Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ *Робототехники и комплексной автоматизации*

КАФЕДРА *Системы автоматизированного проектирования (РК-6)*

ОТЧЕТ О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ

по дисциплине: «Вычислительная математика»

Студент	Петраков Станислав Альбертович
Группа	РК6-56Б
Тип задания	лабораторная работа
Тема лабораторной работы	Использование аппроксимаций для численной оптимизации

Студент _____ **Петраков С.А.**
подпись, дата *фамилия, и.о.*

Преподаватель _____ **Соколов А.П.**
подпись, дата *фамилия, и.о.*

Москва, 2021 г.

Оглавление

Задание на лабораторную работу

Методы аппроксимации, такие как интерполяция и численное интегрирование, часто используются как составные блоки других, более сложных численных методов. В данной лабораторной работе мы рассмотрим одну из старейших задач вариационного исчисления: задачу о брахистохроне, т.е. задачу о кривой наискорейшего спуска. Она состоит в нахождении такой кривой, по которой материальная точка из точки $(x, y) = (0,0)$ достигнет точки $(x, y) = (a, y_a)$ под действием силы тяжести за наименьшее время (здесь и далее ось y направлена вниз). Решением этой задачи является такая кривая $y(x)$, которая минимизирует следующий функционал, являющийся полным временем движения материальной точки:

$$F[y] = \int_0^a \sqrt{\frac{1 + (y'(x))^2}{2gy(x)}}\,dx, \quad (1)$$

где g обозначает ускорение свободного падение, и $y'(x) = \frac{dy}{dx}$. Эта задача имеет аналитическое решение, которым является параметрически заданная циклоида:

$$\begin{bmatrix} x(t) \\ y(t) \end{bmatrix} = C \begin{bmatrix} t - \frac{1}{2}\sin(2t) \\ \frac{1}{2} - \frac{1}{2}\cos(2t) \end{bmatrix}, \quad (2)$$

где $t \in [0;\ T]$ и C, T являются константами, значения которых находятся из граничного условия. В базовой части требуется воспользоваться численным интегрированием для нахождения полного времени движения материальной точки по кривой наискорейшего спуска. В продвинутой части требуется разработать метод для нахождения аппроксимации этой кривой. Здесь и далее принимается $a = 2$ и $y_a = 1$. Константы циклоиды для этого граничного условия равны $C = 1.03439984, T = 1.75418438$.

Требуется (базовая часть):

1. Написать функцию *composite_simpson(a, b, n, f)*, численного интегрирования функции f на интервале $[a, b]$ по n узлам с помощью составной формулы Симпсона.

2. Написать функцию *composite_trapezoid (a, b, n, f)*, численного интегрирования функции f на интервале $[a, b]$ по n узлам с помощью составной формулы трапеций.

3. Рассчитать интеграл 1 для функции $y(x)$, соответствующей кривой наискорейшего спуска, с помощью составной формулы Симпсона и со ставной формулы трапеций для множества значений $n \in [3; 9999]$. Постройте log-log график зависимости абсолютной погрешности численного интегрирования от шага интегрирования для обеих формул.

4. Объясните, каким образом по полученному графику можно определить порядок точности формулы.

5. Для обоих формул сравните порядок, полученный с помощью графика, с аналитическим порядком точности.

6. Существует ли оптимальный шаг интегрирования для данной формулы, минимизирующий достижимую погрешность? Обоснуйте свой ответ

Цель выполнения лабораторной работы

Цель выполнения лабораторной работы – написать алгоритмы для составных формул Симпсона и трапеций, попытаться определить порядок точности формулы, оценить порядок, полученный с помощью графика, с аналитическим порядком точности.

