ЛАБОРАТОРНАЯ РАБОТА №1

НАЗАРОВ РУСТАМ M3132 368563

АНАЛИТИЧЕСКАЯ ЧАСТЬ

10.03.2023 Назаров Рустам М3132 368563

Лабораторная работа

Аналитическая часть.

$f(x) = 2x - x^3 \quad [-1; 2]$

Нашёл несколько точек

x	-1	0	2	1
$f(x)$	-1	0	-4	1

Найдем максимумы и минимумы

$f'(x) = 2 - 3x^2$

$2 - 3x^2 = 0$

$-3x^2 = -2$

$x^2 = \frac{2}{3}$

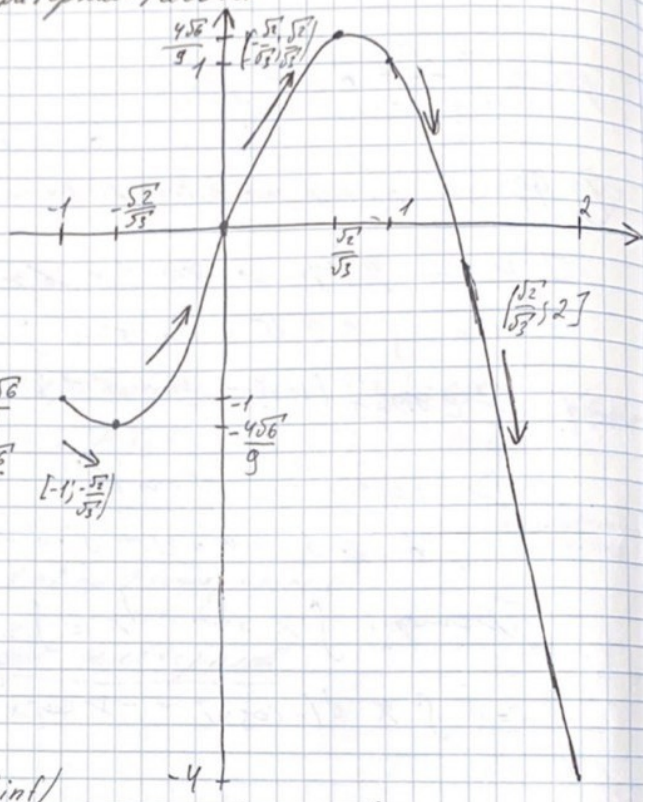
Экстремумы

$x_1 = \frac{\sqrt{2}}{\sqrt{3}} \quad f(x_1) = \frac{4\sqrt{6}}{9}$

$x_2 = -\frac{\sqrt{2}}{\sqrt{3}} \quad f(x_2) = -\frac{4\sqrt{6}}{9}$

$x_3 = -1 \quad f(x_3) = -1$

$x_4 = 2 \quad f(x_4) = -4$



Пусть T - равномерное разбиение на n частей

1) Находим суммы Дарбу

Нижняя:

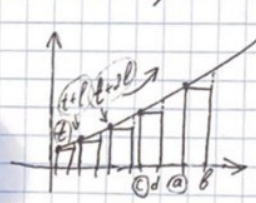
$S_{\underline{T}n} = \sum_{i=1}^{n} m_i \cdot \Delta x$ (m_i - "локальный inf")

верхняя:

$S_T^{\wedge n} = \sum_{i=1}^{n} M_i \cdot \Delta x$ (M_i - "локальный sup")

Как найти локальный inf?

Рассмотрим 2 вида монотонности:



Из рисунка возрастания видно, что, смотря на подотрезки, m_i будет левой точкой подотрезка

Из рисунка убывания видно, что, смотря на подотрезки, m_i будет правой крайней точкой подотрезка

Пусть ширина подотрезка = l, тогда

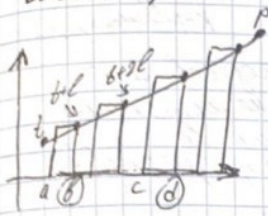
Пусть рассматриваем промежуток $[t; p]$, тогда

$\vec{S_{\underline{T}n}} = \sum_{i=1}^{n} \Delta x \cdot m_i = \sum_{1}^{n} l \cdot f(\underbrace{t + l \cdot (i-1)}_{\text{левая точка "на шаг назад"}})$

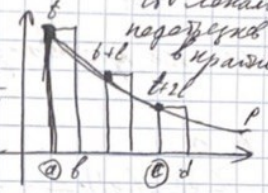
$\vec{S_{\underline{T}n}} = \sum_{i=1}^{n} \Delta x \cdot m_i = \sum_{i=1}^{n} l \cdot f(\underbrace{t + l \cdot i}_{\text{правая точка "шаг вперед"}})$

Как найти локальную sup?

Рассмотрим 2 вида монотонности:



Рассматривая подотрезки, видно, что (по возрастанию), что локальный sup здесь справа, в крайней точке каждого подотрезка



На убывании видно, что локальный inf подотрезка здесь слева, в крайней точке подотрезка.

$$S_T^{\overrightarrow{\lambda n}} = \sum_{i=1}^{n} \Delta x \cdot M_i = \sum_{i=1}^{n} l \cdot f(t + l \cdot i)$$

каждая правая точка

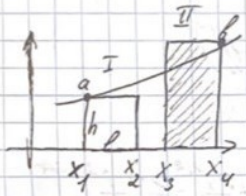
$$S_T^{\overrightarrow{\lambda n}} = \sum_{i=1}^{n} \Delta x \cdot M_i = \sum_{i=1}^{n} l \cdot f(t + l(i-1))$$

каждая левая точка

Почему Δx - это ширина подотрезка?

Почему m_i и M_i - это $f(\ )$ от какой-то точки?

Сумма Дарбу по сути это сумма площадей прямоугольников, где Δx - ширина, m_i / M_i - высота



I нижний: $\Delta x = x_{i+1} - x_i = x_2 - x_1$

$l = x_2 - x_1$

то есть одно и то же

и так как у нас равномерное обрезание то Δx всегда будет один.

~~Красный~~ $h = a$ $a = f(x_1)$

берем точку на OX, вставляем в f, вот и inf $\{m_i\}$ $h = m_i = f(x_i)$

II верхний: аналогично.

В итоге и получается: $S_{\square} = h \cdot l = \Delta x \cdot m_i$

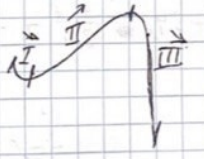
$\downarrow$

$f(x_i)$

Найдем ширину каждого подотрезка, то есть Δx

$$\Delta x = \frac{\text{длина}}{\text{кол-во}} = \frac{2-(-1)}{n} = \frac{3}{n}$$

Так как у нас функция немонотонна, разделим на 3 монотонных промежутка



I - $\left[-1; -\frac{\sqrt{2}}{\sqrt{3}}\right]$

II - $\left(-\frac{\sqrt{2}}{\sqrt{3}}; \frac{\sqrt{2}}{\sqrt{3}}\right)$

III - $\left(\frac{\sqrt{2}}{\sqrt{3}}; 2\right]$

Тогда: $[-1;2]$ $\quad [-1;-\frac{\sqrt2}{\sqrt3}) \quad (-\frac{\sqrt2}{\sqrt3};\frac{\sqrt2}{\sqrt3}) \quad [\frac{\sqrt2}{\sqrt3};2]$ $\quad$ F - равномерное разбиение на $\frac{3-\sqrt6}{9}n$ подотрезков

Нижняя: $\underline{S}_T n = \sum_{i=1}^{n} m_i \Delta x = \underline{S}_F n + \underline{S}_K n + \underline{S}_N n$ $\quad$ N - равномерное разбиение на $\frac{3-\sqrt6}{9}n$ подотрезков

K - равномерное разбиение на $\frac{2\sqrt6}{9}n$ подотрезков

Мы поделили на промежутки, ширина подотрезков Δx не изменилась

Теперь не n, а ко изменилось кол-во подотрезков!

I $\frac{\text{длина}}{\text{ширина } \Delta x} = \left(-\frac{\sqrt2}{\sqrt3} - (-1)\right) : \frac{3}{n} = \frac{(3-\sqrt6)n}{9}$ $\quad \Rightarrow$ левая грань: -1, правая грань: $-\frac{\sqrt2}{\sqrt3}$

II $\left(\frac{\sqrt2}{\sqrt3} - \left(-\frac{\sqrt2}{\sqrt3}\right)\right) : \frac{3}{n} = \frac{2\sqrt6}{9}n$ $\quad \Rightarrow$ Л: $-\frac{\sqrt2}{\sqrt3}$, П: $\frac{\sqrt2}{\sqrt3}$

III $\left(2 - \frac{\sqrt2}{\sqrt3}\right) : \frac{3}{n} = \frac{(6-\sqrt6)n}{9}$ $\quad \Rightarrow$ Л: $\frac{\sqrt2}{\sqrt3}$, П: 2

Тогда: $[-1;2]$

$$\underline{S}_T n = \sum_{i=1}^{\frac{3-\sqrt6}{9}n} (\Delta x \cdot m_i) + \sum_{i=1}^{\frac{2\sqrt6}{9}n} (\Delta x \cdot m_i) + \sum_{i=1}^{\frac{(6-\sqrt6)}{9}n} (\Delta x \cdot m_i) =$$

$$= \frac{3}{n} \cdot \sum_{i=1}^{\frac{(3-\sqrt6)}{9}n} f\left(\underbrace{-1}_{\text{Л.Г.}} + \underbrace{\frac{3}{n}}_{\Delta n} \cdot \underbrace{i}_{\substack{\text{правая} \\ \text{точка}}}\right) + \frac{3}{n} \cdot \sum_{i=1}^{\frac{2\sqrt6}{9}n} f\left(-\frac{\sqrt2}{\sqrt3} + \frac{3}{n}\underbrace{(i-1)}_{\substack{\text{левая} \\ \text{точка}}}\right) + \frac{3}{n} \cdot \sum_{i=1}^{\frac{(6-\sqrt6)}{9}n} f\left(\frac{\sqrt2}{\sqrt3} + \frac{3}{n} \cdot i\right) =$$

$$= \frac{3}{n} \cdot \sum_{i=1}^{\frac{(3-\sqrt6)}{9}n} \left(2\left(-1+\frac{3}{n}\cdot i\right) - \left(-1+\frac{3}{n}\cdot i\right)^3\right) + \frac{3}{n} \cdot \sum_{i=1}^{\frac{2\sqrt6}{9}n} \left(2\left(-\frac{\sqrt2}{\sqrt3} + \frac{3}{n}(i-1)\right) - \left(-\frac{\sqrt2}{\sqrt3} + \frac{3}{n}(i-1)\right)^3\right) +$$

$$+ \frac{3}{n} \cdot \sum_{i=1}^{\frac{6-\sqrt6}{9}n} \left(2\left(\frac{\sqrt2}{\sqrt3} + \frac{3}{n}\cdot i\right) - \left(\frac{\sqrt2}{\sqrt3} + \frac{3}{n}\cdot i\right)^3\right) = (=)$$

$$2\left(-1+\frac{3}{n}\cdot i\right) - \left(-1+\frac{3}{n}\cdot i\right)^3 = -2 + \frac{6i}{n} - \left(-1 + 3\left|\frac{3i}{n}\right| - 3\left(\frac{3i}{n}\right)^2 + \left(\frac{3i}{n}\right)^3\right) = -1 - \frac{3i}{n} + \frac{27i^2}{n^2} - \frac{27i^3}{n^3}$$

$$(=) \frac{3}{n} \cdot \left(-\sum_{1}^{\frac{3-\sqrt6}{9}n}(1) - \frac{3}{n}\cdot\sum_{1}^{\frac{3-\sqrt6}{9}n}(i) + \frac{27}{n^2}\cdot\sum_{1}^{\frac{3-\sqrt6}{9}n}(i^2) - \frac{27}{n^3}\cdot\sum_{1}^{\frac{3-\sqrt6}{9}n}(i^3)\right) + \frac{3}{n}\cdot\left(-\frac{4\sqrt3}{3\sqrt3}\cdot\sum_{i=1}^{\frac{2\sqrt6}{9}n}(1) + \frac{27\sqrt2}{\sqrt3\, n^2}\cdot\sum_{1}^{\frac{2\sqrt6}{9}n}(i^2) - \frac{27}{n^3}\cdot\sum_{1}^{\frac{2\sqrt6}{9}n}(i^3)\right) +$$

$$+ \frac{3}{n}\left(\frac{4\sqrt2}{3\sqrt3}\cdot\sum_{1}^{\frac{6-\sqrt6}{9}n}(1) - \frac{27\sqrt2}{\sqrt3\, n^2}\sum(i) - \frac{27}{n^3}\sum_{1}^{\frac{6-\sqrt6}{9}n}(i^2)\right) (=)$$

$$\sum_{1}^{n} c = \frac{2a_1 + d(n-1)}{2}\cdot n \qquad \sum_{1}^{n} x^2 = \frac{n(n+1)(2n+1)}{6} \qquad \sum x^3 = \frac{n^2(n+1)^2}{4}$$

$$(=) \frac{3}{n}\cdot\Bigg(- \frac{(3-\sqrt6)n}{9} + \frac{27}{n^2}\cdot\frac{(3-\sqrt6)n}{54}\cdot\frac{(3-\sqrt6)n+9}{9} - \frac{27}{n^3}\cdot\frac{(3-\sqrt6)^2 n^2}{81\cdot4}\cdot\left(\frac{(3-\sqrt6)n}{9}+1\right)^2\Bigg) +$$

$$+ \frac{3}{n}\cdot\left(-\frac{4\sqrt2}{3\sqrt3}\cdot\frac{2\sqrt6 n}{9} + \frac{27\sqrt2}{\sqrt3\, n^2}\cdot\frac{2\sqrt6 n}{54}\cdot\frac{2\sqrt6 n+9}{9} - \frac{27}{n^3}\cdot\frac{24n^2}{81\cdot4}\cdot\left(\frac{2\sqrt6 n}{9}+1\right)^2\right) +$$

$$+ \frac{3}{n}\cdot\left(\frac{4\sqrt2}{3\sqrt3}\cdot\frac{(6-\sqrt6)n}{9} - \frac{27\sqrt2}{\sqrt3\cdot n^2}\cdot\frac{6-\sqrt6 n}{54}\cdot\frac{(6-\sqrt6)n+9}{9}\cdot\frac{2(6-\sqrt6)n+9}{9} - \frac{27}{n^3}\cdot\frac{(6-\sqrt6)^2 n^2}{81\cdot4}\cdot\frac{(6-\sqrt6)n+9)^2}{81}\right) =$$

$$= \frac{1}{36n}\left(-4n + \frac{27}{n} - 24\sqrt6 + 54\right) - \frac{4\sqrt6}{3n} + \frac{-10n^2 - 12\sqrt6\, n - 108n - 135}{54n^2} =$$

$$\underline{S}_T n = \frac{-3\sqrt3\, n^2 + 27\sqrt3\, n + 32\sqrt{2n} - 18\sqrt{3n}}{4\sqrt3\, n^2}$$

8 Верхняя:

$$S_\tau^{\wedge n} = \sum_{i=1}^{n} M_i \Delta x = S_{[-1;\frac{\sqrt{2}}{\sqrt{3}}]}^{\wedge n} + S_{[\frac{\sqrt{2}}{\sqrt{3}};\frac{\sqrt{2}}{\sqrt{3}}]}^{\wedge n} + S_{[\frac{\sqrt{2}}{\sqrt{3}};2]}^{\wedge n}$$

$$S_\tau^{\wedge n} = \sum_{i=1}^{\frac{(3-\sqrt{6})n}{9}} (\Delta x \cdot M_i) + \sum_{i=1}^{\frac{2\sqrt{6}n}{9}} (\Delta x \cdot M_i) + \sum_{i=1}^{\frac{(6-5\sqrt{6})n}{9}} (\Delta x \cdot M_i) = \frac{3}{n} \cdot \sum_{i=1}^{\frac{(3-\sqrt{6})n}{9}} f\left(-1+\frac{3}{n}(i-1)\right) + \frac{3}{n} \cdot \sum_{i=1}^{\frac{2\sqrt{6}n}{9}} f\left(-\frac{\sqrt{2}}{\sqrt{3}}+\frac{3}{n}i\right) +$$

$$+ \frac{3}{n} \cdot \sum_{i=1}^{\frac{6-5\sqrt{6}}{9}n} f\left(\frac{\sqrt{2}}{\sqrt{3}}+\frac{3}{n}(i-1)\right)$$

$$S_\tau^{\wedge n} = \frac{3}{n} \cdot \sum_{i=1}^{\frac{(3-\sqrt{6})n}{9}} \left(2\left(-1+\frac{3}{n}(i-1)\right) - \left(-1+\frac{3}{n}(i-1)\right)^3\right) + \frac{3}{n} \cdot \sum_{i=1}^{\frac{2\sqrt{6}n}{9}} \left(2\left(-\frac{\sqrt{2}}{\sqrt{3}}+\frac{3}{n}\cdot i\right) - \left(-\frac{\sqrt{2}}{\sqrt{3}}+\frac{3}{n}\cdot i\right)^3\right) + \frac{3}{n} \cdot \sum_{i=1}^{\frac{6-5\sqrt{6}}{9}n} \left(2\left(\frac{\sqrt{2}}{\sqrt{3}}+\frac{3}{n}(i-1)\right) - \left(\frac{\sqrt{2}}{\sqrt{3}}+\frac{3}{n}(i-1)\right)^3\right)$$

$$S_\tau^{\wedge n} = \frac{3}{n} \cdot \left(\frac{-7n^2+24\sqrt{6}n-54n+27}{108n}\right) + \frac{3}{n}\left(\frac{4\sqrt{6}}{9}\right) + \frac{3}{n}\left(\frac{-10n^2+12\sqrt{6}n+108n-135}{54n}\right) =$$

$$= \frac{-3\sqrt{3}n^2+24\sqrt{3}+32\sqrt{2}n-18\sqrt{3}n}{4\sqrt{3}n^2}$$

2) Критерий Римана интегрируемости функции

$$\forall \varepsilon > 0 : S_\tau^{\wedge n} - S_{-n_\tau} < \varepsilon$$

$$\lim_{n\to\infty} (S^{\wedge n} - S_{-n}) = 0$$

$$\lim_{n\to\infty} \left(\frac{-3\sqrt{3}n^2+24\sqrt{3}+32\sqrt{2}n-18\sqrt{3}n}{4\sqrt{3}n^2} - \frac{-3\sqrt{3}n^2-24\sqrt{3}-32\sqrt{2}n-18\sqrt{3}n}{4\sqrt{3}n^2}\right) =$$

$$= \lim_{n\to\infty} \left(\underbrace{\frac{24\sqrt{3}}{4\sqrt{3}n^2}}_{\to 0} + \underbrace{\frac{64\sqrt{2}n}{4\sqrt{3}n^2}}_{\to 0}\right) = 0 \qquad S_\tau^{\wedge n} - S_{-n_\tau} < \varepsilon$$

$\Downarrow$

функция интегрируема

3) Предложение. Если предел верхней и нижней суммы Дарбу совпадают и конечны, то функция интегрируема.

$$\lim_{n\to\infty} (S_\tau^{\wedge n}) = \lim_{n\to\infty} \left(\frac{-3\sqrt{3}n^2+24\sqrt{3}+32\sqrt{2}n-18\sqrt{3}n}{4\sqrt{3}n^2}\right) = \frac{-3\sqrt{3}}{4\sqrt{3}} = \boxed{-\frac{3}{4}}$$

$$\lim_{n\to\infty} |S_{-n_\tau}| = \lim_{n\to\infty} \left(\frac{-3\sqrt{3}n^2-24\sqrt{3}-32\sqrt{2}n-18\sqrt{3}n}{4\sqrt{3}n^2}\right) = \frac{-3\sqrt{3}}{4\sqrt{3}} = \boxed{-\frac{3}{4}}$$

$\Rightarrow$ интегрируема

4) формула Ньютона-Лейбница $\int_a^b f(x)dx = F(b) - F(a) = \lim$

$$\int (2x - x^3)dx = x^2 - \frac{x^4}{4} = F(2) - F(-1) = 4 - 4 - 1 + \frac{1}{4} = \boxed{-\frac{3}{4}} = \text{совпал}$$

$\Downarrow$

правильно

ПОЛУЧЕННЫЕ КЛЮЧЕВЫЕ ЗНАЧЕНИЯ

1) СУММЫ ДАРБУ

Верхняя сумма

$$S^n = \frac{-3\sqrt{3}n^2 + 27\sqrt{3} + 32\sqrt{2}n - 18\sqrt{3}n}{4\sqrt{3}n^2}$$

Нижняя сумма

$$S_n = \frac{-3\sqrt{3}n^2 - 27\sqrt{3} - 32\sqrt{2}n - 18\sqrt{3}n}{4\sqrt{3}n^2}$$

2) КРИТЕРИЙ РИМАНА

$$\lim_{n \to \infty} S^n - S_n = 0$$

3) ПРЕДЕЛЫ СУММ

$$\lim_{n \to \infty} S^n = -\frac{3}{4} = -0.750$$

$$\lim_{n \to \infty} S_n = -\frac{3}{4} = -0.750$$

4) ФОРМУЛА НЬЮТОНА-ЛЕЙБНИЦА

$$\int_{-1}^{2} f(x)\,dx = \int_{-1}^{2} 2x - x^3\,dx = x^2 - \frac{x^4}{4} = F(2) - F(-1) = 4 - 4 - 1 + \frac{1}{4} = -\frac{3}{4} = -0.750$$

ЧИСЛЕННЫЙ МЕТОД

PYTHON 3.11

```
# Импортируем необходимые пакеты
from random import uniform # Рандом
from texttable import Texttable # Табоица в виде текста
import numpy as np # Вычисления для графиков
import matplotlib.pyplot as plt # Графики
import math # Математика
import matplotlib.patches as patches # Прямоугольники
%matplotlib inline

# Библиотека, по одному индексу из каждого берется информация о текущем рисунке
colors = ['green', 'red', 'orange', 'orange', 'orange', 'orange'] # Библиотека цветов
dots = ['ro', 'go', 'bo', 'bo', 'bo', 'bo'] # Библиотека точек
```

```
name = ['Upper sum', 'Lower sum', 'left', 'right', 'center', 'random']
# Библиотека имен
```

1) Вычисление интегральных сумм с параметрами f(x) [a;b] n разбиениями и оснащением

+

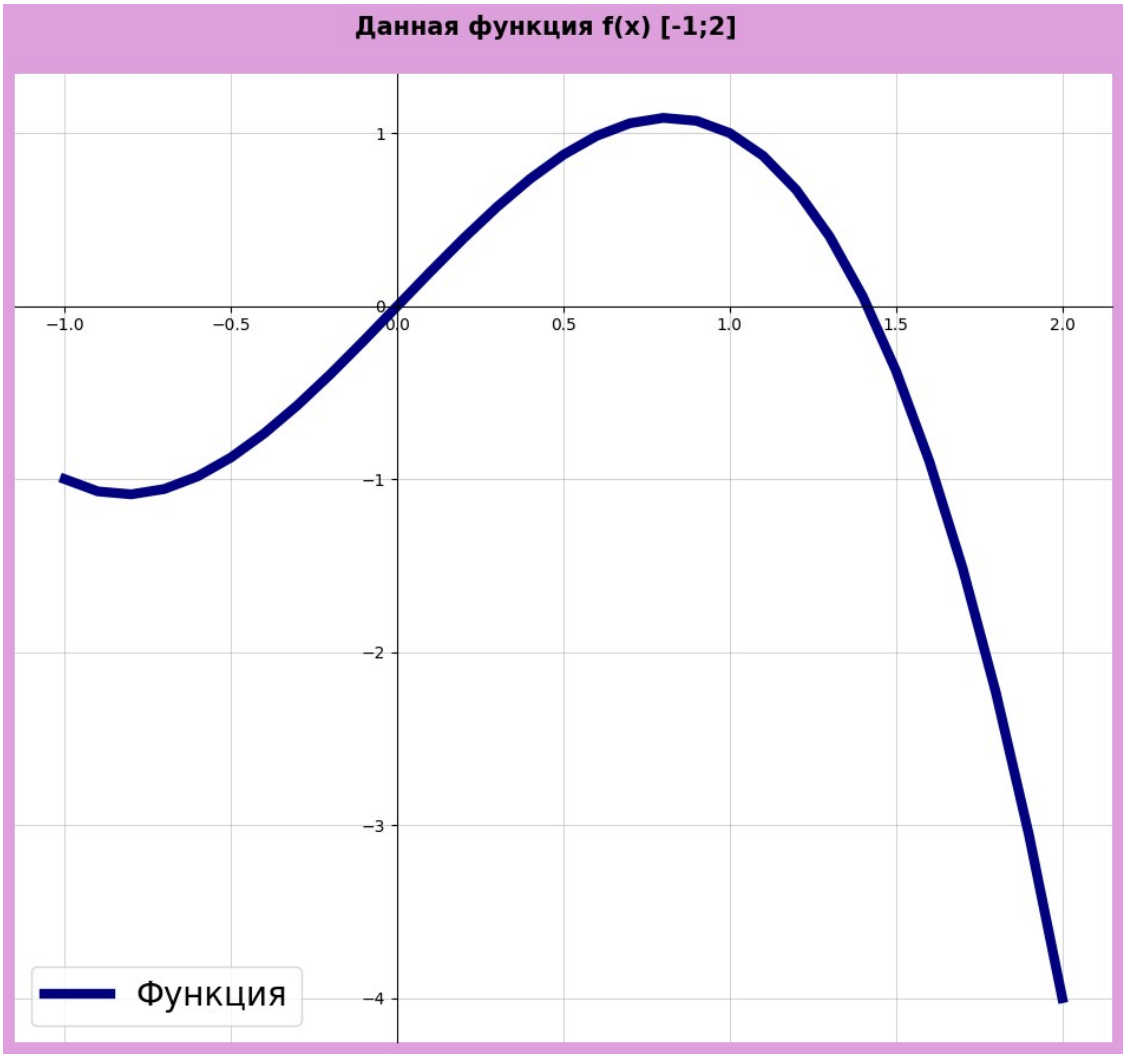
2) Визуализация

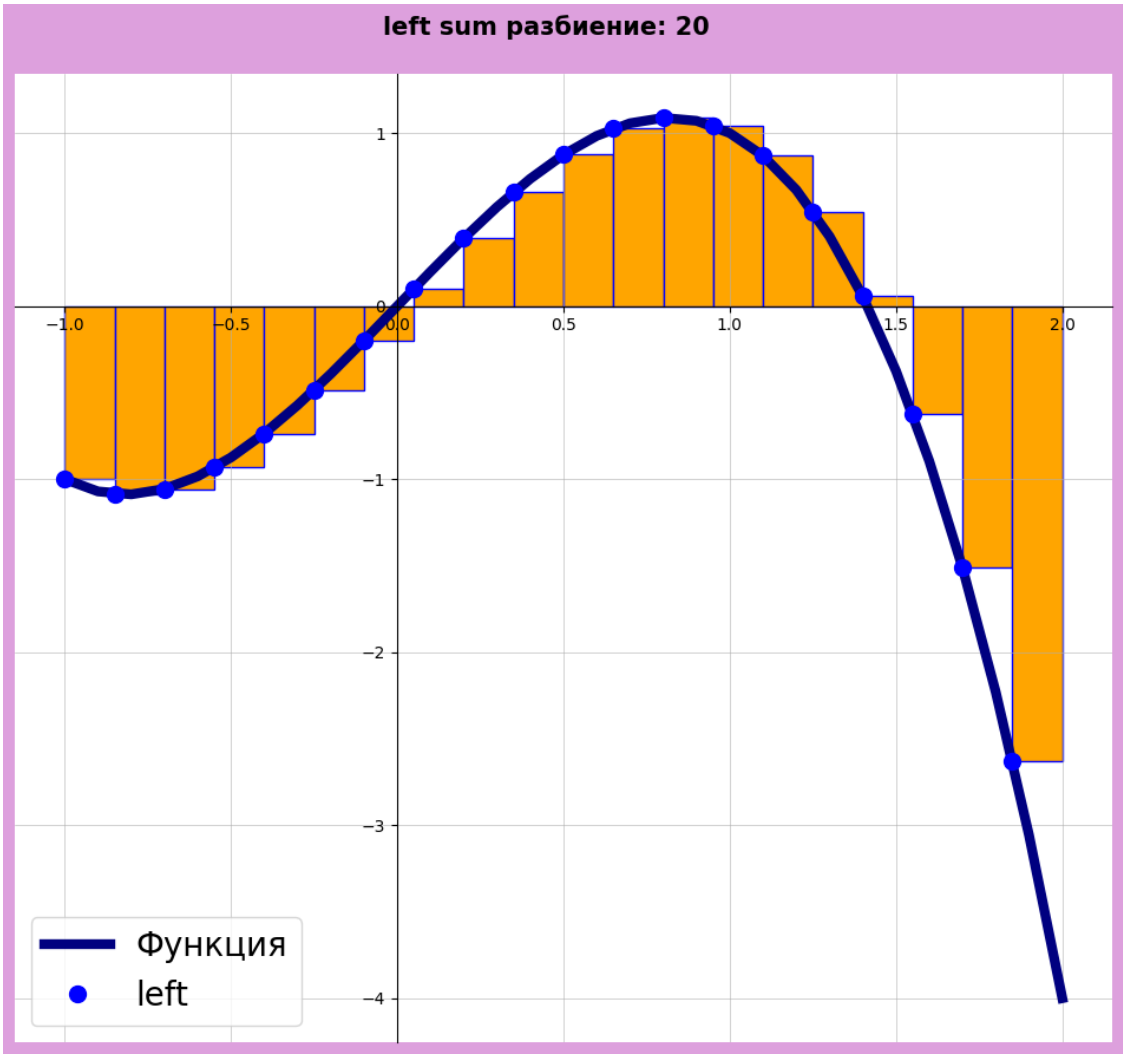
```
# Произвольная функция, можно заменить
def function(x):
    return 2 * x - x ** 3

# Данные для функции
start = -1 # левая граница
end = 2 # правая граница
number = 20 # число разбиений
kind = 'left' # оснащение

# Функция вычисления интегральных сумм по заданному промежутку,
# заданной функциии, заданному разбиения и заданному оснащения
def darbu_sums_methods(f, a, b, n, method='left', draw_funct=True,
draw=True):
    ind = name.index(method) # индекс в библиотеках
    # вычисляем ширину каждого подынтервала
    delta_x = (b - a) / n
    # инициализируем сумму
    sum = 0
    sum_x = [0] * n # массив точек по OX суммы
    square_x = [0] * n # массиов x'ов для секторов (прямоугольников)
    sum_y = [0] * n # массив точек по OУ суммы и прямоугольников,
    # так по высоте растягиваем до точки
    # проходим по каждому подынтервалу
    for i in range(n):
        # выбираем точку на подынтервале в соответствии с заданным
оснащением
        if method == 'left':
            x = a + i * delta_x # Так как начали с 0, нет (i - 1)
просто i
            square_x[i] = x # Левая точка и есть начало сектора
        elif method == 'right':
            x = a + (i + 1) * delta_x # Правая точка суммы
            square_x[i] = x - delta_x # Левая является началом
сектора, поэтому уменьшаем
        elif method == 'center':
            x = a + i * delta_x + delta_x / 2 # Центральная точка
            square_x[i] = x - delta_x / 2 # Уменьшаем на половину,
начало сектора слева
        elif method == 'random':
            x = uniform(a + i * delta_x, a + (i + 1) * delta_x) #
```

```
Рандов от левой до правой
            square_x[i] = a + i * delta_x # Взяли вычисление левой из
левого случая
        else:
            raise ValueError('Invalid method')
        # вычисляем значение функции в выбранной точке
        sum += f(x)
        # Берем точки суммы
        sum_x[i] = x
        sum_y[i] = f(x)
    # Умножаем на дельтту, получаем площадь прямоугольника с высотой
f_x и шириной дельта
    sum *= delta_x
    # Если надо вывести функцию, выводим
    if (draw_funct):
        draw_func(a, b, f)
    if (draw):
        # Выводим полученную сумму
        draw_sum(a, b, f, sum_x, square_x, sum_y, delta_x, ind)
        print(name[ind], sum)
    return sum

# Метод по рисованию данной функции
def draw_func(a, b, f):
    # диапозон
    x = np.arange(a, b + 0.1, 0.1)
    f_x = f(x) # Значения по ОУ

    fig = plt.figure(figsize=(12,10))   # Размер графика
    # Цвет окантовки
    fig.patch.set_facecolor('#DDA0DD')

    # Заголовок
    ax = fig.add_subplot()
    fig.subplots_adjust(top=0.93)
    fig.suptitle(f'Данная функция f(x) [{a};{b}]', fontsize=15,
fontweight='bold')

    # Размер координат осей абсцисс и ординат
    plt.xticks(fontsize = 12)
    plt.yticks(fontsize = 12)

    # Разметка на графике
    plt.grid(axis = 'both', linewidth = 0.4)

    # Выводим график функции
    plt.plot(x, f_x, color='#000080', linewidth=6, label='Функция')

    # Двигаем Оси ОХ ОУ до привычных
    ax.spines['left'].set_position('zero')
```

