Министерство науки и высшего образования Российской Федерации
федеральное государственное бюджетное образовательное учреждение
высшего профессионального 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:	Кутепов Никита Сергеевич
Группа:	РК6-55Б
Тип задания:	лабораторная работа
Тема:	Использование аппроксимаций для численной оптимизации

Студент ____________ Кутепов Н. С.
подпись, дата Фамилия, И.О.

Преподаватель ____________ ____________
подпись, дата Фамилия, И.О.

Москва, 2021

Содержание

Использование аппроксимаций для численной оптимизации

Задание

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:

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

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

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

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 составной 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 поря-док 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, минизимирующий достижимую погрешность? Обоснуйте свой ответ.*

Требуется (продвинутая часть):

1. *Используя кусочно-линейную интерполяцию с равноудаленными узлами, преобразовать задачу о минимизации функционала 1 к полудискретной форме,где аргументами минимизации будут параметры кусочно-линейной интерполяции.*

2. *Далее, используя составную формулу Симпсона, преобразовать задачу к полностью дискретной форме.*

3. *Решить полученную задачу минимизации, используя различные конфигурации дискретизации: с шагом интерполяции и шагом интегрирования от* 10^{-3} *до 1.*

4. *Используя log-log графики и линии уровня, оценить зависимость погрешности решения от шага интерполяции и шага интегрирования.*

Цель выполнения лабораторной работы

Цель выполнения лабораторной работы – реализовать методы численного интегрирования, применить их для нахождения полного времени движения материальной точки по кривой наискоорейшего спуска. Разработать метод для нахождения аппроксимации кривой.

Поставленные задачи

1. Написать функцию composite_simpson(a, b, n, f) численного интегрирования функции f с помощью составной формулы Симпсона.

2. Написать функцию composite_trapezoid(a, b, n, f) численного интегрирования функции f с помощью составной формулы трапеций.

3. Построить log-log график зависимости абсолютной погрешности численного интегрирования от шага интегрирования для обеих формул.

4. Определить порядок точности формулы по графику. Для обеих формул сравнить порядок точности, полученный с помощью графика, с аналитическим.

5. Определить, существует ли оптимальный шаг интегрирования для данной формулы, минимизирующий погрешность.

6. Преобразовать задачу о минимизации функционала к полудискретной форме.

7. Используя составную формулу Симпсона, преобразовать задачу к полностью дискретной форме.

8. Решить полученную задачу минимизации, используя различные конфигурации дискретизации. Оценить зависимость погрешности решения от шага интерполяции и шага интегрирования.

1 Реализация функции composite_simpson(a, b, n, f) численного интегрирования с помощью составной формулы Симпсона.

Рассмотрим составную формулу Симпсона:

$$\int_a^b f(x)dx = \frac{h}{3}\left[f(x_1) + 2\sum_{i=1}^{n/2-1} f(x_{2i+1}) + 4\sum_{i=1}^{n/2} f(x_{2i}) + f(x_{n+1})\right] - \frac{(b-a)h^4}{180} f^{(4)}(\xi), \quad (3)$$

где $x_i = a + (i-1)h$, $h = \frac{b-a}{n}$, $i = 1, ..., n+1$, n - четное число. При этом будем считать остаточный член слишком малым, поэтому в программной реализации он не учитывается. Тогда итоговый вид составной формулы Симпсона будет выглядеть так:

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

Листинг 1. Функция численного интегрирования составной формулой Симпсона.

```
def composite_simpson(a, b, n, f):
    if n % 2 == 1:
        n -= 1
        f.pop(len(f) - 1)

    h = (b - a) / n
    len_f = len(f)
    odd_sum = 0
    even_sum = 0

    for i in range(1, len_f - 1):
        if (i + 1) % 2 == 0:
            even_sum += f[i]
        else:
            odd_sum += f[i]

    f_sim = (h / 3) * (f[0] + 2 * odd_sum + 4 * even_sum + f[len_f - 1])

    return f_sim
```

2 Реализация функции composite_trapezoid(a, b, n, f) численного интегрирования с помощью составной формулы трапеций.