Выполненные задачи

1. Разработан алгоритм вычисления интеграла с помощью составной формулы Симпсона
2. Разработан алгоритм вычисления интеграла с помощью составной формулы трапеции

3. Вычислен интеграл для функции с помощью составных формул Симпсона и трапеций
4. Построен log-log график зависимости абсолютной погрешности численного интегрирования от шага интегрирования для обеих формул
5. Определён порядок точности формулы. Проведён анализ порядка точности, полученного с помощью графика и аналитически

1. Разработан алгоритм вычисления интеграла с помощью составной формулы Симпсона

Из курса лекций известно, что составная формула Симпсона имеет вид:

$$\int_a^b f(x)dx = \frac{h}{3}\left(f(x_1) + 2\sum_{i=1}^{\frac{n}{2}-1} f(x_{2i+1}) + 4\sum_{i=1}^{\frac{n}{2}} f(x_{2i}) + f(x_{n+1}) \right)$$

$$-\frac{(b-a)h^4}{180} f^{(4)}(\xi), \quad (3)$$

Где $x_i = a + (i-1)h$, $h = (b-a)/n$ и $i = 1, \dots, n+1$, где n – четное число. При этом существует такое $\xi \in (a; b)$, то для $f(x) \in C^4[a; b]$.

Для вычислений интеграла 3 опускаем остаточный член, так как он является малым. Он используется в основном для выбора численного метода и его предварительного анализа.

Реализуем вычисление интеграла на интервале [0.001; 2]. Берём интервал не с точки 0, так как в точке 0 имеется разрыв. Для нашей задачи в 3 $f(x)$ является по условию функционалом 1. Вычисление 1 рассмотрено в пункте 3.

Листинг 1 – функция вычисления интеграла с помощью составной формулы Симпсона

```
def compositeSimpson(a: float, b: float, n: int, func) -> float:
    if n % 2 != 0:
        n = n + 1

    x = np.linspace(a, b, n + 1)
    h: float = abs(a - b) / 2

    result: float = func(x[0])
    for xIn in x[2:-1:2]:
        result = result + 2 * func(xIn)
    for xIn in x[1::2]:
        result = result + 4 * func(xIn)
    result = result + func(x[-1])
    result = result * h / 3.
    return result
```

2. Разработан алгоритм вычисления интеграла с помощью составной формулы трапеции

Из курса лекций известно, что составная формула трапеций имеет вид:

$$\int_a^b f(x)dx = \frac{h}{2}\left(f(x_1) + 2\sum_{i=2}^{n} f(x_i) + f(x_{n+1})\right) - \frac{(b-a)h^2}{180} f''(\xi). \quad (4)$$

Где $x_i = a + (i-1)h$, $h = \frac{(b-a)}{n}$ и $i = 1, \dots, n+1$, где $n \in N$. При этом существует такое $\xi \in (a; b)$, то для $f(x) \in C^2[a; b]$.

Для вычислений интеграла 4 опустим остаточный член, так как он является малым. Он используется в основном для выбора численного метода и его предварительного анализа.

Реализуем вычисление интеграла на интервале [0.001; 2]. Берём интервал не с 0, так как в точке 0 имеется разрыв. Для нашей задачи в 4 $f(x)$ является по условию функционалом 1. Вычисление 1 рассмотрено в пункте 3.

Листинг 2 – функция вычисления интеграла с помощью составной формулы трапеций

```
def compositeTrapezoid(a: float, b: float, n: int, func) -> float:
    if n % 2 != 0:
        n = n + 1

    x = np.linspace(a, b, n + 1)
    h: float = abs(a - b) / 2

    result: float = func(x[0])
    for xIn in x[1:-1:1]:
        result = result + 2 * func(xIn)
    result = result + func(x[-1])
    result = result * h / 2.

    return result
```

3. Вычислен интеграл для функции с помощью составных формул Симпсона и трапеций

Для вычисления интеграла с помощью составных формул Симпсона и трапеций для начала нужно понять, как вычислять функционал 1.

Для начала поймём, как вычислять функцию $y(x)$. Для этого импортируем функцию *fsolve* из библиотеки *scipy.optimize*. Функция *fsolve(func, x0, args)* принимает аргументы: *func* – функция, где ищется корень *func(x) = 0*; *args* – дополнительный аргумент. Будем находить конкретное значение t при заданном x и

подставлять его в $y(t)$ в 2. Таким образом находя для каждого значения x соответствующее значение t, найдём $y(t)$.

