

Лабораторная работа «Интеграл Римана»  
(КТ, весна 2021)

Выполнил: Канафин Евгений, М3135

Вариант 75:  $f(x) = \sin x, [0, 4\pi]$

1) Разделим промежуток на  $n$  частей, в каждой части будем брать правую границу, т.е.

$\Delta x_k = \frac{4\pi - 0}{n} = \frac{4\pi}{n}$ ,  $\sigma_n = \sum_{k=1}^n f(x_k) \Delta x_k$ , будем в качестве  $x_k$  брать правую точку промежутка, т.е.

$$x_k = k \cdot \frac{4\pi}{n} :$$

$$\sigma_n = \sum_{k=1}^n \sin k \cdot \frac{4\pi}{n} \cdot \frac{4\pi}{n} = \frac{4\pi}{n} \sum_{k=1}^n \sin\left(\frac{2k}{n} \cdot 2\pi\right) =$$

$$= \left[ \sin x - \text{периодична, период } T=2\pi \right] = \frac{4\pi}{n} \cdot 2 \cdot \sum_{k=1}^n \sin \frac{k}{n} \cdot 2\pi =$$

$= \left[ \sin x - \text{нечетна; } \sin(x) = -\sin(x+\pi) \right]$ , разделим промежуток  $(0; 4\pi]$  на 2 части:  $(0; 2\pi]$  и  $(2\pi; 4\pi]$ ,

$$\sum_{k=1}^n \sin\left(2k \cdot \frac{\pi}{n}\right) = \sum_{k=1}^{\frac{n}{2}} \sin\left(2k \cdot \frac{\pi}{n}\right) + \sum_{k=\frac{n}{2}+1}^n \sin\left(2k \cdot \frac{\pi}{n}\right),$$

$$\text{то } \sum_{k=\frac{n}{2}+1}^n \sin\left(2k \cdot \frac{\pi}{n}\right) = \left[ t = k - \frac{n}{2}, k_0 = \frac{n}{2} + 1 \Rightarrow t_0 = k_0 - \frac{n}{2} = 1, \right. \\ \left. k_e = n, t_e = k_e - \frac{n}{2} = \frac{n}{2} \right] =$$

$$= \sum_{t=1}^{\frac{n}{2}} \sin\left(2\left(t + \frac{n}{2}\right) \cdot \frac{\pi}{n}\right) = \sum_{t=1}^{\frac{n}{2}} \sin\left(\pi + 2t \cdot \frac{\pi}{n}\right) = - \sum_{t=1}^{\frac{n}{2}} \sin\left(2t \cdot \frac{\pi}{n}\right), \text{ т.е.}$$

$$\sum_{k=1}^n \sin\left(2k \cdot \frac{\pi}{n}\right) = \sum_{k=1}^{\frac{n}{2}} \sin\left(2k \cdot \frac{\pi}{n}\right) - \sum_{t=1}^{\frac{n}{2}} \sin\left(2t \cdot \frac{\pi}{n}\right) =$$

$$= \sum_{k=1}^n (\sin(2k \frac{\pi}{n}) - \sin(2k \frac{\pi}{n})) = 0 \Rightarrow \sigma_n = \frac{4\pi}{n} \cdot 2 \cdot 0 = 0 \Rightarrow$$

$$\Rightarrow \lim_{n \rightarrow \infty} \sigma_n = 0$$

Также  $f(x) = \sin(x)$  явл-ся непрерывной,  
а значит, и интегрируемой

По ф. Ньютона - Лейбница:

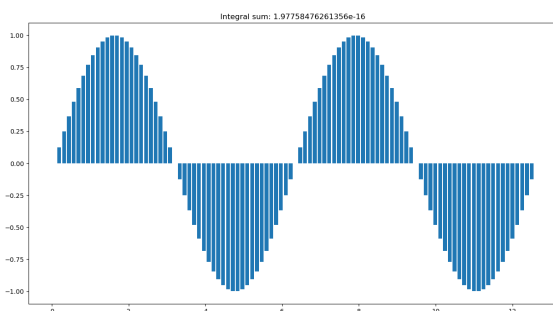
$$\int_0^{4\pi} f(x) dx = F(4\pi) - F(0) = [F - \text{первообр.}]$$

