НИУ ИТМО
Факультет программной инженерии и компьютерных технологий

Отчет по лабораторной работе №3

по дисциплине Вычислительная математика

Студент группы № P32151 Шипулин Павел Андреевич

Преподаватель Машина Екатерина Алексеевна

Санкт-Петербург

2023

Цель работы

Найти приближенное значение определенного интеграла с требуемой точностью различными численными методами.

Вычислительная часть лабораторной работы

Интеграл, согласно варианту:

$$\int_1^3 (-2x^3 - 5x^2 + 7x - 13)\,dx = \left(-\frac{1}{2}x^4 - \frac{5}{3}x^3 + \frac{7}{2}x^2 - 13x\right)\Bigg|_1^3 = -\frac{244}{3}$$
$$= -81.333$$

По формуле Ньютона-Котеса для равноотстоящих узлов ($n = 5$):

$$\int_a^b f(x)\,dx \approx \sum_{k=0}^{n} f\left(a + \frac{k}{n}(b-a)\right) c_n^k$$

$$c_5^0 = c_5^5 = \frac{19(b-a)}{288} \approx 0{,}132$$

$$c_5^1 = c_5^4 = \frac{75(b-a)}{288} \approx 0{,}521$$

$$c_5^2 = c_5^3 = \frac{50(b-a)}{288} \approx 0{,}347$$

$$\int_1^3 (-2x^3 - 5x^2 + 7x - 13)\,dx \approx -81{,}333$$

$$\delta_1 = \left|\frac{-81{,}333 + 81{,}333}{-81{,}333}\right| \approx 0$$

По формуле средних прямоугольников ($n = 10$):

$$\int_a^b f(x)\,dx \approx h\sum_{k=1}^{n} f\left(a + h\left(k - \frac{1}{2}\right)\right), h = \frac{b-a}{n} = 0{,}2$$

$$\int_1^3 (-2x^3 - 5x^2 + 7x - 13)\,dx \approx 0.2[f(1.1) + f(1.3) + \ldots + f(2.9)] \approx -81.22$$

$$\delta_2 = \left|\frac{-81{,}22 + 81{,}333}{-81{,}333}\right| \approx 0{,}0014$$

По формуле трапеций ($n = 10$):

$$\int_a^b f(x)\,dx \approx \frac{h}{2}\sum_{k=1}^{n} f\big(a + h(k-1)\big) + f(a + hk)\,, h = \frac{b-a}{n}$$

$$\int_1^3 (-2x^3 - 5x^2 + 7x - 13)\,dx \approx 0.1[f(1) + f(1.2) + f(1.2) + \cdots + f(3)]$$

$$\approx -81{,}56$$

$$\delta_3 = \left|\frac{-81{,}56 + 81{,}333}{-81{,}333}\right| \approx 0{,}0028$$

По формуле Симпсона ($n = 10$):

$$\int_a^b f(x)\,dx \approx \frac{h}{3}\left[y_0 + 4\sum_{k=1}^{\frac{n}{2}} y_{2k-1} + 2\sum_{k=1}^{\frac{n}{2}-1} y_{2k} + y_n\right], y_k = f(a + hk), h = \frac{b-a}{n}$$

