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

Дискретное

Задание 1

Функция распределения

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

Математическое ожидание

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

Дисперсия

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

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

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

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 [2]:
```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)])
```

Абсолютно непрерывное

Задание 1

Функция распределения

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

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

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

Математическое ожидание

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

Дисперсия

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

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

$+ \frac{2}{1-\theta}(\frac{1}{3}x^3 - \frac{1}{4}x^4)\Big|_\theta^1 = \frac{1}{2\theta}\theta^4 + \frac{2}{1-\theta}(\frac{1}{3} - \frac{1}{4} - \frac{1}{3}\theta^3 + \frac{1}{4}\theta^4) = \frac{1}{2}\theta^3 + \frac{2}{1-\theta}(\frac{1}{12} + \frac{3\theta^4-4\theta^3}{12}) = \frac{1}{2}\theta^3$

$+ \frac{1}{6(1-\theta)}(3\theta^4 - 4\theta^3 + 1) = \frac{1}{6(1-\theta)}(3\theta^4 - 4\theta^3 + 1 + 3\theta^3 - 3\theta^4) = \frac{1}{6(1-\theta)}(1-\theta^3) = \frac{1+\theta+\theta^2}{6}$

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

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)
$\frac{1}{1-\theta}(2x_\gamma - x_\gamma^2 - \theta) \geq \gamma \Rightarrow \frac{1}{1-\theta}(2x_\gamma - x_\gamma^2 - \theta) = \gamma \Leftrightarrow -x_\gamma^2 + 2x_\gamma - \theta = (1-\theta)\gamma \Leftrightarrow -x_\gamma^2 + 2x_\gamma - \theta - \gamma + \theta\gamma$
$= 0 \Rightarrow D = 4 - 4(\theta + \gamma - \theta\gamma) = 4(1 - \theta - \gamma + \theta\gamma) \Rightarrow x_\gamma = \frac{-2\pm2\sqrt{1-\theta-\gamma+\theta\gamma}}{-2} = 1 \pm \sqrt{1-\theta-\gamma+\theta\gamma}$. x_γ
$\in [\theta, 1] \Rightarrow x_\gamma = 1 - \sqrt{1-\theta-\gamma+\theta\gamma}$

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

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 под ниже.

In [75]:
```
import numpy as np
def generate_eta(theta=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

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, которые будут приведены во второй задаче, а здесь, как в пример, показать все выборки по 100 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]

print(*sample_xi[2], sep='\n')
```

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

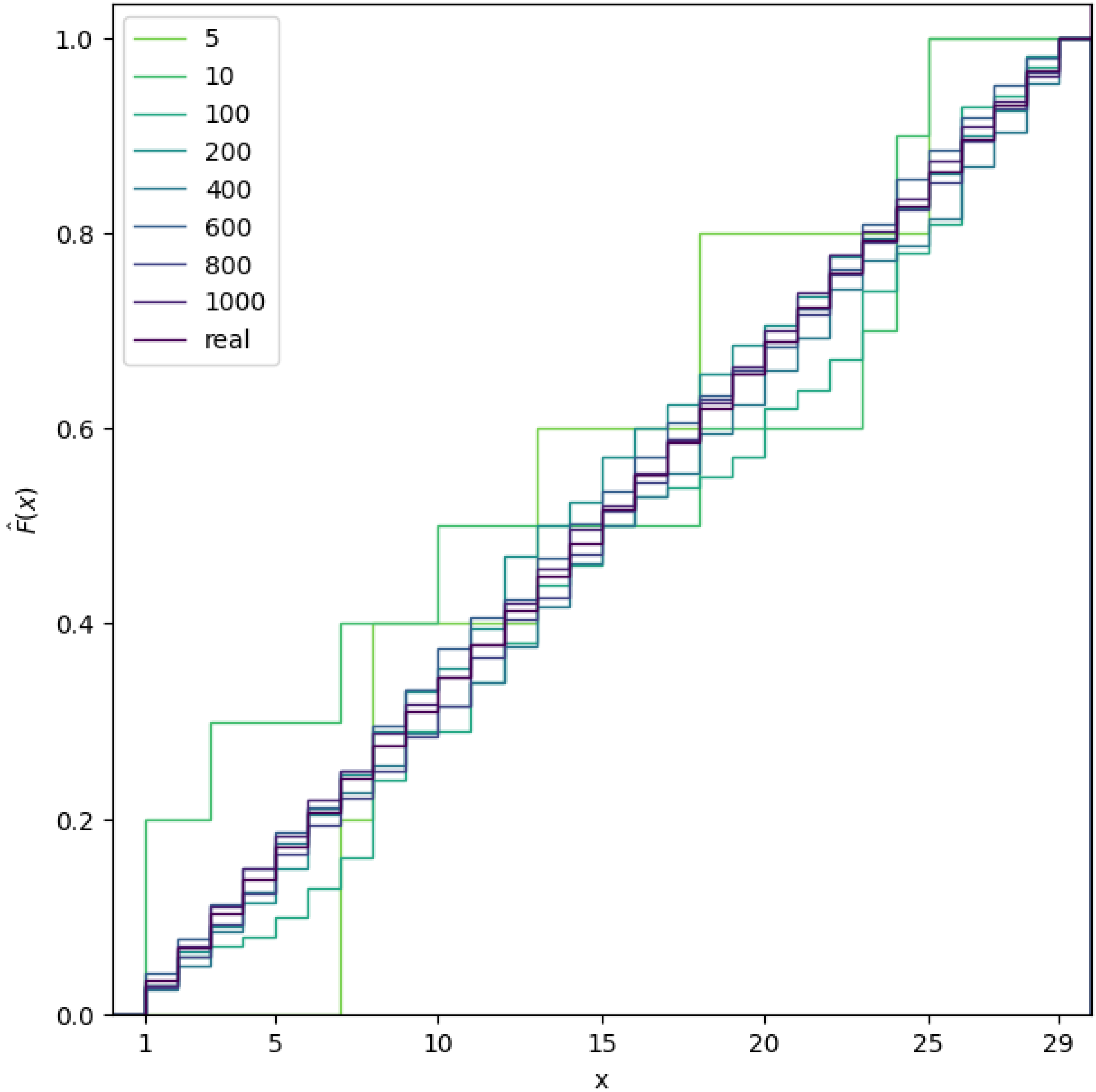
Задание 2

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_real = np.arange(1-1, 29+1+1)
Y_real = xi_distr_real(X_real)

Y = np.array([[[xi_distr(sample_xi[k][j], x) for x in X_real]
      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

diffs = np.array([[[('%.2f' % xi_Dmn(Y[i][k], Y[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]:
```
import matplotlib.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(Y[i][0], np.append(X_real, 30), color = colors[i])
ax[0].stairs(Y_real, np.append(X_real, 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$\\theta$ = 29')
ax[0].legend([*n, 'real'], loc='upper left');

ax[1].table(cellText = diffs[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

$\theta = 29$



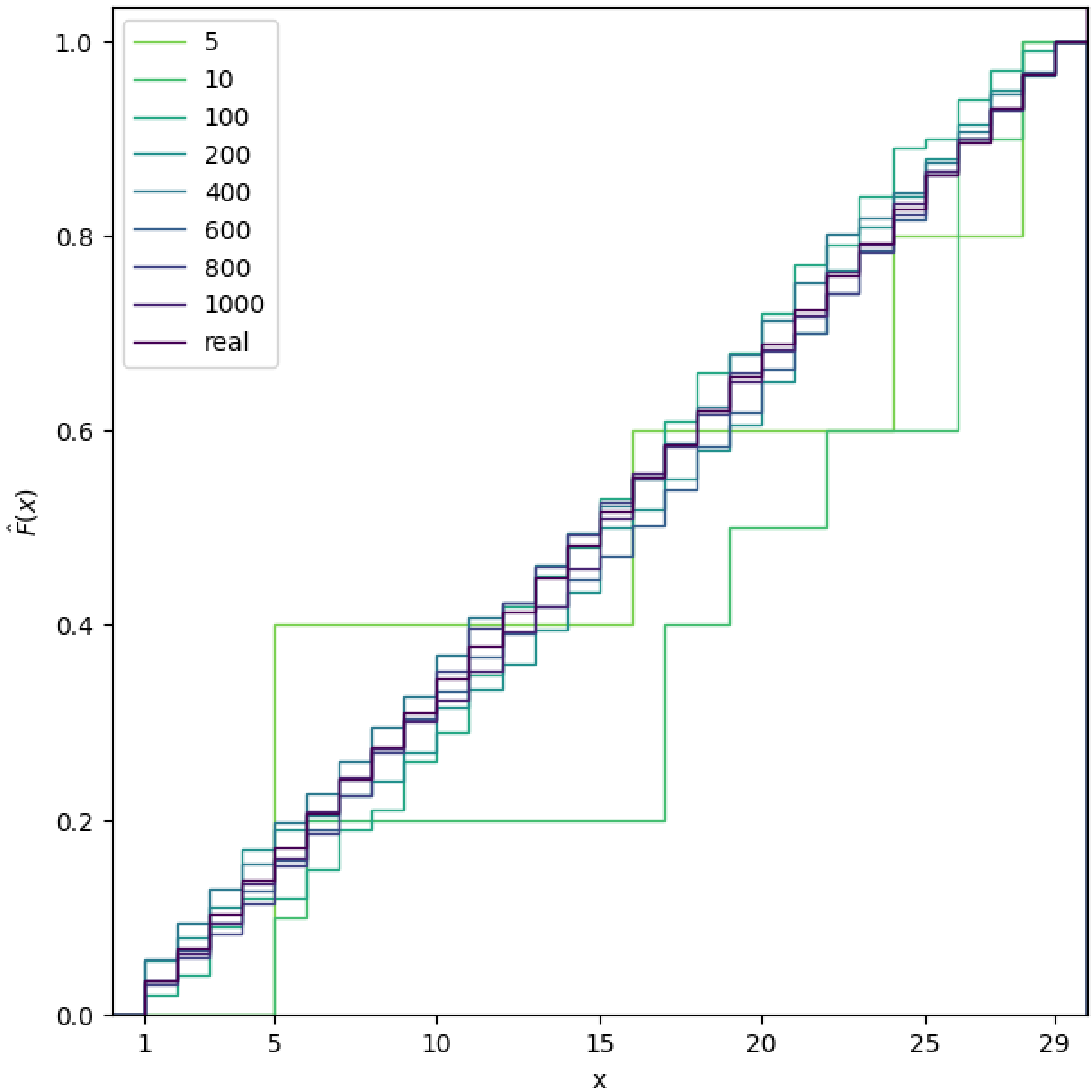
$$D_{m,n} = \sqrt{\frac{nm}{n+m}} \sup_{x \in \mathbb{R}} |F_n(x) - F_m(x)|$$

	5	10	100	200	400	600	800	1000
5	-	0.67	1.77	2.05	2.97	3.70	3.88	4.90
10	-	-	1.70	2.10	3.04	3.23	4.15	4.25
100	-	-	-	1.15	1.13	1.62	1.77	2.39
200	-	-	-	-	1.31	0.78	1.45	1.10
400	-	-	-	-	-	1.21	0.75	1.32
600	-	-	-	-	-	-	1.18	0.65
800	-	-	-	-	-	-	-	0.90
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(Y[i][1], np.append(X_real, 30), color = colors[i])
ax[0].stairs(Y_real, np.append(X_real, 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$\\theta$ = 29')
ax[0].legend([*n, 'real'], loc='upper left');

ax[1].table(cellText = diffs[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
$\theta = 29$

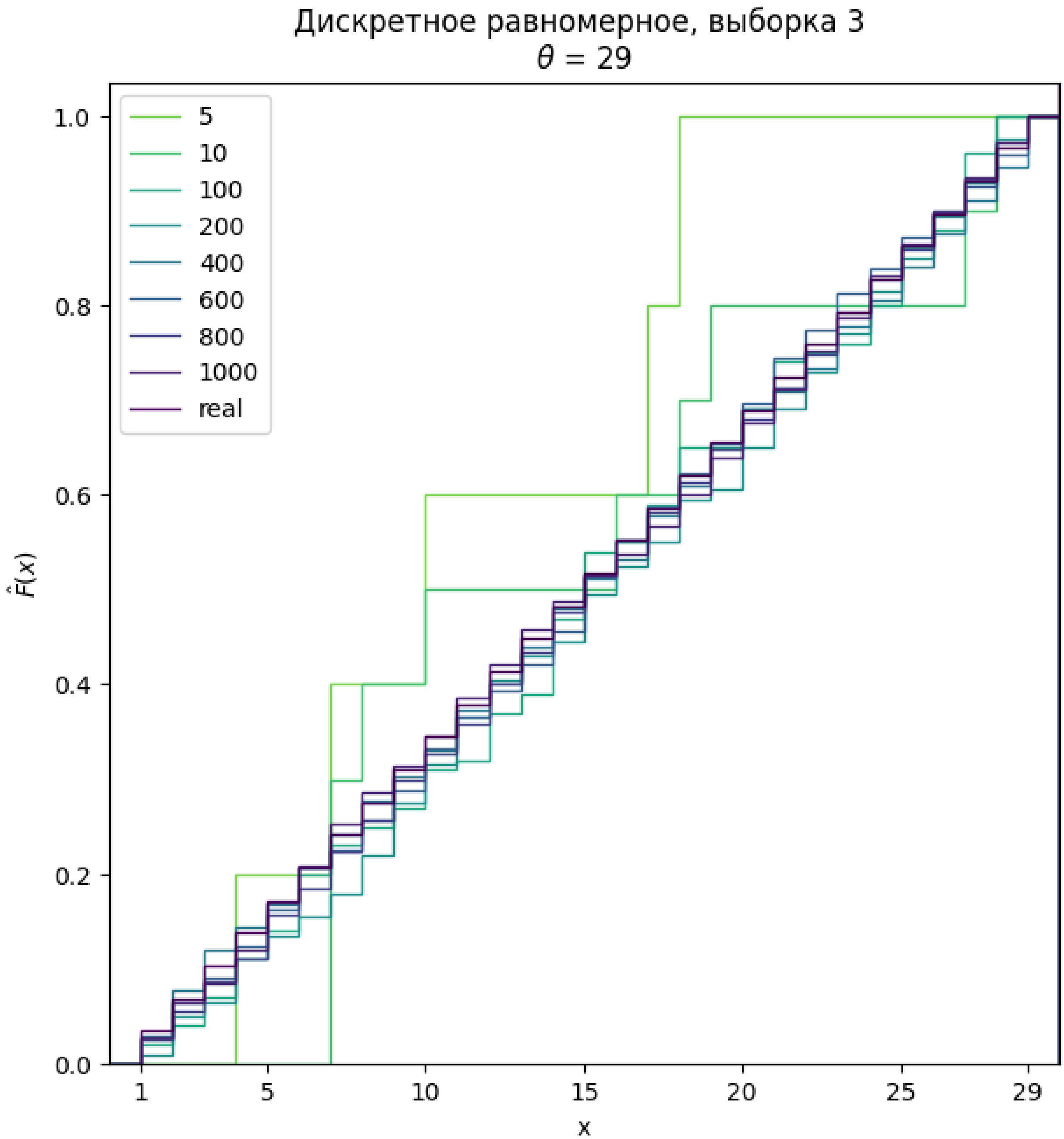


$$D_{m,n} = \sqrt{\frac{nm}{n+m}} \sup_{x \in \mathbb{R}} |F_n(x) - F_m(x)|$$

	5	10	100	200	400	600	800	1000
5	-	0.89	1.98	2.10	3.08	4.19	4.93	5.37
10	-	-	2.47	3.20	4.99	5.25	7.03	7.96
100	-	-	-	0.80	1.20	1.33	1.35	1.41
200	-	-	-	-	1.03	0.72	1.30	1.03
400	-	-	-	-	-	1.05	1.23	1.22
600	-	-	-	-	-	-	1.09	1.18
800	-	-	-	-	-	-	-	0.97
1000	-	-	-	-	-	-	-	-

In [7]:

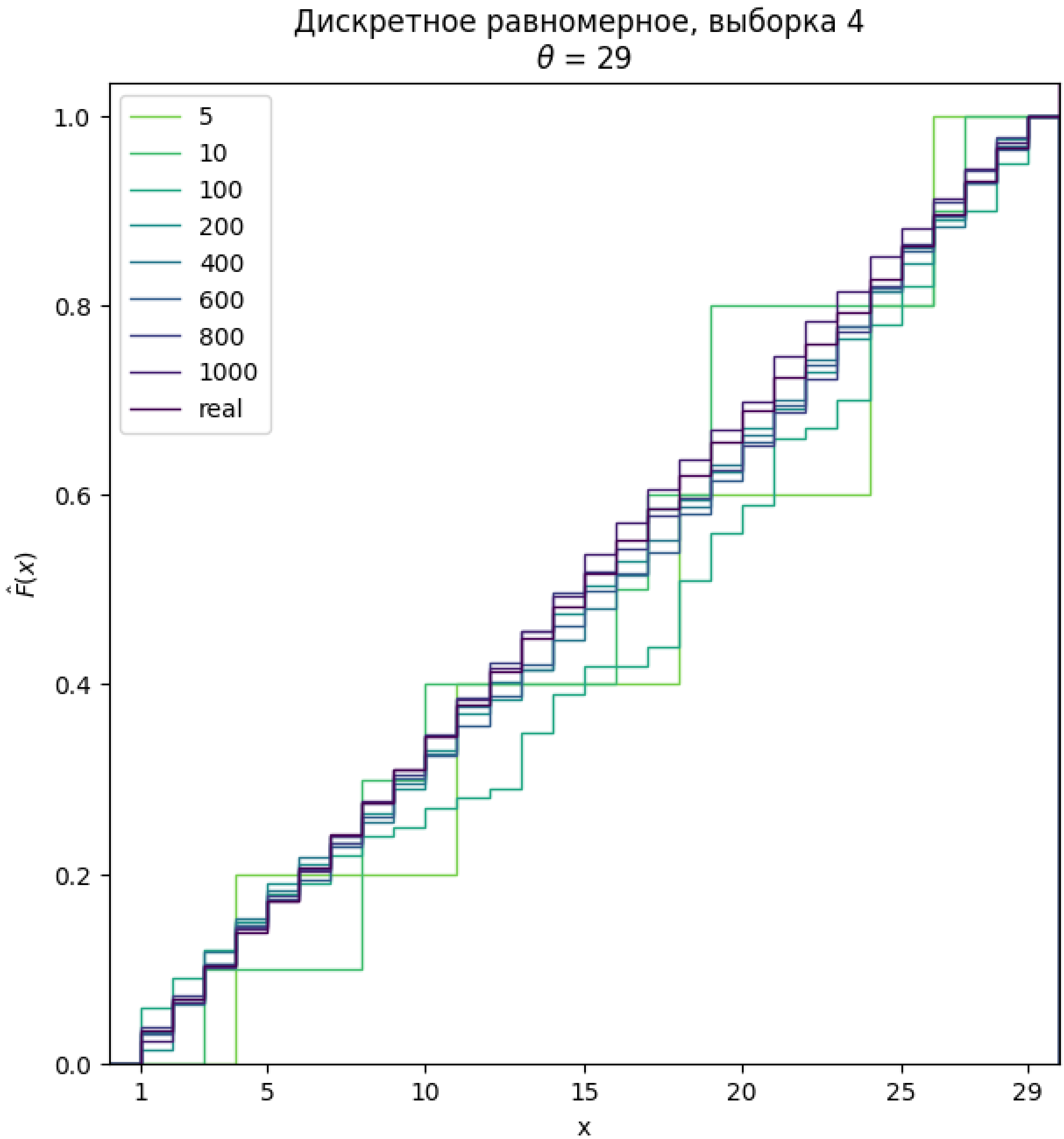
```
# graph[2]
fig, ax = plt.subplots(2,1, figsize=(7,12), height_ratios = [2,1])
for i in range(8):
    ax[0].stairs(Y[i][2], np.append(X_real, 30), color = colors[i])
ax[0].stairs(Y_real, np.append(X_real, 30), color = '#440154')
ax[0].set(xticks = [1]+list(range(5,26,5))+[29], xmargin = 0, ymargin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

ax[1].table(cellText = diffs[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	-	0.67	2.47	4.05	5.52	6.55	7.72	8.94
10	-	-	1.41	1.95	2.93	3.58	3.70	4.67
100	-	-	-	0.55	0.78	0.92	0.88	1.52
200	-	-	-	-	0.88	0.95	0.88	1.63
400	-	-	-	-	-	0.71	0.70	0.78
600	-	-	-	-	-	-	0.65	0.81
800	-	-	-	-	-	-	-	0.69
1000	-	-	-	-	-	-	-	-

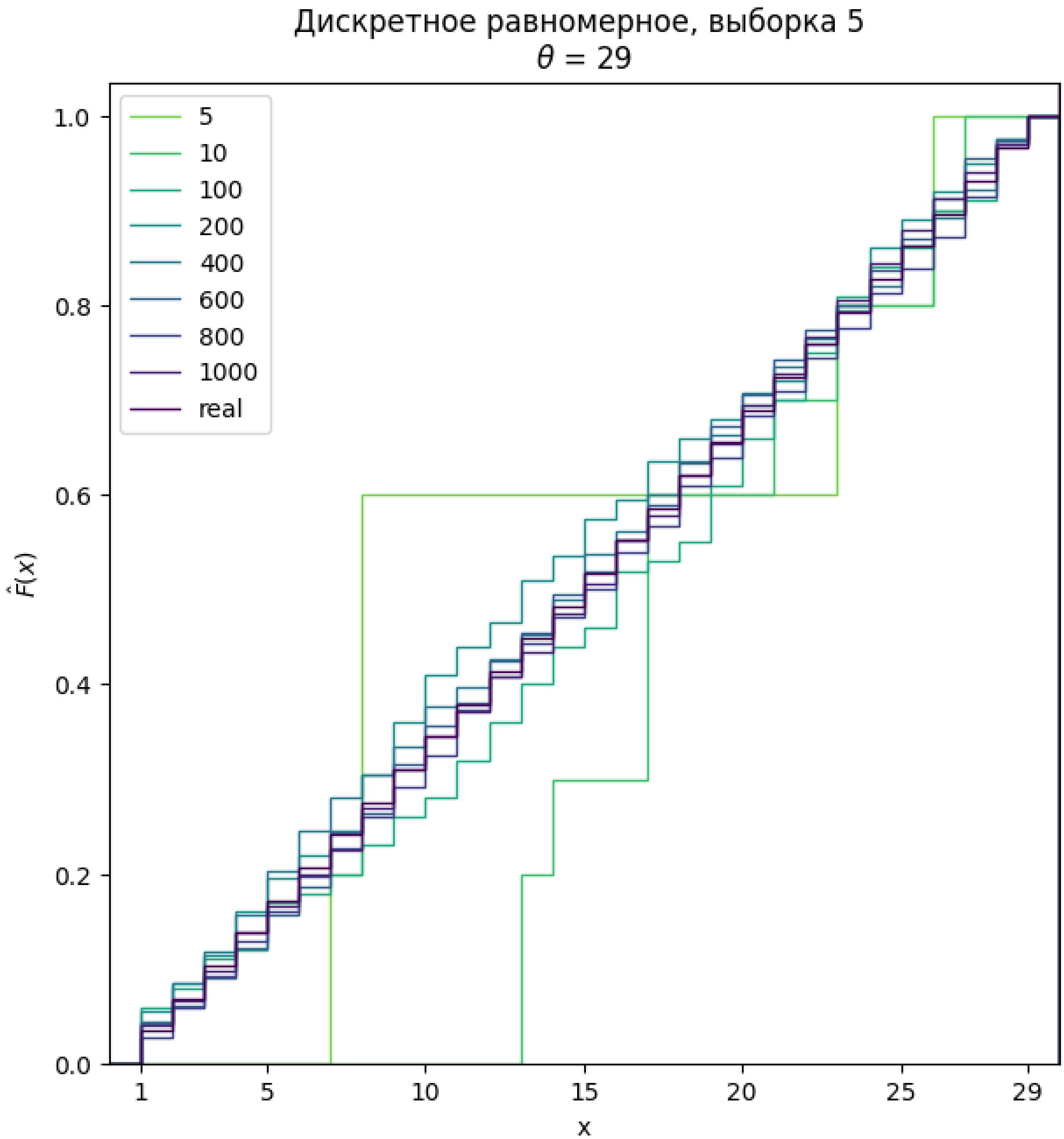
In [8]:
```
# graph[3]
fig, ax = plt.subplots(2,1, figsize=(7,12), height_ratios = [2,1])
for i in range(8):
    ax[0].stairs(Y[i][3], np.append(X_real, 30), color = colors[i])
ax[0].stairs(Y_real, np.append(X_real, 30), color = '#440154')
ax[0].set(xticks = [1]+list(range(5,26,5))+[29], xmargin = 0,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

ax[1].table(cellText = diffs[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	0.85	1.85	2.51	3.09	3.57	4.81
10	-	-	1.70	1.75	2.37	3.20	3.48	3.15
100	-	-	-	1.45	1.59	1.73	2.77	3.71
200	-	-	-	-	0.46	0.78	0.83	1.25
400	-	-	-	-	-	0.42	1.00	1.30
600	-	-	-	-	-	-	0.77	1.48
800	-	-	-	-	-	-	-	1.38
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(Y[i][4], np.append(X_real, 30), color = colors[i])
ax[0].stairs(Y_real, np.append(X_real, 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$\\theta$ = 29')
ax[0].legend([*n, 'real'], loc='upper left');

ax[1].table(cellText = diffs[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	-	1.34	2.62	2.95	4.17	5.80	6.80	7.38
10	-	-	2.55	4.65	6.05	7.36	8.15	9.12
100	-	-	-	1.30	1.38	1.44	1.17	1.59
200	-	-	-	-	0.81	1.04	1.70	1.68
400	-	-	-	-	-	1.01	1.05	1.21
600	-	-	-	-	-	-	0.83	0.47
800	-	-	-	-	-	-	-	0.92
1000	-	-	-	-	-	-	-	-

Задание 3

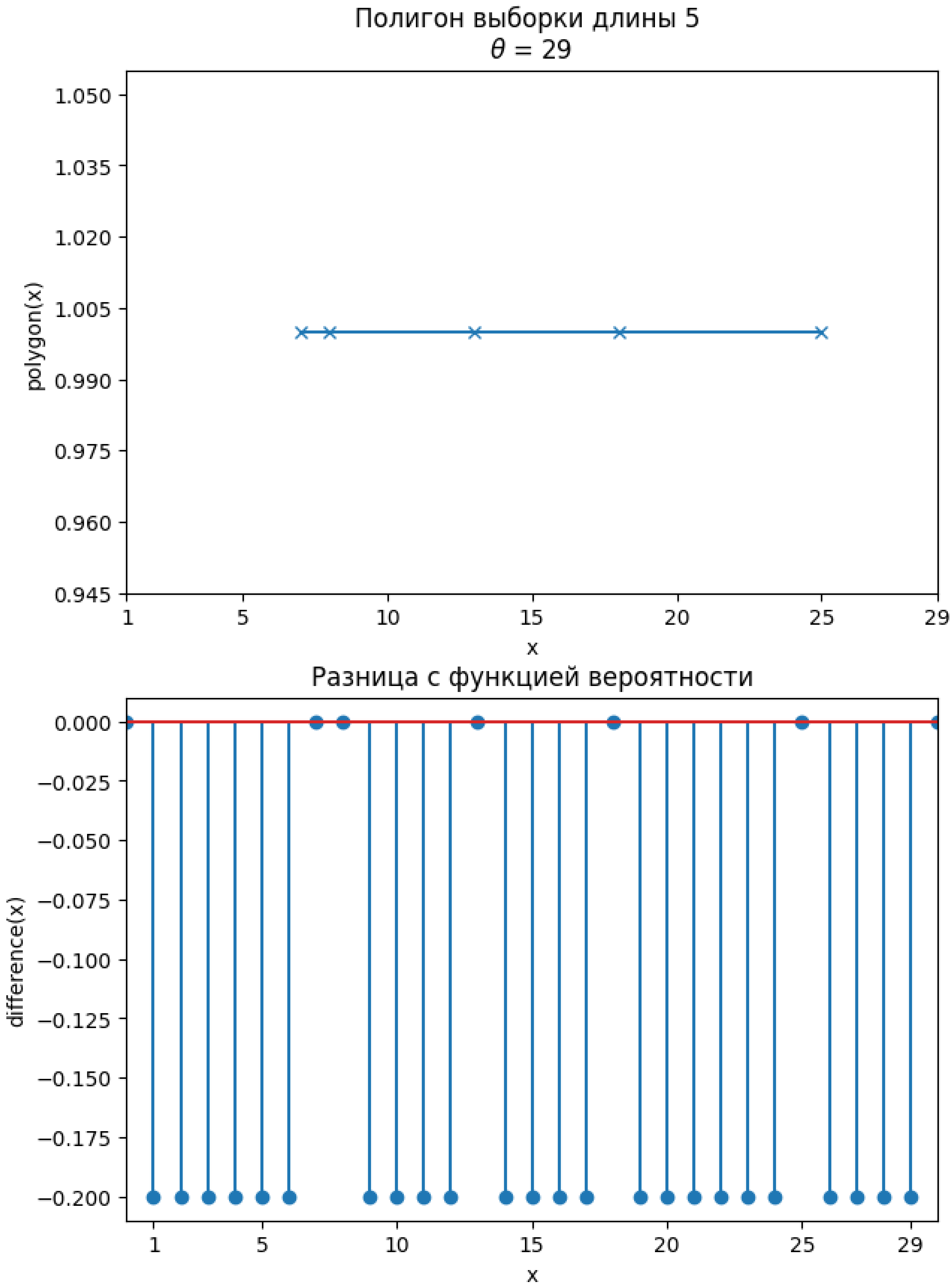
$\hat{F}(x) = \frac{1}{n}\sum_{i=1}^{n} I(x_i \leq x) \Rightarrow P(x_i) = \hat{F}(x_i) - \hat{F}(x_{i-1}) = \frac{1}{n} k\, I(x_i = x_i) = \frac{k}{n}$, где k - частота встречаемости элемента в выборке.

*Я уже потом понял что вероятность по выборке мне находить не надо было, а надо было лишь найти способ как свести полигон частот к функции вероятности, в плане грамотно наложить 1 на другой, чтобы они были схожи, не смотря на огромную разницу в высотах графиков

In [84]:
```
def xi_pilygon(sample, x):
    return np.count_nonzero(sample==x)

def xi_differ_r(sample, x, theta = 29):
    return (xi_pilygon(sample, x)-1)/len(sample)

Y_pol = [[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_dr = [[np.array([xi_differ_r(sample_xi[k][j], x) for x in X_real])
         for j in range(5)] for k in range(len(n))]
#Y_dr = Y - Y_real[None,None,:] # <- чел, для вероятностей
```

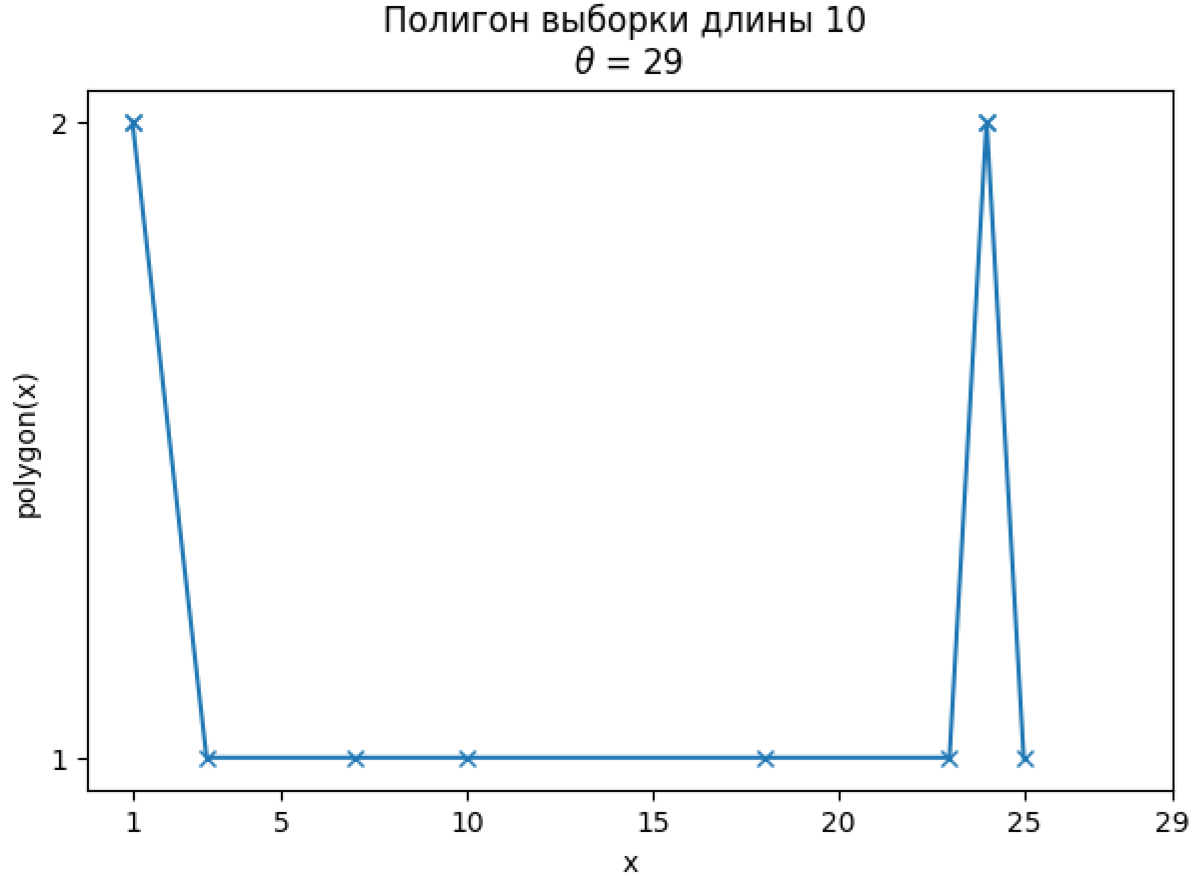
In [108]:

```
from matplotlib.ticker import MaxNLocator

# n=5
fig, ax = plt.subplots(2,1, figsize=(7,10))
ax[0].yaxis.set_major_locator(MaxNLocator(integer=True))
#for i in range(5):
#    ax[0].plot(sample_xi[0][i], Y_pol[0][i], '-x')
ax[0].plot(sample_xi[0][0], Y_pol[0][0], '-x')
ax[0].set(xticks = [1]+list(range(5,26,5))+[29],
          xlabel = 'x', ylabel = 'polygon(x)',
          title = 'Полигон выборки длины 5 \n$\\theta$ = 29')
#ax[0].legend([1, 2, 3, 4, 5], loc='upper left')

#for i in range(5):
#    Y_dr[0][i][0], Y_dr[0][i][30] = 0, 0
#    ax[1].stem(X_real, Y_dr[0][i], 'rgbcm'[i])
Y_dr[0][0][0], Y_dr[0][0][30] = 0, 0
ax[1].stem(X_real, Y_dr[0][0])
ax[1].set(xticks = [1]+list(range(5,26,5))+[29], xmargin = 0,
          xlabel = 'x', ylabel = 'difference(x)',
          title = 'Разница с функцией вероятности');
#ax[1].legend([1, 2, 3, 4, 5], loc='upper left')
```

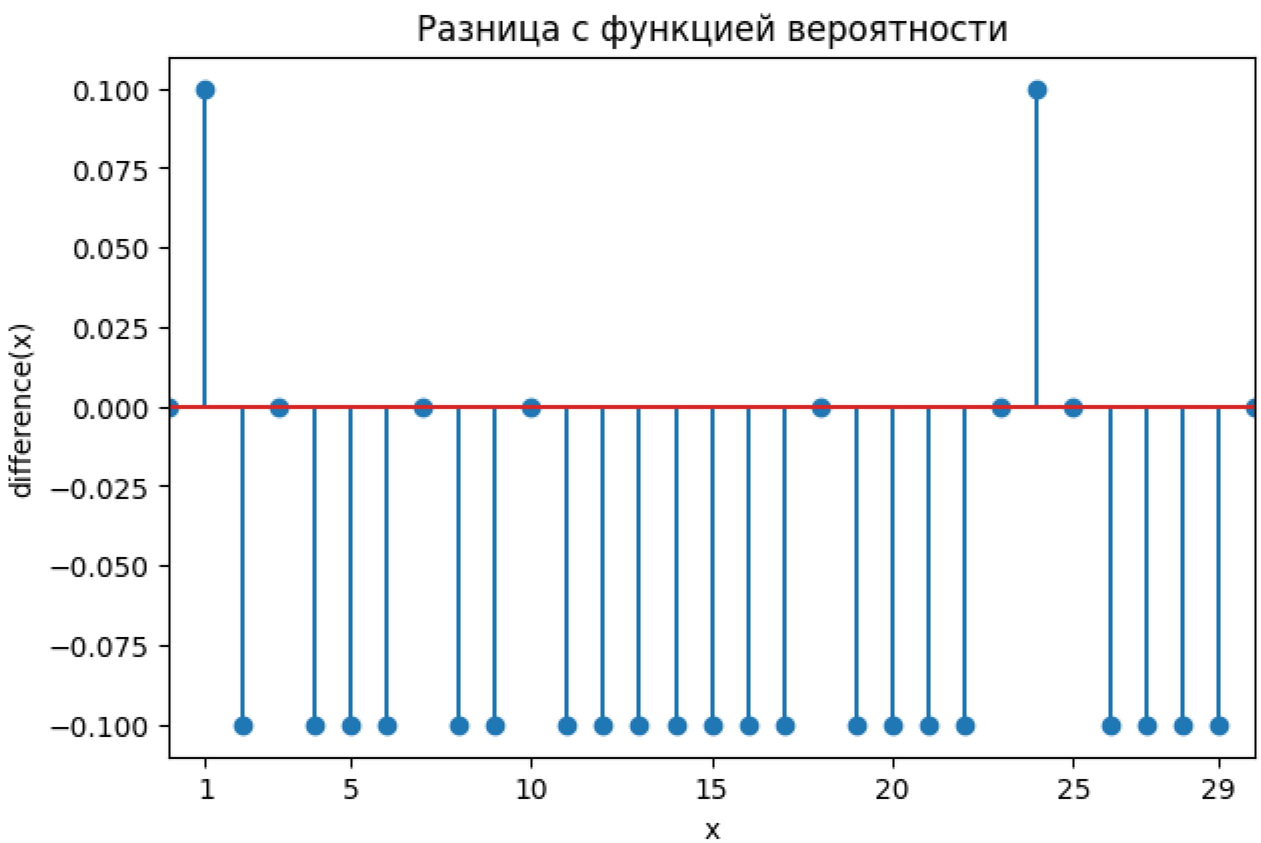

Полигон выборки длины 5
θ = 29
polygon(x)
x
Разница с функцией вероятности
difference(x)
x

In [87]:

```
# n=10
fig, ax = plt.subplots(2,1, figsize=(7,10))
ax[0].yaxis.set_major_locator(MaxNLocator(integer=True))
ax[0].plot(sample_xi[1][0], Y_pol[1][0], '-x')
ax[0].set(xticks = [1]+list(range(5,26,5))+[29],
          xlabel = 'x', ylabel = 'polygon(x)',
          title = 'Полигон выборки длины 10 \n$\\theta$ = 29')

Y_dr[1][0][0], Y_dr[1][0][30] = 0, 0
ax[1].stem(X_real, Y_dr[1][0])
ax[1].set(xticks = [1]+list(range(5,26,5))+[29], xmargin = 0,
          xlabel = 'x', ylabel = 'difference(x)',
          title = 'Разница с функцией вероятности');
```



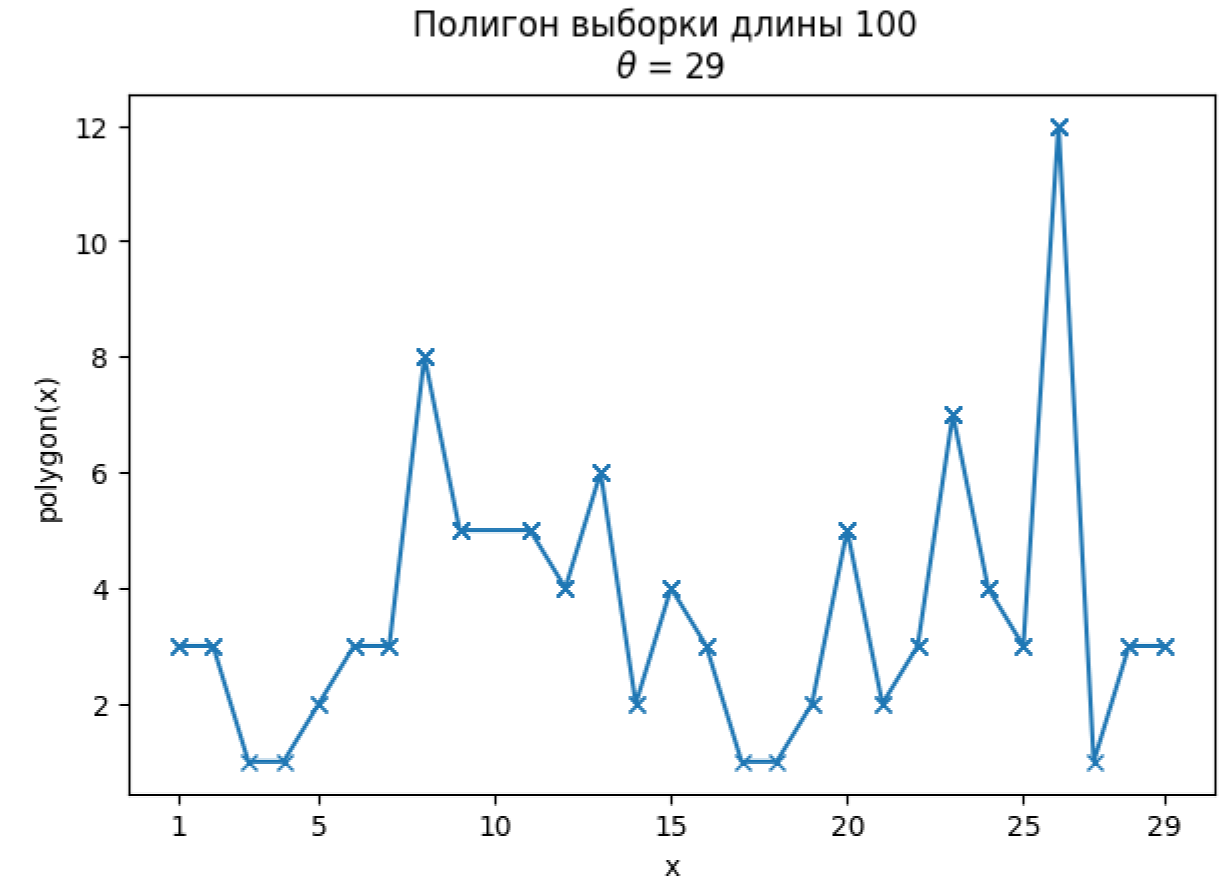
Полигон выборки длины 10
θ = 29
polygon(x)
x

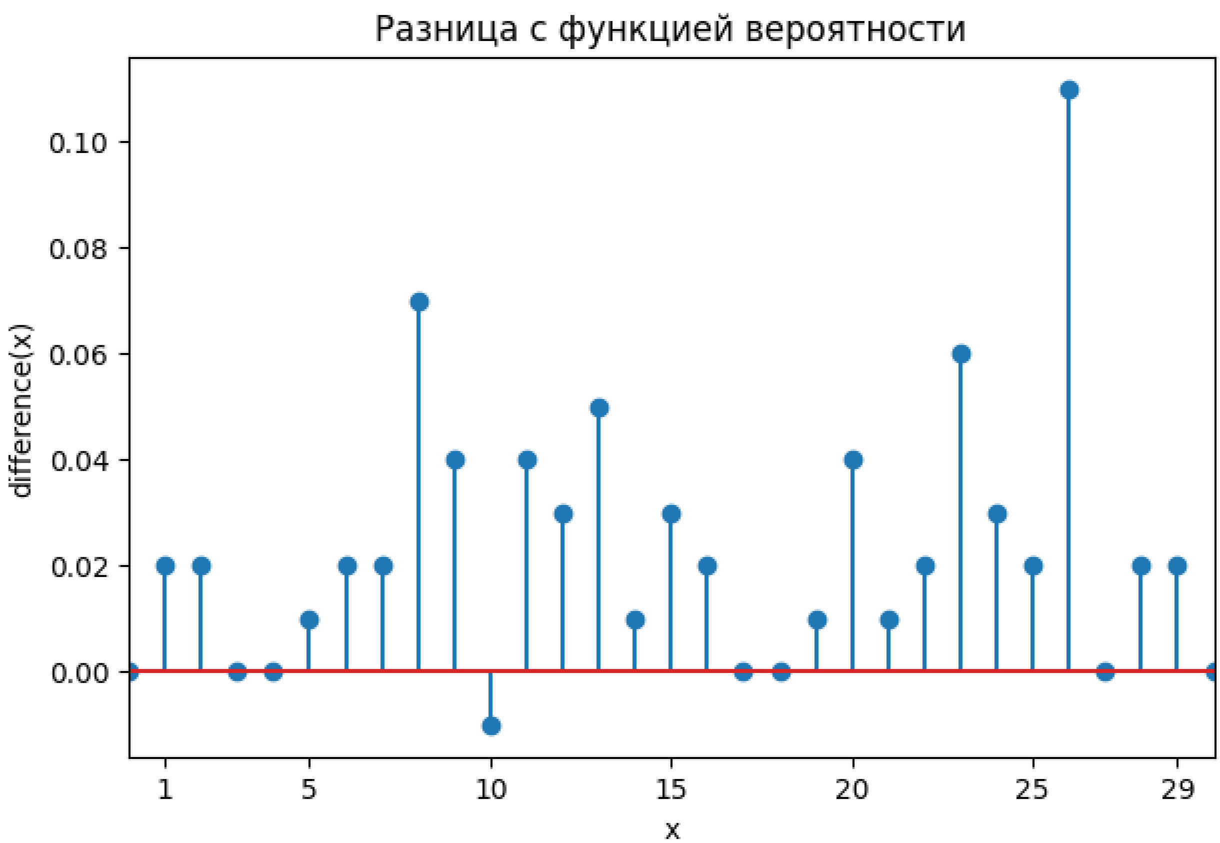


Разница с функцией вероятности
difference(x)
x

In [88]:

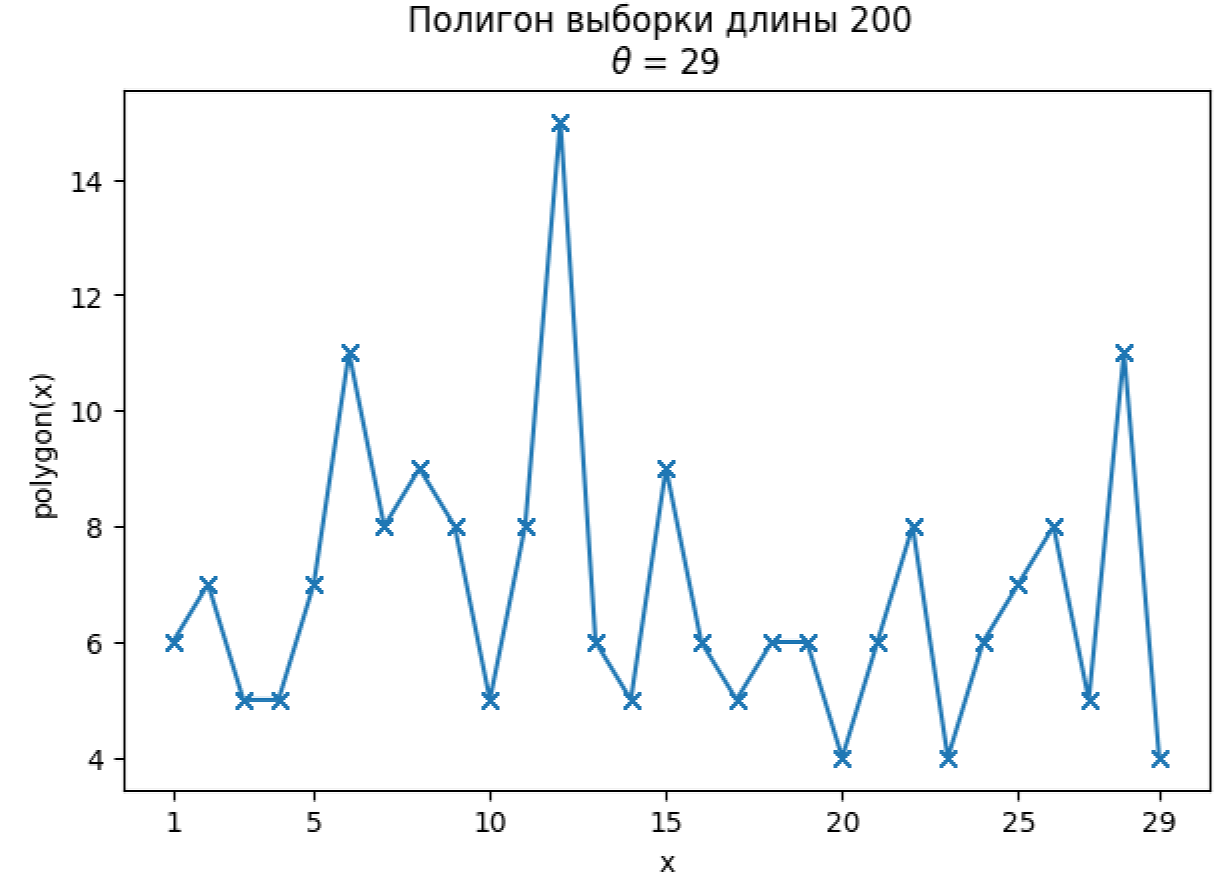
```
# n=100
fig, ax = plt.subplots(2,1, figsize=(7,10))
ax[0].yaxis.set_major_locator(MaxNLocator(integer=True))
ax[0].plot(sample_xi[2][0], Y_pol[2][0], '-x')
ax[0].set(xticks = [1]+list(range(5,26,5))+[29],
          xlabel = 'x', ylabel = 'polygon(x)',
          title = 'Полигон выборки длины 100 \n$\\theta$ = 29')

Y_dr[2][0][0], Y_dr[2][0][30] = 0, 0
ax[1].stem(X_real, Y_dr[2][0])
ax[1].set(xticks = [1]+list(range(5,26,5))+[29], xmargin = 0,
          xlabel = 'x', ylabel = 'difference(x)',
          title = 'Разница с функцией вероятности');
```

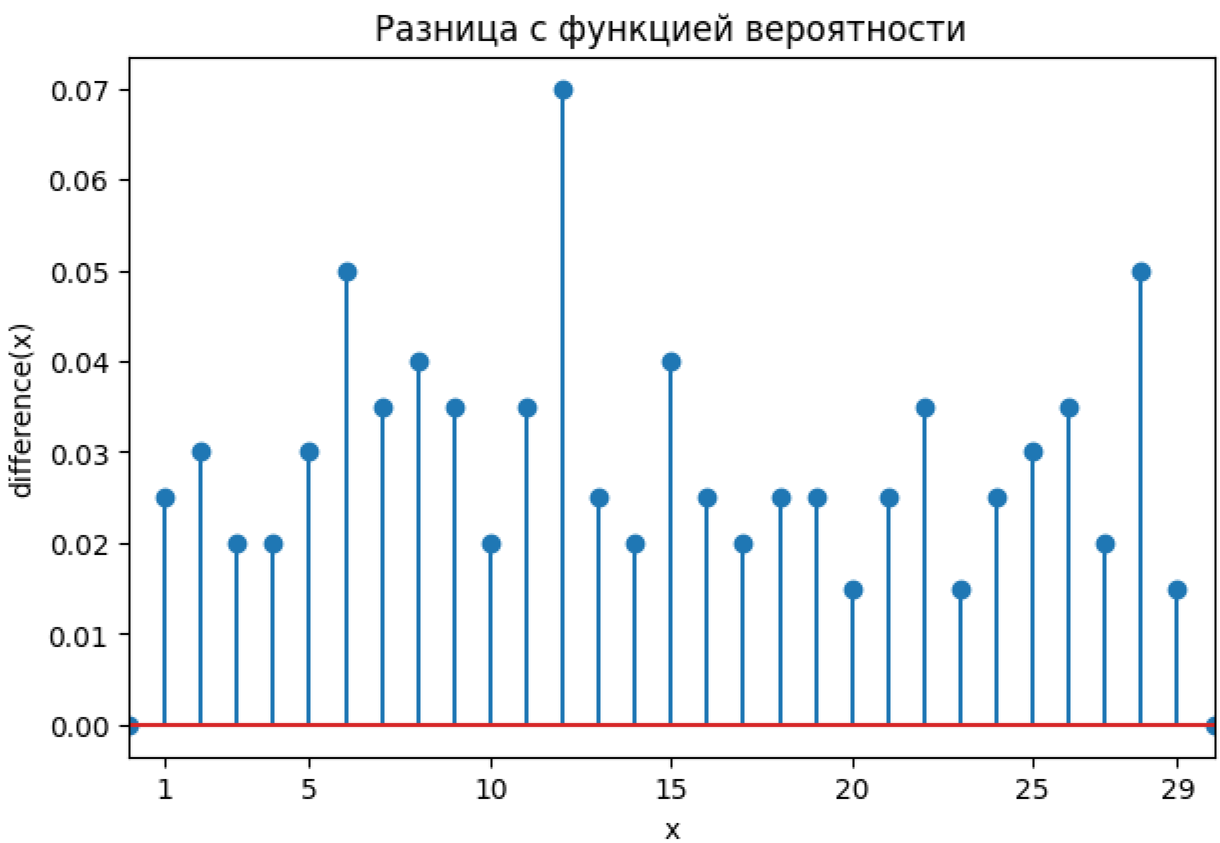

Полигон выборки длины 100
θ = 29
polygon(x)
x


Разница с функцией вероятности
difference(x)
x

In [89]:

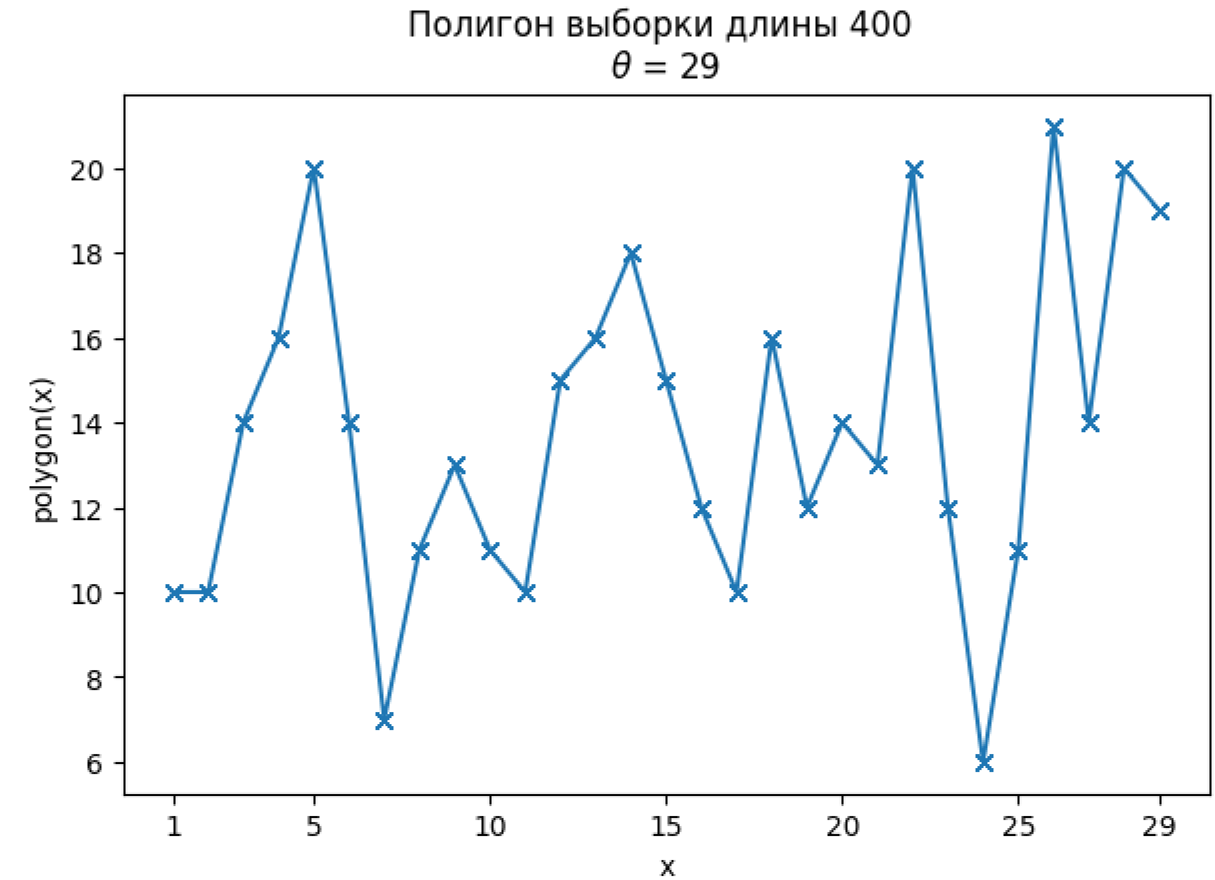
```
# n=200
fig, ax = plt.subplots(2,1, figsize=(7,10))
ax[0].yaxis.set_major_locator(MaxNLocator(integer=True))
ax[0].plot(sample_xi[3][0], Y_pol[3][0], '-x')
ax[0].set(xticks = [1]+list(range(5,26,5))+[29],
          xlabel = 'x', ylabel = 'polygon(x)',
          title = 'Полигон выборки длины 200 \n$\\theta$ = 29')

Y_dr[3][0][0], Y_dr[3][0][30] = 0, 0
ax[1].stem(X_real, Y_dr[3][0])
ax[1].set(xticks = [1]+list(range(5,26,5))+[29], xmargin = 0,
          xlabel = 'x', ylabel = 'difference(x)',
          title = 'Разница с функцией вероятности');
```

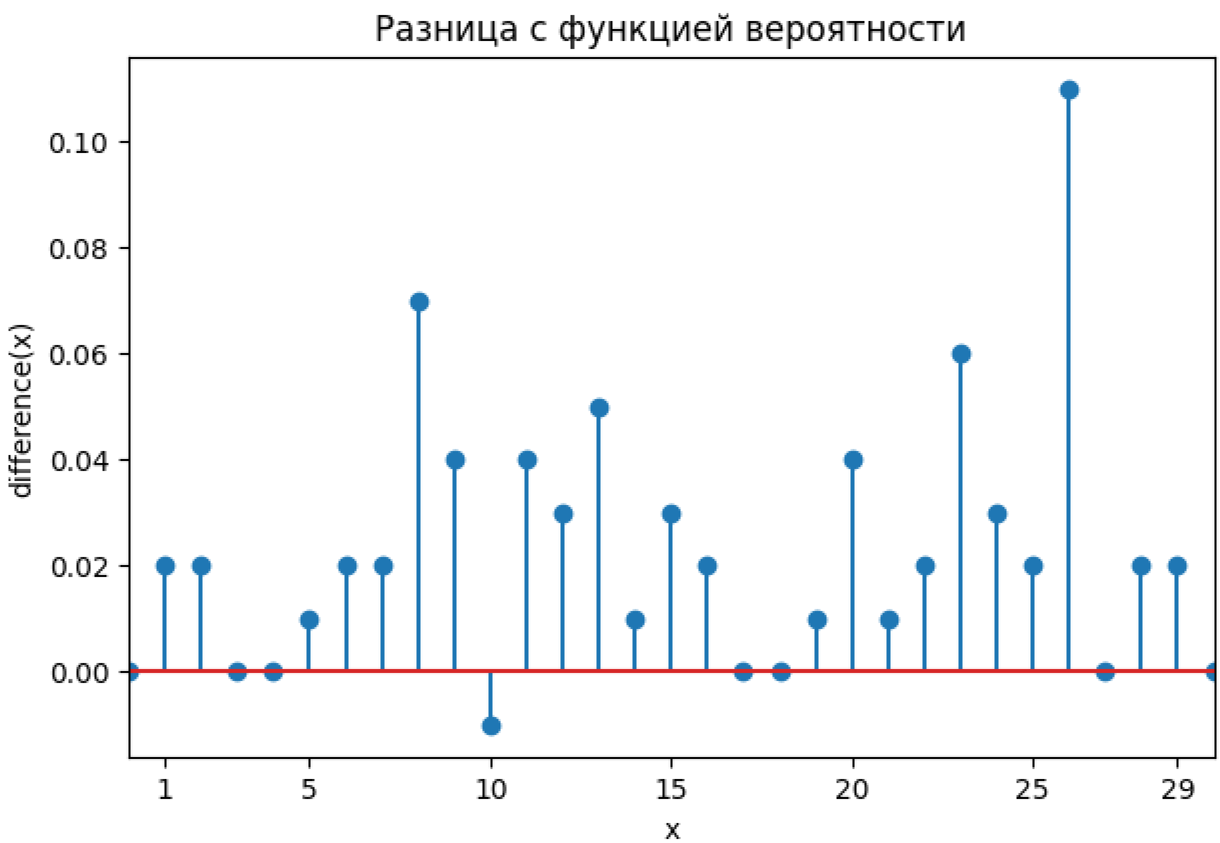

Полигон выборки длины 200
θ = 29
polygon(x)
x


Разница с функцией вероятности
difference(x)
x

In [91]:

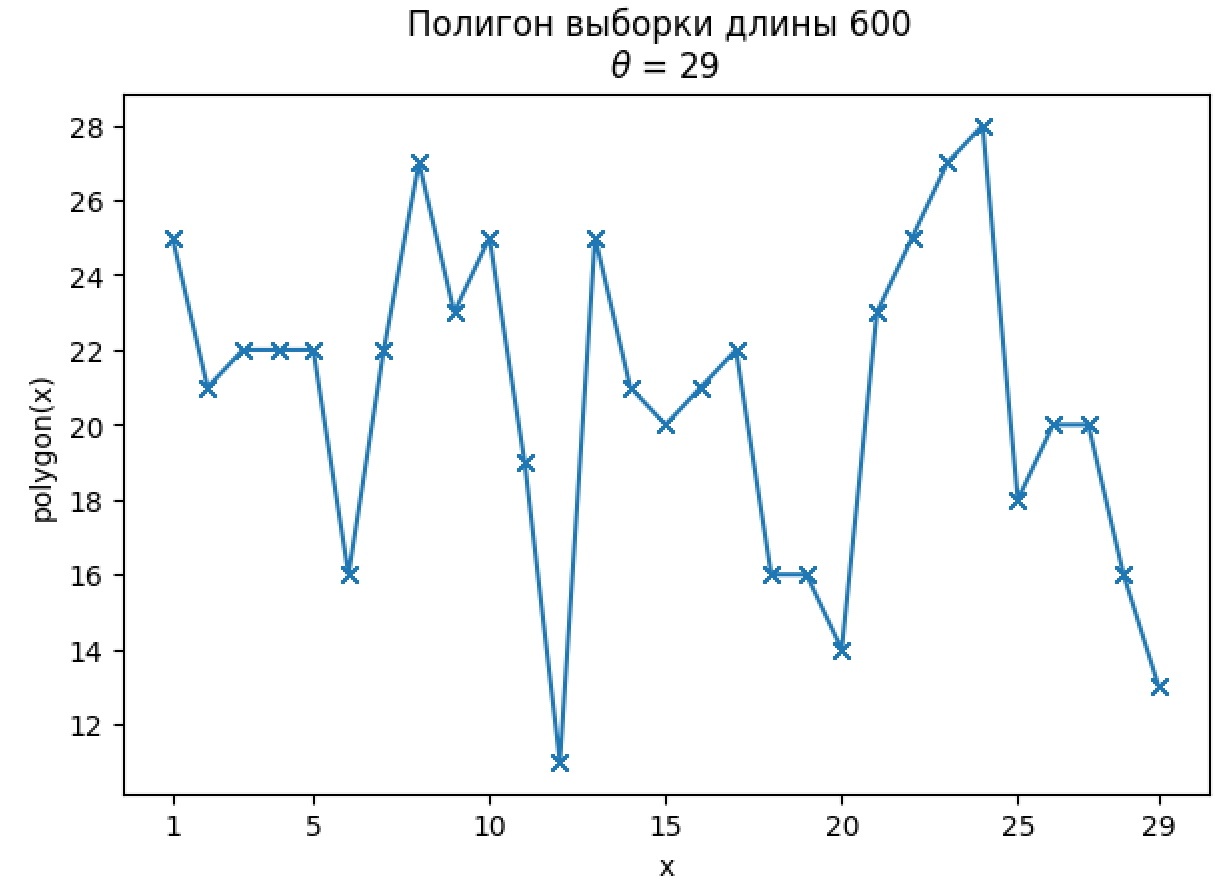
```
# n=400
fig, ax = plt.subplots(2,1, figsize=(7,10))
ax[0].yaxis.set_major_locator(MaxNLocator(integer=True))
ax[0].plot(sample_xi[4][0], Y_pol[4][0], '-x')
ax[0].set(xticks = [1]+list(range(5,26,5))+[29],
          xlabel = 'x', ylabel = 'polygon(x)',
          title = 'Полигон выборки длины 400 \n$\\theta$ = 29')

Y_dr[4][0][0], Y_dr[4][0][30] = 0, 0
ax[1].stem(X_real, Y_dr[2][0])
ax[1].set(xticks = [1]+list(range(5,26,5))+[29], xmargin = 0,
          xlabel = 'x', ylabel = 'difference(x)',
          title = 'Разница с функцией вероятности');
```

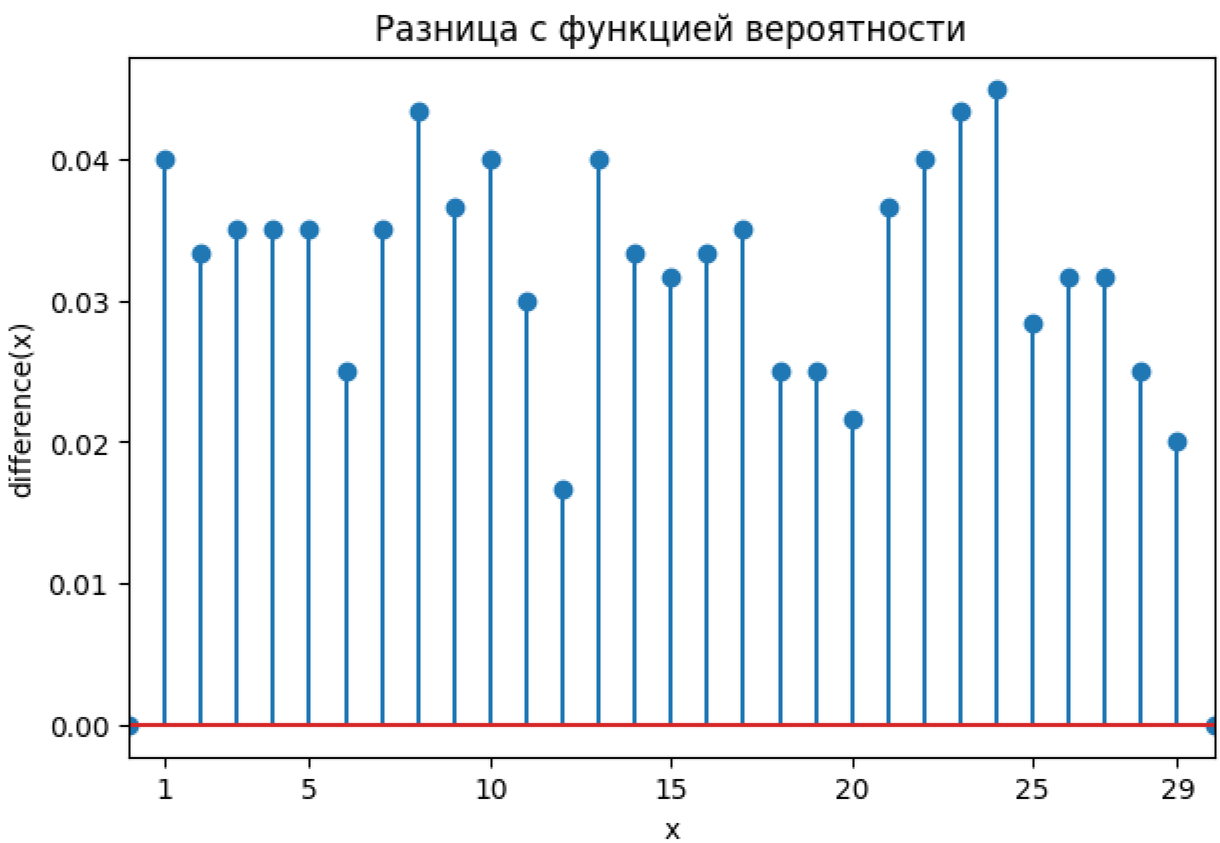

Полигон выборки длины 400
θ = 29
polygon(x)
x
20
18
16
14
12
10
8
6
1
5
10
15
20
25
29


Разница с функцией вероятности
difference(x)
x
0.10
0.08
0.06
0.04
0.02
0.00
1
5
10
15
20
25
29

In [92]:

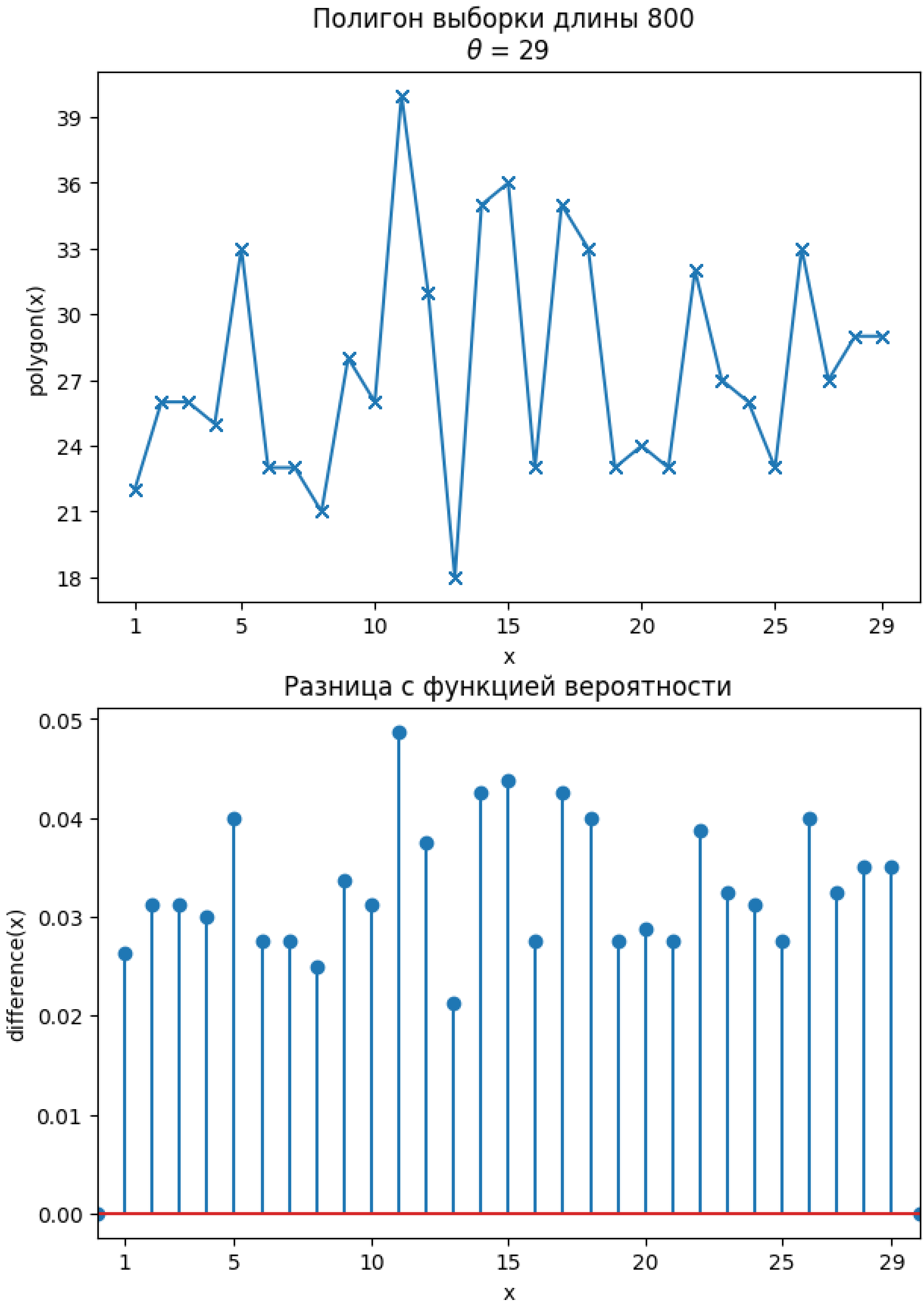
```
# n=600
fig, ax = plt.subplots(2,1, figsize=(7,10))
ax[0].yaxis.set_major_locator(MaxNLocator(integer=True))
ax[0].plot(sample_xi[5][0], Y_pol[5][0], '-x')
ax[0].set(xticks = [1]+list(range(5,26,5))+[29],
          xlabel = 'x', ylabel = 'polygon(x)',
          title = 'Полигон выборки длины 600 \n$\\theta$ = 29')

Y_dr[5][0][0], Y_dr[5][0][30] = 0, 0
ax[1].stem(X_real, Y_dr[5][0])
ax[1].set(xticks = [1]+list(range(5,26,5))+[29], xmargin = 0,
          xlabel = 'x', ylabel = 'difference(x)',
          title = 'Разница с функцией вероятности');
```


Полигон выборки длины 600
θ = 29
polygon(x)
28
26
24
22
20
18
16
14
12
1
5
10
15
20
25
29
x


Разница с функцией вероятности
difference(x)
0.04
0.03
0.02
0.01
0.00
1
5
10
15
20
25
29
x

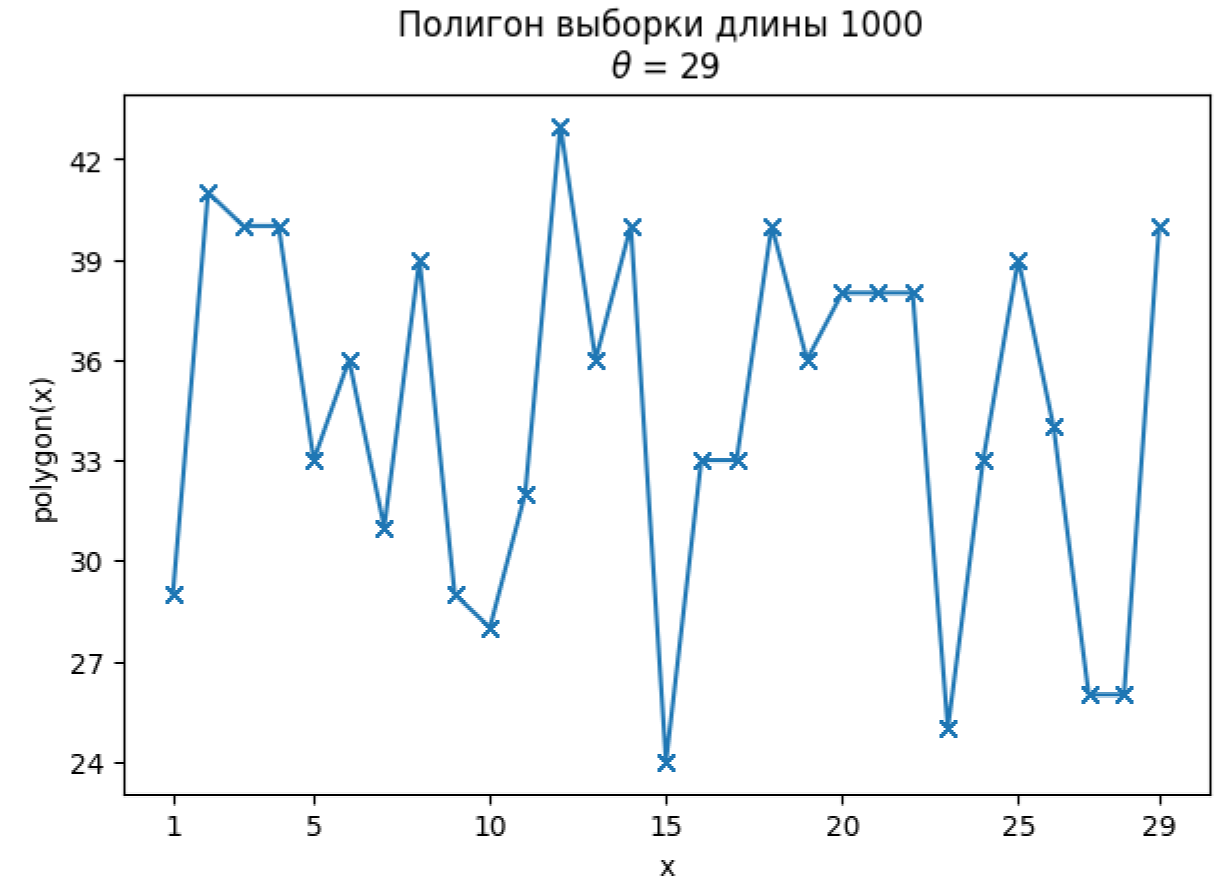
In [93]:

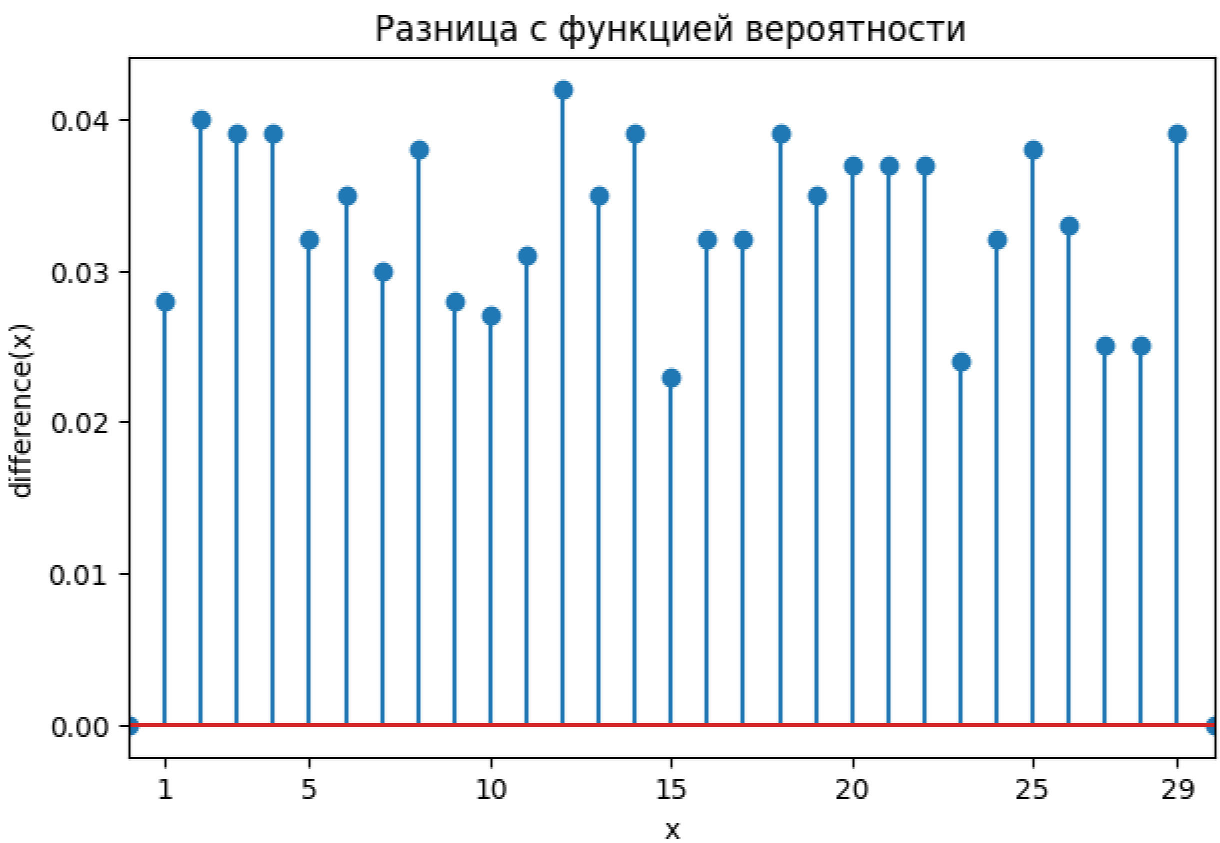
```
# n=800
fig, ax = plt.subplots(2,1, figsize=(7,10))
ax[0].yaxis.set_major_locator(MaxNLocator(integer=True))
ax[0].plot(sample_xi[6][0], Y_pol[6][0], '-x')
ax[0].set(xticks = [1]+list(range(5,26,5))+[29],
          xlabel = 'x', ylabel = 'polygon(x)',
          title = 'Полигон выборки длины 800 \n$\\theta$ = 29')

Y_dr[6][0][0], Y_dr[6][0][30] = 0, 0
ax[1].stem(X_real, Y_dr[6][0])
ax[1].set(xticks = [1]+list(range(5,26,5))+[29], xmargin = 0,
          xlabel = 'x', ylabel = 'difference(x)',
          title = 'Разница с функцией вероятности');
```


Полигон выборки длины 800
θ = 29
polygon(x)
x
Разница с функцией вероятности
difference(x)
x

In [94]:

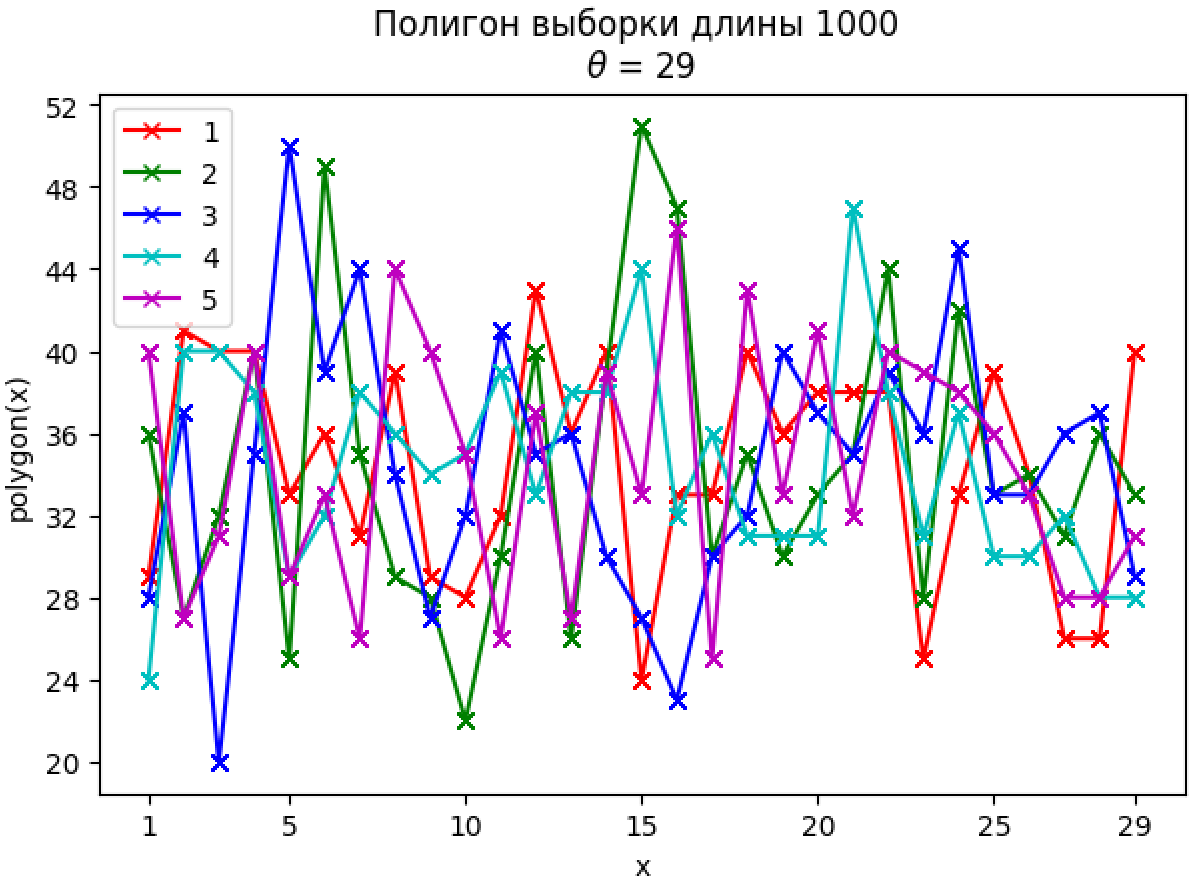
```
# n=1000
fig, ax = plt.subplots(2,1, figsize=(7,10))
ax[0].yaxis.set_major_locator(MaxNLocator(integer=True))
ax[0].plot(sample_xi[7][0], Y_pol[7][0], '-x')
ax[0].set(xticks = [1]+list(range(5,26,5))+[29],
          xlabel = 'x', ylabel = 'polygon(x)',
          title = 'Полигон выборки длины 1000 \n$\\theta$ = 29')

Y_dr[7][0][0], Y_dr[7][0][30] = 0, 0
ax[1].stem(X_real, Y_dr[7][0])
ax[1].set(xticks = [1]+list(range(5,26,5))+[29], xmargin = 0,
          xlabel = 'x', ylabel = 'difference(x)',
          title = 'Разница с функцией вероятности');
```

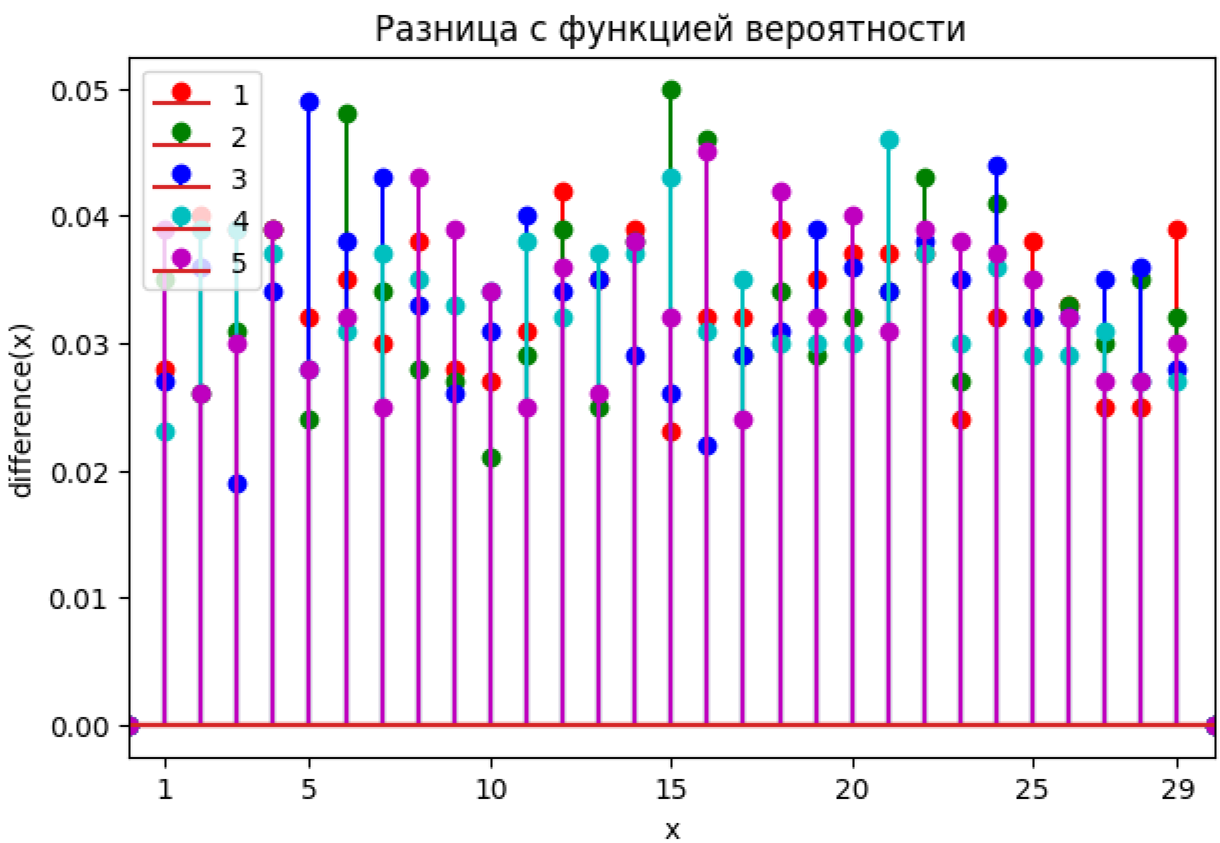

Полигон выборки длины 1000
θ = 29
polygon(x)
x


Разница с функцией вероятности
difference(x)
x

In [98]:

```
# n=1000, all
fig, ax = plt.subplots(2,1, figsize=(7,10))
ax[0].yaxis.set_major_locator(MaxNLocator(integer=True))
for i in range(5):
    ax[0].plot(sample_xi[7][i], Y_pol[7][i], 'rgbcm'[i]+'-x')
ax[0].set(xticks = [1]+list(range(5,26,5))+[29],
          xlabel = 'x', ylabel = 'polygon(x)',
          title = 'Полигон выборки длины 1000 \n$\\theta$ = 29')
ax[0].legend([1, 2, 3, 4, 5], loc='upper left')

for i in range(5):
    Y_dr[7][i][0], Y_dr[7][i][30] = 0, 0
    ax[1].stem(X_real, Y_dr[7][i], 'rgbcm'[i])
ax[1].set(xticks = [1]+list(range(5,26,5))+[29], xmargin = 0,
          xlabel = 'x', ylabel = 'difference(x)',
          title = 'Разница с функцией вероятности')
ax[1].legend([1, 2, 3, 4, 5], loc='upper left');
```





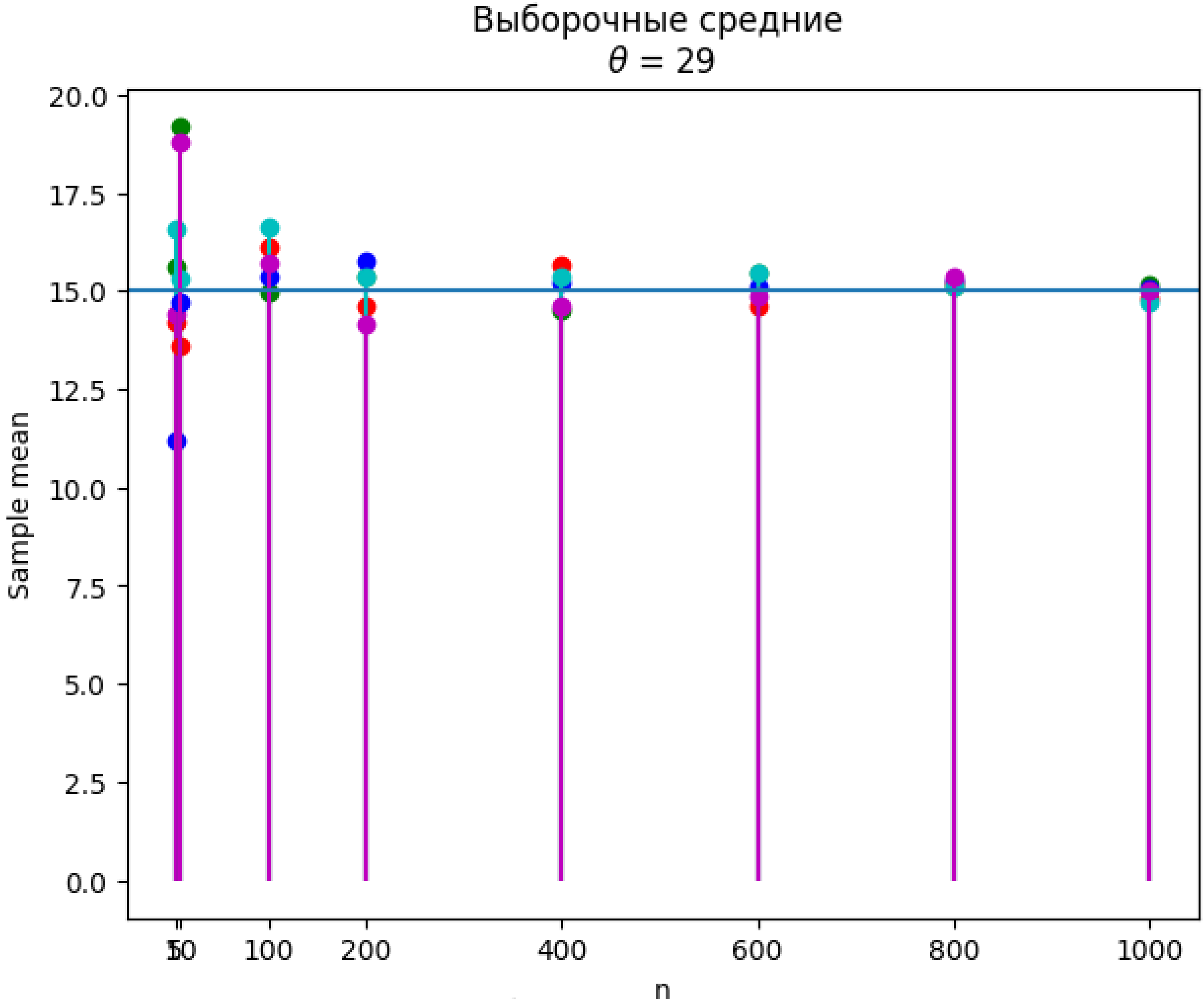
По полигону частот что-то толковое сказать сложно, в силу структуры графика, однако по разнице вероятности эмперической и математической уже можно сделать некоторые выводы (стоит отметить, что второй график напрямую зависит от первого, так что один тут скорее как логическое дополнение к другому). Так, по увеличению количества элементов в выборке можно заметить, что разница в вероятности спадает. Это соответствует теореме о схождении эмперической вероятности к математической (в курсе лекций это теорема о функциях распределения, но одно следует из другого).

Задание 3

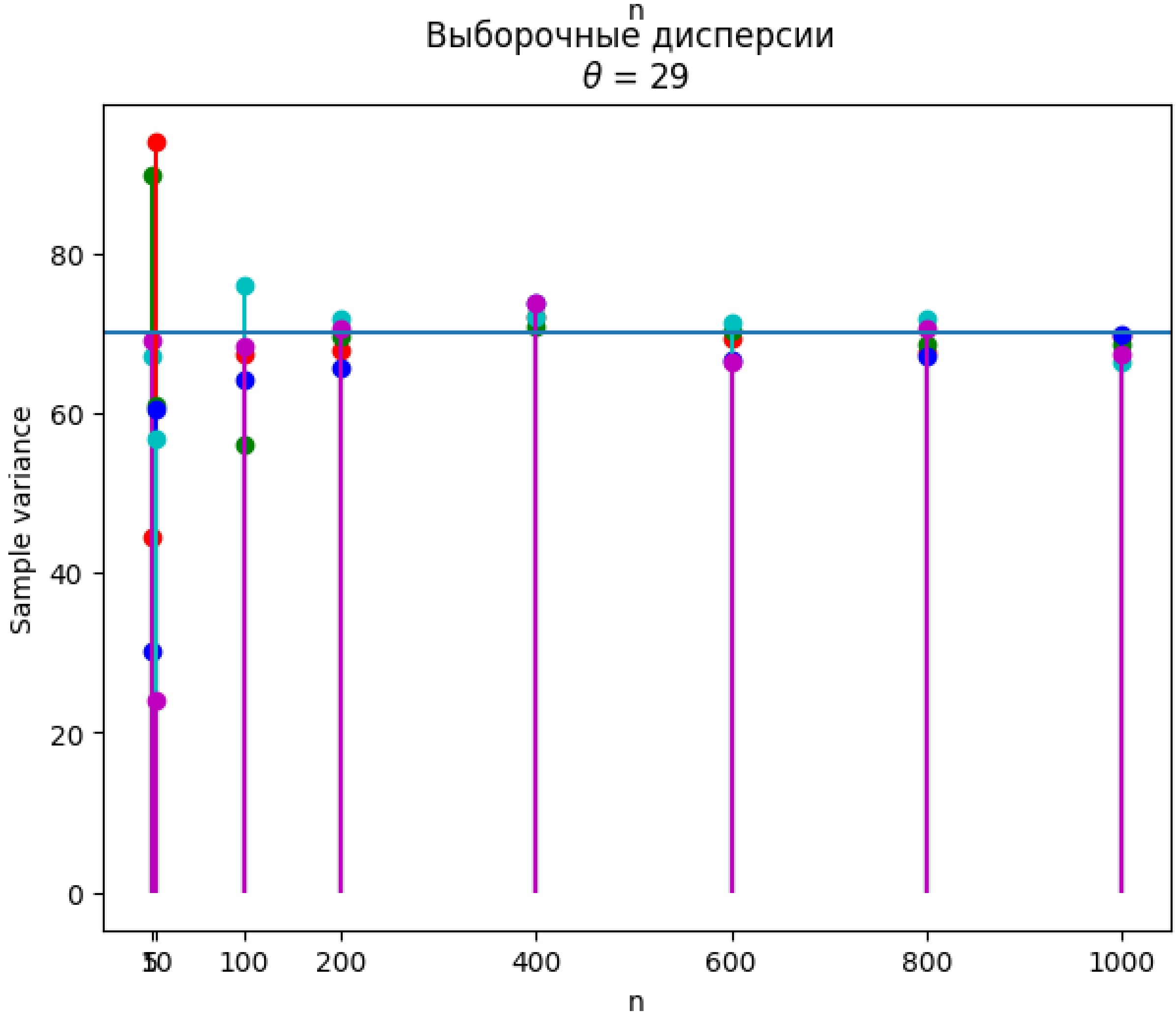
In [109]:
```
def xi_sample_mean(sample):
    return sum(sample)/len(sample)

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

means = np.array([[xi_sample_mean(sample_xi[k][j])for j in range(5)]
         for k in range(len(n))])
variances = np.array([[xi_sample_variance(sample_xi[k][j])for j in range(5)]
             for k in range(len(n))])
expectation = (29+1)/2
variance = (29*29-1)/12
```

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], means[k][j], 'rgbcm'[j])
ax[0].axhline(expectation)
ax[0].set(xticks = n, xlabel = 'n', ylabel = 'Sample mean',
          title = 'Выборочные средние \n$\\theta$ = 29')

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



Выборочные средние
θ = 29
Sample mean
n



Выборочные дисперсии
θ = 29
Sample variance
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 [119]:
```
means_t = np.array([['%.2f' % i for i in j] for j in (means-expectation)])
variances_t = np.array([['%.2f' % i for i in j]
                        for j in (variances-variance)])
fig, ax = plt.subplots(2,1, figsize=(7,7))
ax[0].table(cellText = means_t, 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(expectation));

ax[1].table(cellText = variances_t, 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(variance));
```

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

	1	2	3	4	5
5	-0.80	0.60	-3.80	1.60	-0.60
10	-1.40	4.20	-0.30	0.30	3.80
100	1.14	-0.03	0.39	1.61	0.72
200	-0.40	0.38	0.77	0.35	-0.82
400	0.65	-0.49	0.24	0.38	-0.38
600	-0.40	0.46	0.14	0.46	-0.13
800	0.24	0.10	0.27	0.12	0.35
1000	-0.20	0.16	0.08	-0.26	-0.00

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

	1	2	3	4	5
5	-25.44	19.84	-39.84	-2.96	-0.96
10	24.04	-9.04	-9.59	-13.19	-45.84
100	-2.66	-14.05	-5.86	6.12	-1.56
200	-2.21	-0.50	-4.41	1.72	0.53
400	2.19	0.87	3.71	2.16	3.74
600	-0.59	0.28	-3.36	1.41	-3.69
800	-2.25	-1.45	-2.97	1.78	0.52
1000	-0.34	-1.46	-0.06	-3.63	-2.48

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

Абсолютно непрерывное

Задание 1

```
In [ ]:
```