Листинг 3 – функция для вычисления $y(x)$

```
def yFromX(x: float) -> float:
    def xFromTForRoot(t: float, x: float) -> float:
        C: float = 1.03439984
        return C * (t - 0.5 * np.sin(2 * t)) - x

    C: float = 1.03439984
    t = scipy.optimize.fsolve(func=xFromTForRoot, x0=0, args=x)
    return C * (0.5 - 0.5 * np.cos(2 * t))
```

Для вычисления производной $y(x)$ воспользуемся двумя методами, чтобы понимать в дальнейшем, как лучше и точнее вычислять интеграл.

Для первого метода импортируем функцию *derivative* из библиотеки *scipy.misc*. Функция *scipy.misc.derivative(func, x0, dx, n)* принимает аргументы: *func* – функция, производная которой ищется; *x0* – точка, в которой ищется производная; *dx* – шаг дифференцирования; *n* – порядок производной, по умолчанию равна 1. С помощью этой библиотеки берём производную от $y(x)$.

Листинг 4 – функция для вычисления $y'(x)$ с помощью функции *derivative*

```
def FunctionalSciPy(x: float) -> float:
    g: float = 9.80665
    return np.sqrt((1 + derivative(func=yFromX, x0=x, n=1, dx=0.0000001) ** 2) / (2 * g *
yFromX(x)))
```

Таким образом находя для каждого значения x соответствующее значение t, найдём $y(t)$. Далее с помощью функции *derivative* найдём её производную. Тем самым вычислим функционал 1, подставляя соответствующие точки в 3 и 4.

Для второго метода воспользуемся формулой:

$$y'(x) = \frac{y'(t)}{x'(t)}. \quad (5)$$

Так как мы знаем зависимости $y(t)$ и $x(t)$, найдём их производные и вычислим $y'(x)$ по формуле 5. Найдя $y(x)$ с помощью функции *fsolve* из библиотеки *scipy.optimize*, указанной выше, вычисляем интеграл для функционала 1,

подставляя соответствующие точки в 3 и 4. То есть для каждой точки x на интервале $[0.01;\ 2]$ вычисляем значение функционала 1.

Листинг 5 – функция для вычисления $y'(x)$ по формуле 5

```
4:  def deritativeParametr(x: float) -> float:
5:      def xFromTForRoot(t: float, x: float) -> float:
6:          C: float = 1.03439984
7:          return C * (t - 0.5 * np.sin(2 * t)) - x
8:
9:      C: float = 1.03439984
10:     t: float = scipy.optimize.fsolve(func=xFromTForRoot, x0=0, args=x)
11:
12:     xt: float = C * (1 - np.cos(2 * t))
13:     yt: float = C * np.sin(2 * t)
14:
15:     return yt / xt
```

Сравнивая методы 1 и 2 для вычисления производной, приходим к выводу, что конечные ответы не отличаются друг от друга.

4. Построен log-log график зависимости абсолютной погрешности численного интегрирования от шага интегрирования для обеих формул

Шаг интегрирования одинаковый для обеих составных формул Симпсона и трапеций и имеет вид:

$$h = \frac{b-a}{n}.$$

Где n – количество точек на этом интервале, а a и b – границы интервала интегрирования.

Из курса лекций известно, что абсолютная погрешность интегрирования для составной формулы Симпсона считается как:

$$E = \frac{(b-a)h^4}{180}.$$

Из курса лекций известно, что абсолютная погрешность интегрирования для составной формулы трапеций считается как:

$$E = \frac{(b-a)h^2}{12}.$$

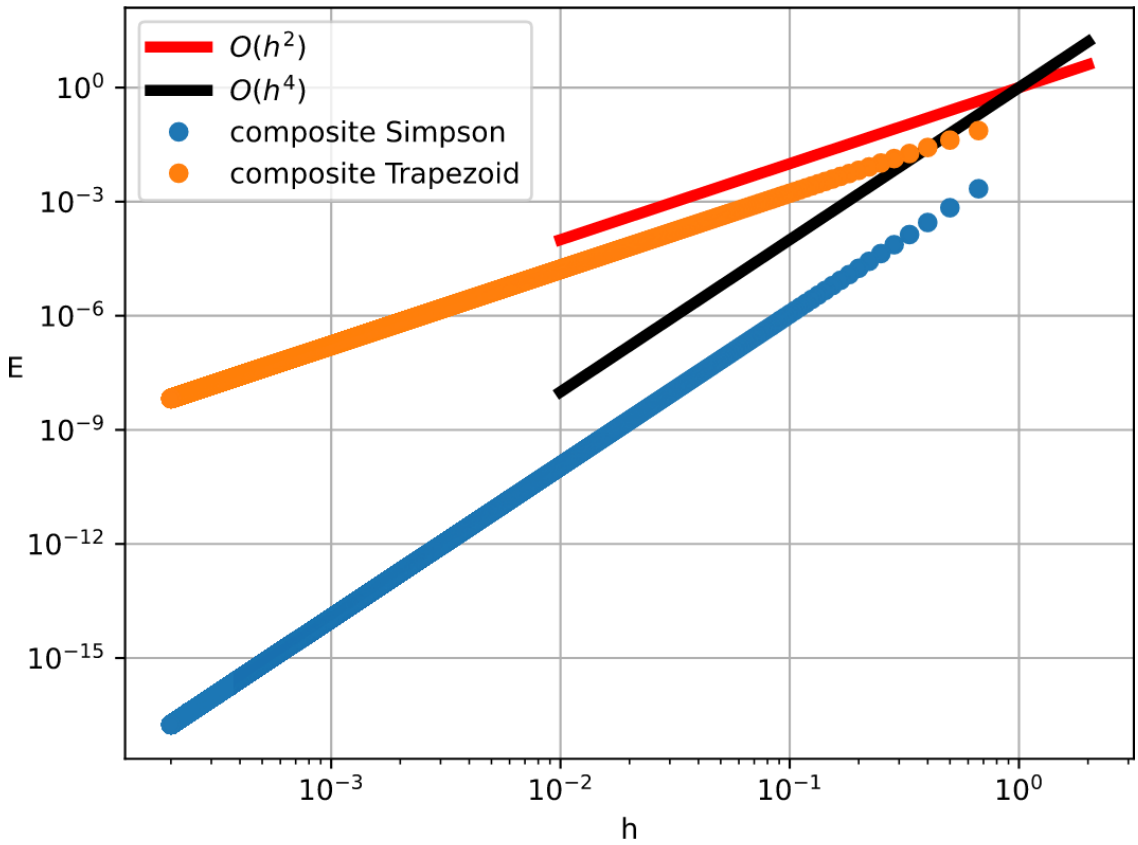

Рис. 1 – Абсолютная погрешность численного интегрирования от шага интегрирования

5. Определён порядок точности формулы. Проведён анализ порядка точности, полученного с помощью графика и аналитически

По графику, изображенному на рис. 1, видно, что абсолютная погрешность, вычисленная с помощью составной формулы Симпсона, пропорциональная $O(h^4)$. Это означает, что порядок точности составной формулы Симпсона, равняется h^4, что совпадает с аналитическим порядком точности.

Также по графику, изображенному на рис. 1, видно, что абсолютная погрешность, вычисленная с помощью составной формулы трапеций, пропорциональная $O(h^2)$. То есть порядок точности составной формулы трапеций равняется h^2, что также совпадает с аналитическим порядком точности.

Для оптимального шага интегрирования нужно взять маленький шаг интегрирования h.

Заключение

В лабораторной работе были реализованы составные формулы Симпсона и трапеций. Вычислены интегралы для функционала с помощью этих составных формул. По выведенному графику оказалось, что чем больше шаг интегрирования, тем выше погрешность вычислений приближенного значения интеграла. Если сделать шаг интегрирования меньше машинного эпсилон, то погрешность при вычислениях будет больше погрешности математической. То есть уменьшение шага меньше машинного эпсилон приведёт к резкому увеличению ошибки вычисления.

Список использованных источников

1. Першин А.Ю. Лекции по курсу «Вычислительная математика. Москва, 2018-2021, С. 140.