$$\int_1^3 (-2x^3 - 5x^2 + 7x - 13)\,dx \approx -81{,}333$$

$$\delta_4 = \left|\frac{-81{,}33 + 81{,}333}{-81{,}333}\right| \approx 0$$

Код численных методов

```
class AnyCompIntegrate:
    def __init__(self, name):
        self.name = name
        self.n_max = 2 ** 20

    def integrate(self, f, a, b, n, epsilon):
        pass

    def __str__(self):
        return self.name

    def runge_rule(self, s_n, s_half_n, p=2):
        return abs(s_n - s_half_n) / (2 ** p - 1)

    def set_n_max(self, n):
        self.n_max = n

    def get_n_max(self):
        return self.n_max

    def is_n_too_big(self, n):
        return n >= self.n_max

from Labs.Lab3.data.table import Table
from Labs.Lab3.integrals.AnyCompIntegrate import AnyCompIntegrate


class MiddleRectanglesMethod(AnyCompIntegrate):
    def __init__(self):
        super().__init__("Интегрирование методом средних прямоугольников")

    def integrate(self, f, a, b, n, epsilon):
        result = Table(head=["Номер шага", "n", "S_n", "|S_n*2 - S|"])
        if a > b:
            a, b = b, a

        steps = 1
        h = (b - a) / n
        x = [(a + h * i) for i in range(n + 1)]
        y = [f((x[i] + x[i + 1]) / 2) for i in range(n)]

        s_n = sum(y) * h
        runge_num = epsilon + 1
        result.add_row(row=[steps, n, s_n, "-"])

        while (runge_num > epsilon) and (not self.is_n_too_big(n)):
            steps += 1

            new_x = []
            for i in range(n):
                new_x.append(x[i])
                new_x.append((x[i] + x[i + 1]) / 2)
            new_x.append(x[-1])
            x = new_x

            n = len(x) - 1
            h = (b - a) / n
            y = [f((x[i] + x[i + 1]) / 2) for i in range(n)]

            s_half_n = s_n
```

```python
            s_n = sum(y) * h
            runge_num = self.runge_rule(s_n, s_half_n, p=2)
            result.set_cell(-1, 3, runge_num)
            result.add_row(row=[steps, n, s_n, "-"])

        return result


class LeftRectanglesMethod(AnyCompIntegrate):
    def __init__(self):
        super().__init__("Интегрирование методом левых прямоугольников")

    def integrate(self, f, a, b, n, epsilon):
        result = Table(head=["Номер шага", "n", "S_n", "|S_n*2 - S|"])
        if a > b:
            a, b = b, a

        steps = 1
        h = (b - a) / n
        x = [(a + h * i) for i in range(n + 1)]
        y = [f(x[i]) for i in range(n)]

        s_n = sum(y) * h
        runge_num = epsilon + 1
        result.add_row(row=[steps, n, s_n, "-"])

        while (runge_num > epsilon) and (not self.is_n_too_big(n)):
            steps += 1

            new_x = []
            for i in range(n):
                new_x.append(x[i])
                new_x.append((x[i] + x[i + 1]) / 2)
            new_x.append(x[-1])
            x = new_x

            n = len(x) - 1
            h = (b - a) / n
            y = [f(x[i]) for i in range(n)]

            s_half_n = s_n
            s_n = sum(y) * h
            runge_num = self.runge_rule(s_n, s_half_n, p=2)
            result.set_cell(-1, 3, runge_num)
            result.add_row(row=[steps, n, s_n, "-"])

        return result


class RightRectanglesMethod(AnyCompIntegrate):
    def __init__(self):
        super().__init__("Интегрирование методом правых прямоугольников")

    def integrate(self, f, a, b, n, epsilon):
        result = Table(head=["Номер шага", "n", "S_n", "|S_n/2 - S|"])
        if a > b:
            a, b = b, a

        steps = 1
        h = (b - a) / n
        x = [(a + h * (i + 1)) for i in range(n + 1)]
        y = [f(x[i]) for i in range(n)]

        s_n = sum(y) * h
```

```python
