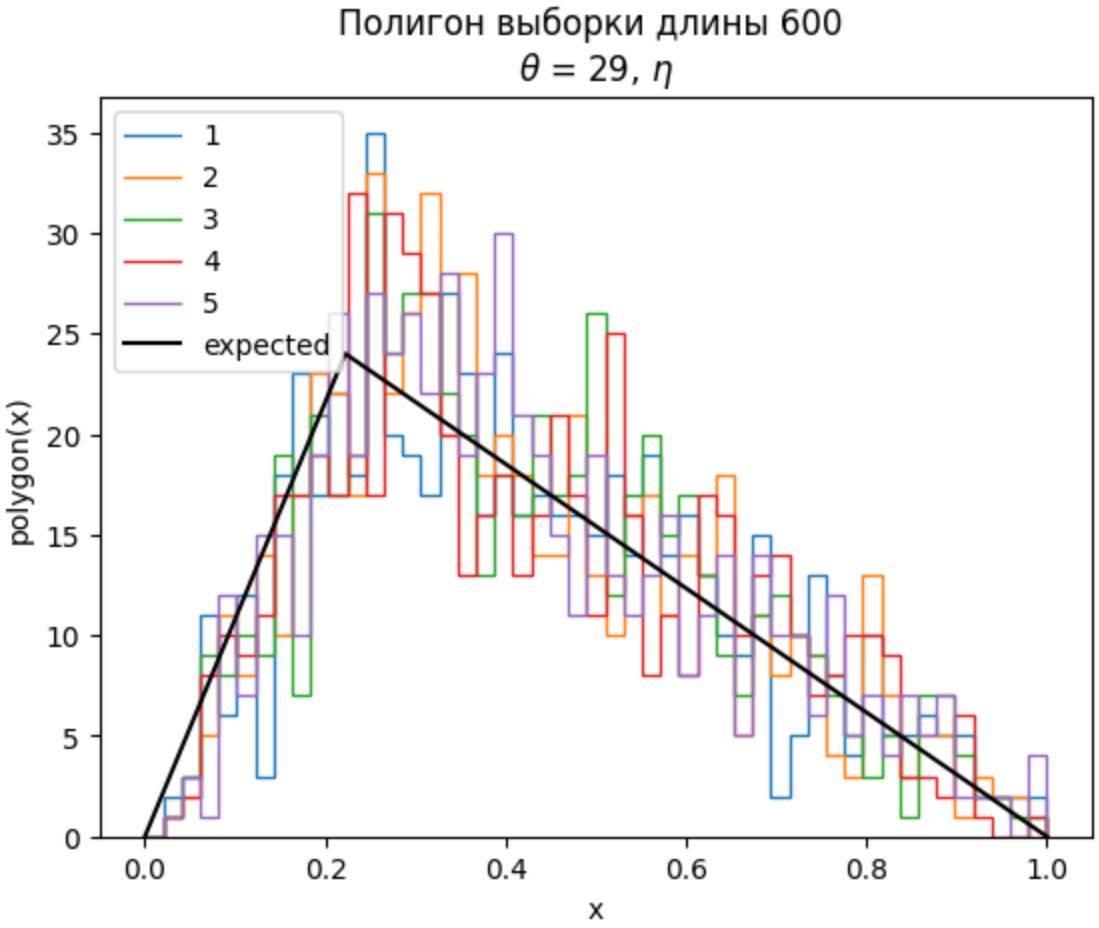
In [114]:
```
# n=200
for i in range(5):
    plt.stairs(Y_poleta[3][i], np.append(X_poleta, 1))
plt.plot(X_realeta, posibilityeta*200/50, 'k')
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 200 \n$\\theta$ = 29, $\\eta$')
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



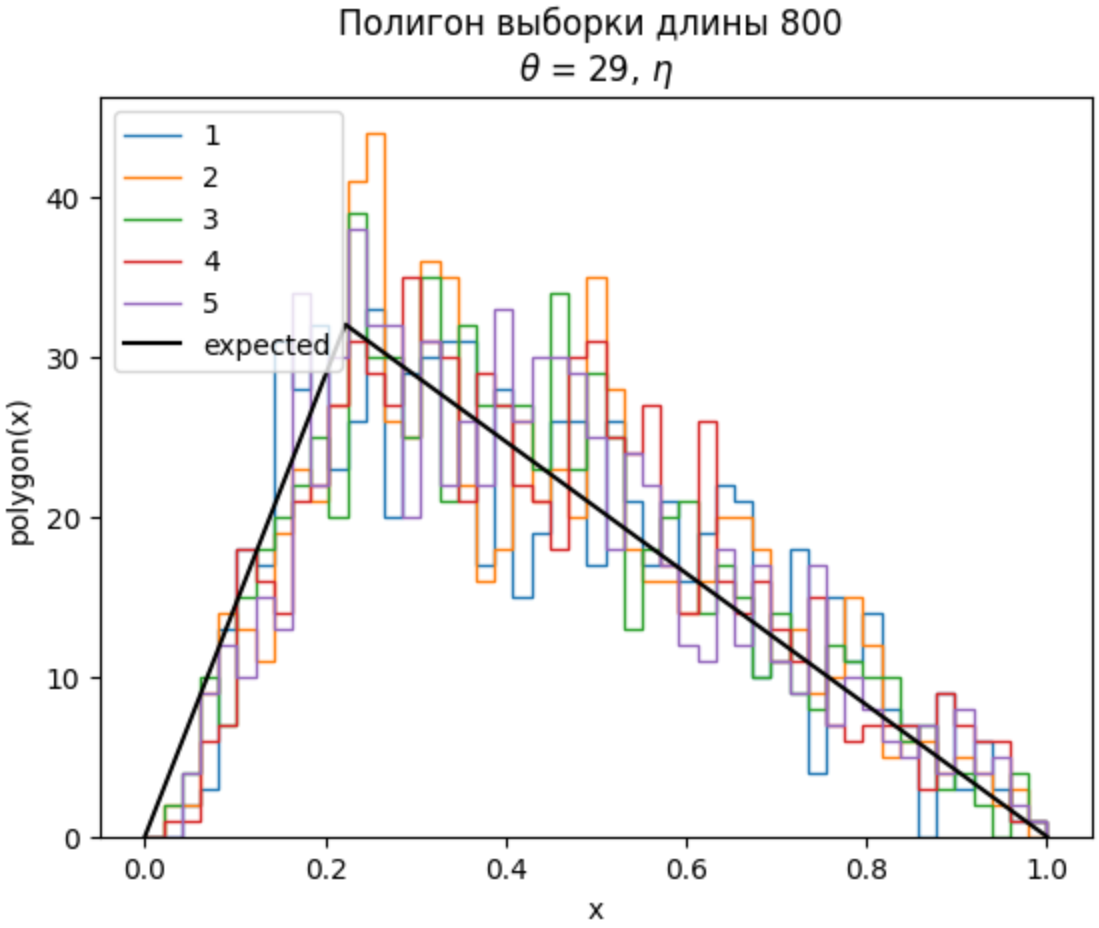
In [115]:
```
# n=400
for i in range(5):
    plt.stairs(Y_poleta[4][i], np.append(X_poleta, 1))
plt.plot(X_realeta, posibilityeta*400/50, 'k')
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 400 \n$\\theta$ = 29, $\\eta$')
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