Рассмотрим составную формулу трапеций:

$$\int_a^b f(x)dx = \frac{h}{2}\left[f(x_1) + 2\sum_{i=2}^{n} f(x_i) + f(x_{n+1})\right] - \frac{(b-a)h^2}{12}f''(\xi), \tag{5}$$

где $x_i = a + (i-1)h$, $h = \frac{b-a}{n}$, $i = 1, ..., n+1$, $n \in N$. При этом будем считать остаточный член слишком малым, поэтому в программной реализации он не учитывается. Тогда итоговый вид составной формулы трапеций будет выглядеть так:

$$\int_a^b f(x)dx = \frac{h}{2}\left[f(x_1) + 2\sum_{i=2}^{n} f(x_i) + f(x_{n+1})\right] \tag{6}$$

Листинг 2. Функция численного интегрирования составной формулой трапеций.

```
def composite_trapezoid(a, b, n, f):
    h = (b - a) / n
    len_f = len(f)
    sum = 0

    for i in range(1, len_f - 1):
        sum += f[i]

    f_sim = (h / 2) * (f[0] + 2 * sum + f[len_f - 1])

    return f_sim
```

3 Построение log-log графика зависимости абсолютной погрешности численного интегрирования от шага интегрирования

Введем определение абсолютной погрешности. Абсолютной погрешностью приближенного значения a называют величину $\Delta(a^*)$, которая определена как:

$$\Delta(a^*) = |a - a^*|, \tag{7}$$

где a - точное значение.

Для нахождения абсолютной погрешности посчитаем истинное значение интеграла (1):

$$F[y] = \int_0^a \sqrt{\frac{1 + (y'(x))^2}{2gy(x)}}dx \tag{8}$$

$$dx = \frac{1}{2}C[2dt - 2cos(2t)dt] = C[1 - cos(2t)]dt$$

$$dy = \frac{1}{2}C[0 - sin(2t)2dt] = Csin(2t)dt$$

$$y^{'}(x) = \frac{dy}{dx} = \frac{dy}{dt}\frac{dt}{dx} = \frac{Csint(2t)dt}{C[1 = cos(2t)]dt} = \frac{sin(2t)}{1 - cos(2t)}$$

$$F[y] = \frac{1}{\sqrt{2g}}\int_0^{t_a}\sqrt{\frac{1 + (\frac{sin(2t)}{1 - cos(2t)^2})}{\frac{1}{2}C[1 - cos(2t)]}}C[1 - cos(2t)]dt \tag{9}$$

$$F[y] = \frac{1}{\sqrt{2g}}\int_0^{t_a}\sqrt{\frac{1 - 2cos(2t) + cos^2(2t) + sin^2(2t)}{1 - cos(2t)}2Cdt} \tag{10}$$

$$= \frac{1}{\sqrt{2g}}\int_0^{t_a}\sqrt{\frac{2(1 - cos(2t))}{1 - cos(2t)}2Cdt} = \sqrt{\frac{2C}{g}}\int_o^{t_a} dt$$

Так как верхний предел интегрирования $t_a = T$, для g = 10:

$$F[y] = \sqrt{\frac{2C}{g}}T \approx 0,7978742693373146 \tag{11}$$

Для осуществления программной реализации перейдем к интегрированию по t, соответственно для вычисления значения интеграла по формулам Симпсона и трапеций используем (10). Тогда абсолютная погрешность будет вычислятся по формуле (7), где a - значение, полученное по формуле (11), a^* - значение, полученное по формуле Симпсона/трапеций.

Запишем формулу Симпсона, при этом: $x_i = a + (i - 1)h$, $h = \frac{b-a}{n}$, $i = 1, ..., n + 1$, где n - четное число:

$$\int_a^b f(x) = \frac{h}{3}\left[f(x_1) + 2\sum_{i=1}^{n/2-1} f(x_{2i+1}) + 4\sum_{i=1}^{n/2} f(x_{2i}) + f(x_{n+1}) + \frac{(b-a)h^4}{180}f^4(\xi)\right] \tag{12}$$

Запишем формулу трапеций, при этом: $x_i = a + (i - 1)h$, $h = \frac{b-a}{n}$, $i = 1, ..., n + 1$, где $n \in N$:

$$\int_a^b f(x) = \frac{h}{2}\left[f(x_1) + 2\sum_{i=2}^{n/2-1} f(x_i) + f(x_{n+1}) + \frac{(b-a)h^2}{12}f''(\xi)\right] \tag{13}$$