        runge_num = epsilon + 1
        result.add_row(row=[steps, n, s_n, "-"])

        while (runge_num > epsilon) and (not self.is_n_too_big(n)):
            steps += 1

            new_x = []
            for i in range(n):
                new_x.append(x[i])
                new_x.append((x[i] + x[i + 1]) / 2)
            new_x.append(x[-1])
            x = new_x

            n = len(x) - 1
            h = (b - a) / n
            y = [f(x[i + 1]) for i in range(n)]

            s_half_n = s_n
            s_n = sum(y) * h
            runge_num = self.runge_rule(s_n, s_half_n, p=2)
            result.set_cell(-1, 3, runge_num)
            result.add_row(row=[steps, n, s_n, "-"])

        return result

from Labs.Lab3.data.table import Table
from Labs.Lab3.integrals.AnyCompIntegrate import AnyCompIntegrate


class SimpsonMethod(AnyCompIntegrate):
    def __init__(self):
        super().__init__("Интегрирование методом Симпсона")

    def integrate(self, f, a, b, n, epsilon):
        result = Table(head=["Номер шага", "n", "S_n", "|S_n*2 - S|"])
        if a > b:
            a, b = b, a

        steps = 1
        h = (b - a) / n
        x = [(a + h * i) for i in range(n + 1)]
        y = [f(x[i]) for i in range(n + 1)]

        s_n = (y[0]
               + sum([y[i * 2 - 1] * 4 for i in range(1, n // 2)])
               + sum([y[i * 2] * 2 for i in range(1, n // 2)])
               + y[-1]) * h / 3
        runge_num = epsilon + 1
        result.add_row(row=[steps, n, s_n, "-"])

        while (runge_num > epsilon) and (not self.is_n_too_big(n)):
            steps += 1

            new_x = []
            for i in range(n):
                new_x.append(x[i])
                new_x.append((x[i] + x[i + 1]) / 2)
            new_x.append(x[-1])
            x = new_x

            n = len(x) - 1
            h = (b - a) / n
            y = [f(x[i]) for i in range(n + 1)]
```

```
            s_half_n = s_n
            s_n = (y[0]
                   + sum([y[i * 2 - 1] * 4 for i in range(1, n // 2)])
                   + sum([y[i * 2] * 2 for i in range(1, n // 2)])
                   + y[-1]) * h / 3
            runge_num = self.runge_rule(s_n, s_half_n, p=2)
            result.set_cell(-1, 3, runge_num)
            result.add_row(row=[steps, n, s_n, "-"])

        return result

from Labs.Lab3.data.table import Table
from Labs.Lab3.integrals.AnyCompIntegrate import AnyCompIntegrate


class TrapezesMethod(AnyCompIntegrate):
    def __init__(self):
        super().__init__("Интегрирование методом трапеций")

    def integrate(self, f, a, b, n, epsilon):
        result = Table(head=["Номер шага", "n", "S_n", "|S_n*2 - S|"])
        if a > b:
            a, b = b, a

        steps = 1
        h = (b - a) / n
        x = [(a + h * i) for i in range(n + 1)]
        y = [f(x[i + 1]) for i in range(n - 1)]
        s_n = (sum(y) + (f(x[0]) + f(x[-1])) / 2) * h
        runge_num = epsilon + 1
        result.add_row(row=[steps, n, s_n, "-"])

        while (runge_num > epsilon) and (not self.is_n_too_big(n)):
            steps += 1

            new_x = []
            for i in range(n):
                new_x.append(x[i])
                new_x.append((x[i] + x[i + 1]) / 2)
            new_x.append(x[-1])
            x = new_x

            n = len(x) - 1
            h = (b - a) / n
            y = [f(x[i]) for i in range(1, n - 1)]

            s_half_n = s_n
            s_n = (sum(y) + (f(x[0]) + f(x[-1])) / 2) * h
            runge_num = self.runge_rule(s_n, s_half_n, p=2)
            result.set_cell(-1, 3, runge_num)
            result.add_row(row=[steps, n, s_n, "-"])

        return result
```