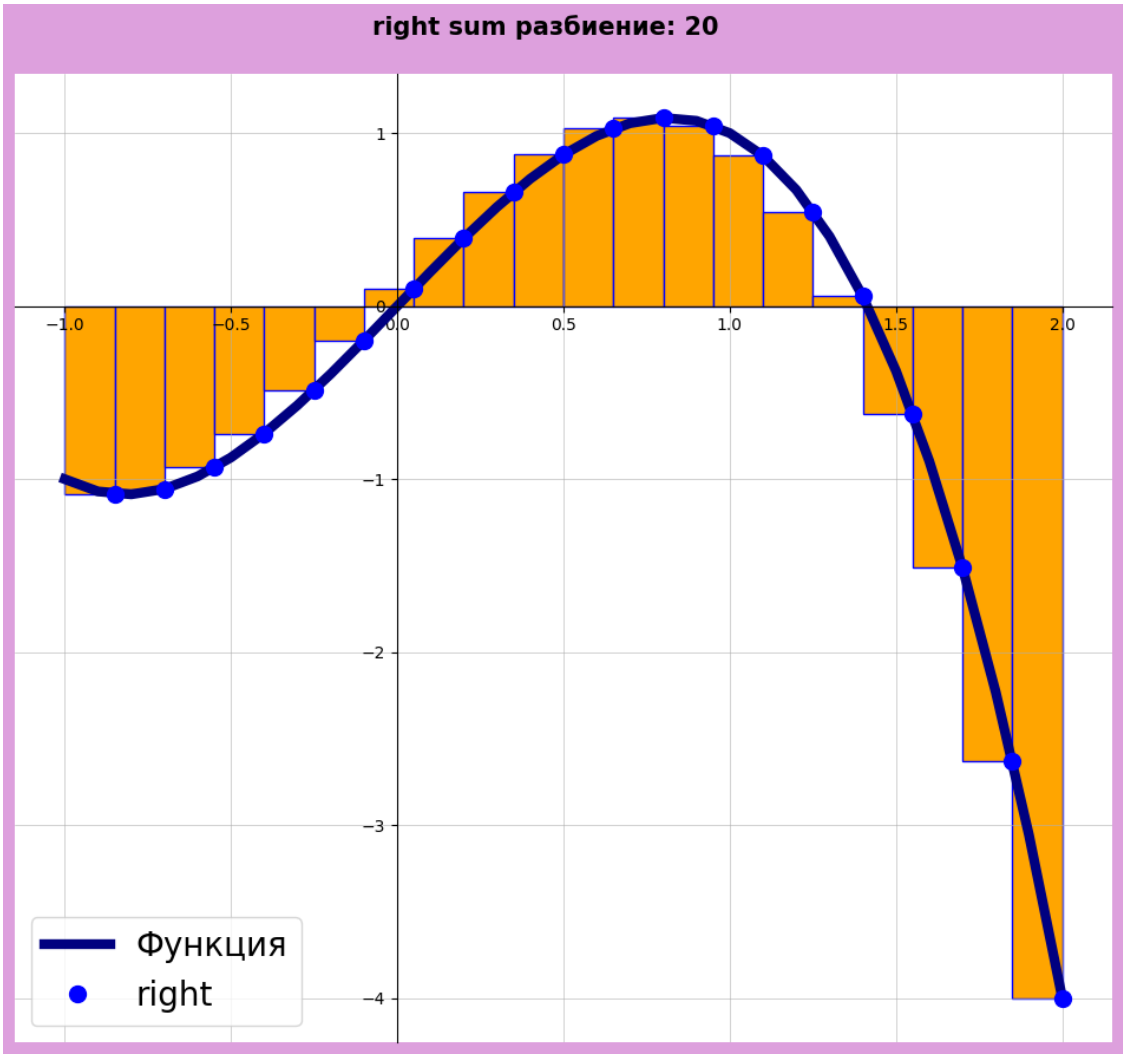
```
    ax.spines['bottom'].set_position('zero')
    ax.spines['right'].set_color('none')
    ax.spines['top'].set_color('none')
    ax.xaxis.set_ticks_position('bottom')
    ax.yaxis.set_ticks_position('left')

    # Выведем Легенду
    plt.legend(loc=3, prop={'size': 20})

    # Вывод полученного графика
    plt.show()

# Метод по рисованию суммы
def draw_sum(a, b, f, sum_x, square_x, sum_y, delta_x, ind):
    x = np.arange(a, b + 0.1, 0.1) # диапозон по OX
    f_x = f(x) # Диапозон по OY

    fig = plt.figure(figsize=(12,10))   # Размер графика
    # Цвет окантовки
    fig.patch.set_facecolor('#DDA0DD')

    # Заголовок
    ax = fig.add_subplot()
    fig.subplots_adjust(top=0.93)
    fig.suptitle(f'{name[ind]} sum разбиение: {len(square_x)}',
fontsize=15, fontweight='bold')

    # Размер координат осей абсцисс и ординат
    plt.xticks(fontsize = 12)
    plt.yticks(fontsize = 12)

    # Разметка на графике
    plt.grid(axis = 'both', linewidth = 0.4)

    # Выводим график функции
    plt.plot(x, f_x, color='#000080', linewidth=6, label='Функция')
    # Выводим точки суммы
    plt.plot(sum_x, sum_y, dots[ind], markersize=10, label=name[ind])

    # Секторы (прямоугольники)
    for i in range(len(sum_x)):
        ax.add_patch(
        patches.Rectangle(
            (square_x[i], 0),
            delta_x,
            sum_y[i],
            edgecolor = 'blue',
            facecolor = colors[ind],
            fill=True
         ) )
```

```
    # Оси в центр
    ax.spines['left'].set_position('zero')
    ax.spines['bottom'].set_position('zero')
    ax.spines['right'].set_color('none')
    ax.spines['top'].set_color('none')
    ax.xaxis.set_ticks_position('bottom')
    ax.yaxis.set_ticks_position('left')

    # Выведем Легенду
    plt.legend(loc=3, prop={'size': 20})

    # Вывод полученного графика
    plt.show()

# Выводим заданную произовльную функцию с заданными параметрами
darbu_sums_methods(function, start, end, number, kind)
```

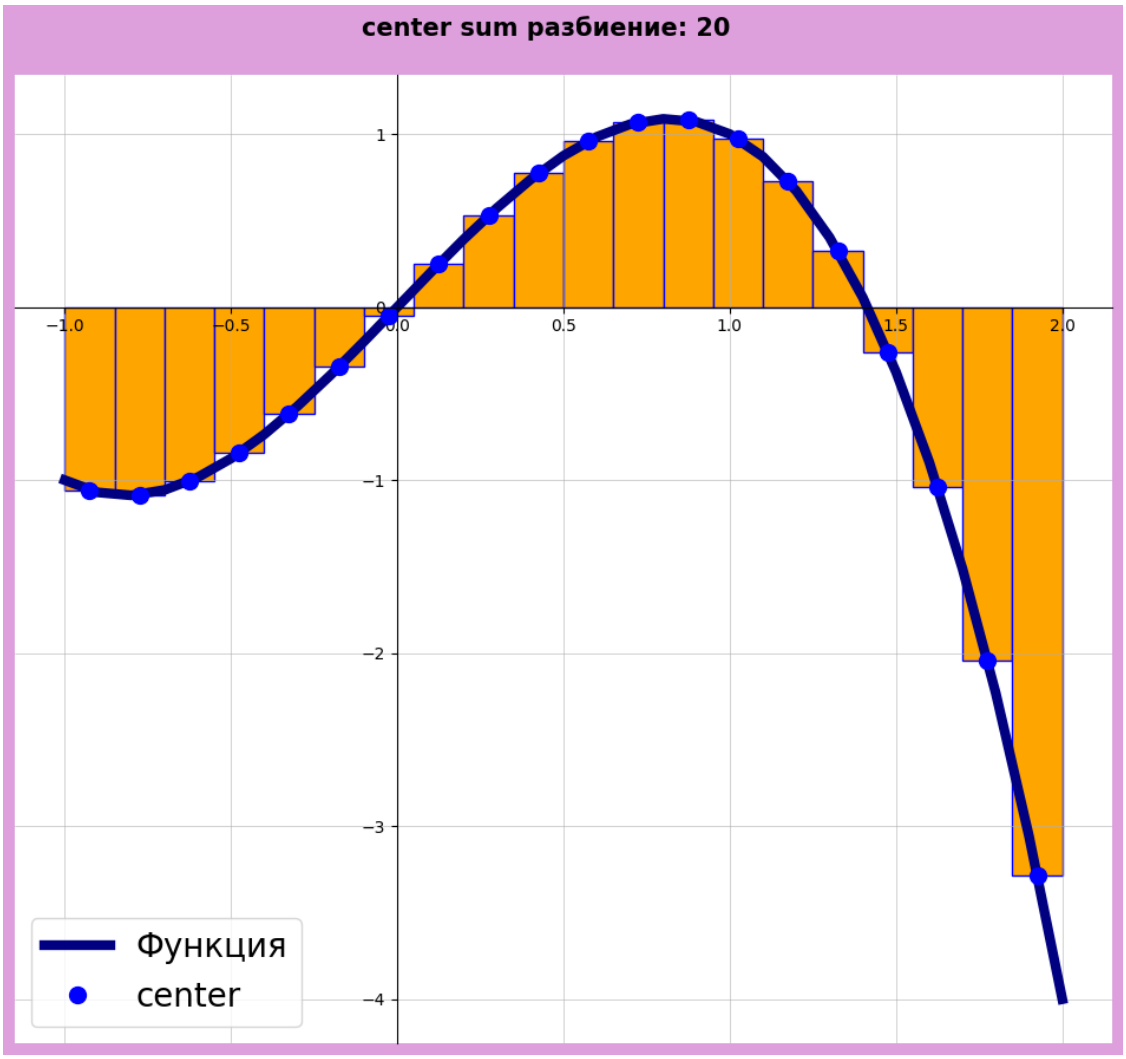





```
left -0.5418749999999996

-0.5418749999999996

# Так как по умолчанию в коде left, Вывожу и остальные оснащения с заданными параметрами
# Правое оснащение
kind = 'right'
darbu_sums_methods(function, start, end, number, kind, False)
```



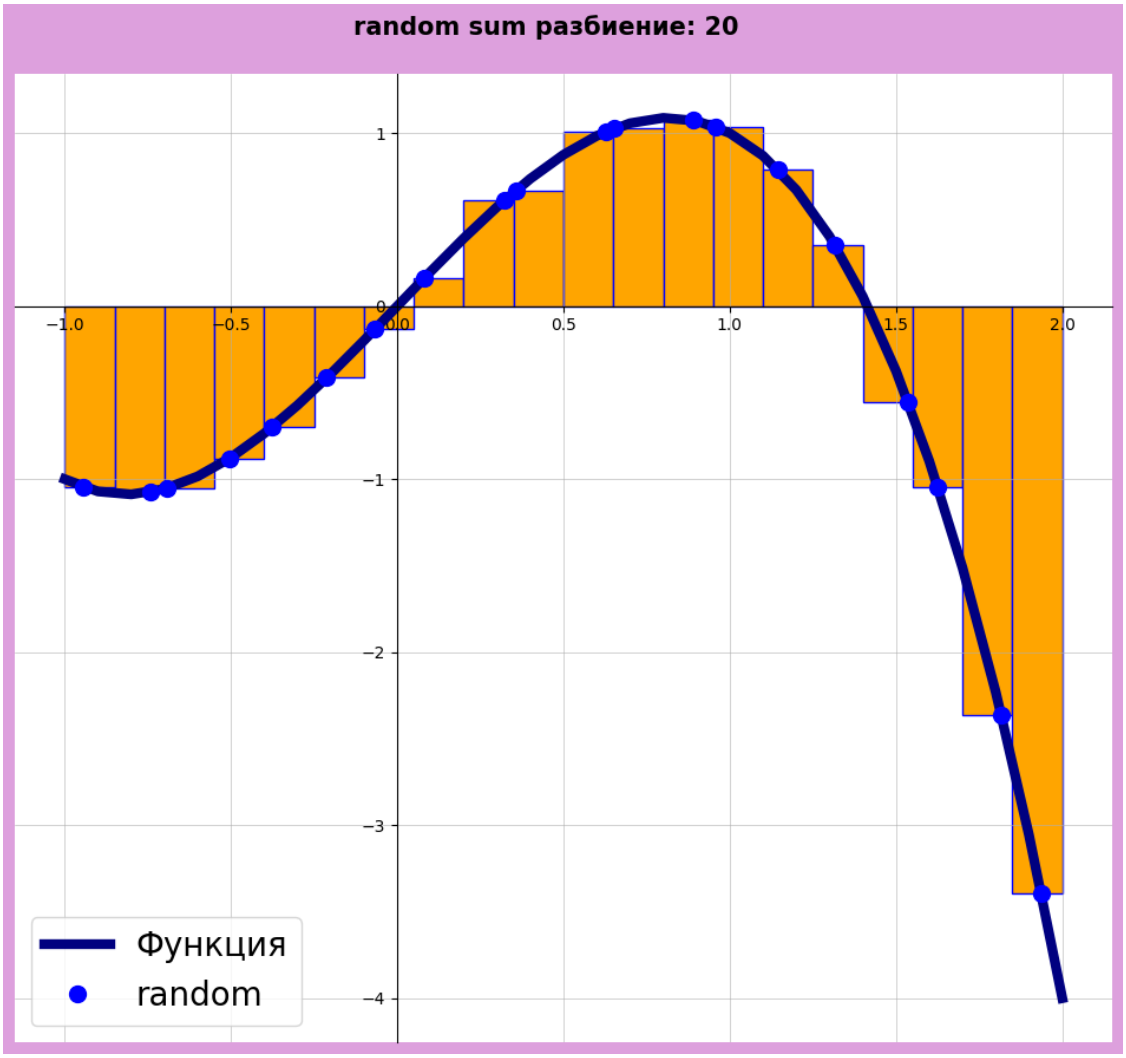

```
right -0.9918749999999995

-0.9918749999999995

# Центральное оснащение
kind = 'center'
darbu_sums_methods(function, start, end, number, kind, False)
```




```
center -0.7415624999999996
```

```
-0.7415624999999996
```

```
# Рандомное оснащение
kind = 'random'
darbu_sums_methods(function, start, end, number, kind, False)
```




```
random -0.8897955238718195

-0.8897955238718195
```

3) Сравнение результатов своей функции

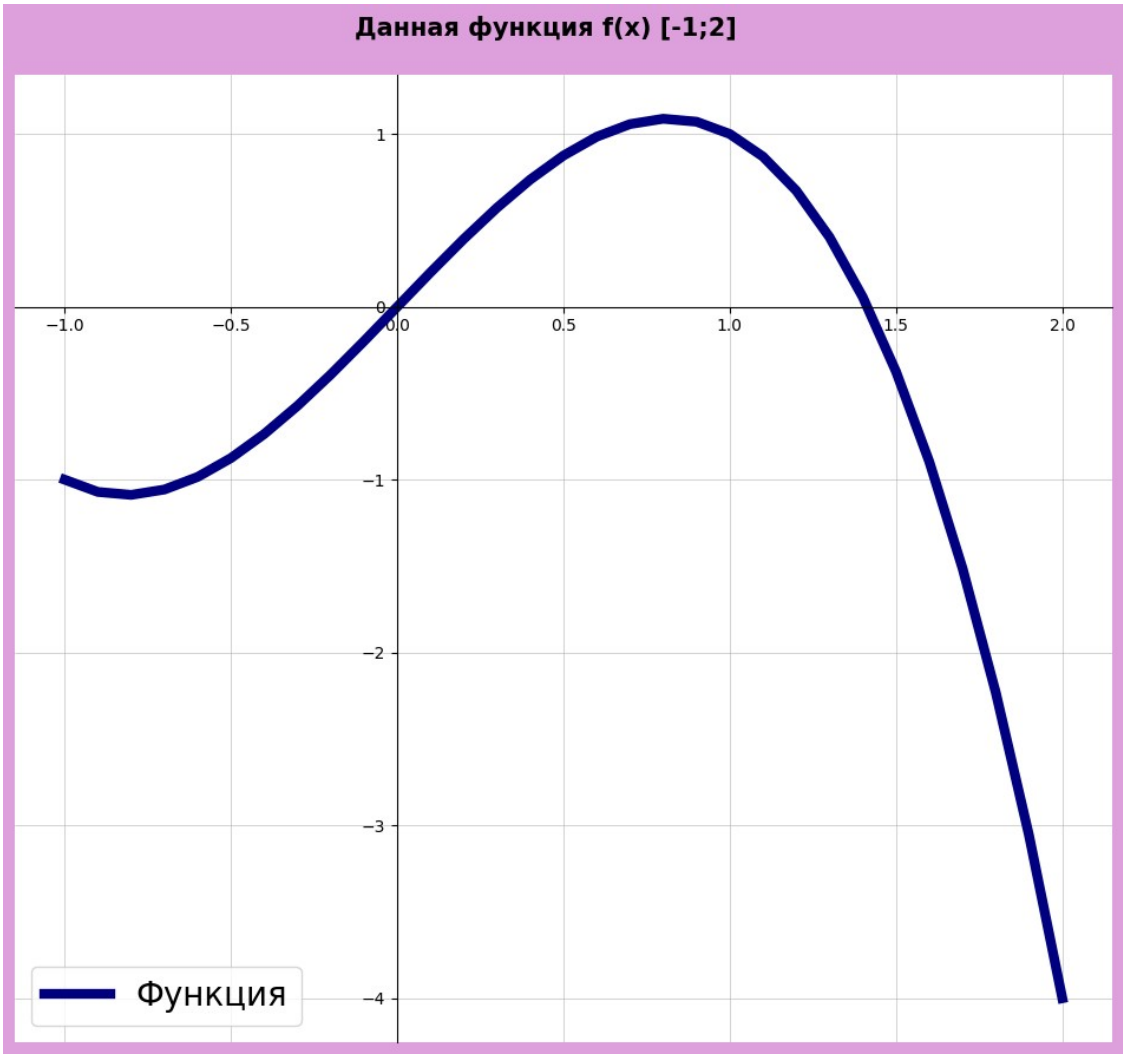
```python
# Цикл с шагом float
def frange(a, b, jump):
    while a <= b:
        yield a
        a += jump

# Моя функция
def my_function(x):
    return 2 * x - x ** 3

# Вычисление верхней и нижней сумм Дарбу,
# чтобы проверить правильность выбора точек в аналитическом,
# по всему подотрезку выбираю реальный локальный inf/sup
def darbu_sums_correct(f, a, b, n, draw_funct=True, draw=True):
```

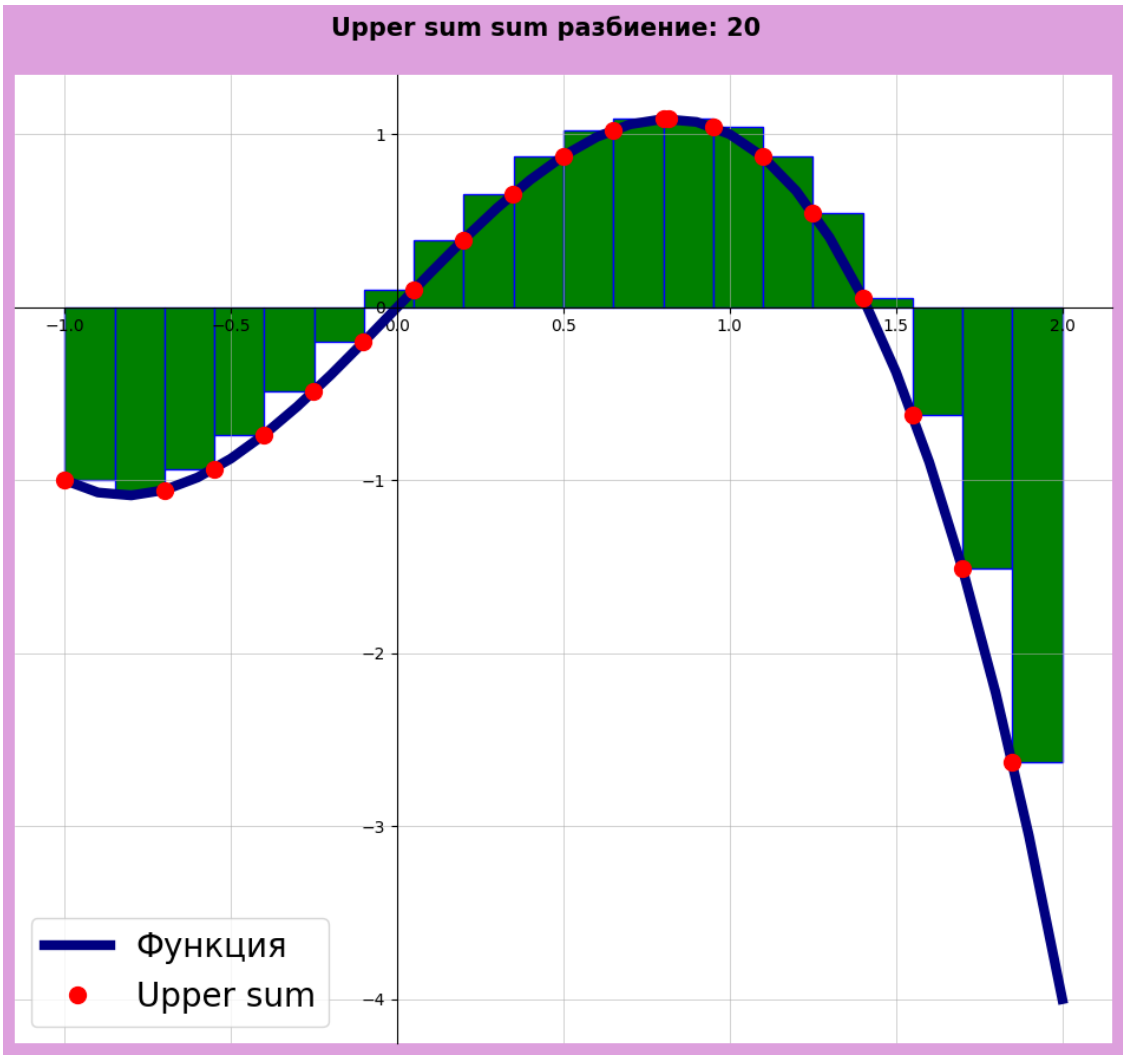
```
    # Дельта, ширита подотрезка
    delta_x = (b - a) / n
    # все крайние точки одотрезков
    x_values = [a + i * delta_x for i in range(n + 1)]
    # Инициализация сумм
    up_sum = 0
    low_sum = 0
    # Точки для секторов и сум
    square_x = [0] * n
    sup_values_x = [0] * n
    inf_values_x = [0] * n
    sup_values_y = [0] * n
    inf_values_y = [0] * n

    # Циклом бежим по всем точкам
    for i in range(n):
        # берем две точки, это и следующую
        x_0 = x_values[i]
        x_1 = x_values[i + 1]
        # Диапозон точек [x0;x1] с шагом длина подотрезкаа / 100
        arr_x = [x for x in frange(x_0, x_1, delta_x / 100)]
        # По этим точкам высчитываем точки на OY
        arr_y = [f(x) for x in arr_x]
        # Максимум из точек
        sup = max(arr_y)
        # Минимум из точек
        inf = min(arr_y)
        # Прибавляем к суммам
        up_sum += sup
        low_sum += inf

        # Берем точку для прямоугольника
        square_x[i] = x_0
        # Берем точки для рисунка сумм
        sup_values_x[i] = arr_x[arr_y.index(sup)]
        sup_values_y[i] = sup
        inf_values_x[i] = arr_x[arr_y.index(inf)]
        inf_values_y[i] = inf
    # Домножаем на дельту, получаем сумму площадей каждого
прямоугольника
    up_sum *= delta_x
    low_sum *= delta_x
    # Если надо то выводим функцию
    if (draw_funct):
        draw_func(a, b, f)
    if (draw):
        # Выводим полученные суммы
        draw_sum(a, b, f, sup_values_x, square_x, sup_values_y,
delta_x, 0)
        draw_sum(a, b, f, inf_values_x, square_x, inf_values_y,
```

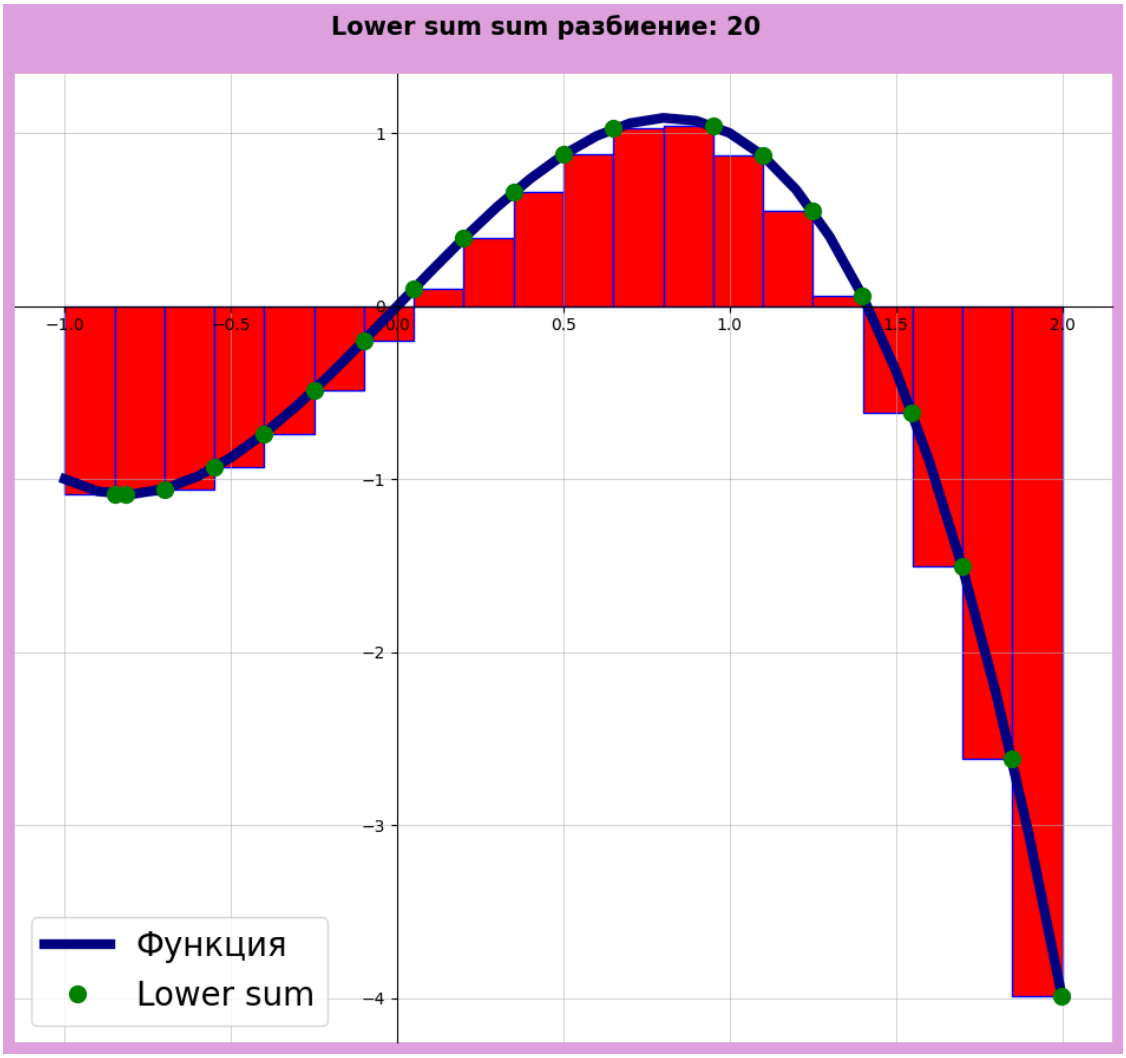
```
delta_x, 1)
        print('Upper sum: ', up_sum)
        print('Lower sum: ', low_sum)
    return (up_sum, low_sum)
```

Тут мы видим, что в аналитечской части правильно разбили на промежутки и правильно выбрали точки, они и правда являются крайними и слева, там где было слева, справа, где было справа

```
# Вывожу свою функцию с правильными точками верхней и нижней сумм
darbu_sums_correct(my_function, -1, 2, number)
```



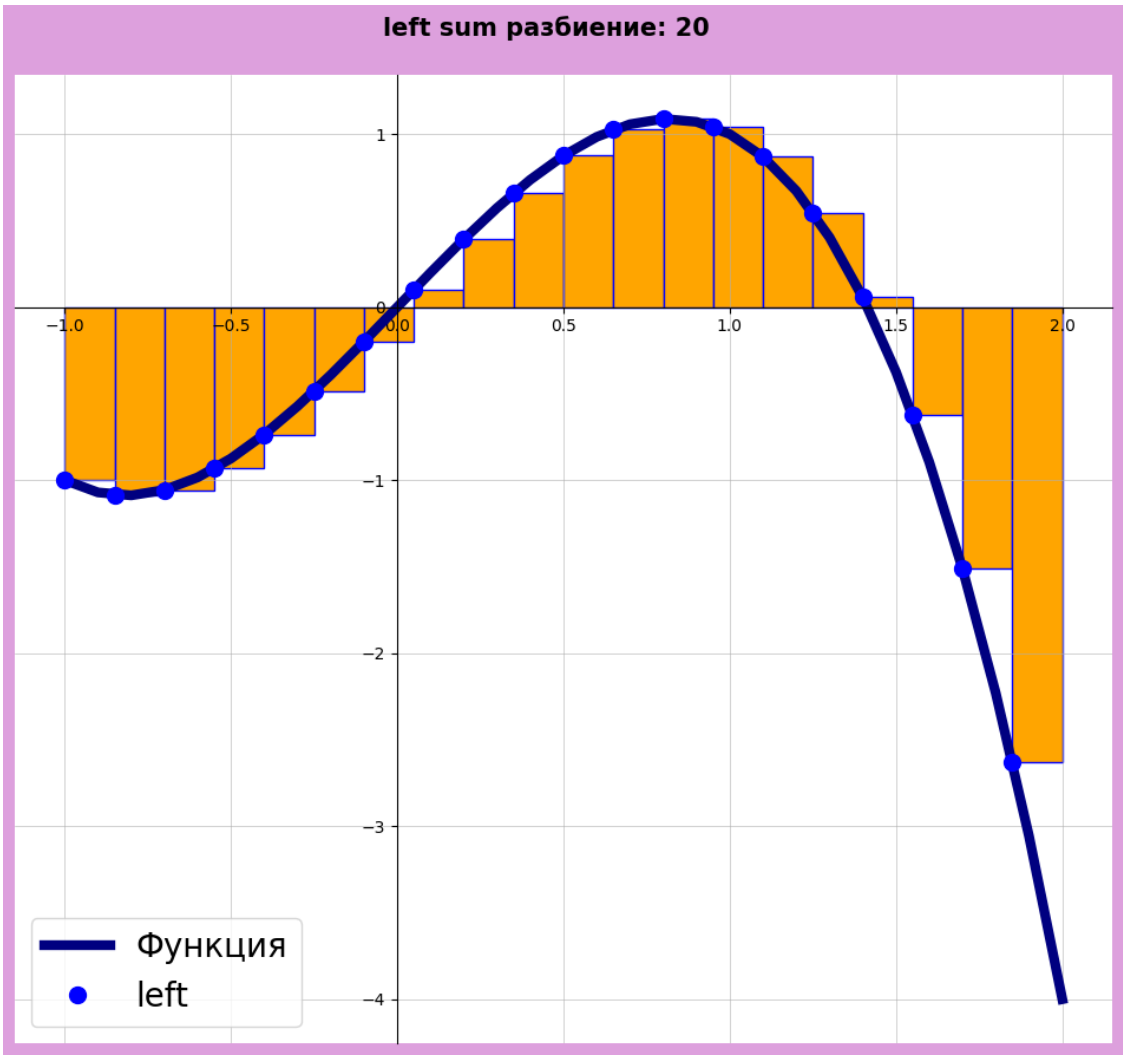

Upper sum sum разбиение: 20
1
0
−1
−2
−3
−4
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
Функция
Upper sum



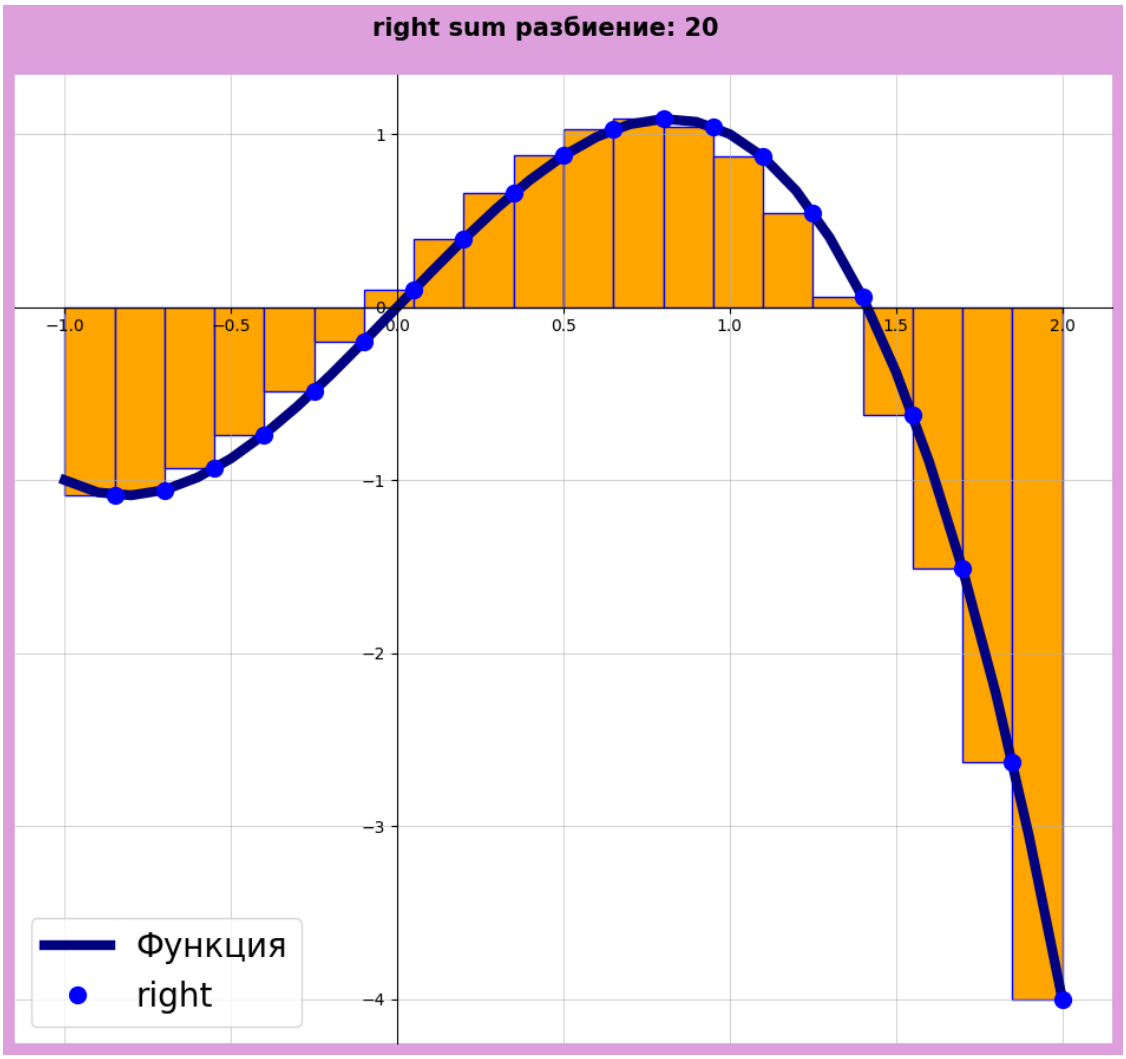

```
Upper sum:  -0.2173360092937519
Lower sum:  -1.3097581426312803

(-0.2173360092937519, -1.3097581426312803)
```

```python
# Сравнение результатов, вывод таблицы
def campare_values():
    # Создаю таблицу
    t = Texttable()
    # Добавляю строку заголовков
    # Буду выводить:
    # N : Число разбиений
    # LEFT : Вывод при левом оснащении
    # RIGHT : Вывод при правом оснащении
    # CENTER : Вывод при центральном оснащении
    # RANDOM : ВЫвод при рандомном оснащении
    # CORRECT LOW : Вывод "правильной" нижней суммы
    # CORRECT UP : Вывод "правильной" верхней суммы
    # ANALITIC RES -3/4 : Вывод цели, к чему стремятся все верхние
```

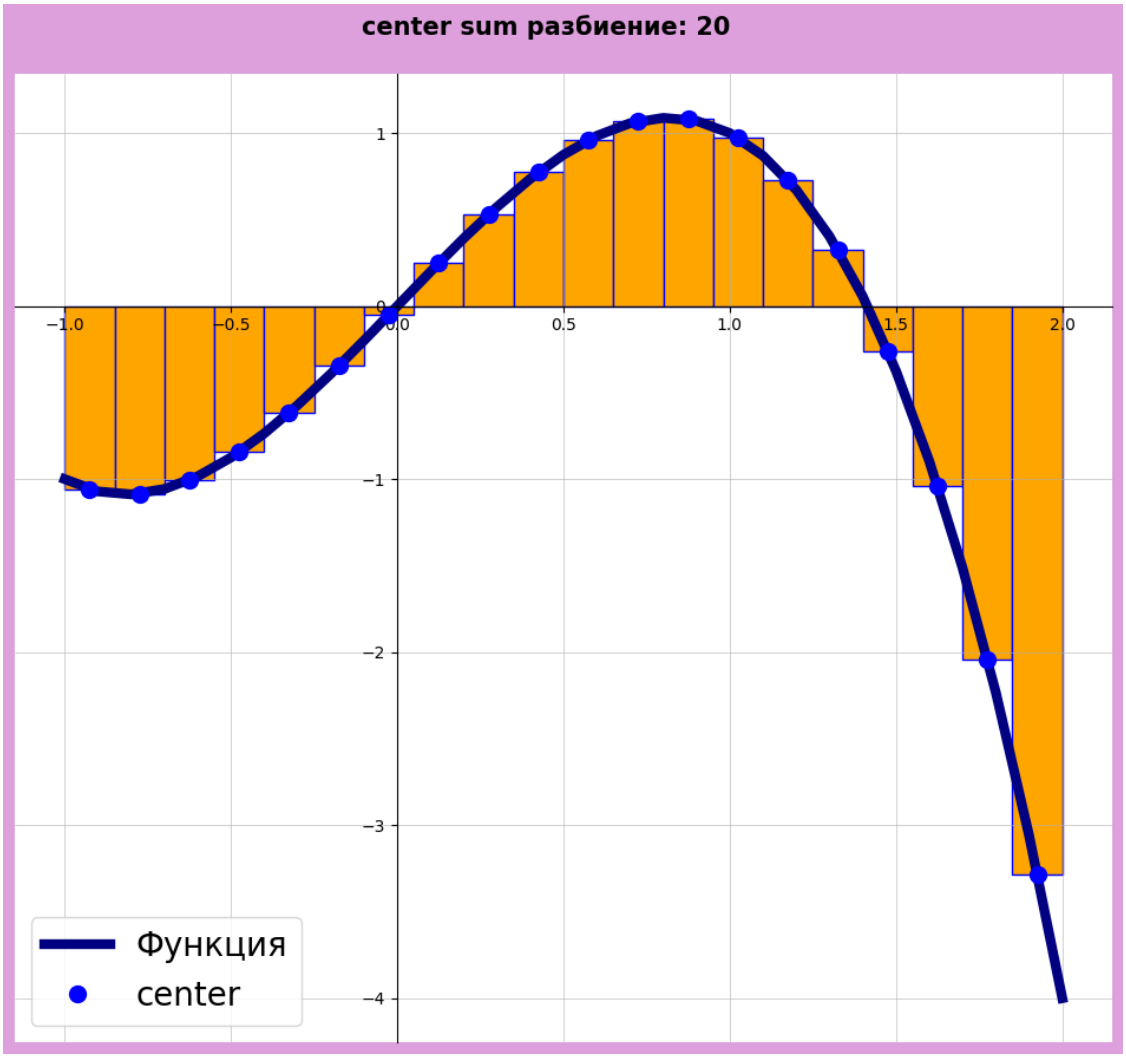
```
значения
    t.add_row(['N', 'LEFT', 'RIGHT', 'CENTER', 'RANDOM', 'COR\nLOW',
'COR\nUP', 'ANAL\nRES\n-3/4'])

    # Вывожу результаты с разбиением 1-752 с шагом 250
    for i in range(20, 171, 25):

        kind = 'left'
        left = darbu_sums_methods(function, start, end, i, kind,
False)

        kind = 'right'
        right = darbu_sums_methods(function, start, end, i, kind,
False)

        kind = 'center'
        centre = darbu_sums_methods(function, start, end, i, kind,
False)

        kind = 'random'
        random = darbu_sums_methods(function, start, end, i, kind,
False)

        correct = darbu_sums_correct(function, -1, 2, i, False)

        # Добавляем строку
        t.add_row([i, left, right, centre, random, correct[1],
correct[0], -3/4])
    # Вывод таблицы
    return t.draw()

table = campare_values() # Вызов метода сравнения
```