где  $f, f(x) = \sin x \Rightarrow F(x) = -\cos(x)$

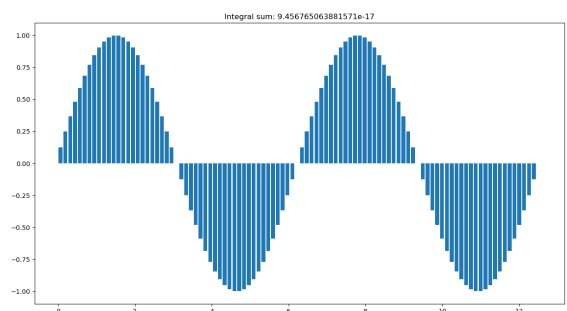
$$= -\cos(4\pi) + \cos(0) = -1 + 1 = 0$$

## Скриншоты

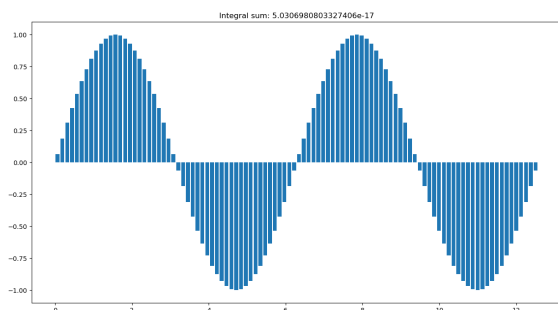
Количество точек  $n == 100$



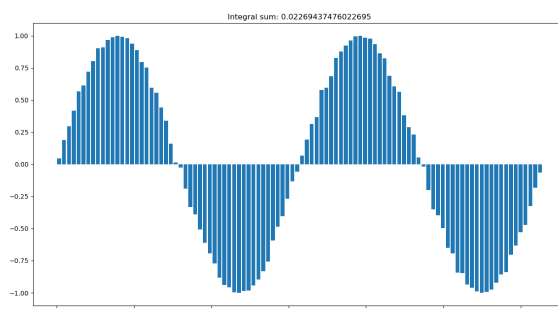
Тип оснащения == левые точки



Тип оснащения == правые точки

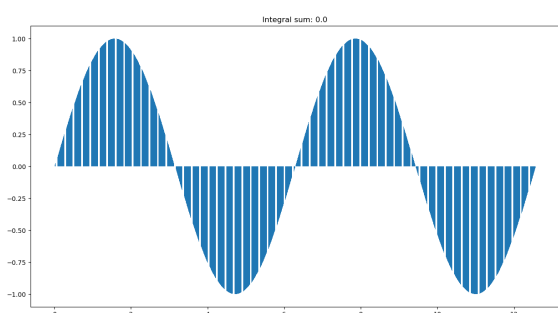


Тип оснащения == центральные точки

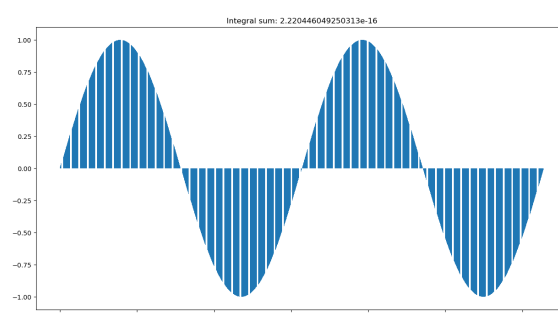


Тип оснащения == случайные точки

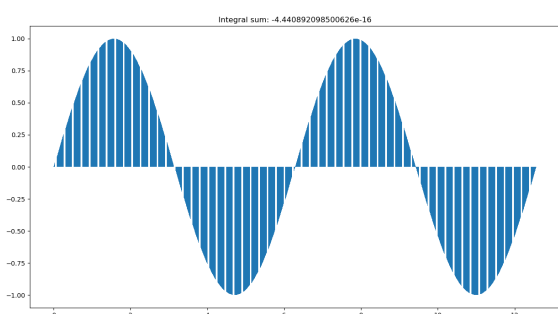
Количество точек n == 1000



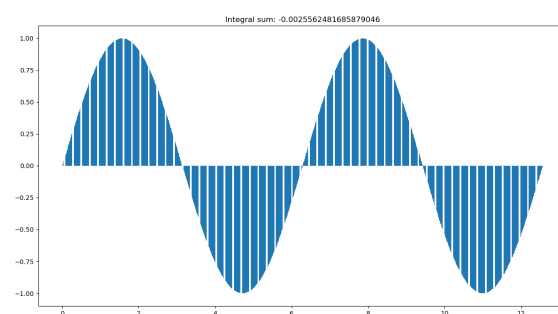
Тип оснащения == левые точки



Тип оснащения == правые точки



Тип оснащения == центральные точки



Тип оснащения == случайные точки

Примечание: из-за особенностей рендеринга в matplotlib на больших n изображение отображается странно. Для разглядывания деталей рекомендую запустить программу локально

## Программа

Текст программы, а также инструкции могут быть найдены по адресу [https://github.com/evjeny/matan\\_integration](https://github.com/evjeny/matan_integration)

## Примечание

Для корректной работы программы должен быть установлен Python  $\geq 3.6$ , а также библиотеки matplotlib и numpy. Для создания окружения я рекомендую использовать Anaconda (<https://docs.anaconda.com/anaconda/install/>), создать окружение можно следующей командой:

```
conda create -n integrator python=3.8 matplotlib numpy
```

```

import argparse
import numpy as np
import matplotlib.pyplot as plt

def left_choicer(a, b):
    return a

def right_choicer(a, b):
    return b

def mid_choicer(a, b):
    return (a + b) / 2

def random_choicer(a, b):
    return np.random.uniform(a, b)

