МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение
высшего образования «Санкт-Петербургский политехнический университет
Петра Великого»

Институт компьютерных наук и кибербезопасности

Высшая школа технологий искусственного интеллекта

Направление: 02.03.01 Математика и компьютерные науки

Отчет о выполнении курсовой работы
«Решение краевой задачи методом стрельбы»
Дисциплина «Вычислительная математика»
Вариант 14.

Выполнил студент группы
№5130201/30001 __________ Мелещенко С.И.

Проверил __________ Пак В. Г.

Санкт-Петербург, 2024

Содержание

Введение

В численных методах решения задач часто встречаются задачи, которые требуют нахождения решений для нелинейных уравнений, вычисления интегралов и решения дифференциальных уравнений. Эти задачи имеют широкое применение в различных областях науки и техники. В рамках данной работы рассматривается комплексная задача, включающая несколько этапов: нахождение корня нелинейного уравнения с помощью метода бисекции, численное интегрирование с использованием метода трапеций и решение системы дифференциальных уравнений методом стрельбы. Эти методы позволяют найти приближенные решения для сложных задач, которые невозможно решить аналитически.

Постановка задачи.

Задача состоит из нескольких этапов, каждый из которых решается с использованием различных численных методов:

1. Необходимо найти минимальный положительный корень уравнения на заданном интервале с помощью метода бисекции.

2. Требуется вычислить величину A, которая зависит от найденного значения корня.

3. Необходимо провести численное интегрирование функции на заданном интервале с помощью метода трапеций.

4.Решение системы дифференциальных уравнений методом стрельбы, где необходимо подобрать параметр, который удовлетворяет заданному условию на границе.

Данные для моего варианта:

14	$\left(\int_2^8 \sin(\frac{1}{x^2})\ dx - 0{,}373961\right)^6$	$2{,}3089132x^*$	$\sin x + \cos(2x) = 1$, минимальный положительный корень

1 Ход работы

Постановка вычислительной задачи

1. Нахождение корня уравнения

Необходимо найти корень нелинейного уравнения на определенном интервале. Для этого применяется метод бисекции, который позволяет точно и эффективно найти решение, разделяя интервал пополам и уточняя корень на каждой итерации.

2. Вычисление величины A

После нахождения корня, вычисляется величина, которая зависит от этого значения. Это позволяет перейти к следующему этапу.

3. Численное интегрирование

Необходимо вычислить определенный интеграл функции на заданном интервале. Для этого используется метод трапеций, который аппроксимирует интеграл путем разбиения области на небольшие участки и нахождения площади трапеций.

4. Решение системы дифференциальных уравнений методом стрельбы

Необходимо решить систему дифференциальных уравнений с определенными граничными условиями. Для этого используется метод стрельбы, который включает решение системы для разных значений начальных условий и подбор необходимого параметра, чтобы удовлетворить заданным условиям.

Разберемся, как решать данное задание вручную.

1. Уравнение и граничные условия

Нам нужно решить нелинейное дифференциальное уравнение второго порядка на интервале

[0,1] с заданными граничными условиями:

$$\frac{d^2y}{dx^2} = y^2 - 1; \ \ y(0) = 0; \ y(1) = 1.$$

2. Метод стрельбы Метод стрельбы заключается в следующем:

a. Начальные условия: Подбираем начальное значение для $\frac{dy}{dx}$.

b. Решение дифференциального уравнения: Сначала решаем дифференциальное уравнение с этим выбранным значением s.

c. Проверка граничных условий: После того как мы решим задачу для некоторого s, проверим, выполняются ли граничные условия, то есть y(1)=1.

d. Корректировка s: Если y(1) =1, то корректируем значение s с помощью метода Ньютона, который будет искать значение s, при котором

y(1)=1.

e. Итерации: Повторяем шаги 2 и 4 до тех пор, пока ошибка не станет достаточно малой (например, y(1)1<e, где e — заданная точность).

3. Этапы решения

3.1. Нахождение минимального положительного корня для вычисления A = 2,3089132x*

Мы используем уравнение для нахождения значения

$$\sin x + \cos(2x) = 1,$$

минимальный положительный корень

a. Сначала записываем уравнение

sin(x)+cos(2x)=1.

b. Далее находим корень уравнения с помощью численных методов (например, методом бисекции или Ньютона), получая значение x* . Это значение нужно будет использовать для вычисления A.