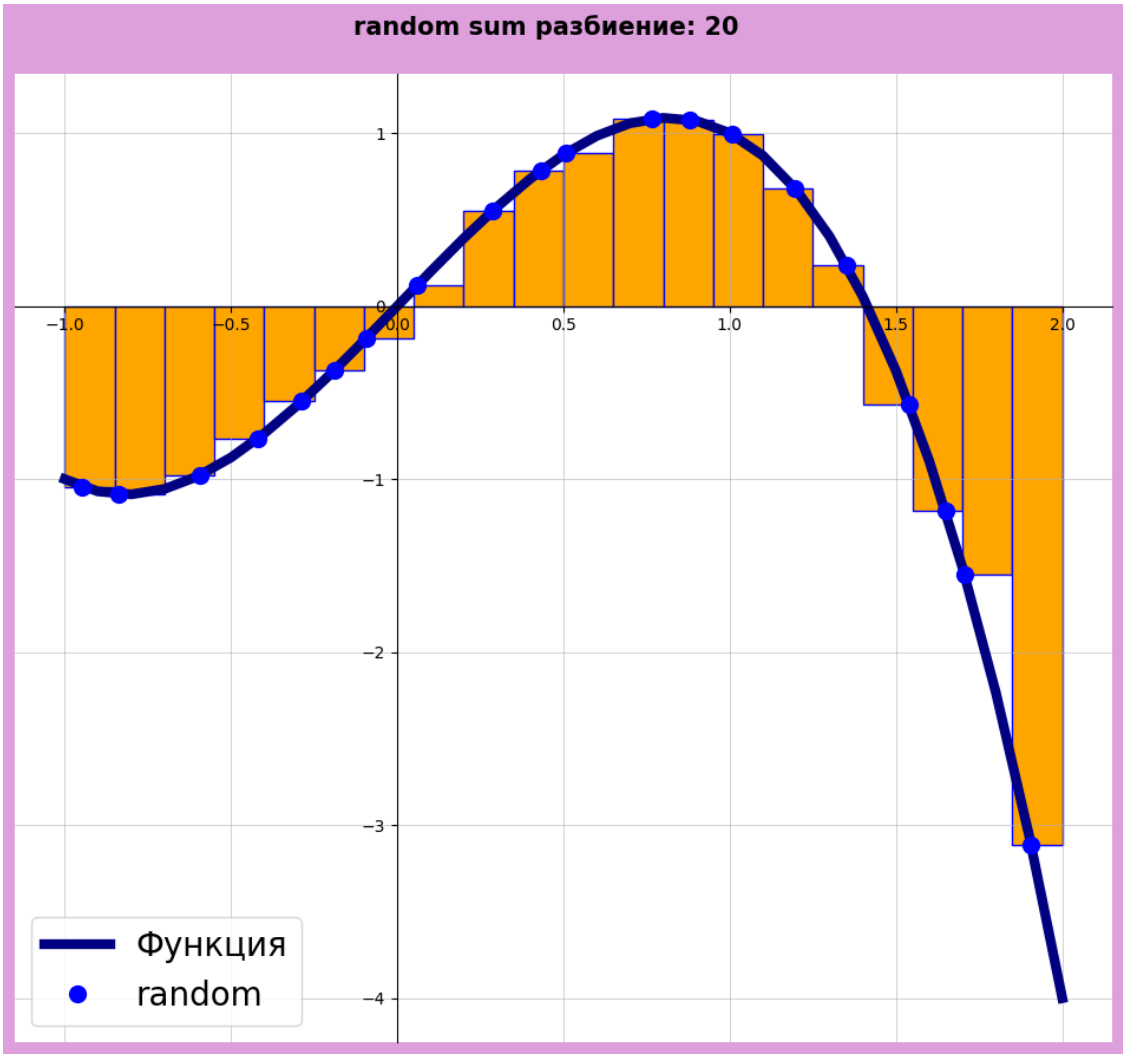
left -0.5418749999999996



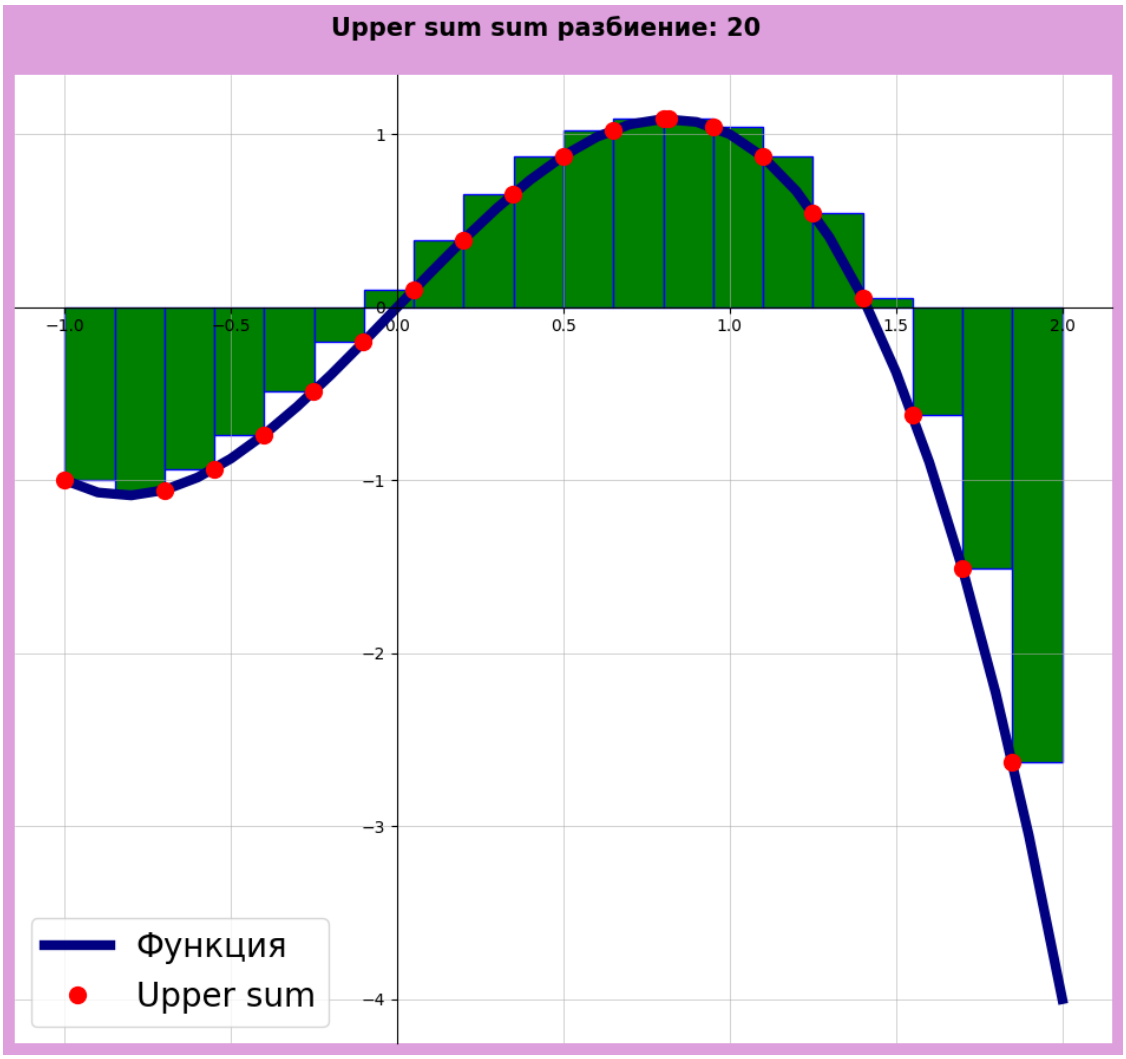
right -0.9918749999999995

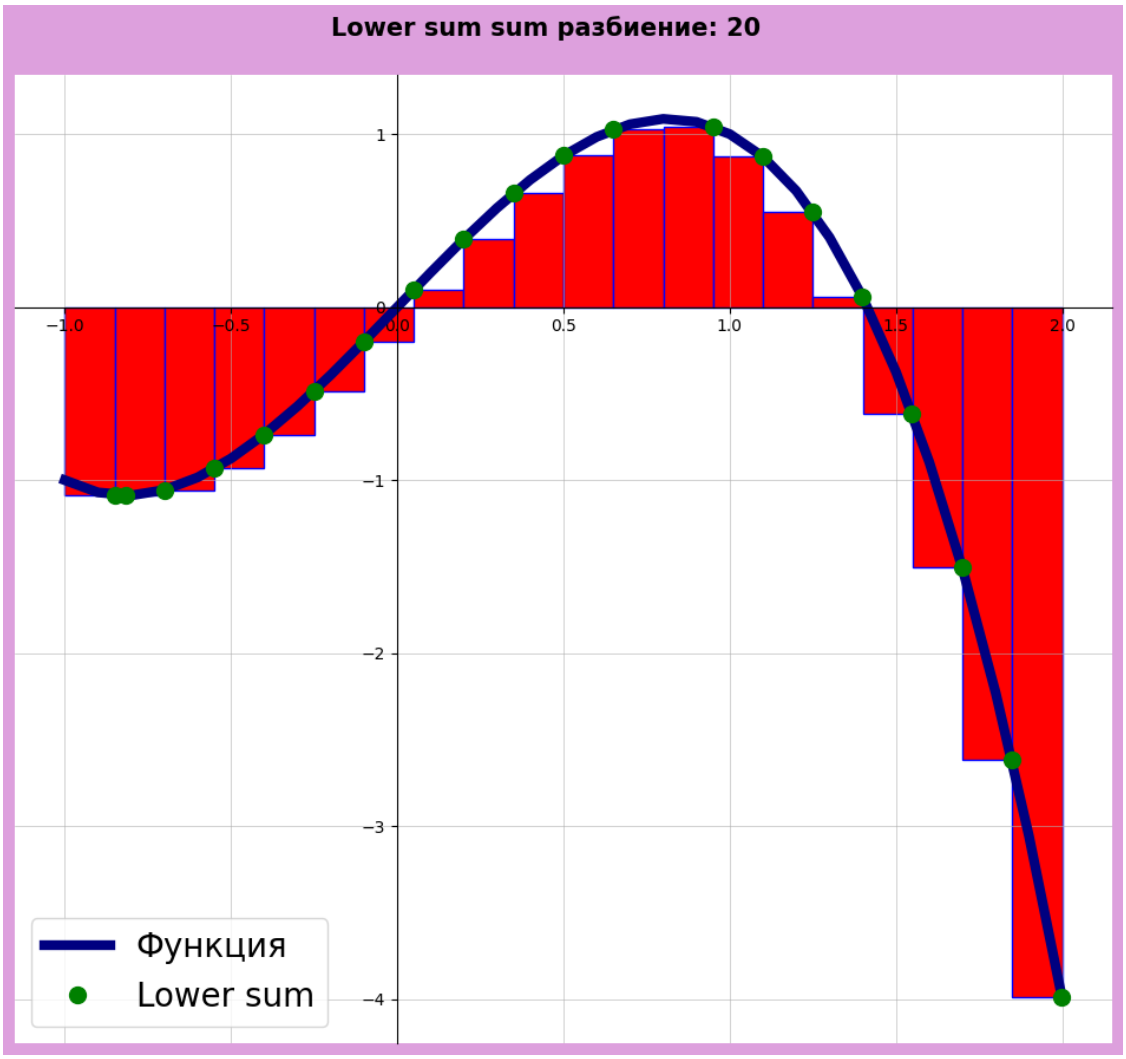


center -0.7415624999999996

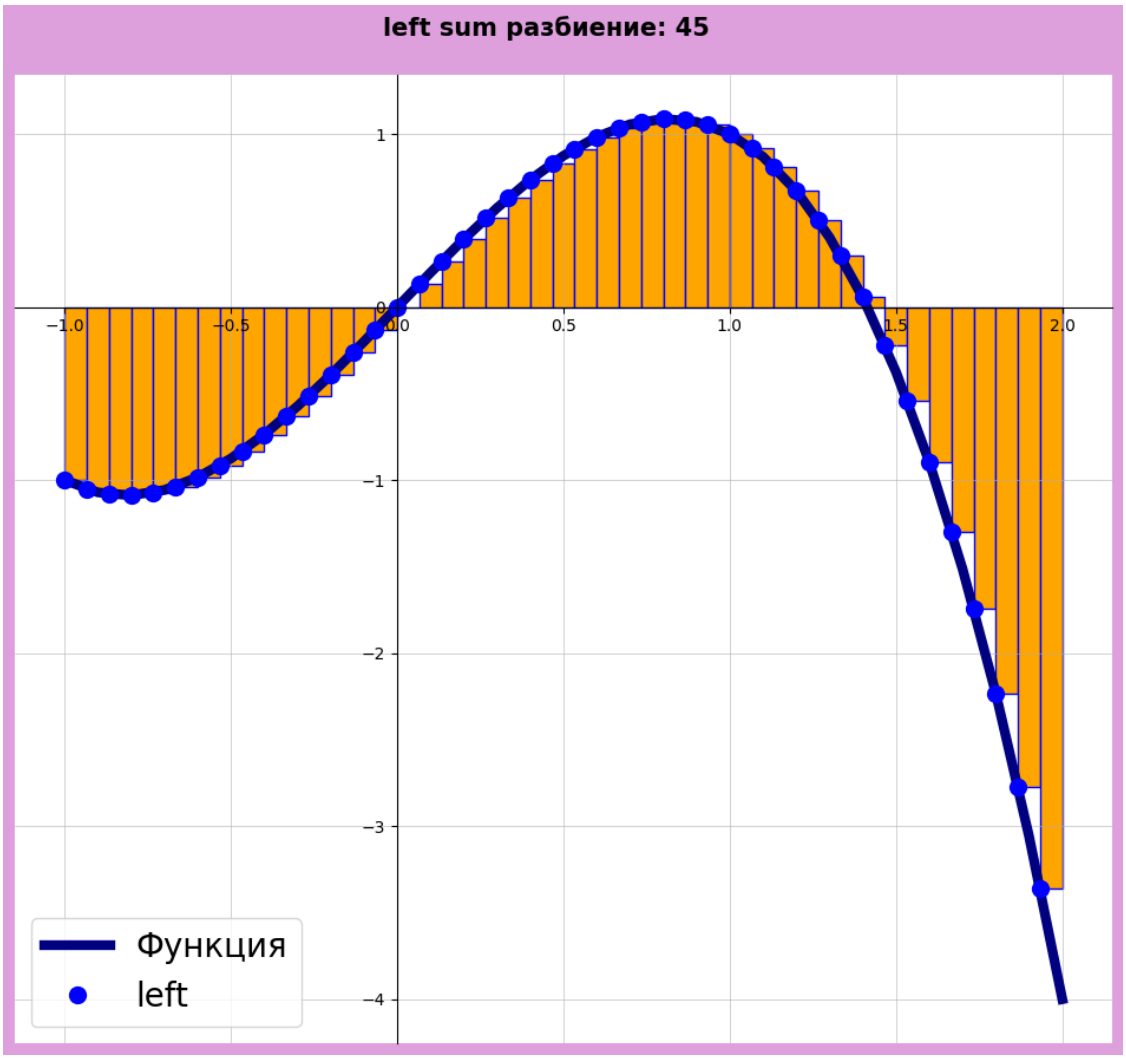


random -0.7473545921627011

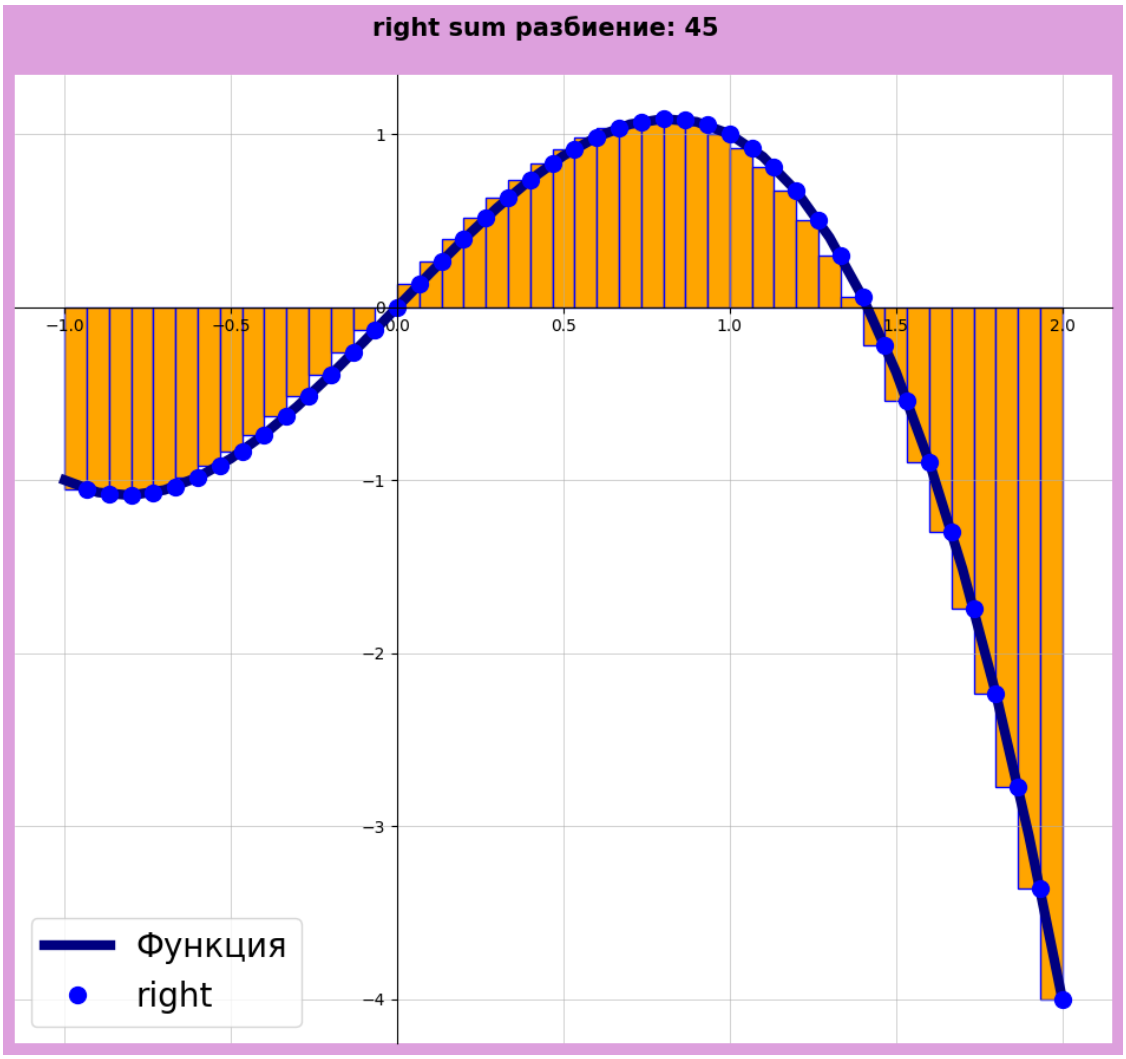

Upper sum sum разбиение: 20
1
0
−1
−2
−3
−4
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
Функция
Upper sum



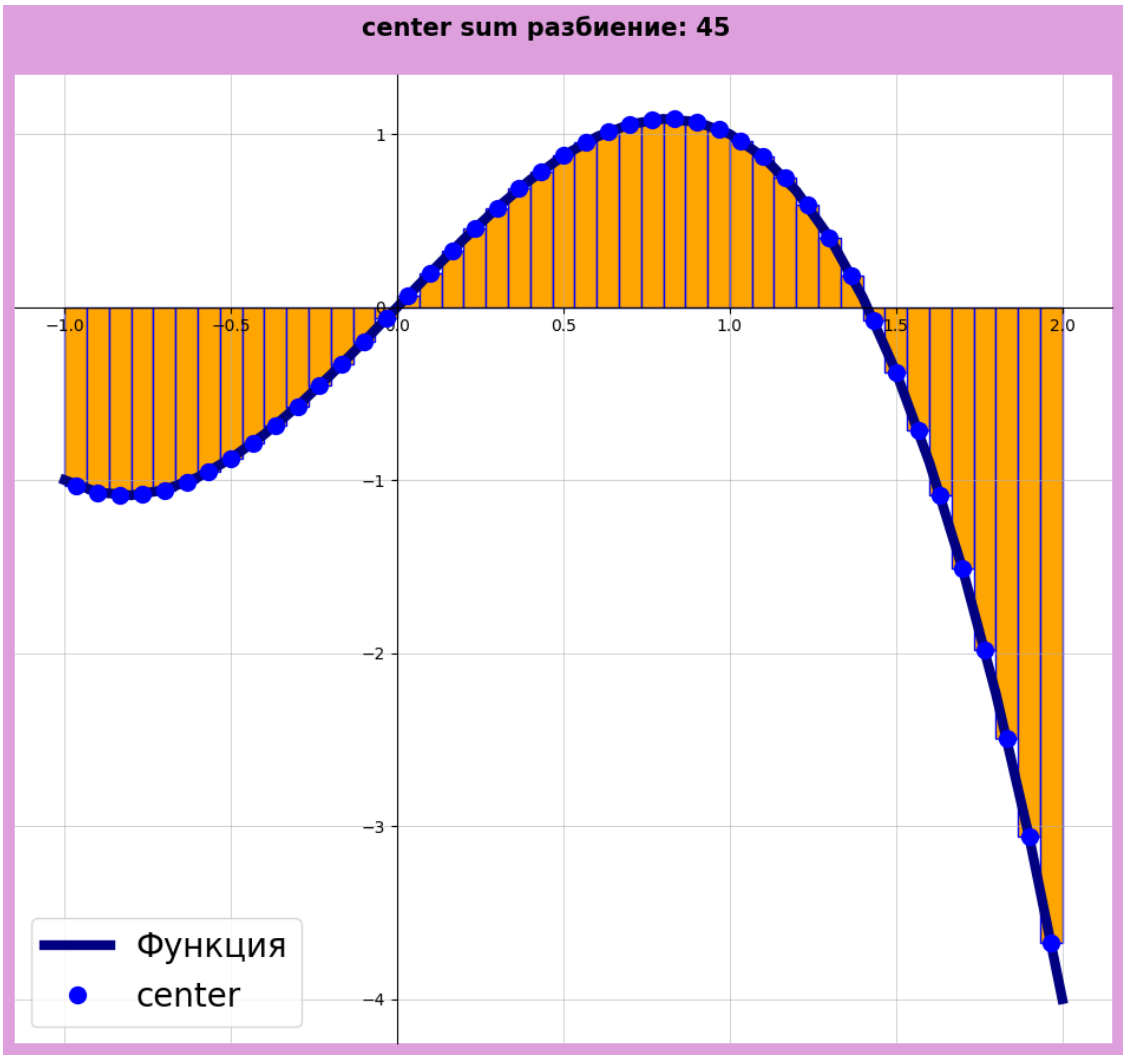

```
Upper sum:  -0.2173360092937519
Lower sum:  -1.3097581426312803
```



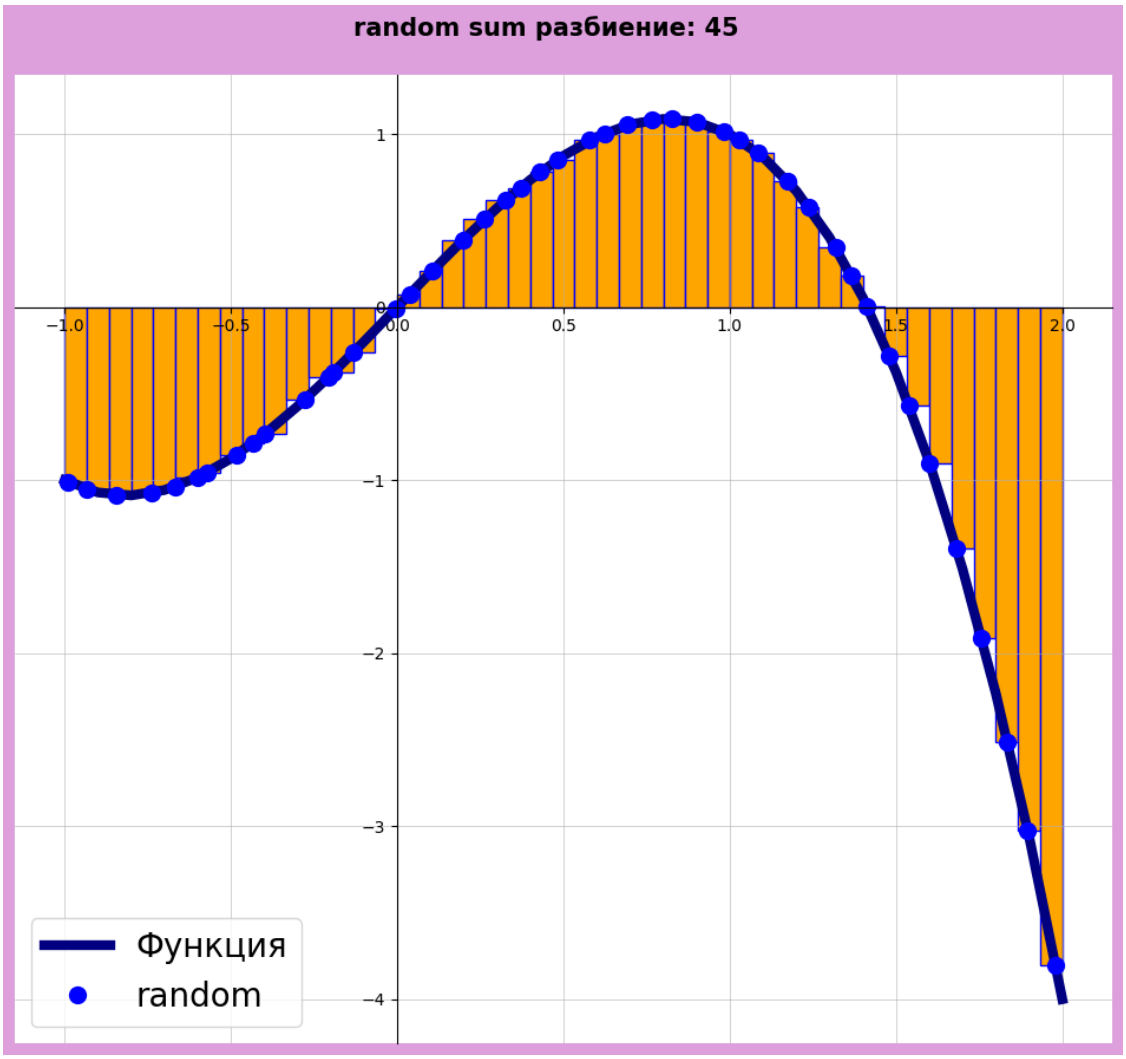
left -0.6533333333333331



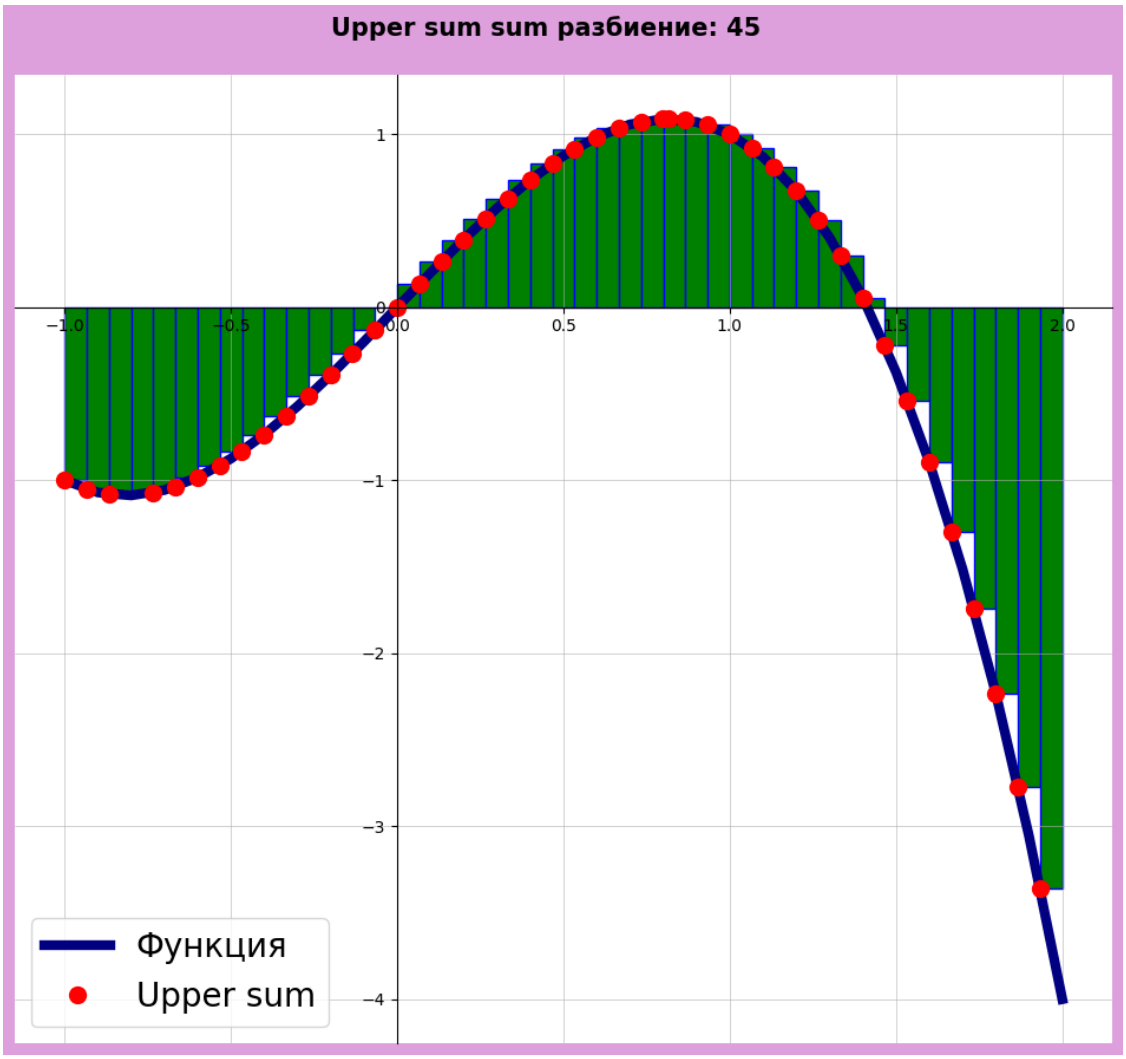
right -0.8533333333333332

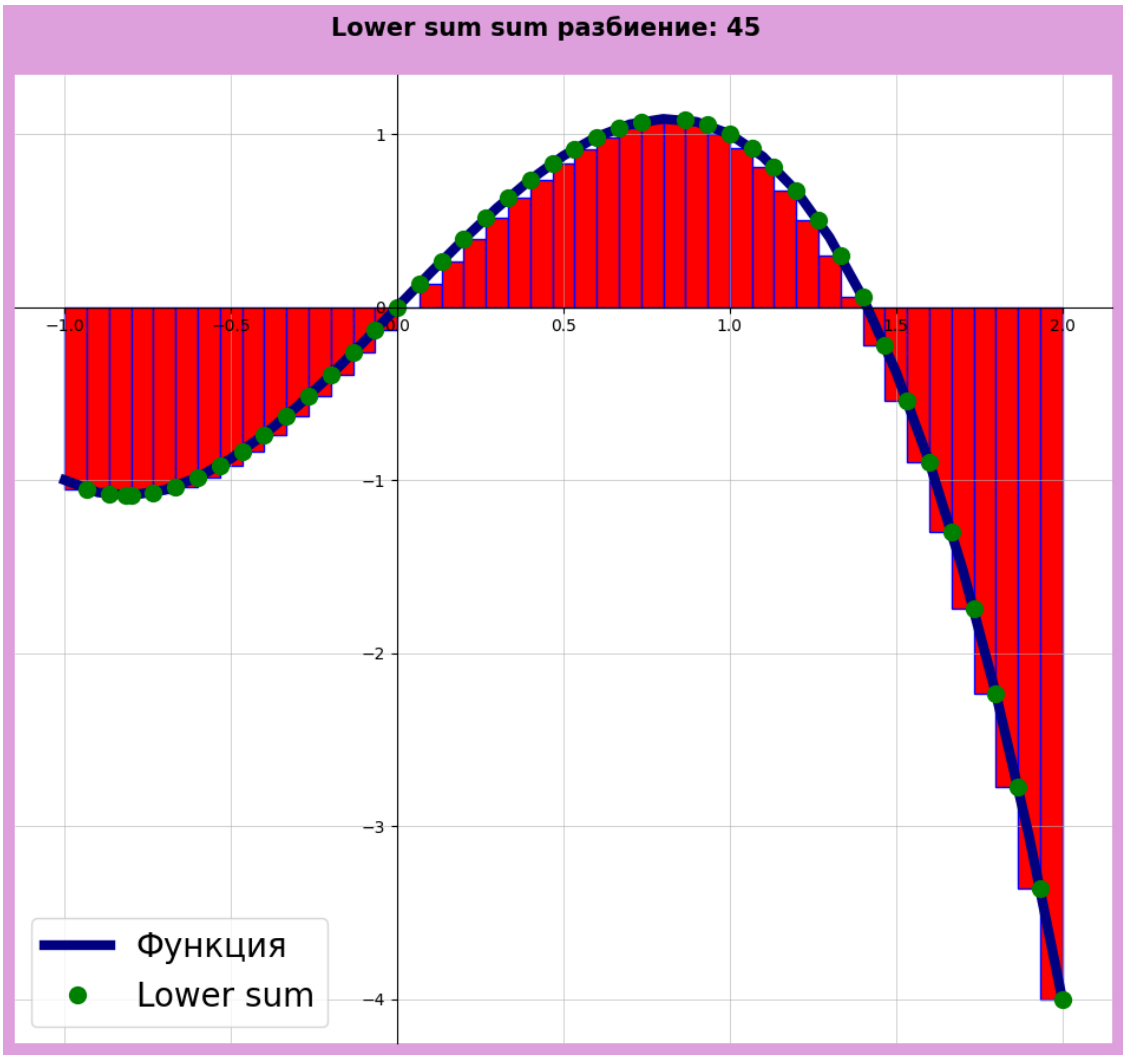


center -0.7483333333333336

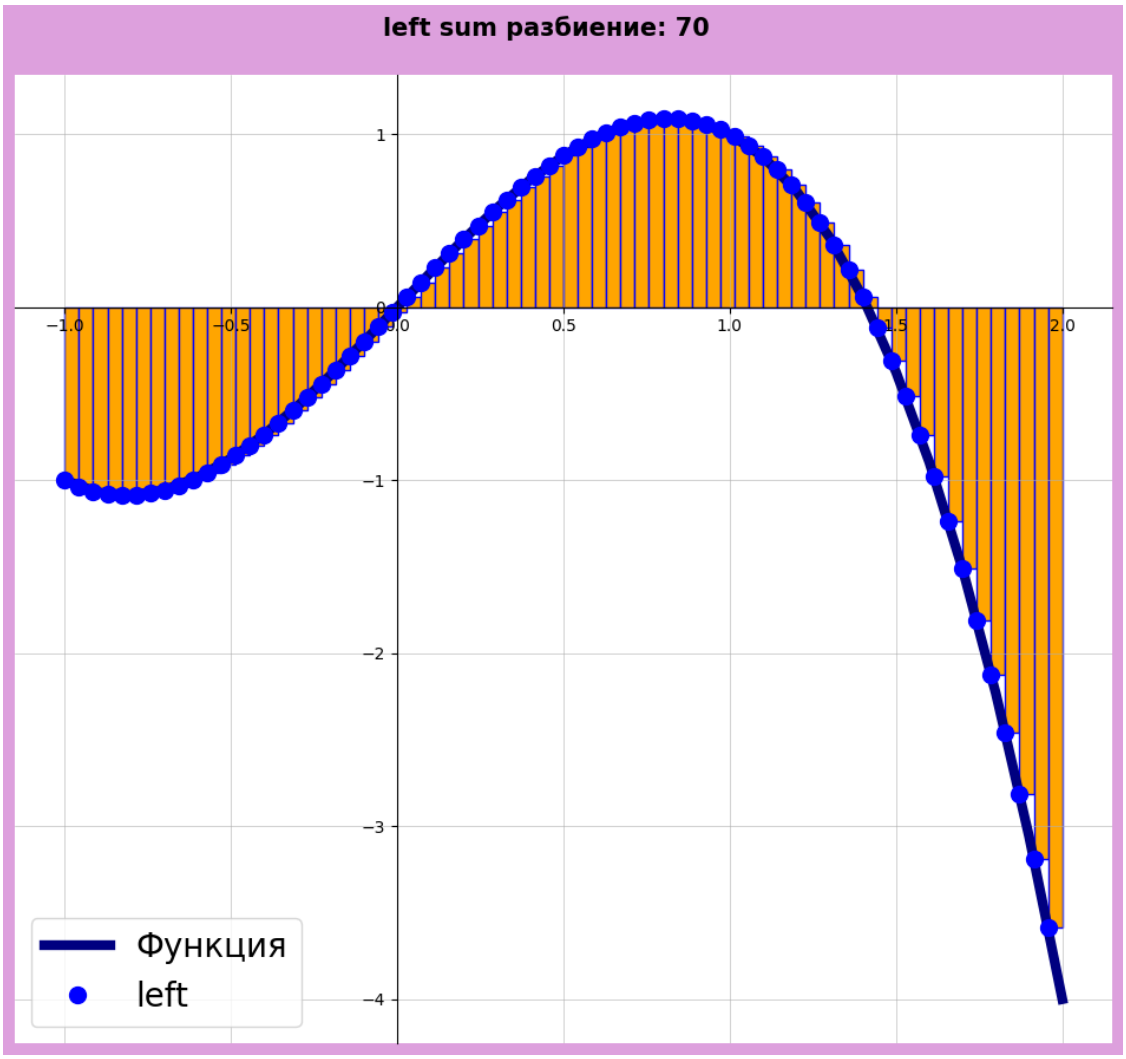


random -0.6967912126726417

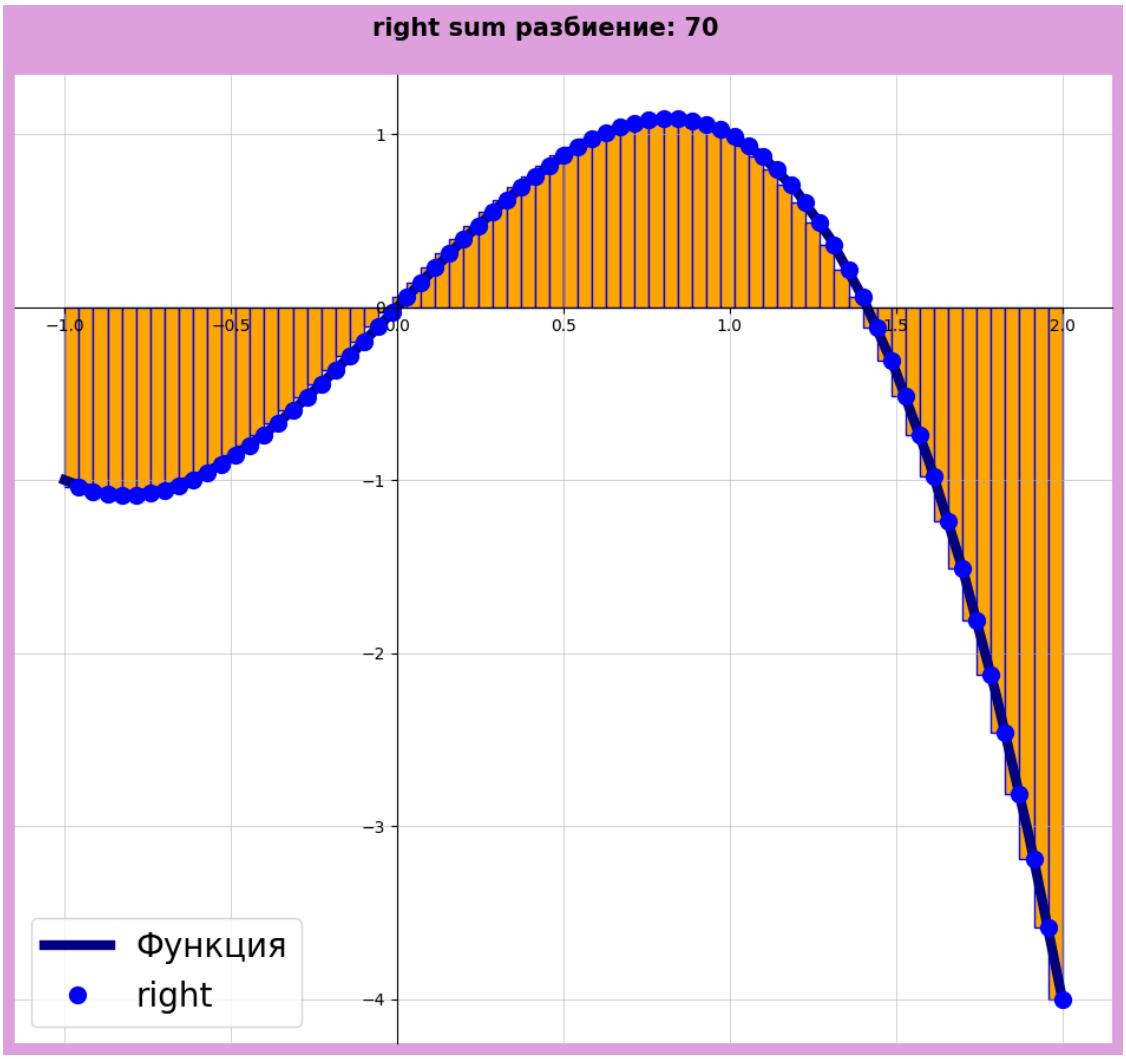

Upper sum sum разбиение: 45
1
0
−1
−2
−3
−4
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
Функция
Upper sum