def integrate(f, x_left, x_right, n_points, choicer, save_to=None):
    borders = np.linspace(x_left, x_right, n_points + 1)
    # segments in format [(begin_1, end_1), ..., (begin_n, end_n)]
    segments = np.hstack([
        borders[:-1].reshape(-1, 1),
        borders[1:].reshape(-1, 1)
    ])
    # choosed points in format [x_1, ... x_n]
    dots = np.apply_along_axis(lambda row: choicer(row[0], row[1]), 1, segments)
    # [f(x) for x in dots]
    ys = f(dots)

    # sum of y_i * dx_i
    integral_sum = np.sum(ys * (segments[:, 1] - segments[:, 0]))

    fig, ax = plt.subplots(1, figsize=(15, 8))
    ax.bar(segments.mean(axis=1), ys, width=10/n_points)
    ax.set_title(f"Integral sum: {integral_sum}")

    if save_to:
        plt.savefig(save_to)
    else:
        plt.show()

def main():
    parser = argparse.ArgumentParser(description="Integrator by evjeny. Integrates f(x)=sin(x) on [0, 4Pi]")
    parser.add_argument("-n", type=int, default=100, help="Number of points to split the interval")
    parser.add_argument("-e", type=str, default="mid", help="Type of equipment, must be one of: left, right, mid, random")
    parser.add_argument("--save-to", type=str, default=None, help="Path to save the plot")
    args = parser.parse_args()

    if args.n <= 0:
        print("n must be greater than 0!")
        return

    if args.e not in ["left", "right", "mid", "random"]:
        print("e must be one of: left, right, mid, random !")
        return

    choicers = {
        "left": left_choicer,
        "right": right_choicer,
        "mid": mid_choicer,
        "random": random_choicer
    }

    integrate(np.sin, 0, 4 * np.pi, args.n, choicers.get(args.e), args.save_to)

if __name__ == "__main__":
    main()

```