Пусть $F[y] = \frac{1}{\sqrt{2g}}\sqrt{\frac{1 + (\frac{sin(2t)}{1 - cos(2t)^2})}{\frac{1}{2}C[1 - cos(2t)]}}C[1 - cos(2t)]$ интегрируемая функция, рассмотрим пределы интегрирования. Так как функция имеет разрыв на отрезке [0;2] в x = 0, допустим, что нижняя граница интегрирования равна 1e-7, тогда истинное значение интеграла (1) будет вычисляться следующим образом:

$$F[y] = \sqrt{\frac{2C}{g}}(T - 10^{-7}) \tag{14}$$

Перейдем к программной реализации, сгенерируем значения t, при этом a = 1e-7, b = T, количество узлов $n \in [3; 9999]$ создадим с шагом 100, программно вычислим истинное значение интеграла. Каждую итерацию вычисляем абсолютную погрешность и шаг интегрирования, после чего строим точки, где координата x - шаг интегрирования, а y - абсолютная погрешность.

Листинг 3. Построение log-log графика.

```
n = [i for i in range(3, 10000, 100)]
  f_real = get_real(a)

  for item in n:
      h = (b - a) / item
      y = abs(f_real - composite_simpson(a, b, item, create_f(a, b, item)))
      plt.scatter(h, y, color="darkturquoise", s=7, label="simpson")
      y = abs(f_real - composite_trapezoid(a, b, item, create_f(a, b, item)))
      plt.scatter(h, y, color="yellowgreen", s=7, label="trapezoid")
```

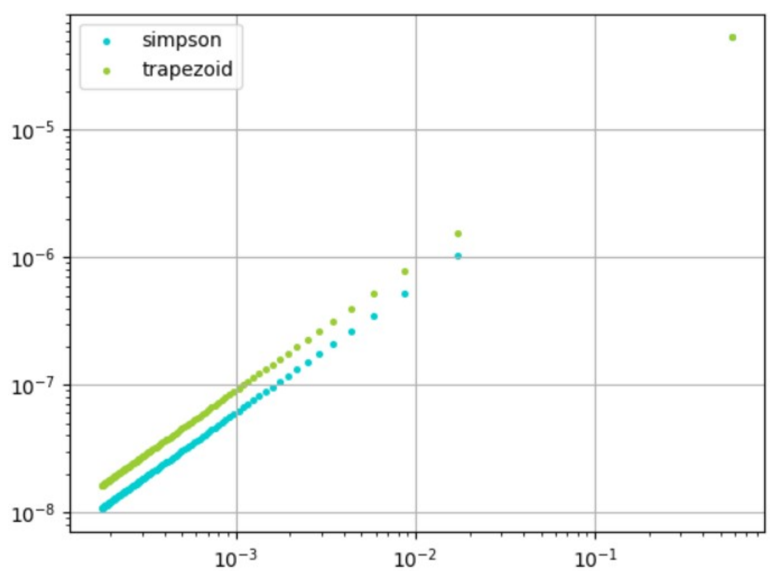

Рис. 1. log-log график зависимости абсолютной погрешности численного интегрирования от шага интегрирования.

4 Определение порядка точности формулы по графику. Сравнение порядка точности, полученного с помощью графика, с аналитическим.

Дадим определение порядка точности. Порядок точности - наибольшая степень полинома, для которой численный метод даёт точное решение задачи. Иными словами, говорят, что численный метод имеет порядок точности d, если его остаток R_n равен нулю для любого полинома степени d, но не равен нулю для полинома степени d + 1. Обозначим порядок точности как $O(h)$. В рамках лабороторной работы порядок точности представляет зависимость точности от величины шага. На рис. 5 представлен log-log график, поэтому порядок точности определяется как тангенс угла наклона графика, так как при логарифмировании степень точности численного метода становится тангенсом угла наклона.

$$tg(\alpha) = \frac{lg10^{-6} - lg10^{-7}}{lg10^{-2} - lg10^{-3}} = 1 \tag{15}$$

Заметим, что график для формулы Симпсона параллен графику для формулы трапеций, соответственно углы равны.