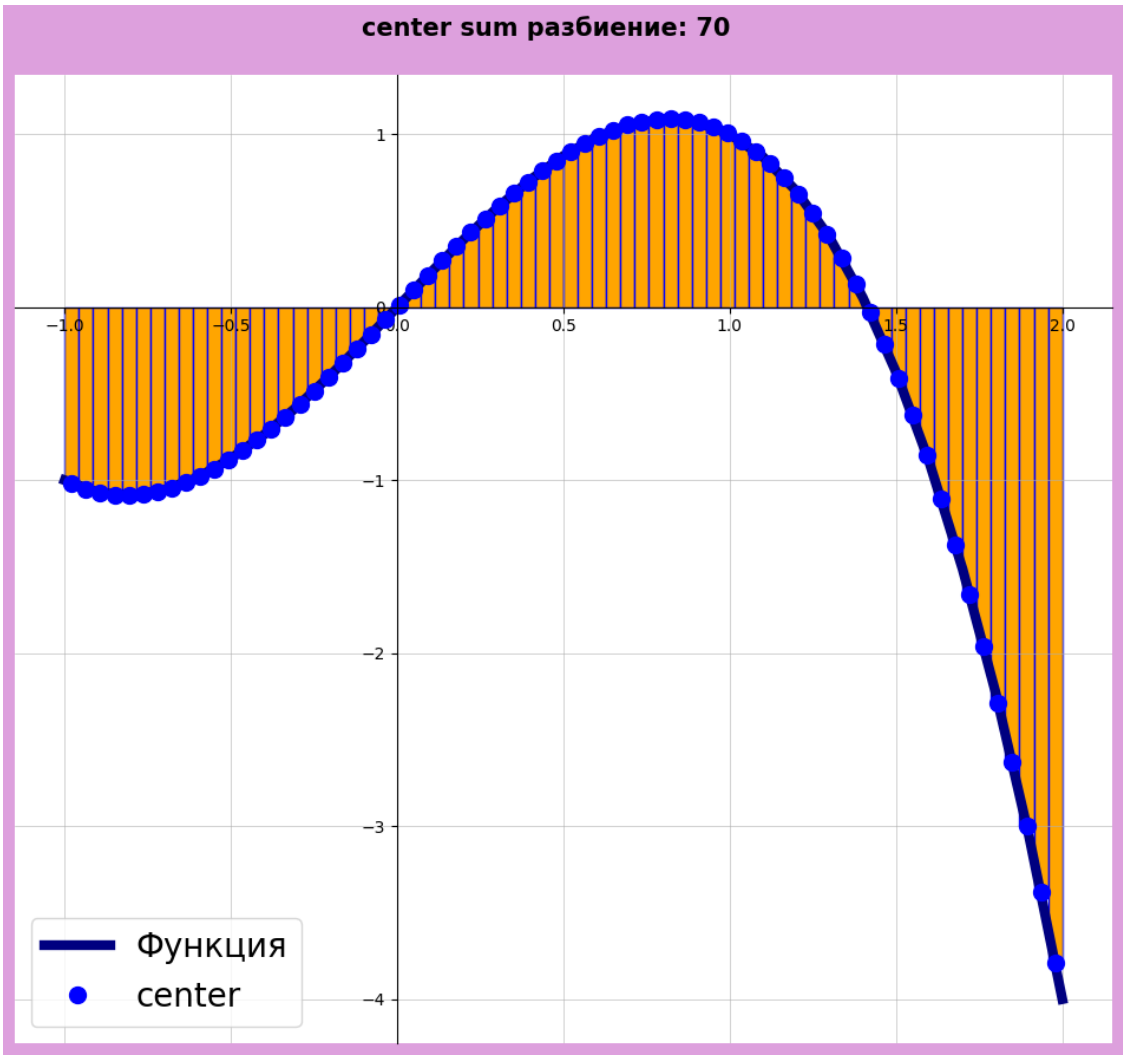

```
Upper sum:  -0.5090735409185181
Lower sum:  -0.9983227431110727
```



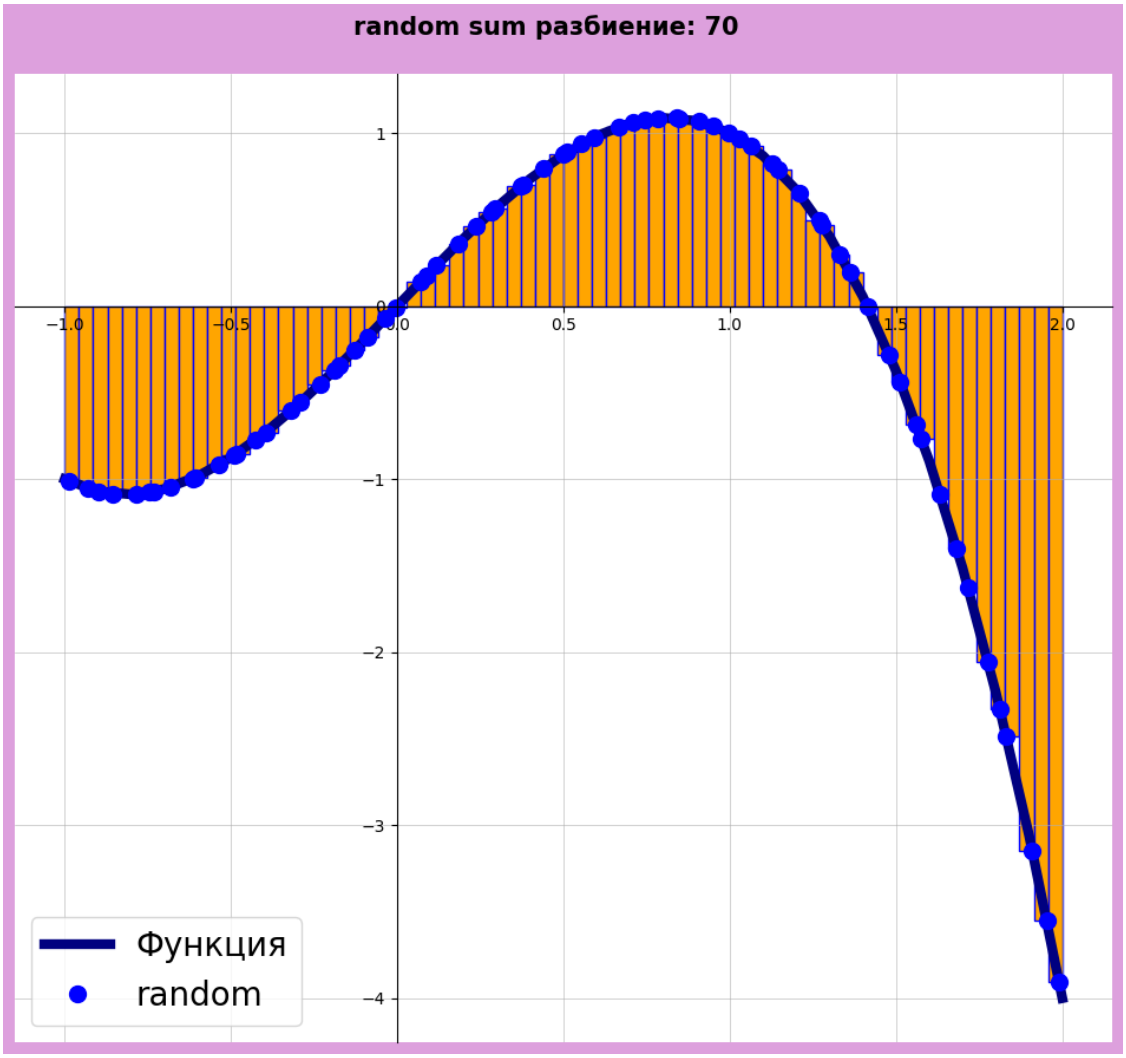
left -0.6870918367346942



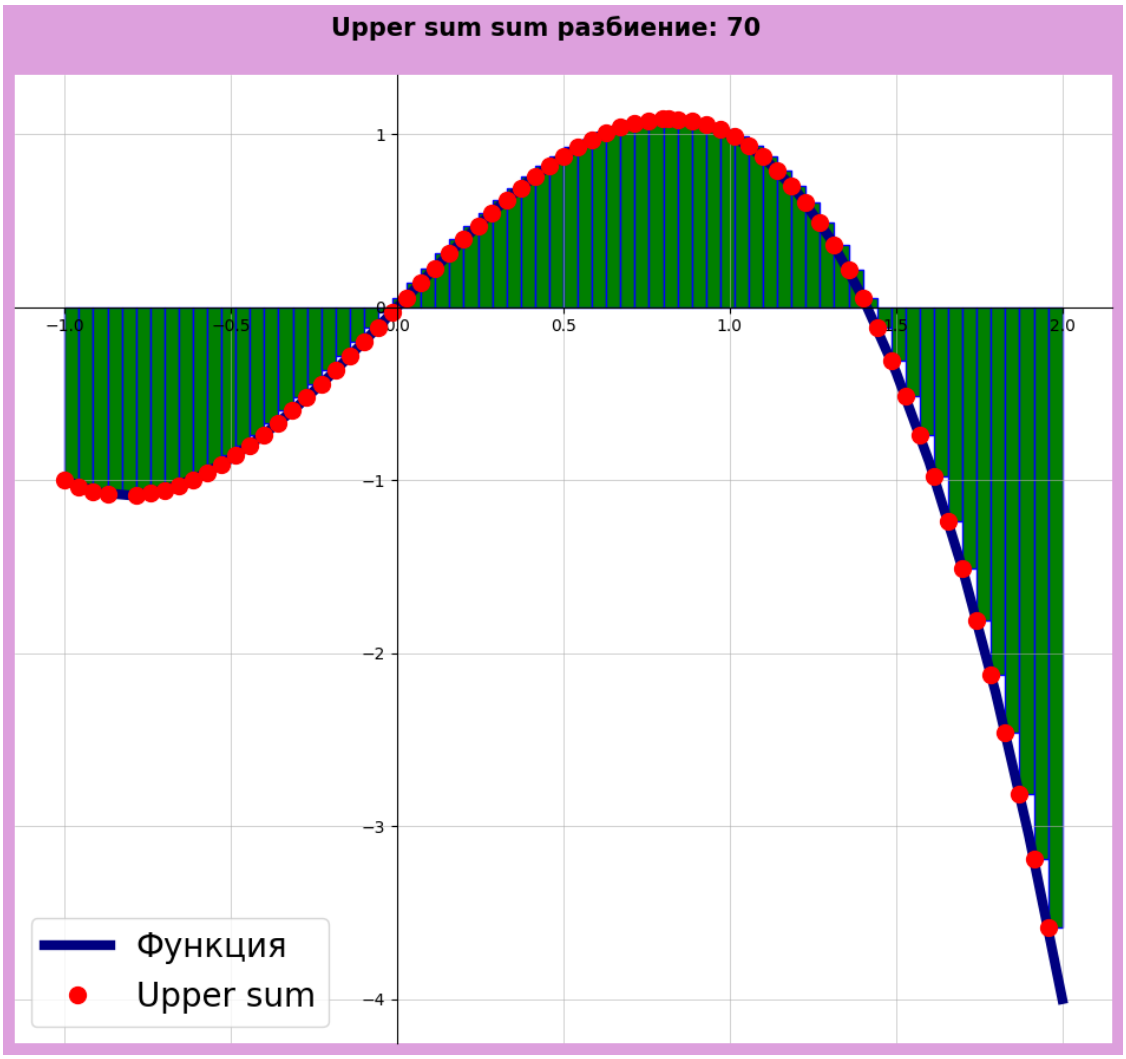
right -0.8156632653061228

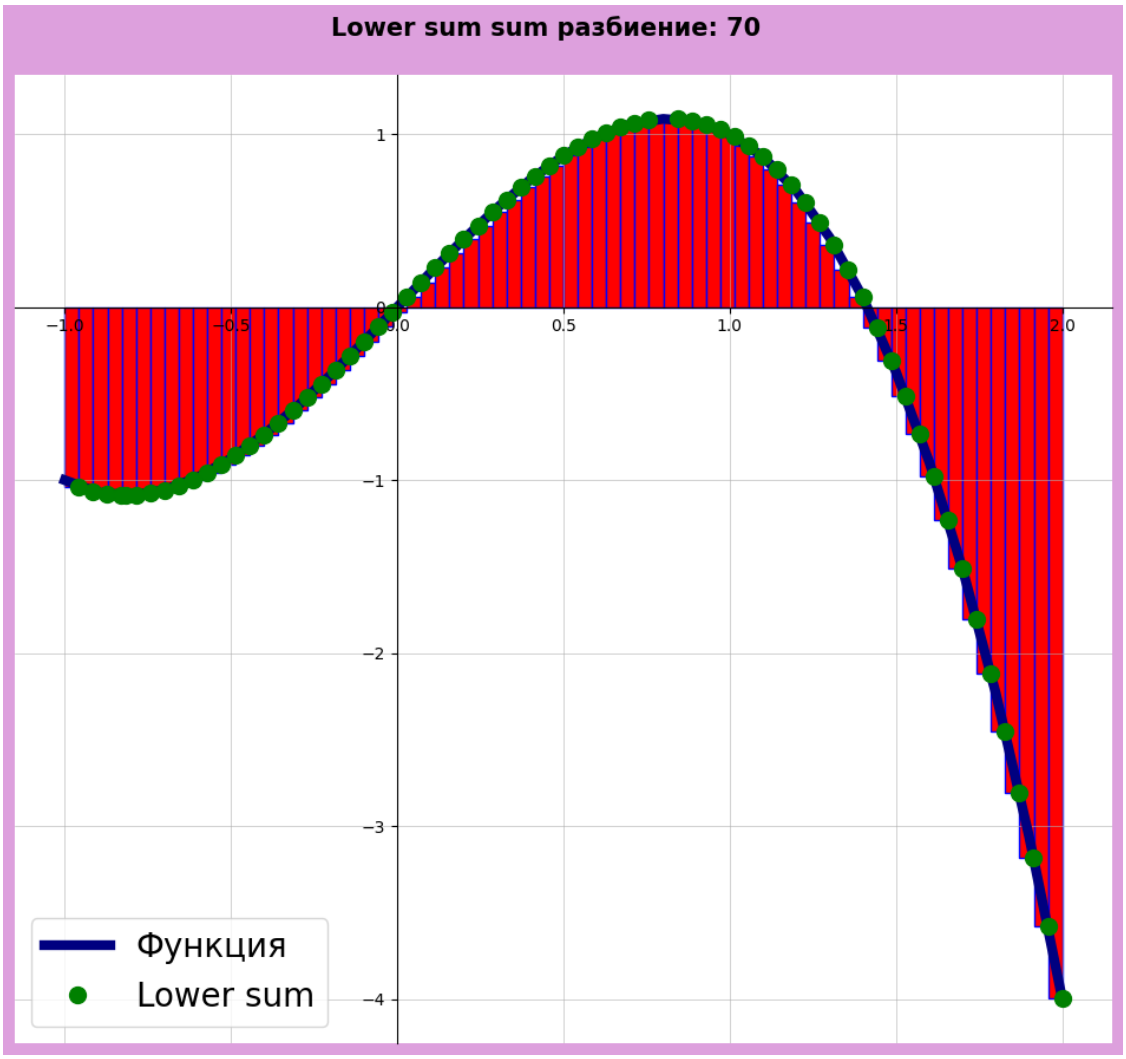


center -0.7493112244897959

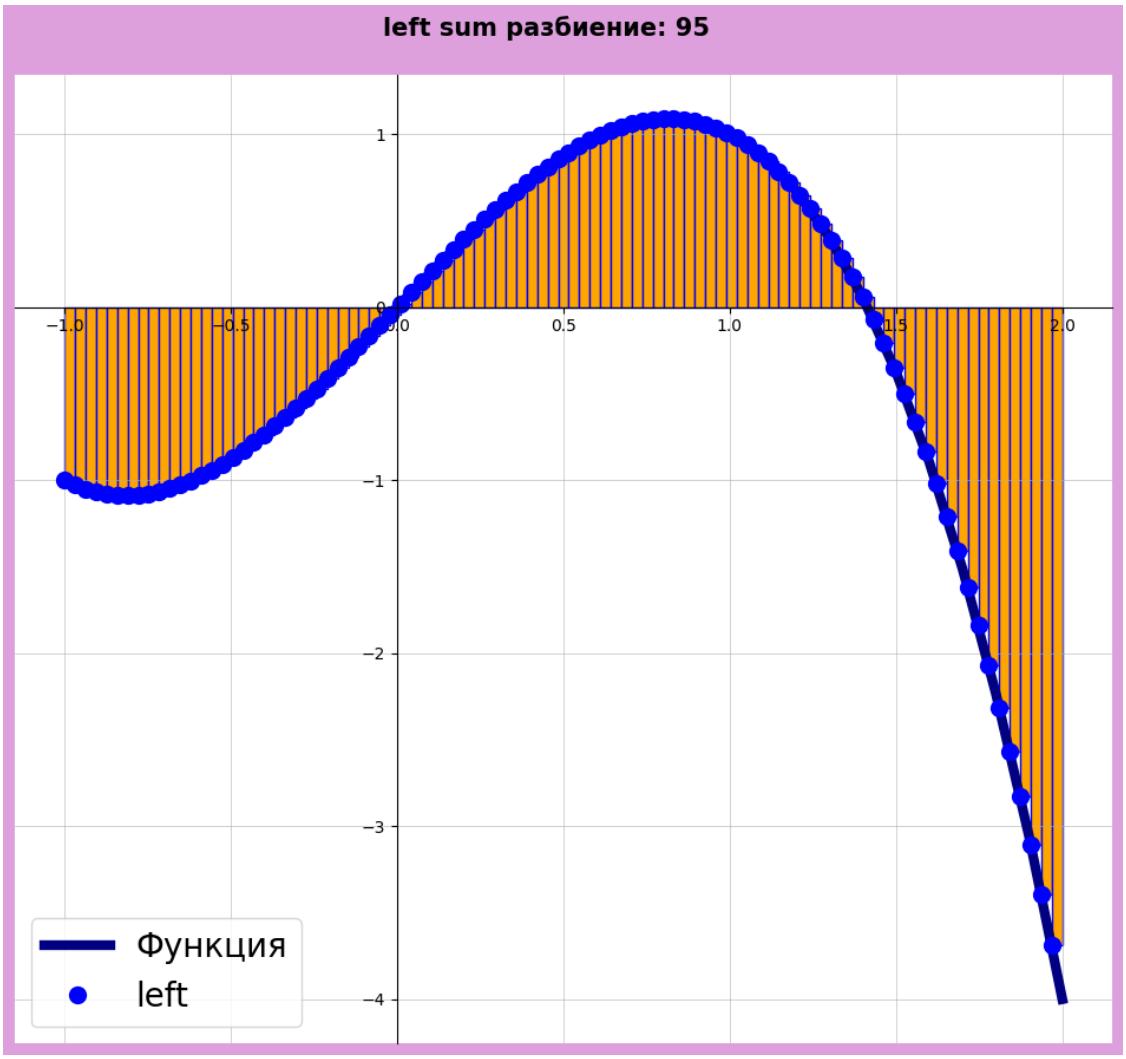


random -0.7596064119076746

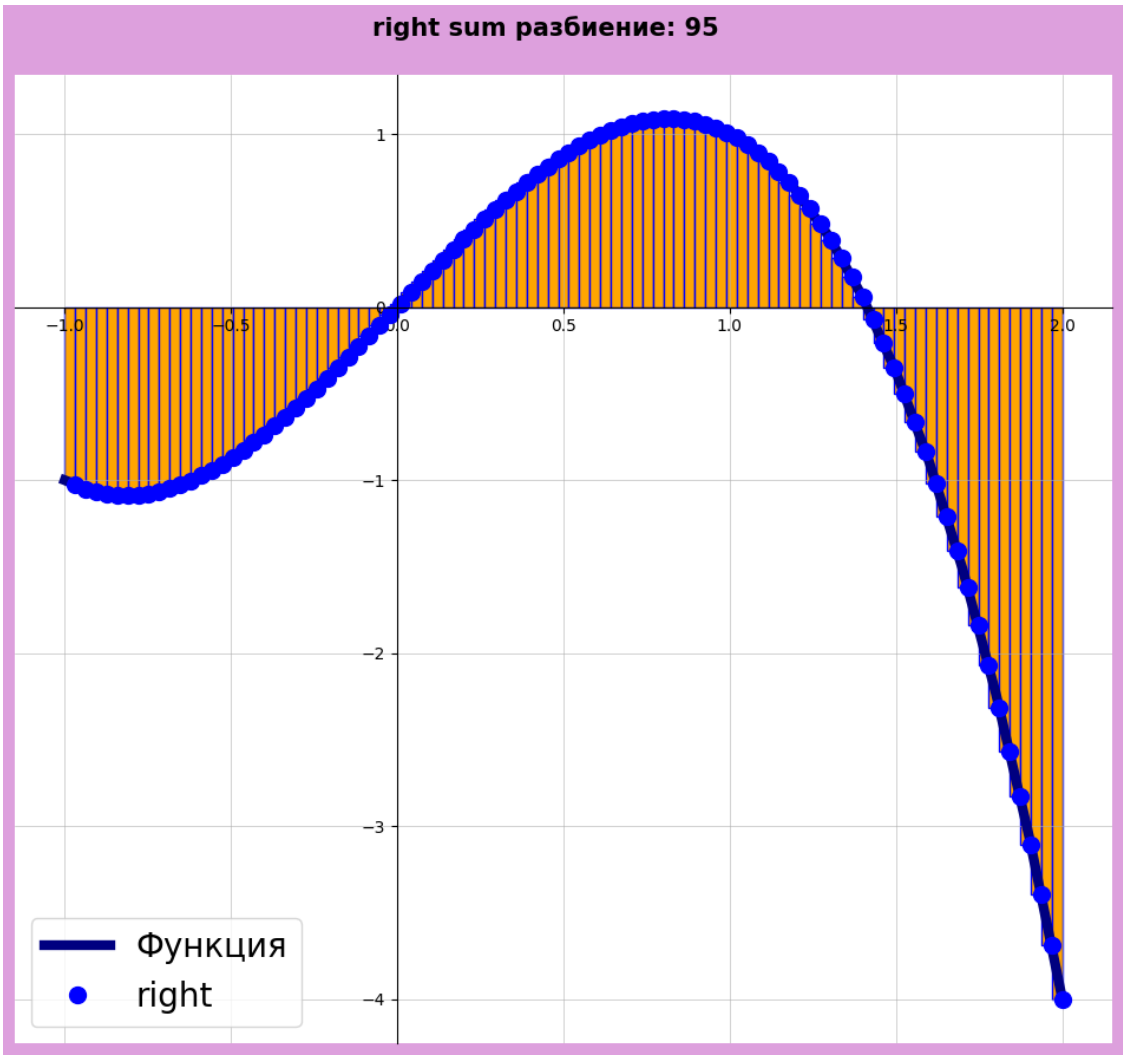

Upper sum sum разбиение: 70
1
0
−1
−2
−3
−4
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
Функция
Upper sum



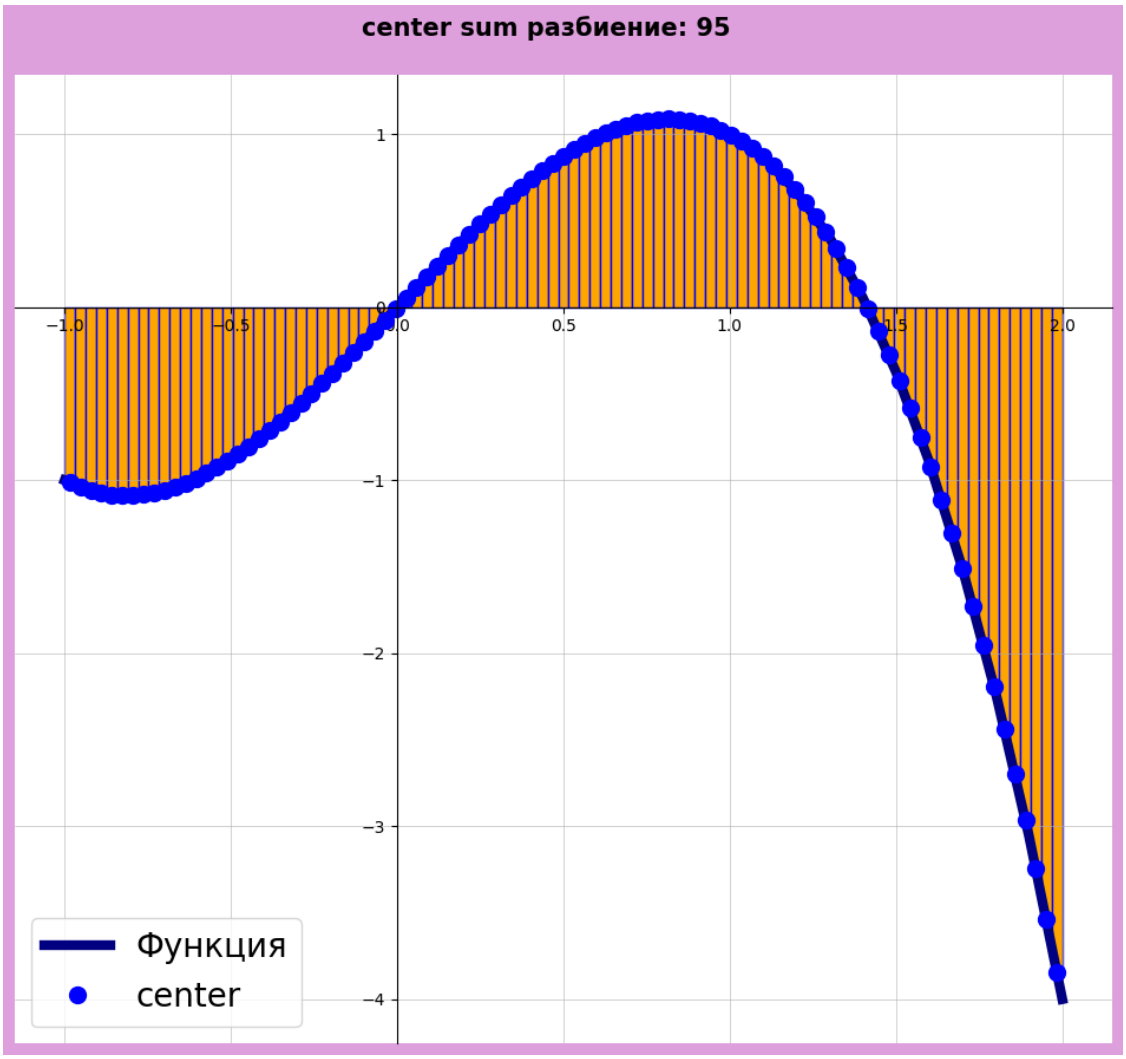

```
Upper sum:  -0.5945166550245716
Lower sum:  -0.9066467486014251
```



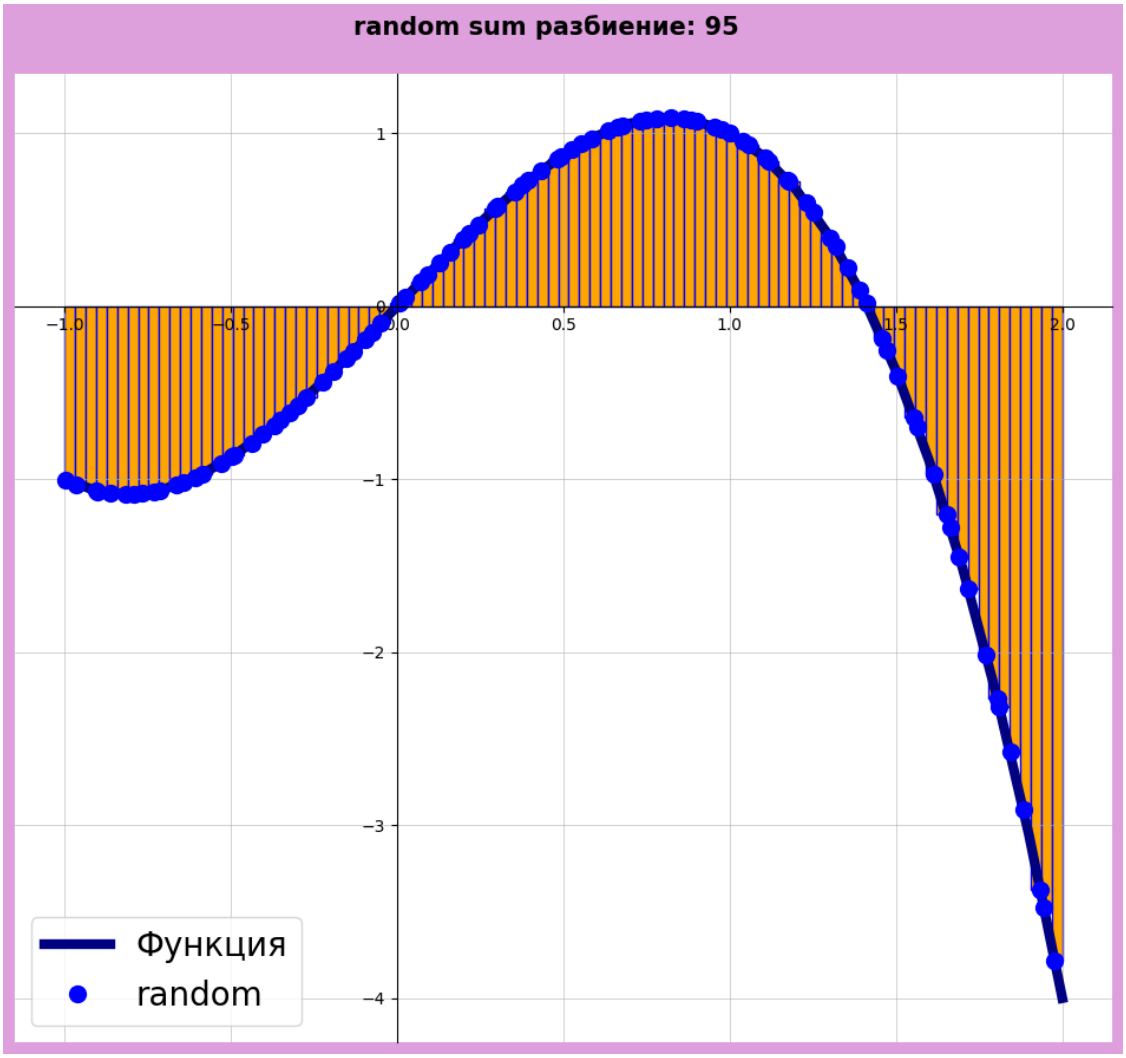
left -0.7033795013850419



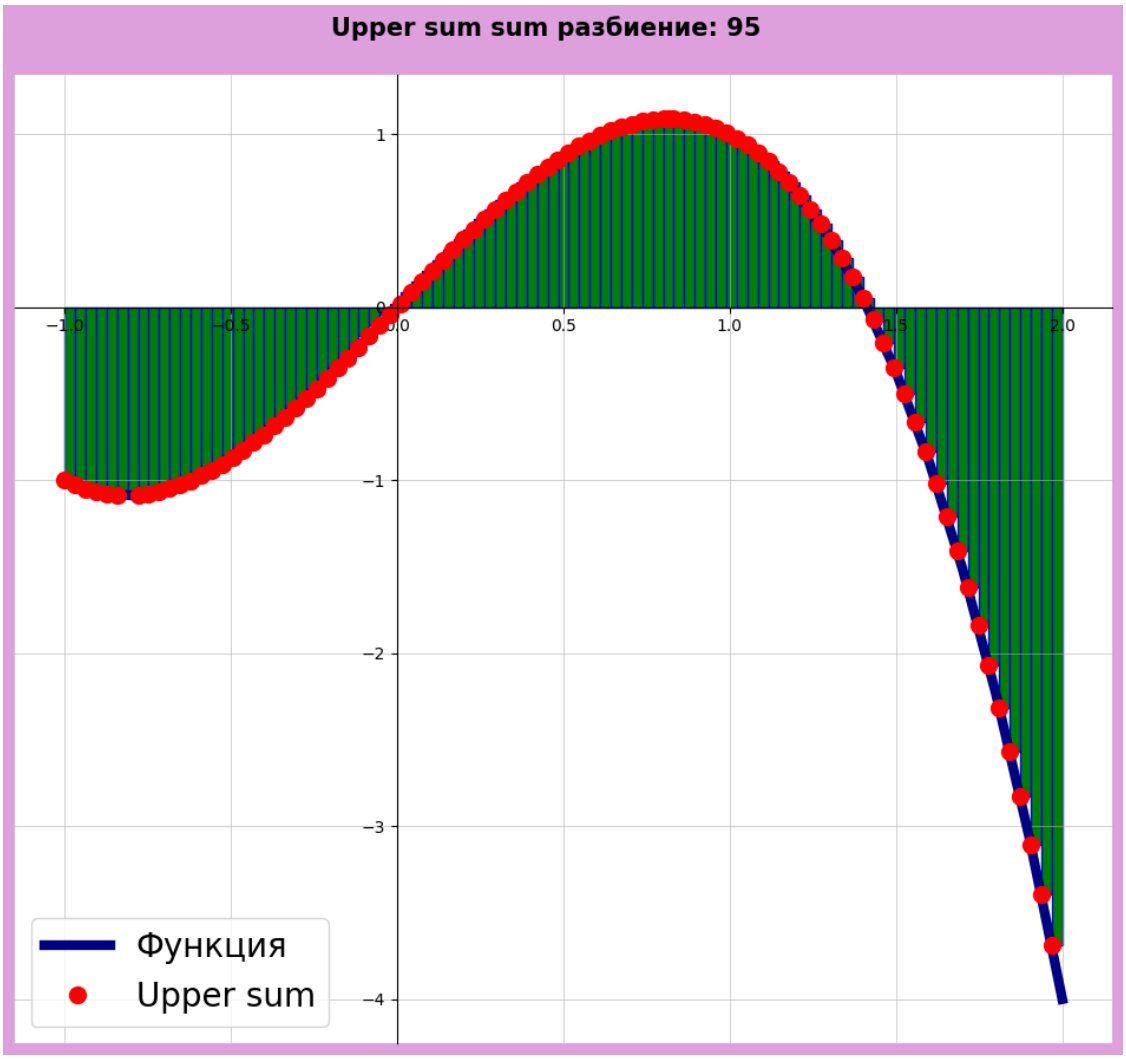
right -0.798116343490305

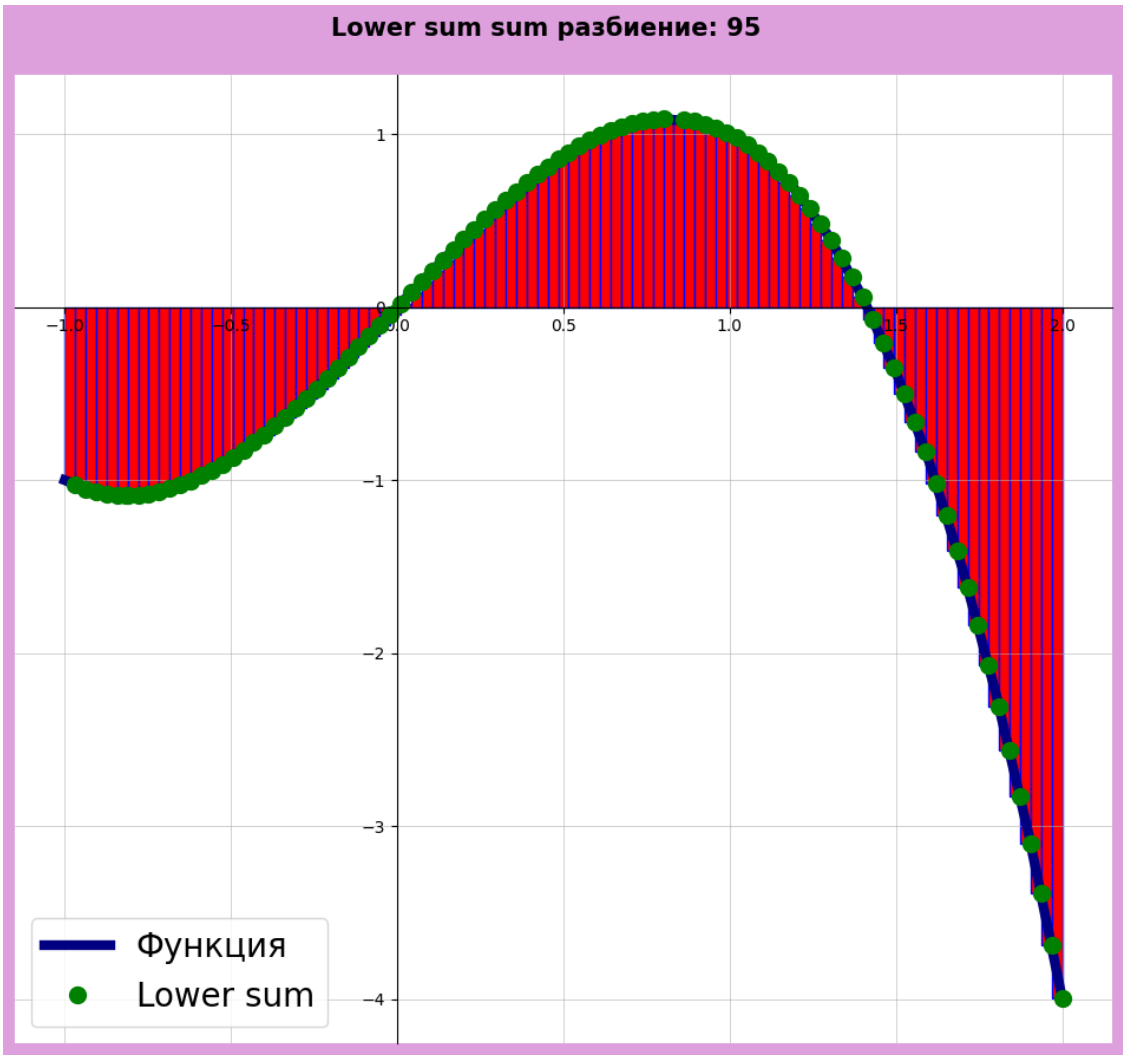


center -0.7496260387811639

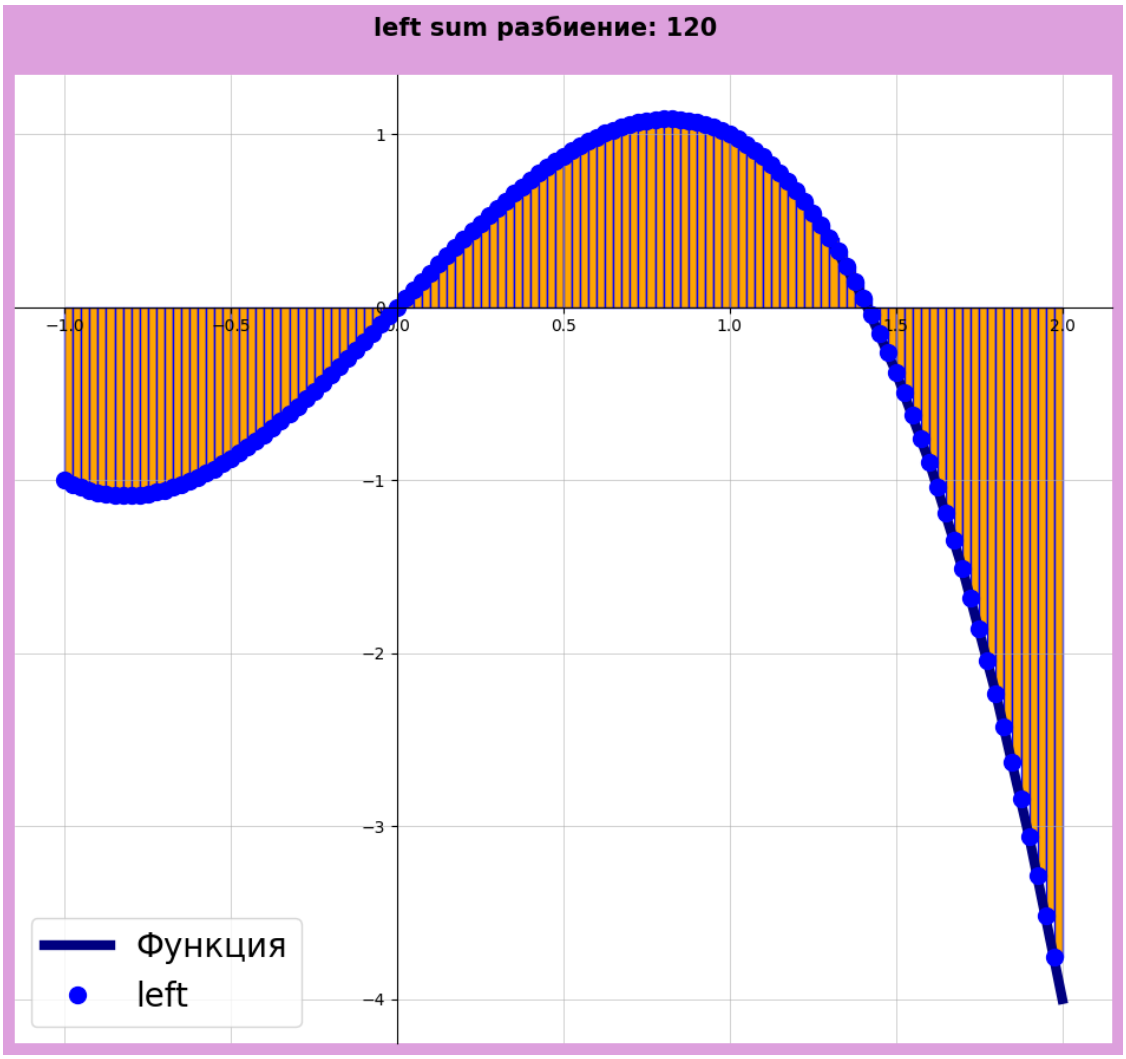


random -0.7395572004021873

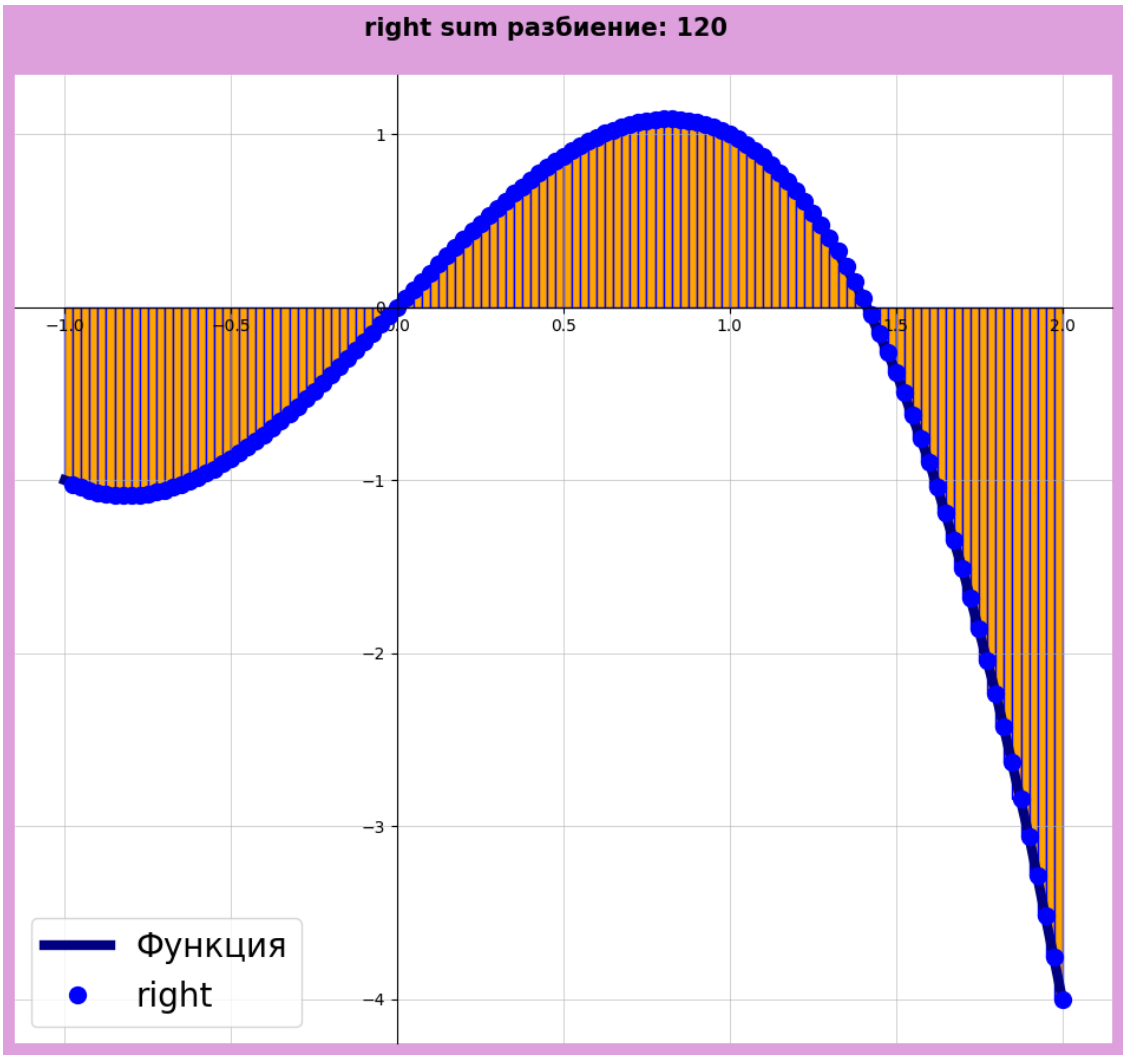

Upper sum sum разбиение: 95
1
0
−1
−2
−3
−4
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
Функция
Upper sum



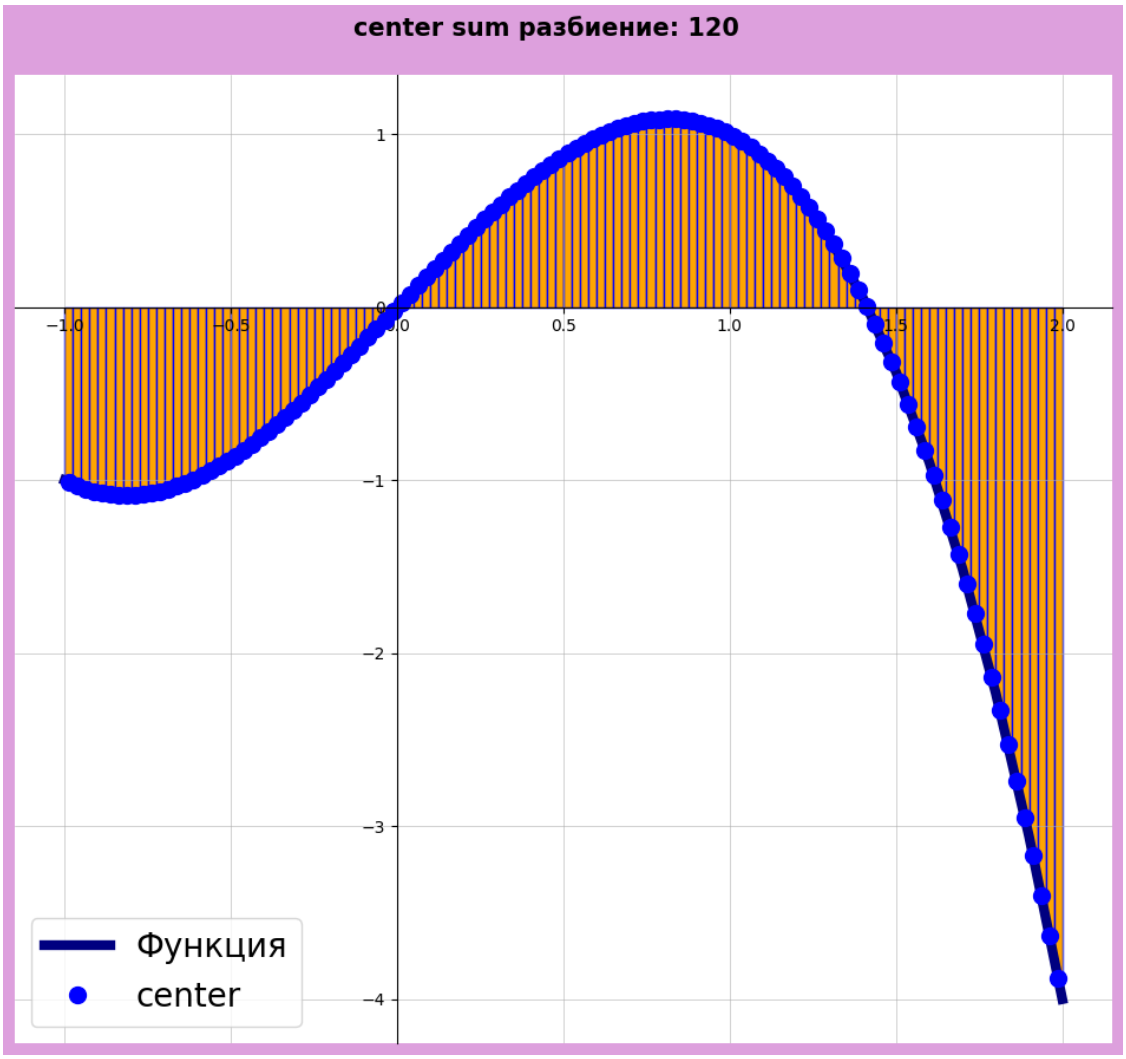

```
Upper sum:  -0.6351290065594849
Lower sum:  -0.8652291880060025
```