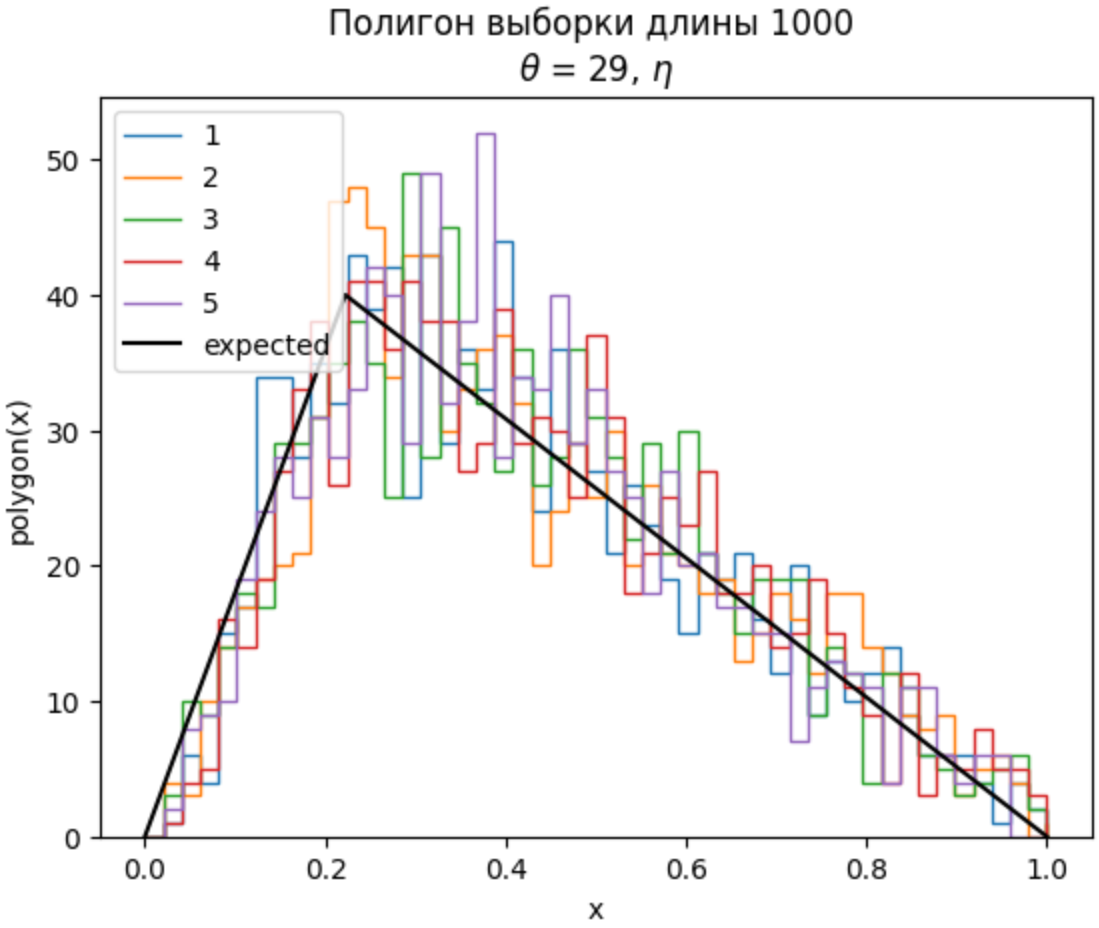
In [116]:
```
# n=600
for i in range(5):
    plt.stairs(Y_poleta[5][i], np.append(X_poleta, 1))
plt.plot(X_realeta, posibilityeta*600/50, 'k')
plt.xlabel('x')
plt.ylabel('polygon(x)')
plt.title('Полигон выборки длины 600 \n$\\theta$ = 29, $\\eta$')
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



In [117]:
```
# 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$\\theta$ = 29, $\\eta$')
plt.legend([1, 2, 3, 4, 5, 'expected'], loc='upper left');
```



In [118]:
```
# 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$\\theta$ = 29, $\\eta$')
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 [103]:

```
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+(1/4.5))/3
varianceeta = (1-(1/4.5)+(1/4.5)**2)/18
means_difeta = np.array([['%.4f' % i for i in j] for j in (meanseta-expectationet
a)])
variances_difeta = np.array([['%.4f' % i for i in j]
                        for j in (varianceseta-varianceeta)])
```