Результат выполнения программы

Пример 1

```
[Info]: Введите комманду:
```

```
lab3_good
[Input]: lab3_good
[Info]: Уравнения для исследования
[Info]:
+-------+---------------------------------------+
| Номер |                             Уравнение |
+-------+---------------------------------------+
|     1 |                          5*cos(x) + x |
+-------+---------------------------------------+
|     2 |       (e ** (-x ** 2)) / (pi ** 0.5) |
+-------+---------------------------------------+
|     3 | ln[(x*sin(x))**2 + cos(x)**2 - 0.5] |
+-------+---------------------------------------+

[Info]: Введите номер уравнения:
2
[Input]: 2
[Info]: первая граница интервала:
-1000
[Input]: -1000
[Info]: вторая граница интервала
1000
[Input]: 1000
[Info]: Введите точность для правила Рунге:
0.001
[Input]: 0.001
```

[Info]: Методы для решения

[Info]:

Номер	Метод
1	Интегрирование методом левых прямоугольников
2	Интегрирование методом средних прямоугольников
3	Интегрирование методом правых прямоугольников
4	Интегрирование методом трапеций
5	Интегрирование методом Симпсона

[Info]: Введите номер метода

4

[Input]: 4

[Info]: Результат работы метода. Интегрирование методом трапеций

Номер шага	n	S_n	|S_n*2 - S|
1	4	282.095	47.016

2	8	141.047	23.508
3	16	70.524	11.754
4	32	35.262	5.877
5	64	17.631	2.938
6	128	8.815	1.469
7	256	4.408	0.735
8	512	2.204	0.351
9	1024	1.151	0.050
10	2048	1.000	0.000
11	4096	1.000	-

[Info]: Итоговое значение определенного интеграла: 1.0 +- 2.1349218726059622e-05

[Info]: Лабораторная работа 3, вычисление собственного интеграла - завершена

Пример 2

```
F:\Programming\python\CalcMath\venv\Scripts\python.exe
F:/Programming/python/CalcMath/Labs/Lab3/main.py

[Info]: Введите комманду:

lab3_bad

[Input]: lab3_bad

[Info]: Уравнения для исследования

[Info]:

+-------+-----------------------+

| Номер |             Уравнение |

+-------+-----------------------+

|     1 |             |ln(|x|)| |

+-------+-----------------------+

|     2 |            sin(x) / x |

+-------+-----------------------+

|     3 | 1 / ((x + 1) * (x - 2)) |

+-------+-----------------------+


[Info]: Введите номер уравнения:

3

[Input]: 3

[Info]: Введите смещение от точек разрыва

0.001

[Input]: 0.001

[Info]: первая граница интервала:

-2
```

[Input]: -2

[Info]: вторая граница интервала

0

[Input]: 0

[Info]: Точка разрыва: -1

[Info]: На интервале 1 точек разрыва. Аналитически, интеграл может расходиться

[Info]: Введите точность для правила Рунге:

0.001

[Input]: 0.001

[Info]: Методы для решения

[Info]:

Номер	Метод
1	Интегрирование методом левых прямоугольников
2	Интегрирование методом средних прямоугольников
3	Интегрирование методом правых прямоугольников
4	Интегрирование методом трапеций
5	Интегрирование методом Симпсона

[Info]: Введите номер метода

2

[Input]: 2

[Info]: Результат работы метода на интервале: [-2.0; -1.001]. Интегрирование методом средних прямоугольников

Номер шага	n	S_n	|S_n*2 - S|
1	4	1.016	0.075
2	8	1.239	0.073
3	16	1.458	0.069
4	32	1.664	0.062
5	64	1.850	0.051
6	128	2.003	0.036
7	256	2.110	0.020
8	512	2.170	0.009
9	1024	2.195	0.003

10	2048	2.204	0.001
11	4096	2.206	-

[Info]: Результат работы метода на интервале: [-0.999; 0.0]. Интегрирование методом средних прямоугольников

Номер шага	n	S_n	|S_n*2 - S|
1	4	-1.246	0.075
2	8	-1.470	0.073
3	16	-1.688	0.069
4	32	-1.895	0.062
5	64	-2.081	0.051
6	128	-2.233	0.036
7	256	-2.340	0.020

8	512	-2.400	0.009
9	1024	-2.426	0.003
10	2048	-2.434	0.001
11	4096	-2.437	-

[Info]: Итоговое значение определенного интеграла: -0.2308268378013718 +- 0.0015755744772691926

[Info]: Лабораторная работа 3, вычисление несобственного интеграла в смысле главного значения – завершена

Выводы

Изучил численные методы интегрирования.