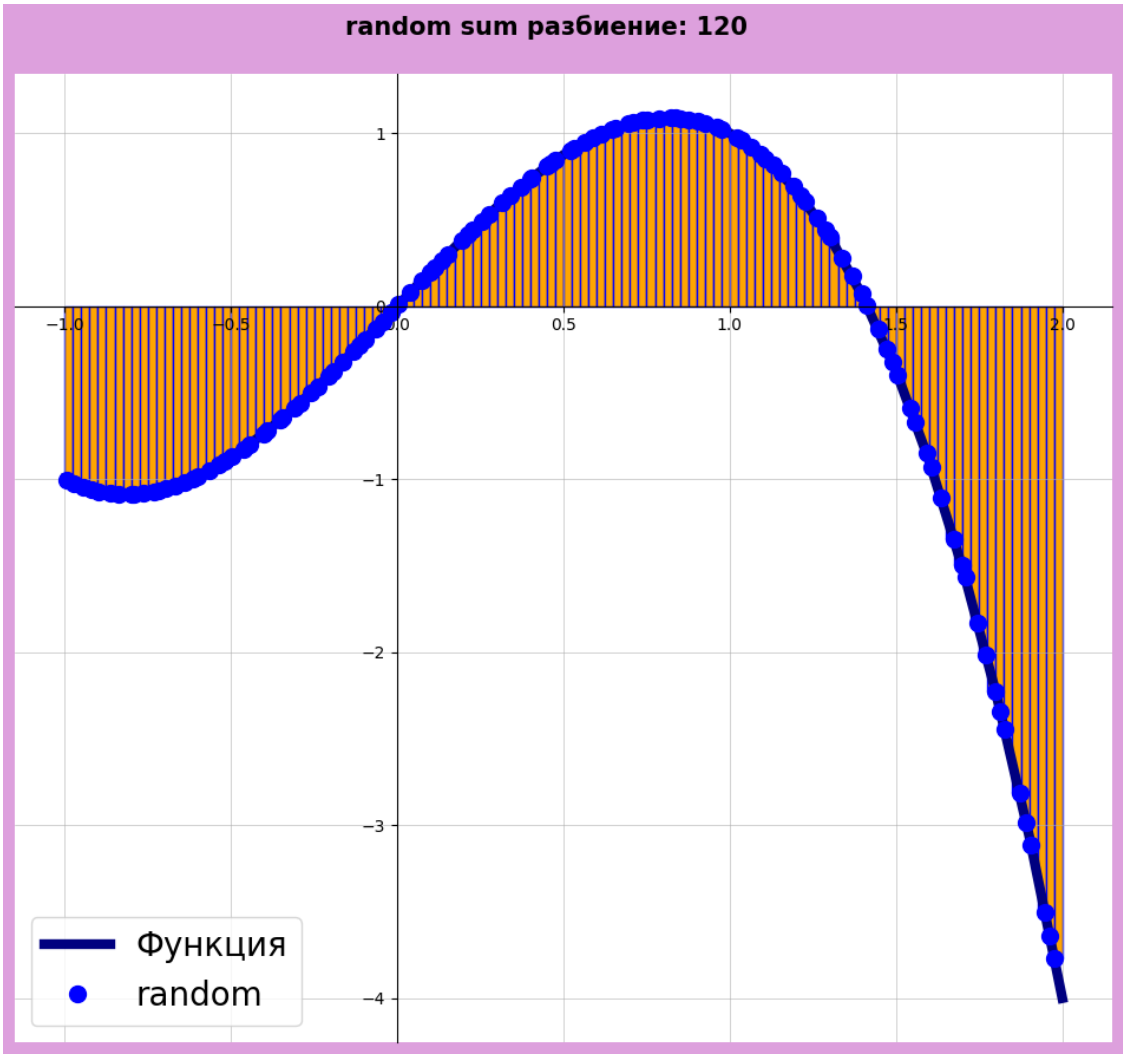
left -0.7129687500000008



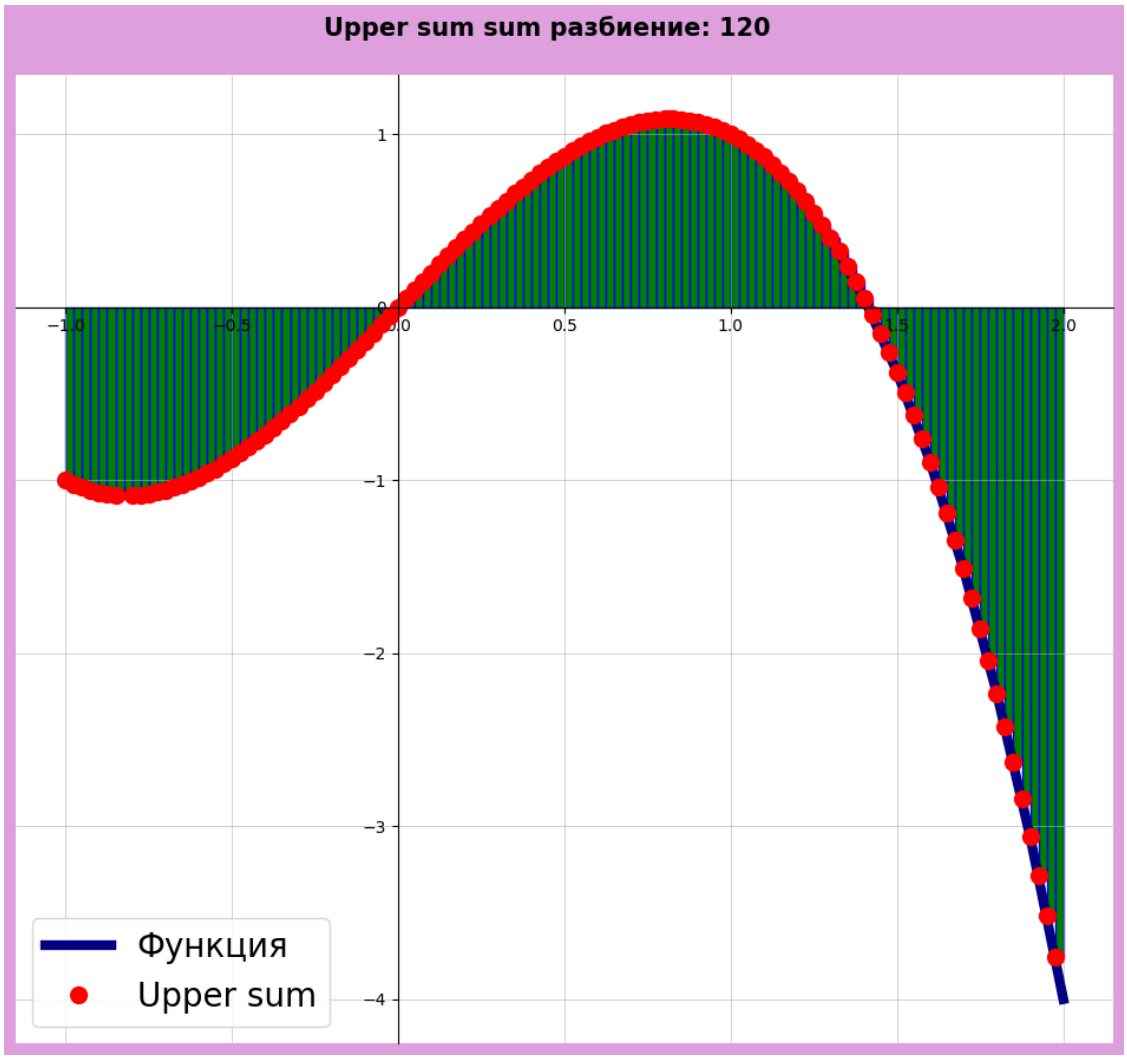
right -0.787968750000001

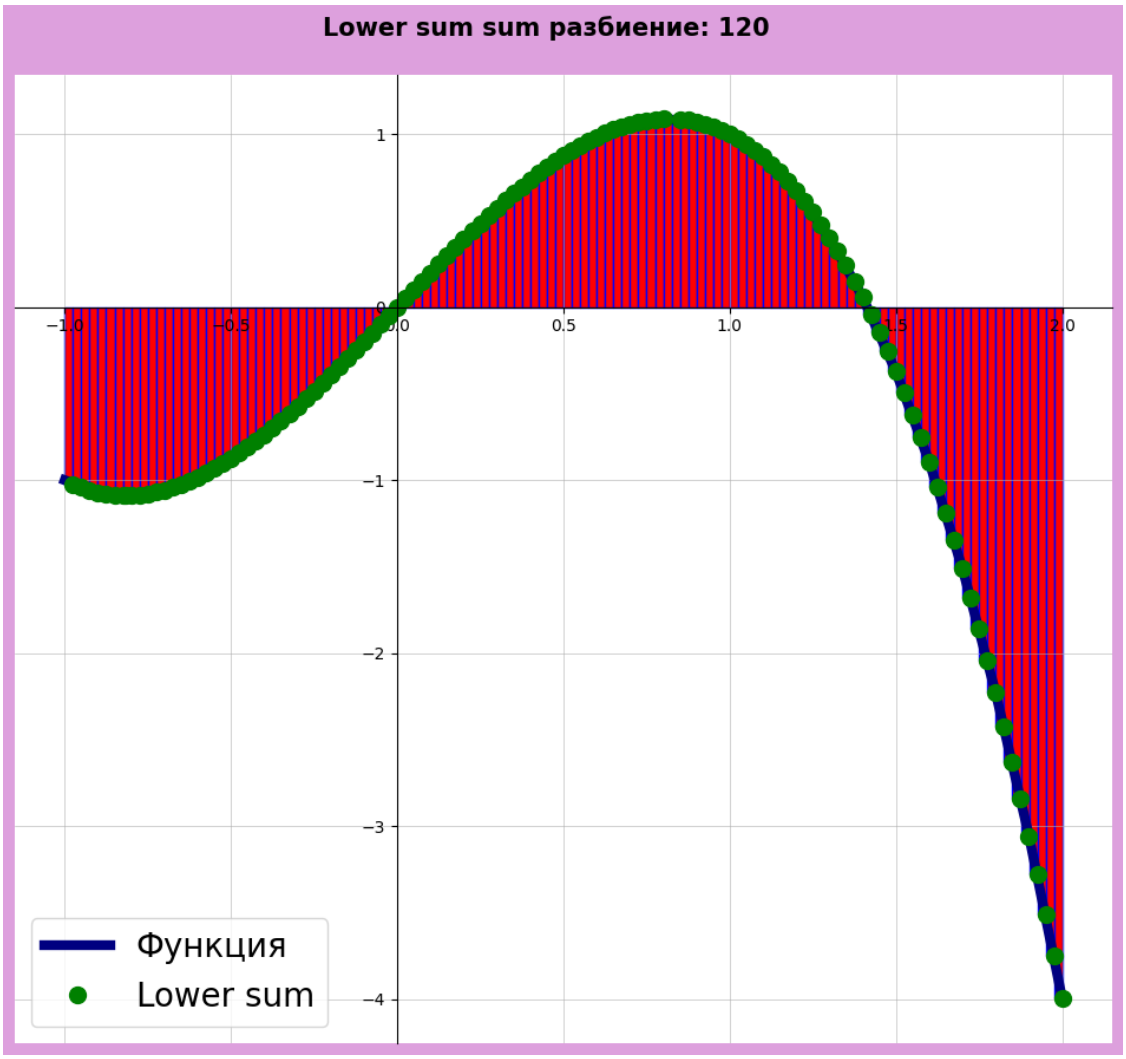


center -0.7497656250000007

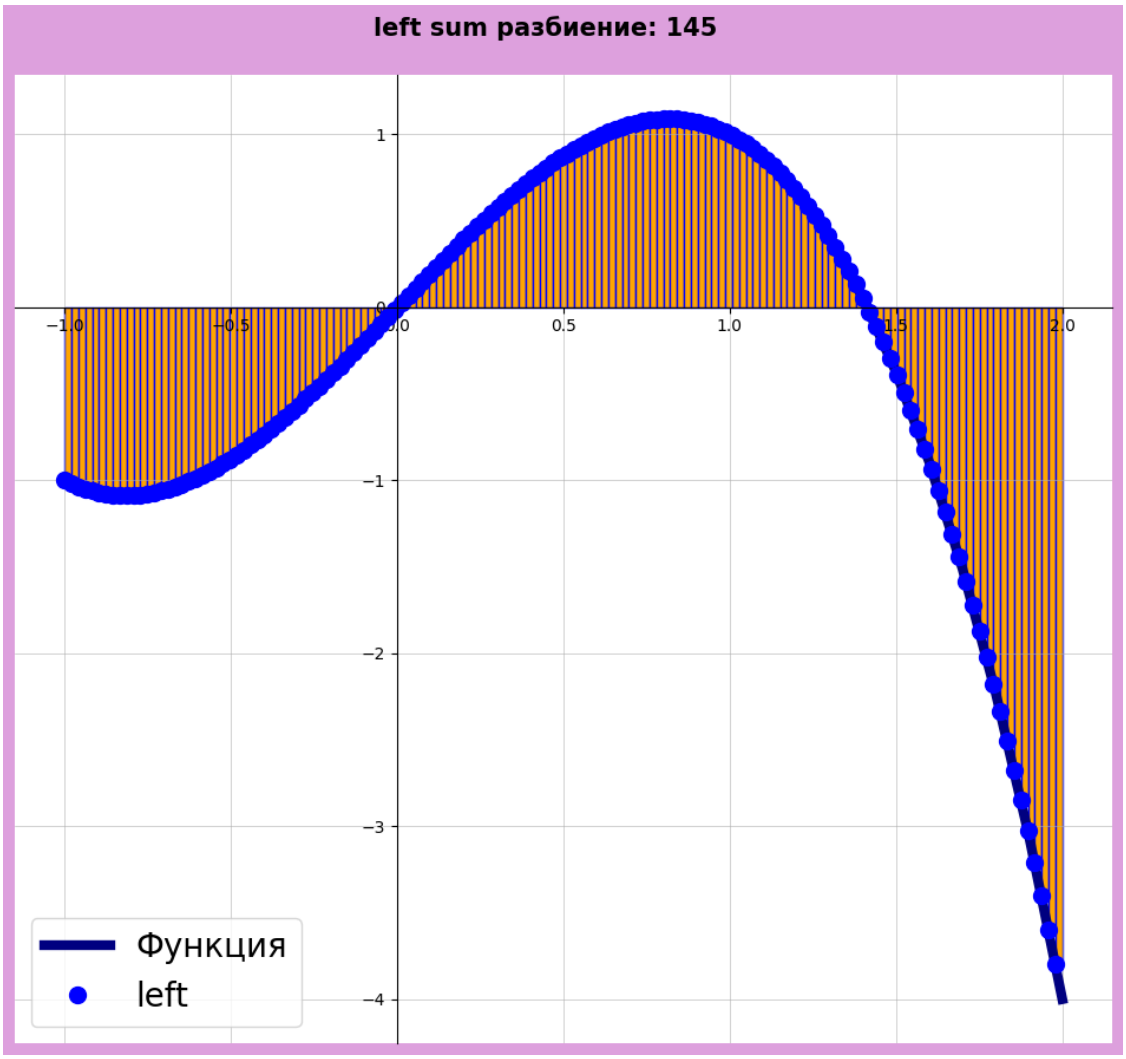


random -0.7589191988001868

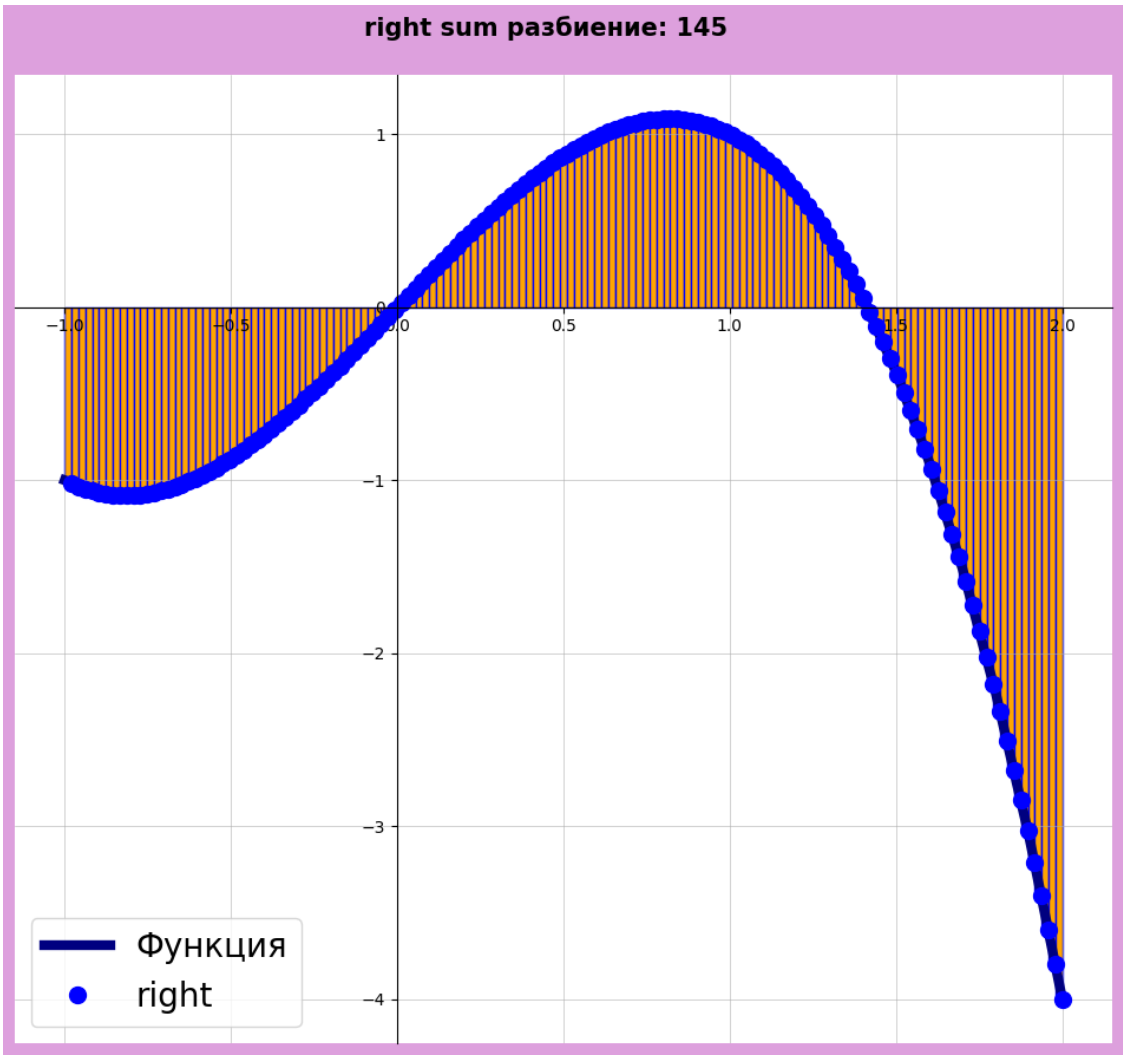

Upper sum sum разбиение: 120
1
0
−1
−2
−3
−4
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
Функция
Upper sum



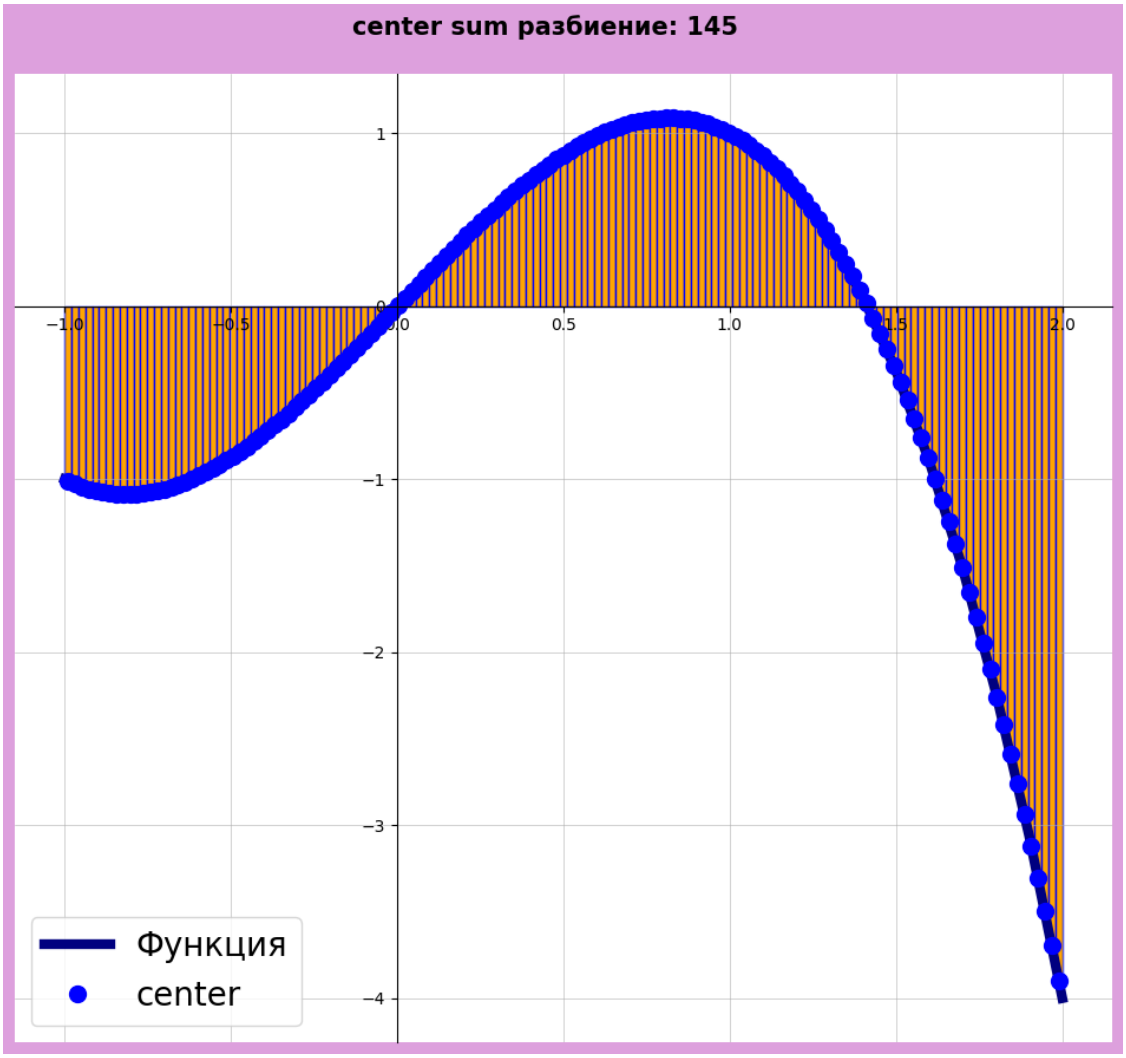

```
Upper sum:  -0.6586366579640665
Lower sum:  -0.8411194925250426
```



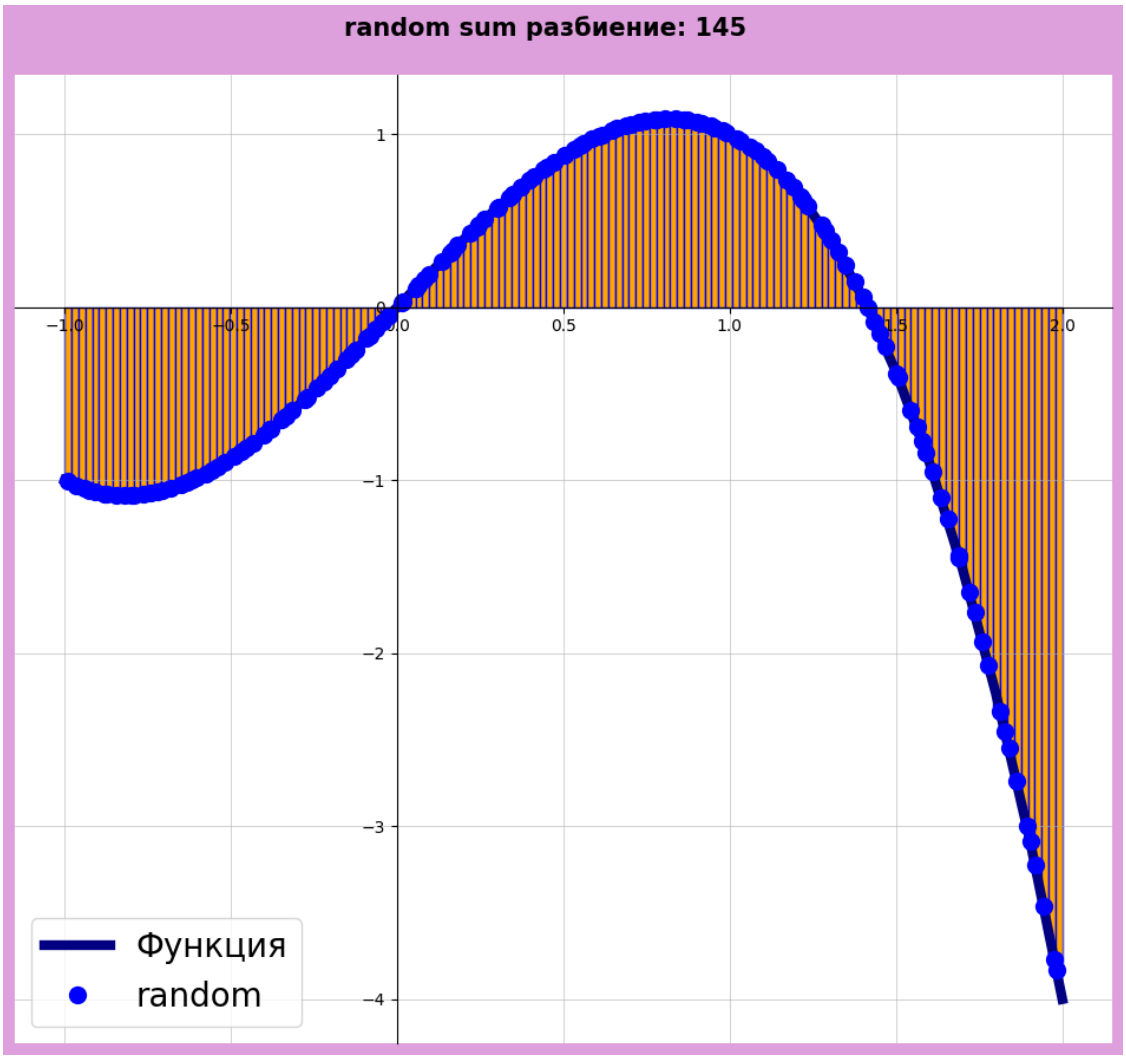
left -0.7192865636147443



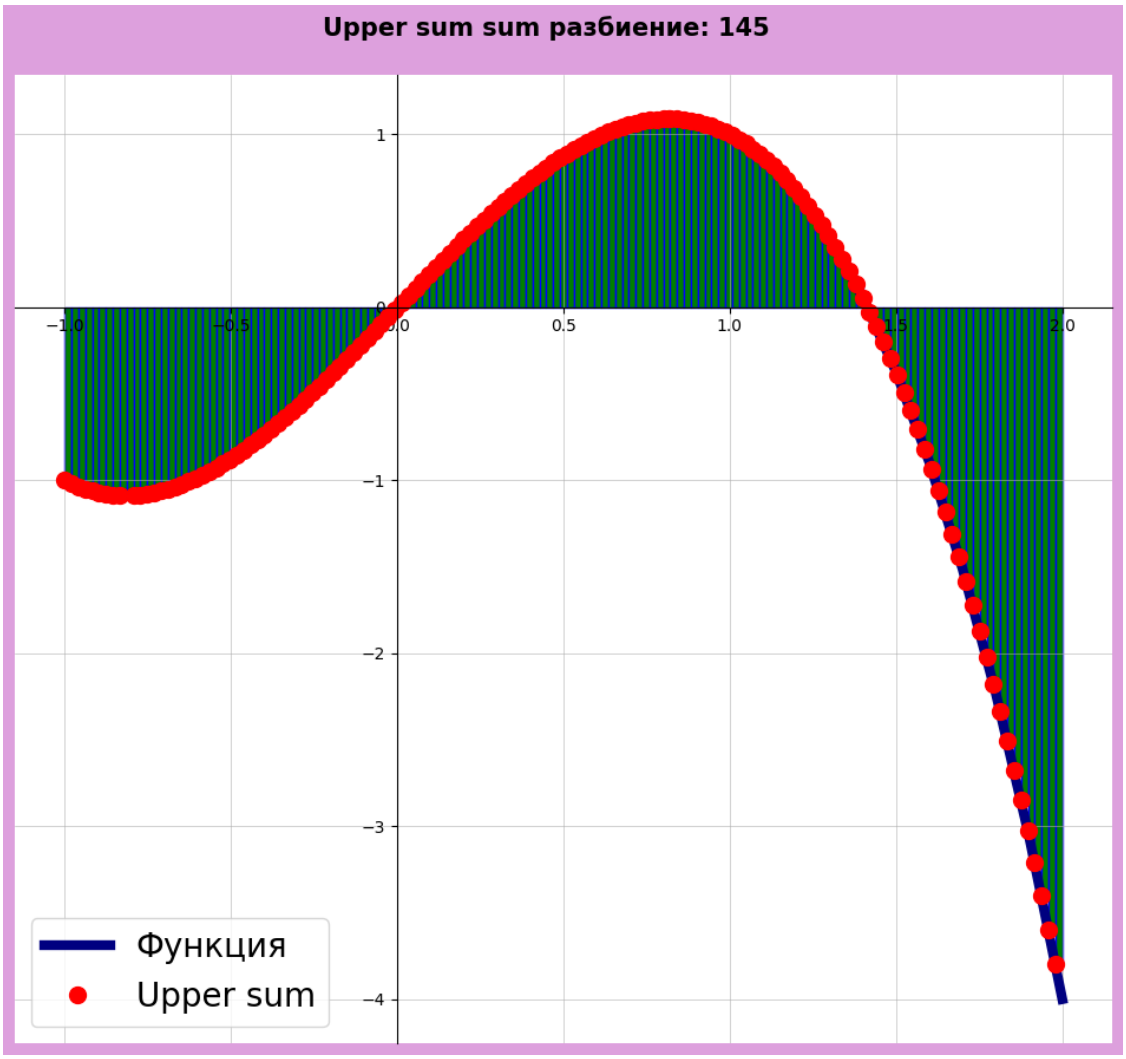
right -0.7813555291319854

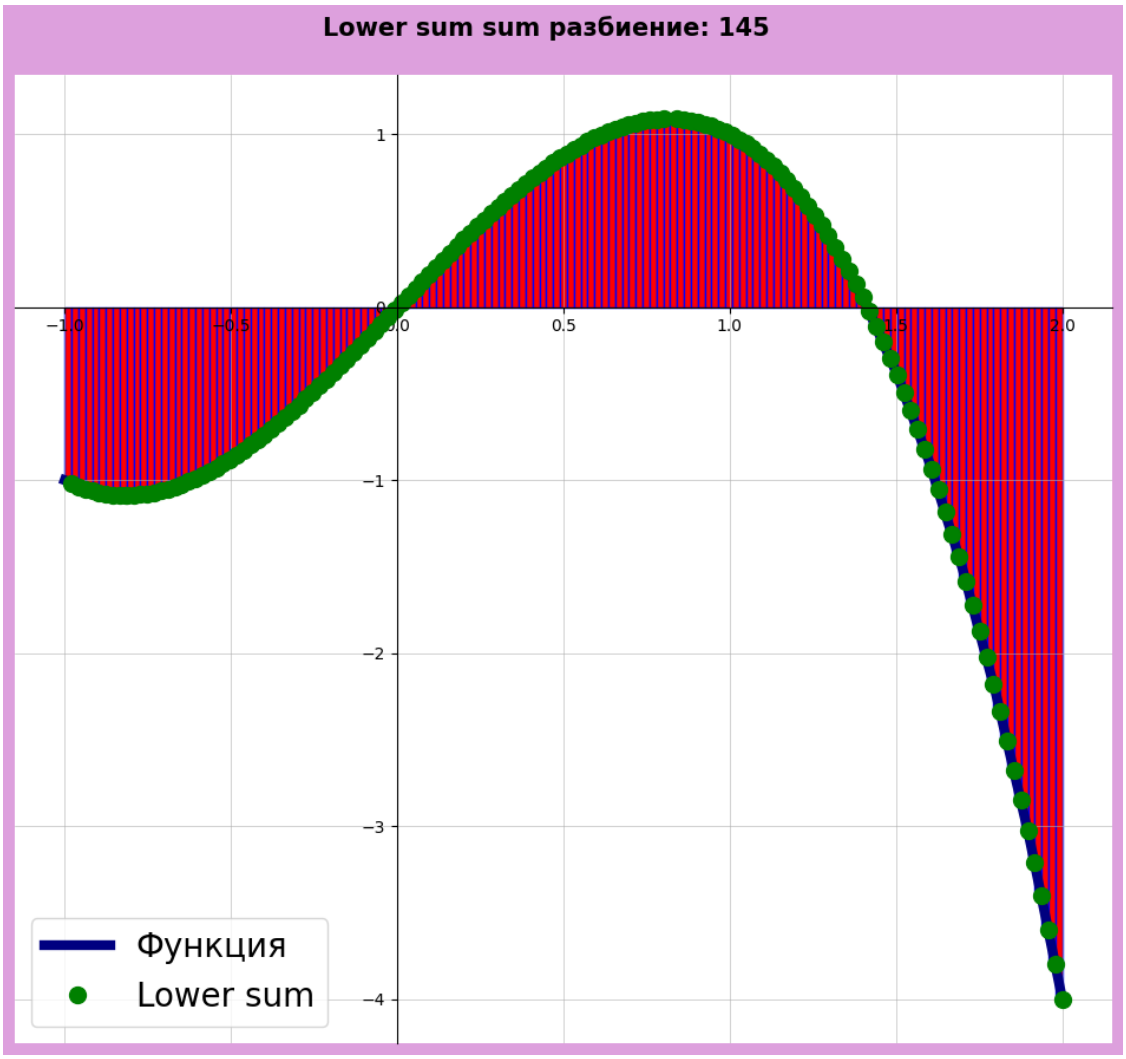


center -0.7498394768133165

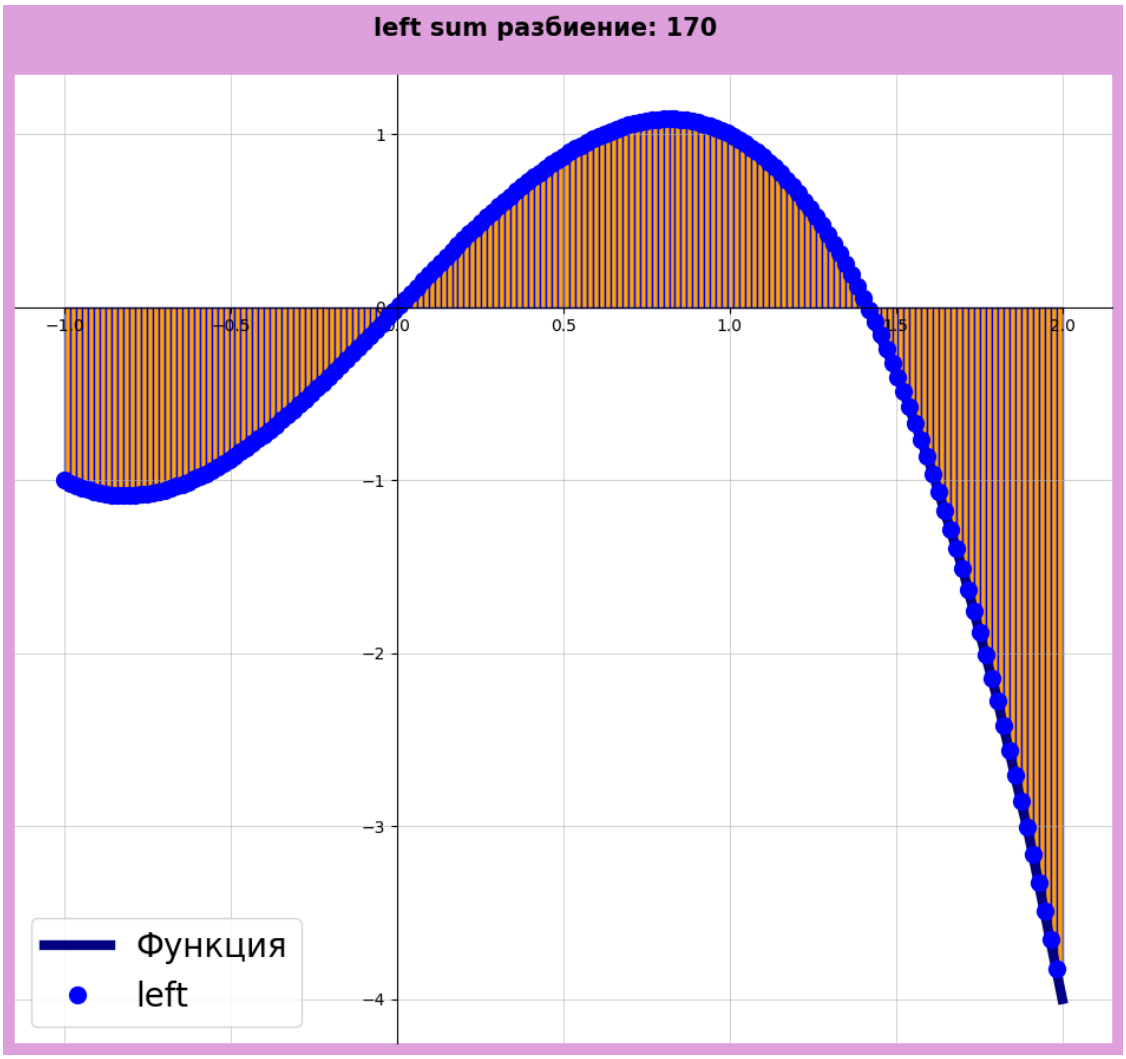


random -0.75032385973748

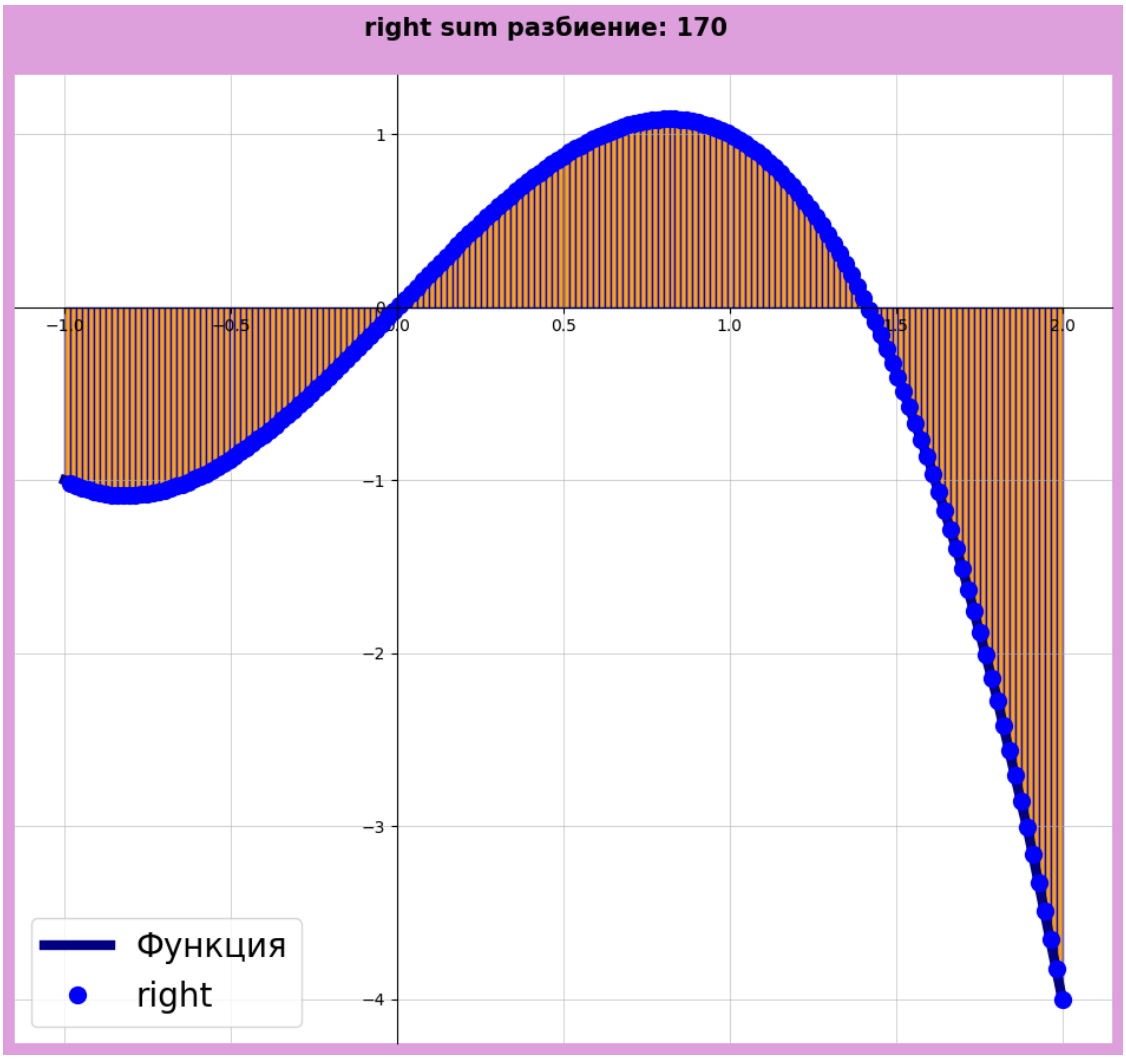

Upper sum sum разбиение: 145
1
0
−1
−2
−3
−4
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
Функция
Upper sum