In [104]:
```
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');
```

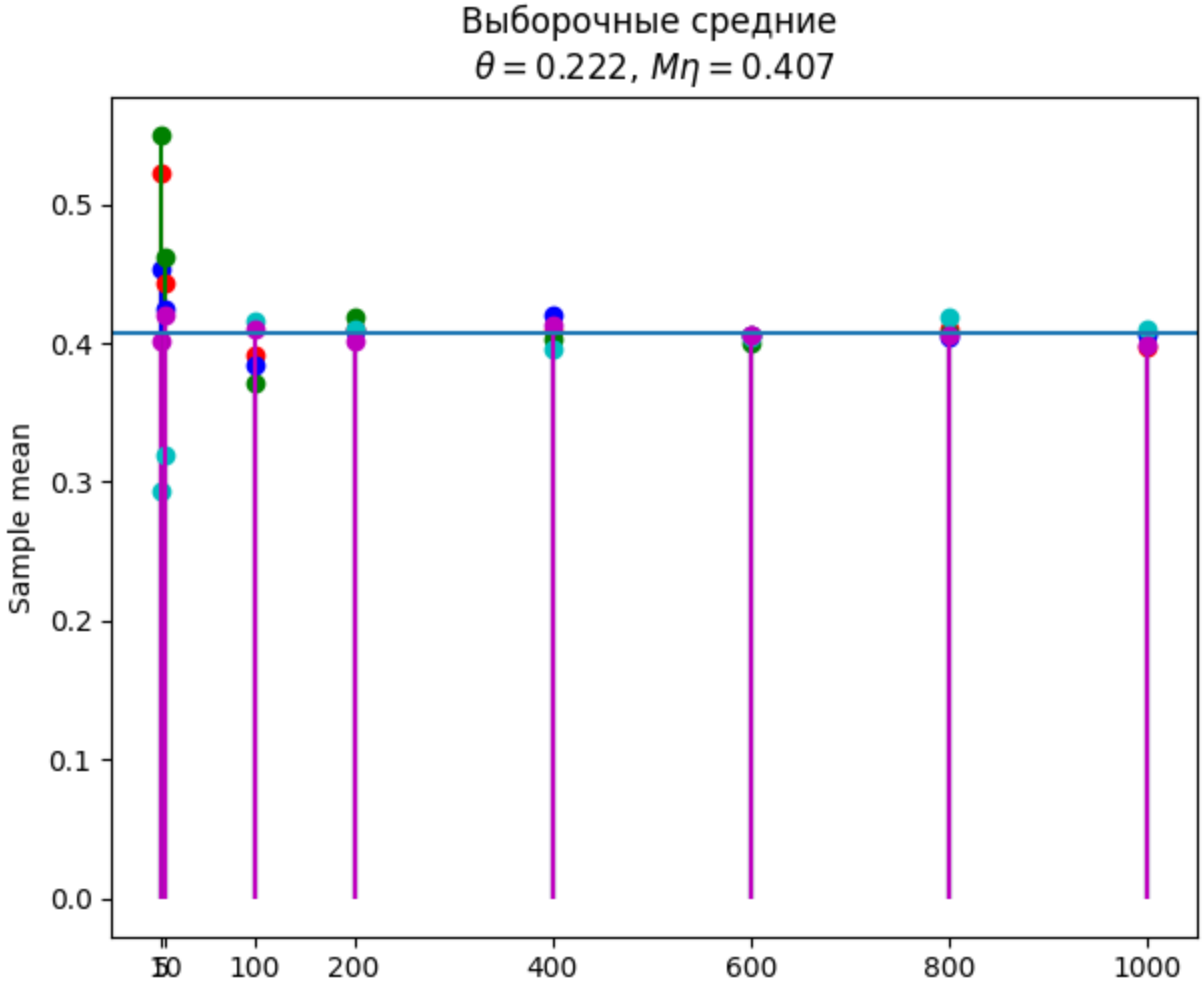
Выборочные средние

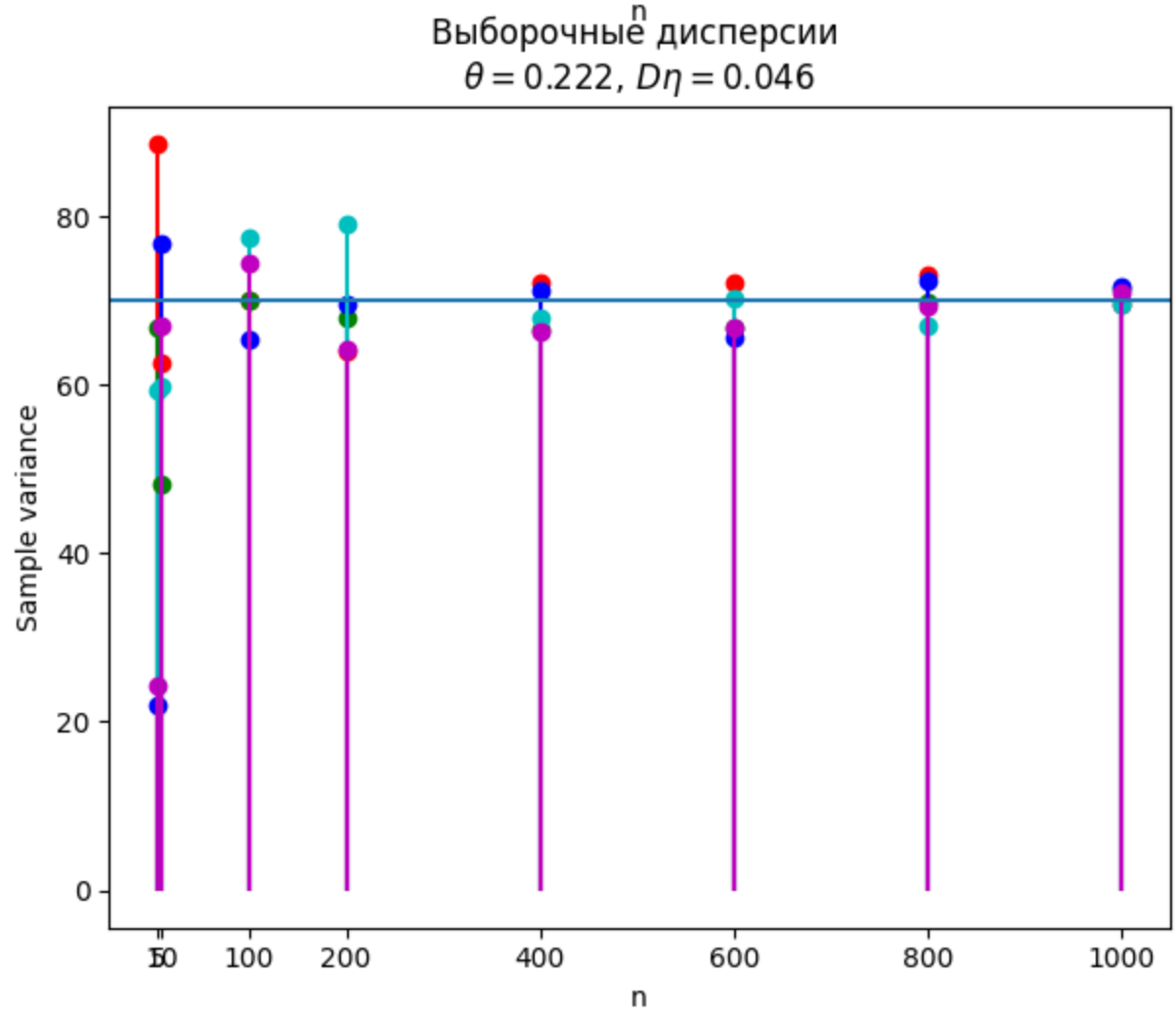
	1	2	3	4	5
5	0.5224	0.5502	0.4530	0.2939	0.4012
10	0.4436	0.4621	0.4240	0.3199	0.4197
100	0.3908	0.3712	0.3846	0.4164	0.4098
200	0.4101	0.4184	0.4082	0.4106	0.4018
400	0.4092	0.4035	0.4204	0.3957	0.4124
600	0.4054	0.4000	0.4061	0.4039	0.4051
800	0.4095	0.4079	0.4043	0.4188	0.4064
1000	0.3977	0.4077	0.4058	0.4098	0.3991