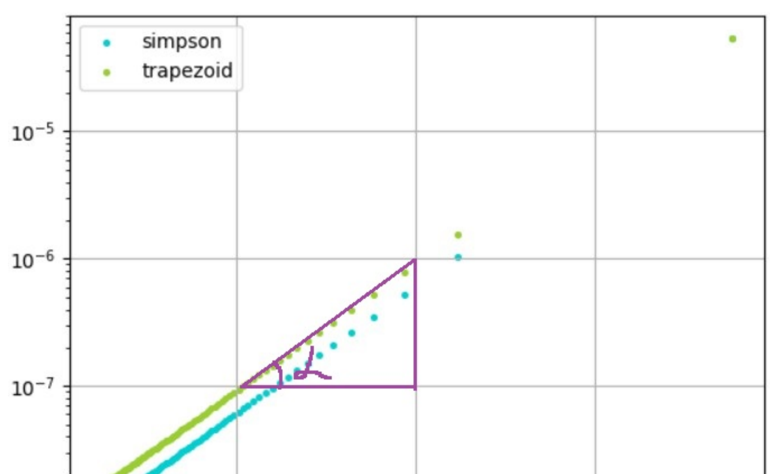

Рис. 2. Определение порядка точности как тангенс угла наклона.

Получаем O(h) = 1 для формулы Симпсона и для формулы трапеций, заметим, что аналитический порядок точности для формулы Симпсона равен 4, для формулы трапеций - 2(определяется как степень остаточного члена). Такое различие объясняется недостаточной гладкостью функции. Введем определение гладкости функции. Гладкая функция — это функция, имеющая непрерывную производную на всем множестве определения. Рассматривают также гладкие функции высших порядков, а именно, функция с порядком гладкости r имеет непрерывную производную порядка r. Первая производная функции $F[y] = \frac{1}{\sqrt{2g}}\sqrt{\frac{1+(\frac{sin(2t)}{1-cos(2t)^2})}{\frac{1}{2}C[1-cos(2t)]}}C[1-cos(2t)]$ равна:

$$F^{'}[y] = \frac{2cos(2x)(1-cos(2x)) - 4cos(2x)sin^2(2x)}{\sqrt(2)\sqrt(g)(1-cos^2(2x))^2\sqrt(\frac{sin(2x)}{1-cos^2(2x)} - 1)} \tag{16}$$

Построим график производной(16), используя программную реализацию на языке программирования Python. В данном случае реализованы две функции: нахождение значения производной в заданной точке и функция построения графика, которая итеративно находит значения производной в точках, сгенерированных с помощью np.linspace().

Листинг 4. Построение графика производной(16).

```
def diff_f(x):
    g = 10
    try:
        nom = 2 * np.cos(2 * x) * (1 - np.cos(2 * x) ** 2) - 4 * np.cos(2 * x) * np.sin(2 *
            x) ** 2
        dom = np.sqrt(2) * np.sqrt(g) * ((1 - np.cos(2 * x) ** 2) ** 2) * np.sqrt(np.sin(2 *
            x) / (1 - np.cos(2 * x) ** 2) + 1)
        f = nom / dom
        if f > 40 or f < -40:
            raise ValueError
    except ValueError:
        print("ERROR")
        return None

    return f


def plot_diff_f():
    x = np.linspace(-4, 4, 5000)
    y = []
    for item in x:
        y.append(diff_f(item))

    plt.plot(x, y, color="navy", label="diff f")
    plt.legend()
    plt.grid(True)
    plt.show()
```

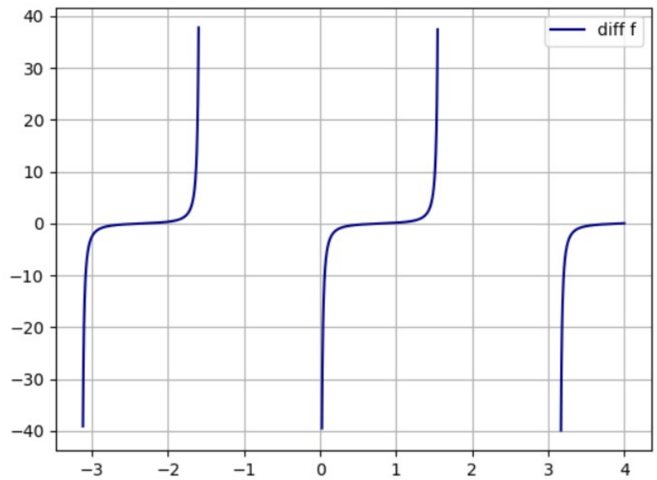

Рис. 3. График производной(16).

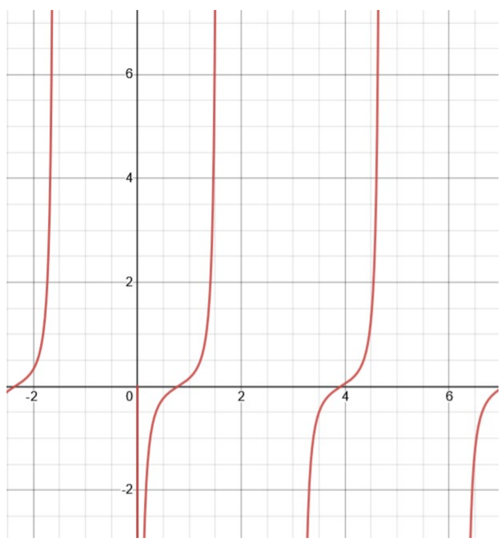

Рис. 4. График производной(16), построенный в Desmos.

Заметим, что функция терпит разрыв, в точках $x = 0 \pm i\Pi$, где $i = 0, 1, \ldots$, соответственно функция(16) не является непрерывной на участке [0;2], поэтому можно утверждать, что $F[y] = \frac{1}{\sqrt{2g}} \sqrt{\frac{1+(\frac{sin(2t)}{1-cos(2t)^2})}{\frac{1}{2}C[1-cos(2t)]}} C[1 - cos(2t)]$ недостаточно гладкая, отсюда такая разница в порядках точности. Именно разрыв функции поспособствовал изменению нижней границы интегрирования при работе с численными методами.

5 Определение оптимального шага интегрирования для заданной формулы

По сравнению с дифференцированием, операция интегрирования способна к стабилизации вычислительной погрешности. Интуитивное объяснение этого эффект заключается в том, что интегрирование предполагает суммирование близких значений, в то время как дифференцирование вычисляет их разность.

Рассмотрим составную формулу Симпсона и предположим, что значение f(x) в точке x_i вычисляется с погрешностью округления e_i:

$$f(x_i) = \tilde{f}(x_i) + e_i, i = 1, ..., n + 1 \tag{17}$$

Тогда полная погрешность округления, накапливаемая составной формулой Симпсона, может быть оценена следующим образом:

$$e(h) = \frac{h}{3}\left[e_1 + 2\sum_{i=1}^{n/2-1} e_{2i+1} + 4\sum_{i=1}^{n/2} e_{2i} + e_{n+1}\right] \le \frac{h}{e}\left[|e_1| + 2\sum_{i=1}^{n/2-1} |e_{2i+1}| + 4\sum_{i=1}^{n/2} |e_{2i}| + |e_{n+1}|\right]. \tag{18}$$

Предположим, что погрешность округления ограничена, например, машинным эпсилон: $|e_i| \le \epsilon$, $i = 1, ..., n + 1$. Тогда полная погрешность погрешность оценивается как:

$$e(h) \le \frac{h\epsilon}{3}\left[1 + 2(\frac{n}{2} - 1) + 4\frac{n}{2} + 1\right] = nh\epsilon = (b - a)\epsilon. \tag{19}$$

Этот результат показывает, что верхняя грань для накопленной погрешности округления не зависит от n или h, что означает, что увеличение числа подотрезков не приводит к дестабилизации полной погрешности. Действительно, можно заметить, что полная погрешность интегрирования падает до тех пор, пока она не достигнет значения, сравнимого с машинным эпсилон, после чего уменьшение погрешности становится невозможным, и она стабилизируется на уровне машинного эпсилон.

6 Преобразование задачи о минимизации функционала к полудискретной форме

Рассмотрим функционал:

$$F[y] = \int_0^a \sqrt{\frac{1 + (y'^{(x)})^2}{2gy(x)}} dx \tag{20}$$

Задача минимизации состоит в нахождении:

$$y^*(x) = argminF[y] \tag{21}$$

Осуществим кусочно-линейную интерполяцию, т.е. график интерполированной функции будет представлен линейной комбинацией прямых, уравнения которых заданы для каждого участка на интервале [0;a].

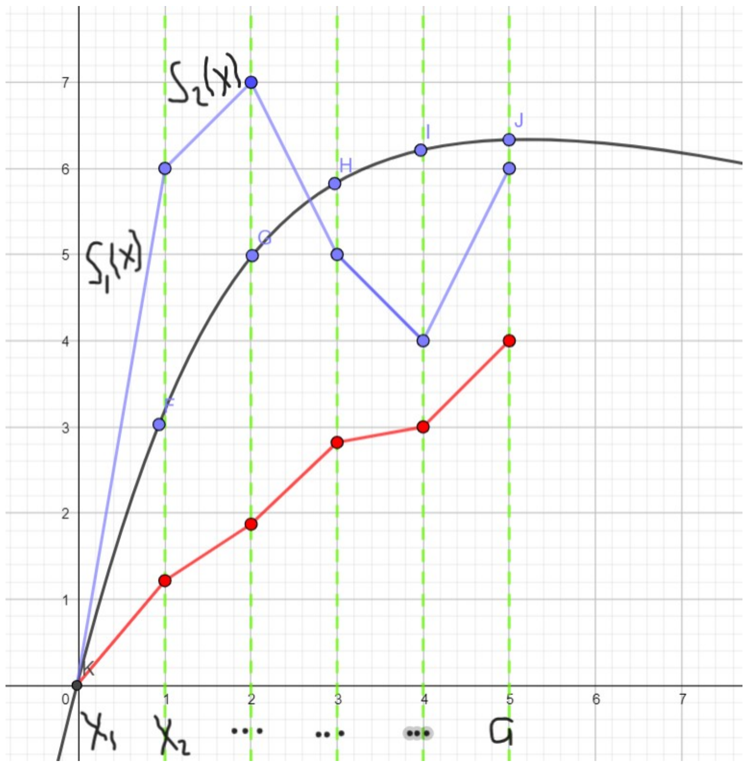

Рис. 5. Кусочно-линейная интерполяция функции(20).

Тогда, учитывая, что $y(x) = \sum_{i=1}^{n-1} S_i(x) = \sum_{i=1}^{n-1} \alpha_i + \beta_i x$, функционал будет представлен в виде суммы интегралов:

$$y^{'(x)=\beta_i}$$

$$F[y] = \sum_{i=1}^{n-1} \sqrt{\frac{1+\beta_i^2}{2g}} \int_{x_i}^{x_{i+1}} \frac{dx}{\sqrt{\alpha_i + \beta_i x}} = H(\alpha, \beta)$$

Для решения данной задачи необходимо найти α_i и β_i, рассмотрим задачу минимизации при нахождении коэффициентов прямой, тогда задача интерполяции будет "вложена"в задачу оптимизации. Найдем производные $H(\alpha, \beta)$ по α_i и β_i, приравняем их к нулю, после чего выражаем соответствующие коэффициенты. Следует учесть граничные условие, т.к. именно на граничных узлах заданы значения соответствующих ординат. Также запишем уравнение для сопряжения соседних узлов в итоговую СЛАУ.

$$\begin{cases} \frac{\delta H}{\delta \alpha_i} = 0, \\ \frac{\delta H}{\delta \beta_i} = 0 \end{cases}$$

$$\frac{\delta H}{\delta \alpha_j} = \sum_{j=1}^{n-1} \sqrt{\frac{1+\beta_i^2}{2g}} \int_{x_i}^{x_{i+1}} \frac{\delta}{\delta \alpha_j} \frac{dx}{\sqrt{\alpha_i + \beta_i x}} = \sqrt{\frac{1+\beta_j^2}{2g}} \int_{x_j}^{x_{j+1}} (\alpha_j + \beta_j x)^{-\frac{3}{2}} dx$$

$$\frac{\delta H}{\delta \beta_j} = \sum_{i=1}^{n-1} \int_{x_i}^{x_{i+1}} \frac{\delta}{\delta \beta_j} \left[\left(\frac{1+\beta_i^2}{2g(\alpha_u + \beta_i x)} \right)^2 \right] = \int_{x_j}^{x_{j+1}} \left(\frac{\alpha_j + \beta_j x}{2g(1+\beta_j^2)} \right)^2 \frac{\beta + j\alpha_j - x}{2(\alpha_j + \beta_j x)^2} dx$$

$$= \int_{x_j}^{x_{j+1}} \left(\frac{\alpha_j + \beta_j x}{2g(1+\beta_j^2)} \right)^{\frac{1}{2}} \frac{\beta_j \alpha_j - x}{2(\alpha_j + \beta_j x^2)} dx$$

Тогда итоговая система, содержащая граничные условия и условия сопряжения соседних узлов будет выглядеть следующим образом:

$$\begin{cases} \sqrt{\frac{1+\beta_j^2}{2g}} \int_{x_j}^{x_{j+1}} (\alpha_j + \beta_j x)^{-\frac{3}{2}} dx = 0, j = 1, 2, ..., n-1 \\ \int_{x_j}^{x_{j+1}} \left(\frac{\alpha_j + \beta_j x}{2g(1+\beta_j^2)} \right)^{\frac{1}{2}} \frac{\beta_j \alpha_j - x}{2(\alpha_j + \beta_j x^2)} dx = 0 \\ S_j(x_{j+1}) = S_{j+1}(x_{j+1}), j = 1, .., n-2 \\ S_1(0) = 0 \\ S_{n-1}(a) = y_a \end{cases} \tag{22}$$

7 Преобразование задачи к полностью дискретной форме с помощью составной формулы Симпсона

Решение системы уравенений(22) позволит перейти в полудискретной форме, однако нахождение коэффициентов прямых довольно нетривиальная задача, выразить их "в лоб"невозможно, использование численных методов для упрощения записи не дало никаких результатов(использовалась составная формула Симпсона).

В случае решения данного СЛАУ(22) осуществляется переход к полудискретной форме, тогда используя составную формулу Симпсона для нахождения интеграла(20) можно перейти к полностью дискретной форме.

8 Решение задачи минимизации, используя различные конфигурации дискретизации. Оценка зависимости погрешности решения от шага интерполяции и шага интегрирования

Для минимизации воспользуемся модулем scipy.optimize, который включает в себя следующие функции:

1. Условной и безусловной минимизации скалярных функций нескольких переменных (minim) с помощью различных алгоритмов (симплекс Нелдера-Мида, BFGS, сопряженных градиентов Ньютона, COBYLA и SLSQP)

2. Глобальной оптимизации (например: basinhopping, diff_evolution)

3. Минимизация остатков МНК (least_squares) и алгоритмы подгонки кривых нелинейным МНК (curve_fit)

4. Минимизации скалярной функций одной переменной (minim_scalar) и поиска корней (root_scalar)

5. Многомерные решатели системы уравнений (root) с использованием различных алгоритмов (гибридный Пауэлла, Левенберг-Марквардт или крупномасштабные методы, такие как Ньютона-Крылова).

Будем варьировать шаг интерполяции и шаг интегрирования от 10^{-3} до 1, на каждой итерации находим погрешность и строим соответствующие точки(x - шаг интегрирования, y - шаг интерполяции, z - погрешность). По полученному графику оценим зависимость погрешности решения от шага интерполяции и шага интегрирования.

9 Заключение

1. В ходе лабораторной работы были реализованы численные методы, осуществлена оценка зависимости абсолютной погрешности численного интегрирования от шага интегрирования.

2. Решение интегрировать по t привело к несовпадению аналитического порядка точности с порядком точности, полученного с помощью графика, вследствие чего была исследована производная подыинтегральной функции. Заданная формула, как и ожидалось, оказалась вычислительно устойчивой, что сказалось на отсутствии оптимального шага интегрирования.

3. Преобразование задачи минимизации функционала к полудискретной форме свелось к получению суммы интеграла путем кусочно-линейной интерполяции, которая стала частью процесса оптимизации. В итоге была получена СЛАУ, решение которой позволяет получить полудискретную форму задачи минимизации.

4. В виду отсутсвия решения СЛАУ построить график зависимости погрешности от шага интегрирования и шага интерполяции невозможно, следовательно и оценить эту зависимость можно лишь эмпирически.

Список использованных источников

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

2. https://habr.com

Выходные данные

Кутепов Н. С.. Отчет о выполнении лабораторной работы по дисциплине «Вычислительная математика». [Электронный ресурс] — Москва: 2021. — 16 с. URL: `https://sa2systems.ru:88` *(система контроля версий кафедры РК6)*

2021, осенний семестр