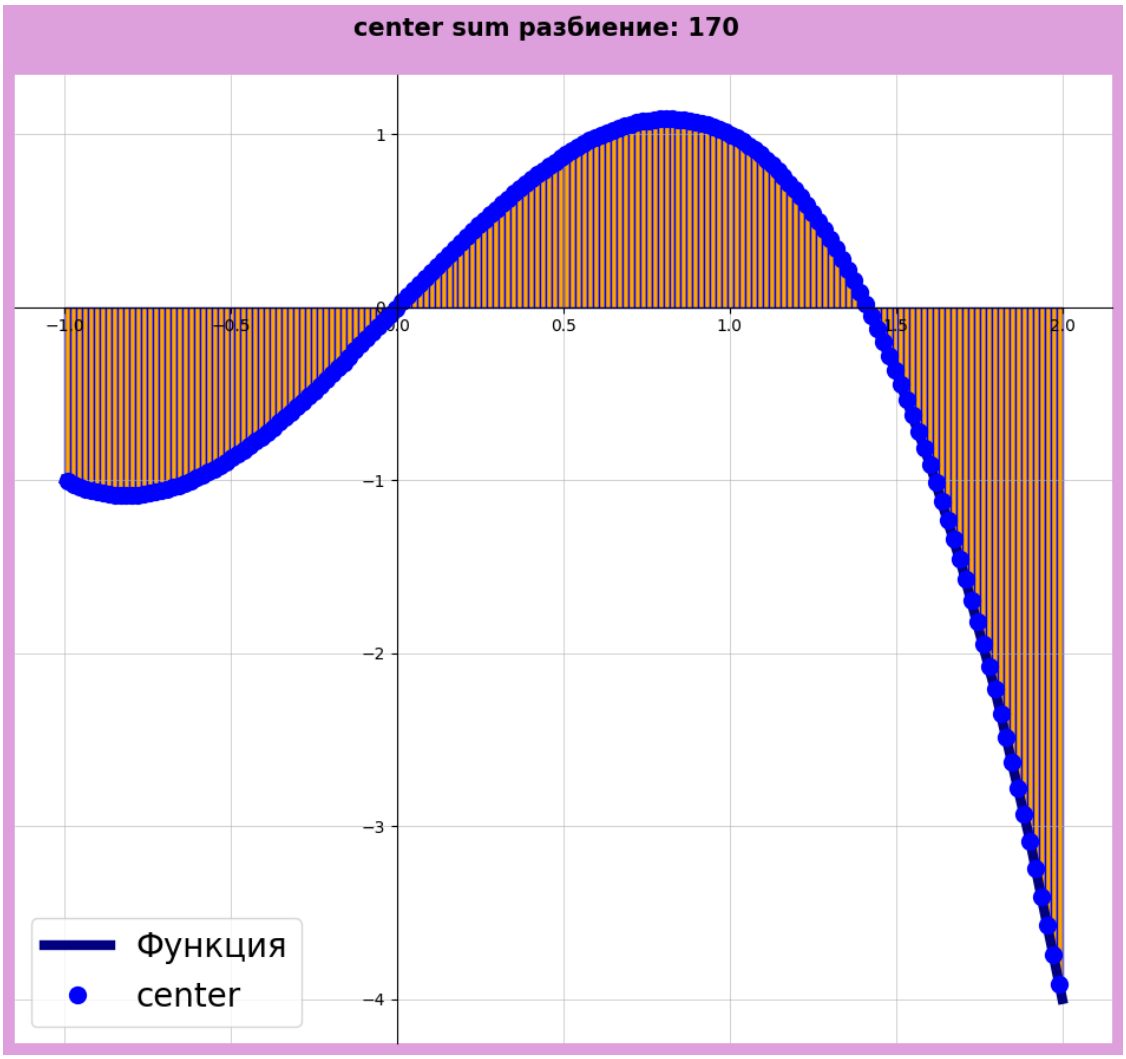

```
Upper sum:  -0.6743529709400741
Lower sum:  -0.8264027237872119
```



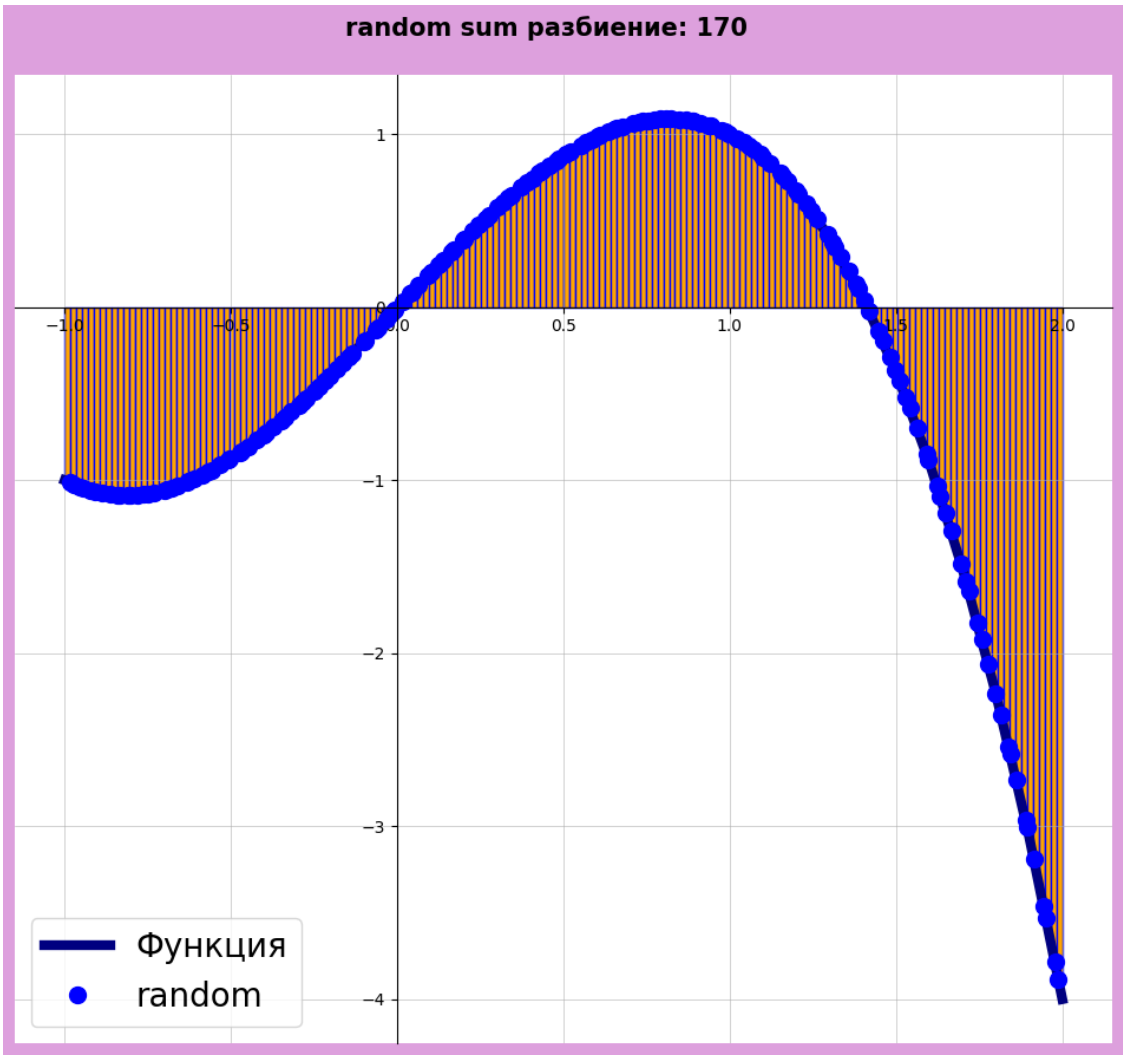
left -0.7237629757785468



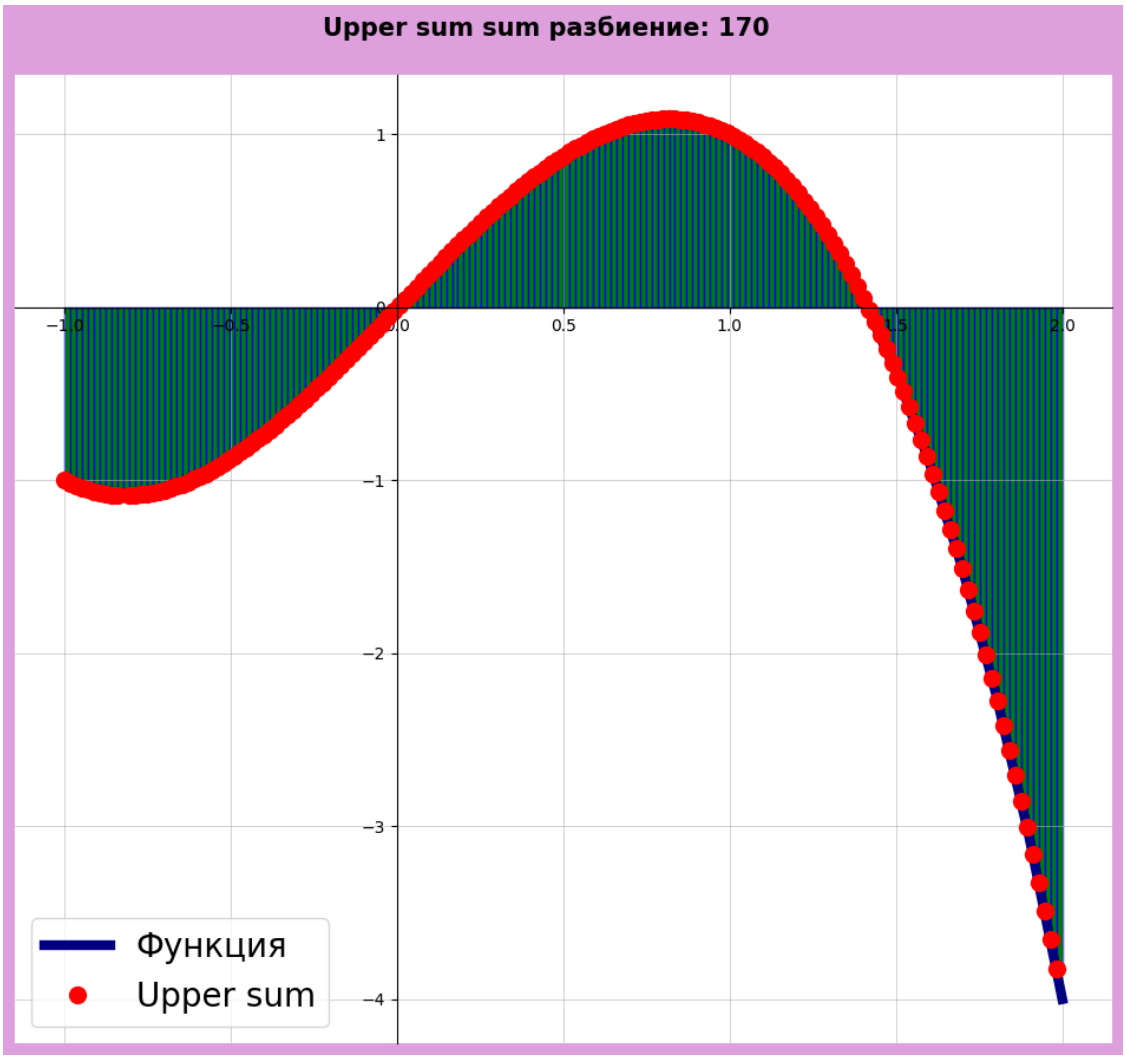
right -0.776704152249135

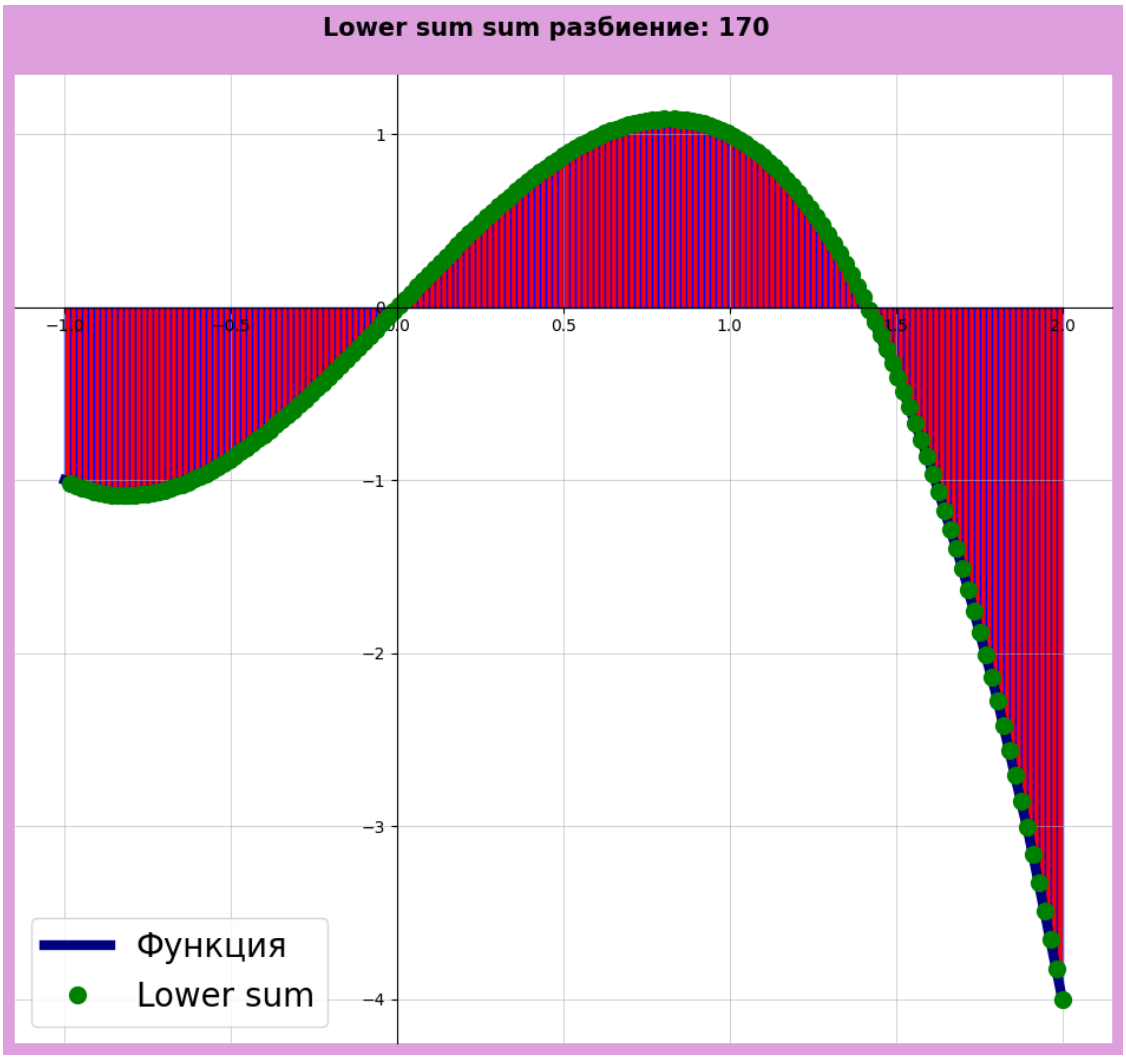


center -0.7498832179930793



random -0.7433657483772804


Upper sum sum разбиение: 170
1
0
−1
−2
−3
−4
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
Функция
Upper sum




```
Upper sum:  -0.6855271346922833
Lower sum:  -0.8150972129696125
```

```
print(table)
```

```
+-----+--------+--------+--------+--------+--------+--------+--------+
| N   | LEFT   | RIGHT  | CENTER | RANDOM | COR    | COR    | ANAL   |
|     |        |        |        |        | LOW    | UP     | RES    |
|     |        |        |        |        |        |        | -3/4   |
+-----+--------+--------+--------+--------+--------+--------+--------+
| 20  | -0.542 | -0.992 | -0.742 | -0.747 | -1.310 | -0.217 | -0.750 |
+-----+--------+--------+--------+--------+--------+--------+--------+
| 45  | -0.653 | -0.853 | -0.748 | -0.697 | -0.998 | -0.509 | -0.750 |
+-----+--------+--------+--------+--------+--------+--------+--------+
| 70  | -0.687 | -0.816 | -0.749 | -0.760 | -0.907 | -0.595 | -0.750 |
+-----+--------+--------+--------+--------+--------+--------+--------+
| 95  | -0.703 | -0.798 | -0.750 | -0.740 | -0.865 | -0.635 | -0.750 |
+-----+--------+--------+--------+--------+--------+--------+--------+
| 120 | -0.713 | -0.788 | -0.750 | -0.759 | -0.841 | -0.659 | -0.750 |
```

```
+-----+--------+--------+--------+--------+--------+--------+--------+
| 145 | -0.719 | -0.781 | -0.750 | -0.750 | -0.826 | -0.674 | -0.750 |
+-----+--------+--------+--------+--------+--------+--------+--------+
| 170 | -0.724 | -0.777 | -0.750 | -0.743 | -0.815 | -0.686 | -0.750 |
+-----+--------+--------+--------+--------+--------+--------+--------+
```

Вывод: чем больше разбиение, тем ближе значение площади к реальной точной площади из аналитическоой части, к -0.750

Среднее значение самое точное, так как работает, почти как настоязий интеграл

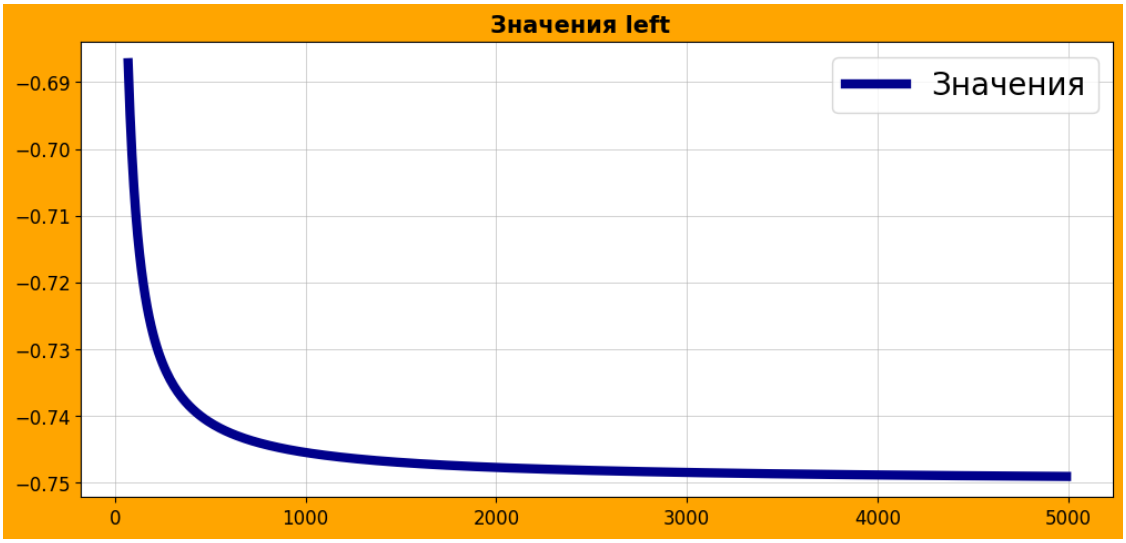
Рандом на втором месте, так как при сужении границ, случайному числу почти ничего не остается, чем быть как среднее значение

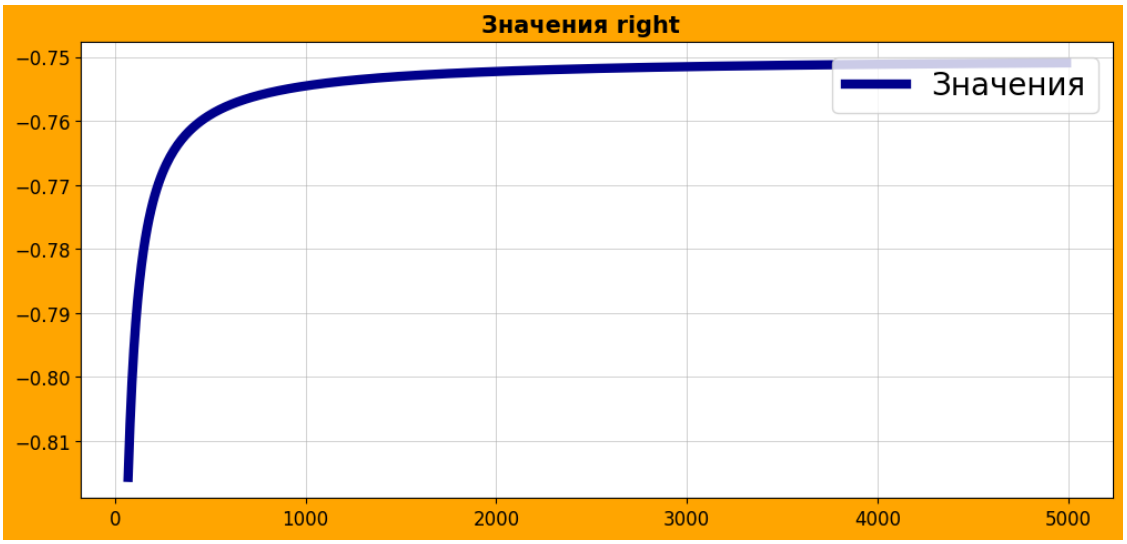
Правый, Левый, Нижняя, Верхняя постепенно идут к цели, так как берут крайние значения, максимумы или минимумы, в итоге далеки от среднего значения

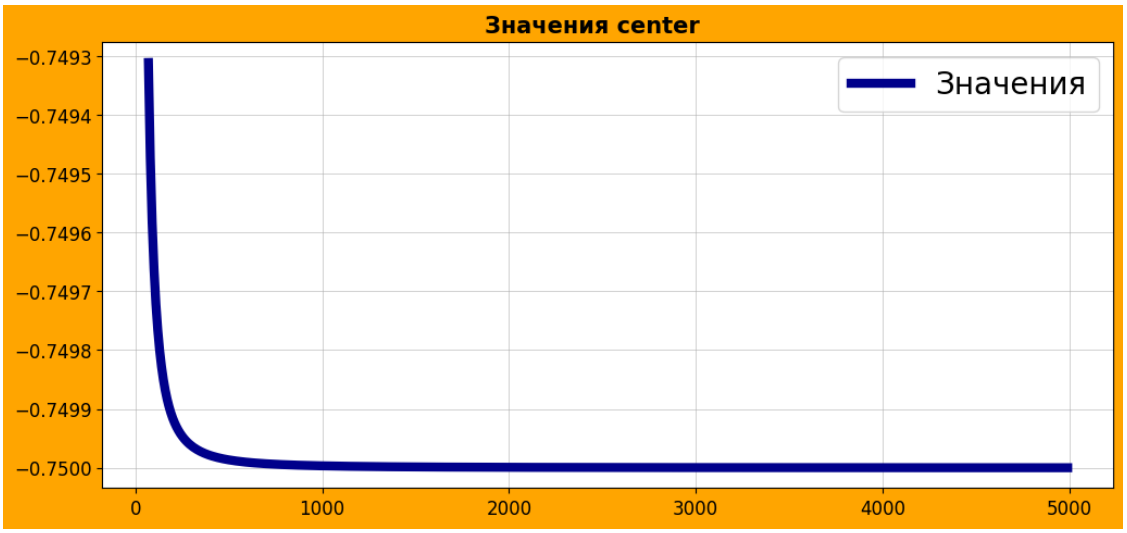
Покажу, что все функции и правда стремятся к -3/4 (-0.750)

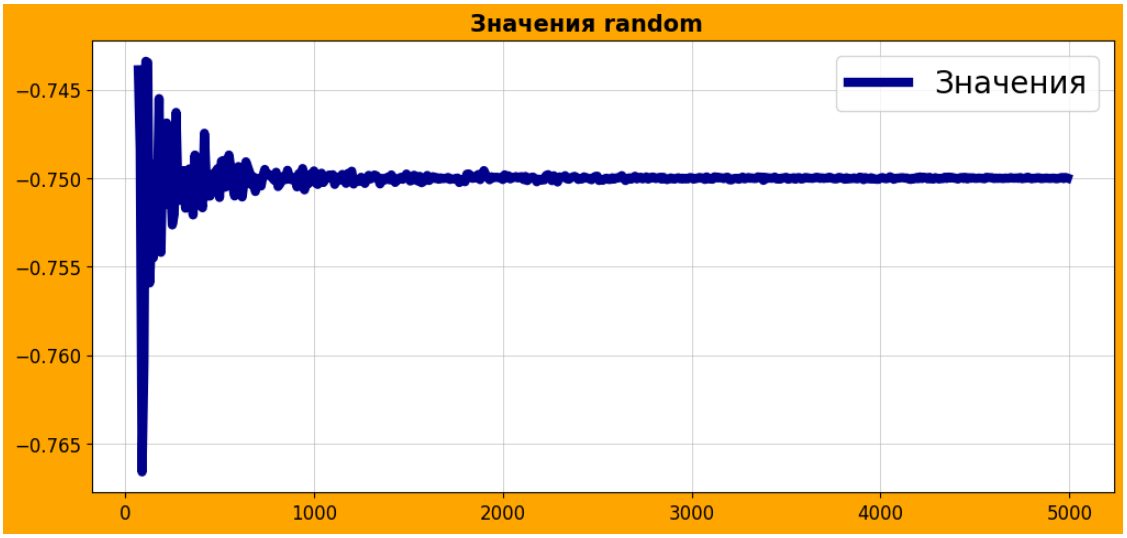
```
def draw_middle_values():
    # Средние значения
    middle_x = [0] * 493
    middle_y = [0] * 493
    pos = 0
    # Вывожу результаты с разбиением 1-752 с шагом 250
    for i in range(70, 5000, 10):
        res = darbu_sums_methods(function, start, end, i, kind, False,
False)
        middle_x[pos] = i
        middle_y[pos] = res
        pos += 1
    x = middle_x
    y = middle_y

    fig = plt.figure(figsize=(12,5))   # Размер графика
    # Цвет окантовки
    fig.patch.set_facecolor('orange')

    # Заголовок
    ax = fig.add_subplot()
    fig.subplots_adjust(top=0.93)
    fig.suptitle(f'Значения {kind}', fontsize=15, fontweight='bold')

    # Размер координат осей абсцисс и ординат
    plt.xticks(fontsize = 12)
    plt.yticks(fontsize = 12)

    # Разметка на графике
    plt.grid(axis = 'both', linewidth = 0.4)
```

```
    # Выводим график функции
    plt.plot(x, y, color='darkblue', linewidth=6, label='Значения')

    # Выведем Легенду
    plt.legend(loc=1, prop={'size': 20})

    # Вывод полученного графика
    plt.show()

kind = 'left'
draw_middle_values()
kind = 'right'
draw_middle_values()
kind = 'center'
draw_middle_values()
kind = 'random'
draw_middle_values()
```




Значения right
Значения
−0.75
−0.76
−0.77
−0.78
−0.79
−0.80
−0.81
0
1000
2000
3000
4000
5000


Значения center
Значения
−0.7493
−0.7494
−0.7495
−0.7496
−0.7497
−0.7498
−0.7499
−0.7500
0
1000
2000
3000
4000
5000


Значения random
Значения
−0.745
−0.750
−0.755
−0.760
−0.765
0
1000
2000
3000
4000
5000