3.2. Вычисление B

Теперь вычислим значение B. Для этого нам нужно вычислить интеграл:

$$\frac{d^2y}{dx^2} = y^2 - 1; \;\; y(0) = 0; \; y(1) = 1.$$

Этот интеграл можно вычислить численно. Обычно для таких интегралов используется метод трапеций, Симпсона или метод Гаусса.

Пределы интегрирования: Интеграл берётся на интервале [2,8].

Для того чтобы решить этот интеграл вручную, потребуется выполнить несколько шагов, но в реальных условиях его проще решить с помощью численного интегрирования, как это и будет сделано в коде.

3.3. Метод стрельбы

a. Начальные условия: Мы начинаем с выбора некоторого начального значения для s=y(0). Например, s=0.

b. Решение дифференциального уравнения: Мы решаем систему дифференциальных уравнений:

$$\frac{dy}{dx} = z$$

$$\frac{dz}{dx} = y^2 - 1$$

с начальными условиями y(0)=0 и z(0)=s. Решение будет зависеть от s.

c. Проверка: Получаем y(1), проверяем, соответствует ли оно граничному условию y(1)=1.

d. Корректировка: Если y(1) отличается от 1, то корректируем s с использованием метода Ньютона.

Метод Ньютона заключается в том, что мы итеративно изменяем s с учётом отклонения y(1) от 1, пока не достигнем нужной точности.

$$s_{new} = s - \frac{f(s)}{f'(s)}$$

где $f(s) = y(1) - 1$, а $f'(s)$ — производная функции $f(s)$ по s.

Корректировка происходит по формуле:

e. Повторение: После каждого обновления s мы снова решаем систему и проверяем y(1).

3.4. Итоговое решение

После нескольких итераций метода Ньютона мы находим подходящее значение s, при котором y(1)=1, и получаем решение краевой задачи.

4. Построение графика

После нахождения решения для

y(x) можно построить график зависимости y(x) на интервале [0,1].

2 Особенности реализации

Для упрощения работы была использована среда Engee.

Напишем код, основанный на методических указаниях, для решения данного задания и вывода графика (см. Приложение).

1. Код находит A через минимальный корень x* и вычисляет B по обновлённому интегралу.

2. Метод стрельбы решает краевую задачу.

3. Построен график решения y(x).

1. Поиск минимального положительного корня x*:

Функция equation(x) определяет математическое выражение, корень которого мы ищем. Метод бисекции (функция bisection_method) используется для нахождения корня уравнения на интервале [0,]. Метод заключается в делении интервала пополам и проверке знака функции на каждом подинтервале, что позволяет сузить область поиска корня.

2. Вычисление значения A:

После нахождения корня x , вычисляется значение A по формуле .

3. Вычисление значения B (численное интегрирование):

Функция integrand(x) определяет подынтегральную функцию. Используется метод трапеций (функция trapezoidal_integration) для численного интегрирования этой функции на интервале [2.0,8.0]. Метод трапеций делит интервал на равные части и суммирует значения функции в этих точках с учётом коэффициентов.

4. Решение системы дифференциальных уравнений методом стрельбы:

Функция shooting_system! описывает систему дифференциальных уравнений с двумя переменными.

Для решения этой системы используется метод Эйлера (функция euler_method), который итеративно решает систему на интервале [0,1] с заданными начальными условиями.

5. Метод Ньютона для подбора параметра s:

Метод Ньютона (функция shooting_method) используется для подбора значения параметра s, чтобы удовлетворить условию y(1)=1. В каждой итерации метода вычисляется ошибка и корректируется значение s, пока ошибка не станет достаточно малой.

6. Построение графика:

В конце, при успешном нахождении решения, строится график функции y(x), который является решением краевой задачи, с помощью библиотеки Plots.

3 Результаты работы программы

Были получены все результаты (см. Рис. 1, Рис. 2).

```
Минимальный положительный корень x* = 2.617992879306086
Вычисленное значение A = 6.044718316535828
Вычисленное значение B = 0.007357980751645128
Итерация метода Ньютона:
Итерация 1: s = 3.0260381486437367, y(1) = 3.2397067373109847, error = 2.2397067373109847
Итерация 2: s = 1.5909140138113678, y(1) = 1.2402442728940244, error = 0.2402442728940244
Итерация 3: s = 1.3967725279466219, y(1) = 1.0037680873410597, error = 0.0037680873410597115
Итерация 4: s = 1.3936294129275077, y(1) = 1.000000971843642, error = 9.718436420058651e-7
Решение найдено: s = 1.3936294129275077, y(1) = 1.000000971843642
```

Рис. 1: Расчеты

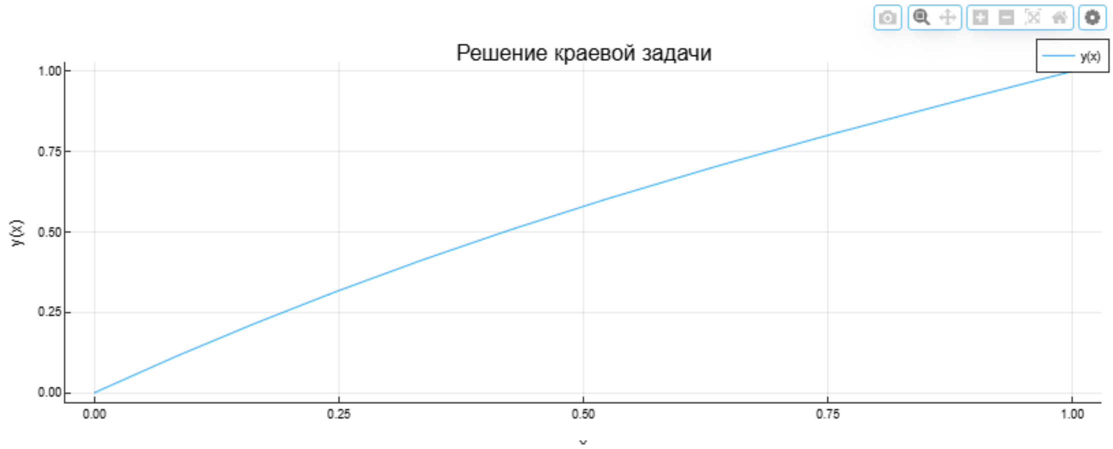

Рис. 2: График

4 Оценка погрешностей

1. Нахождение минимального положительного корня x*

Метод бисекции позволяет получить корень с заданной точностью, которая определяется параметром погрешности tol tol. Погрешность метода бисекции можно оценить как половину длины интервала на последней итерации. В данном случае, если корень был найден на интервале [a,b], то погрешность будет меньше или равна $\frac{b-a}{2}$.

2. Вычисление величины A

Для оценки погрешности в вычислении A можно использовать следующее приближенное выражение:

$$\Delta A = \left|\frac{\partial A}{\partial x^*}\right| \cdot \Delta x^* = 2.3089132 \cdot \Delta x^*$$

3. Численное интегрирование Для метода трапеций, где B — результат интегрирования функции, погрешность метода можно оценить как:

$$\Delta B = \frac{(b-a)^3}{12n^2} \cdot |f''(\xi)|$$

4. Метод Ньютона для подбора параметра s

На каждом шаге метод Ньютона улучшает точность решения, и погрешность постепенно уменьшается:

Итерация 1:

s=3.0260381486437367, погрешность 2.2397067373109847

Итерация 2:

s=1.5909140138113678, погрешность 0.2402442728940244

Итерация 3:

s=1.3967725279466219, погрешность 0.0037680873410597115

Итерация 4:

s=1.3936294129275077, погрешность 9.718436420058651

Заключение

В результате работы

В ходе выполнения работы были решены задачи, связанные с нахождением корней уравнений, численным интегрированием и решением системы дифференциальных уравнений методом стрельбы. В результате работы были получены следующие результаты:

Минимальный положительный корень: Был найден минимальный положительный корень уравнения с помощью метода бисекции. Полученное значение:

x* = 2.617992879306086

Погрешность метода бисекции зависит от точности параметра tol tol, и её можно уменьшить, увеличив количество итераций или сузив начальный интервал.

Вычисление значения A: С использованием найденного корня было вычислено значение

A=6.044718316535828

При малой погрешности корня погрешность в вычислении A также будет мала.

Численное интегрирование для нахождения значения B: Было использовано численное интегрирование методом трапеций для вычисления значения

B=0.007357980751645128

Погрешность интегрирования зависит от количества разбиений n и второй производной функции, которая интегрируется. Для повышения точности можно увеличить количество разбиений n.

Решение системы дифференциальных уравнений методом стрельбы: Для нахождения параметра s был использован метод Ньютона. После нескольких итераций метод сходился к значению:

s=1.3936294129275077

Погрешность на последней итерации составила:

$\Delta y = 9.718436420058651 10^{-7}$

Это подтверждает, что решение достигло высокой точности, и погрешность на последней итерации была крайне мала.

Приложение

Листинг 1: Код программы

```
using Plots

# 1. Поиск минимального положительного корня x*
function equation(x)
    return sin(x) + cos(2*x) - 1
end

# Метод бисекции
function bisection_method(f, a, b, tol=1e-6, max_iters=100)
    if f(a) * f(b) > 0
        println("Функция имеет одинаковые знаки на концах
        интервала. Метод бисекции не применим.")
        return nothing
    end
    for i in 1:max_iters
        c = (a + b) / 2
        if abs(f(c)) < tol
            return c
        end
        if f(c) * f(a) < 0
            b = c
        else
            a = c
        end
    end
    return (a + b) / 2
end

# Поиск корня
x_star = bisection_method(equation, 0.0,    )
println("Минимальный положительный корень x* = $x_star")

# 2. Вычисление A
A = 2.3089132 * x_star
println("Вычисленное значение A = $A")

# 3. Вычисление B (численное интегрирование)
function integrand(x)
```

```
    return (sin(1 / x^2) - 0.373962)^6
end

# Метод трапеций для численного интегрирования
function trapezoidal_integration(f, a, b, n)
    h = (b - a) / n
    sum = (f(a) + f(b)) / 2
    for i in 1:n-1
        sum += f(a + i*h)
    end
    return sum * h
end

# Вычисление B
n = 1000  # количество разбиений
B = trapezoidal_integration(integrand, 2.0, 8.0, n)
println("Вычисленное значение B = $B")

# 4. Система дифференциальных уравнений и метод стрельбы

function shooting_system!(y, z, x)
    dy = z
    dz = y^2 - 1
    return dy, dz
end

# Метод Рунге-Кутты для более точного решения системы
function runge_kutta_method(f, s, xspan, n)
    x0, x1 = xspan
    h = (x1 - x0) / n
    x_vals = [x0]
    y_vals = [0.0]  # Начальные условия: y(0) = 0
    z_vals = [s]    # Начальные условия: z(0) = s

    for i in 1:n
        x = x_vals[end] + h
k1y, k1z = f(y_vals[end], z_vals[end], x)
k2y, k2z = f(y_vals[end] + h * k1y / 2,
z_vals[end] + h * k1z / 2, x + h / 2)
k3y, k3z = f(y_vals[end] + h * k2y / 2,
z_vals[end] + h * k2z / 2, x + h / 2)
k4y, k4z = f(y_vals[end] + h * k3y, z_vals[end]
+ h * k3z, x + h)
```

```
y_next = y_vals[end] + h * (k1y + 2*k2y +
2*k3y + k4y) / 6
z_next = z_vals[end] + h * (k1z + 2*k2z +
2*k3z + k4z) / 6

        push!(x_vals, x)
        push!(y_vals, y_next)
        push!(z_vals, z_next)
    end

    return x_vals, y_vals
end

# Метод Ньютона для подбора параметра s
function shooting_method(A, B, tol=1e-6, max_iters=100)
    s = (A + B) / 2
    println("Итерация метода Ньютона:")

    for i in 1:max_iters
        # Решаем систему для текущего значения s
        x_vals, y_vals = runge_kutta_method(shooting_system!,
        s, (0.0, 1.0), 100)
        y1 = y_vals[end]

        error = y1 - 1
        println("Итерация $i: s = $s, y(1) = $y1,
        error = $error")

        if abs(error) < tol
            println("Решение найдено: s = $s, y(1) = $y1")
            return s, x_vals, y_vals
        end

        # Шаг метода Ньютона для корректировки s
        delta_s = 1e-5
        x_vals_delta, y_vals_delta =
        runge_kutta_method(shooting_system!, s + delta_s,
        (0.0, 1.0), 100)
        y1_delta = y_vals_delta[end]
        derivative = (y1_delta - y1) / delta_s

        s -= error / derivative
```

```
    end

    error("Метод Ньютона не сошелся за $max_iters итераций")
end

# 5. Запуск метода стрельбы
s, x_vals, y_vals = shooting_method(A, B)

# 6. Построение графика
plot(x_vals, y_vals, label="y(x)", xlabel="x", ylabel="y(x)",
title="Решение краевой задачи")
```