Выборочные дисперсии

	1	2	3	4	5
5	0.0253	0.0302	0.0142	0.0207	0.0240
10	0.0339	0.0365	0.0105	0.0428	0.0532
100	0.0314	0.0313	0.0423	0.0410	0.0474
200	0.0467	0.0449	0.0393	0.0484	0.0434
400	0.0440	0.0421	0.0477	0.0467	0.0479
600	0.0471	0.0445	0.0437	0.0455	0.0452
800	0.0476	0.0455	0.0439	0.0445	0.0450
1000	0.0458	0.0477	0.0459	0.0459	0.0429

In [106]:
```
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$\\theta = %.3f, \\,M\\eta = %.3f$'%((1/
4.5),expectationeta))

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$\\theta = %.3f, \\,D\\eta = %.3f$'%((1
/4.5),varianceeta));
```


Выборочные средние
$\theta = 0.222$, $M\eta = 0.407$
Sample mean
0.0
0.1
0.2
0.3
0.4
0.5
10
100
200
400
600
800
1000
n


Выборочные дисперсии
$\theta = 0.222$, $D\eta = 0.046$
Sample variance
0
20
40
60
80
10
100
200
400
600
800
1000
n

In [108]:
```
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);
```

	1	2	3	4	5
5	0.1150	0.1428	0.0456	-0.1135	-0.0062
10	0.0362	0.0547	0.0166	-0.0875	0.0123
100	-0.0166	-0.0363	-0.0228	0.0090	0.0024
200	0.0027	0.0110	0.0008	0.0032	-0.0056
400	0.0018	-0.0039	0.0130	-0.0117	0.0050
600	-0.0020	-0.0074	-0.0013	-0.0035	-0.0023
800	0.0021	0.0005	-0.0031	0.0114	-0.0010
1000	-0.0097	0.0003	-0.0016	0.0024	-0.0083

	1	2	3	4	5
5	-0.0206	-0.0157	-0.0318	-0.0252	-0.0219
10	-0.0120	-0.0094	-0.0354	-0.0031	0.0073
100	-0.0146	-0.0146	-0.0036	-0.0050	0.0014
200	0.0007	-0.0010	-0.0067	0.0025	-0.0025
400	-0.0019	-0.0039	0.0017	0.0008	0.0019
600	0.0012	-0.0014	-0.0022	-0.0005	-0.0007
800	0.0016	-0.0004	-0.0021	-0.0015	-0.0010
1000	-0.0002	0.0017	-0.0000	-0.0001	-0.0031

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

In [109]:
```
Осталось это отредачить чтобы в страничку влезало..
```

```
  Cell In [109], line 1
    Осталось это отредачить чтобы в страничку влезало..
                ^
SyntaxError: invalid syntax
```

In [ ]: