МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский Авиационный Институт»
(Национальный Исследовательский Университет)

Институт: №8 «Информационные технологии и прикладная математика»
Кафедра: 806 «Вычислительная математика и программирование»

Курсовая работа по курсу «Численные методы»

Группа: М8О-407Б-21

Студент: А. В. Крючков

Преподаватель: Ю.В. Сластушенский

Оценка:

Дата: 01.12.2024

Москва, 2024

1 Тема

Вычисление несобственных интегралов численными методами.

2 Задание

1. Изучить теоретические основы несобственных интегралов, классификацию особенностей и методы их численного вычисления.
2. Исследовать классические методы численного вычисления интегралов, такие как метод прямоугольников, метод трапеций и метод Симпсона, и проанализировать их применимость для решения задач с несобственными интегралами.
3. Разработать программу для численного вычисления несобственных интегралов с использованием различных методов.
4. Выполнить сравнительный анализ точности и эффективности выбранных методов, используя в качестве примера конкретный интеграл.

3 Листинг кода

```python
import math
import numpy as np


INF = 1e10


# Функция интегрирования
def f(x):
    return np.exp(-np.abs(x))


# Высчитываем интеграл f(x)dx на интервале [l; r]
используя метод центральных прямоугольников с шагом=h
def integrate_rectangle_method(f, l, r, h):
    n = int((r - l) / h)
    s = 0
    for i in range(n):
        s += h * f(l + h * (i + 0.5))

    return s
```

```python
# Высчитываем интеграл f(x)dx на интервале [l; r]
используя метод трапеций с шагом=h
def integrate_trapezoid_method(f, l, r, h):
    n = int((r - l) / h)
    x = [l + i * h for i in range(n + 1)]
    s = 0
    for i in range(n):
        s += h * (f(x[i]) + f(x[i + 1])) / 2
    return s


# Высчитываем интеграл f(x)dx на интервале [l; r]
используя метод парабол (формула Симпсона) с шагом=h
def integrate_parabola_method(f, l, r, h):
    n = int((r - l) / (h / 2) )

    x = [l + i * (h / 2) for i in range(1, n)]

    s1, s2 = 0, 0
    for i in range(len(x)):
        if i % 2:
            s1 += f(x[i])
        else:
            s2 += f(x[i])

    return (h / 2) / 3 * (f(l) + 2 * s1 + 4 * s2 + f(r))


# Вычисляем несобственный интеграл (1 типа), преобразуя
его в определённый интеграл
def integrate_with_definite_integral(integration_method,
f, l, r, h=0.01, eps=1e-6):
    def f_new(t):
        return (1. / t ** 2) * f(1. / t)

    result = 0
    if r == INF:
        new_r = max(eps, l)
        result += integration_method(f_new, eps, 1. /
new_r - eps, h)
```

```
    else:
        new_r = r
    if l == -INF:
        new_l = min(-eps, r)
        result += integration_method(f_new, 1. / new_l +
eps, -eps, h)
    else:
        new_l = l
    if new_l < new_r:
        result += integration_method(f, new_l, new_r, h)
    return result


# Вычисляем несобственный интеграл f(x)dx (1 типа),
используя предельный переход. Возвращаем: результат
интегрирования и количество итераций
def integrate_lim(integration_method, f, l, r, h=0.1,
eps=1e-6):
    result = 0
    iters = 0
    # Если правая граница равна бесконечности (INF), мы
начинаем интегрирование с точки cur_x = max(l, 0) и
    # продолжаем до тех пор, пока разница между текущим
значением интеграла и следующим не станет меньше eps.
    # После завершения цикла добавляем площадь правого
треугольника, образованного последней точкой и конечной
точкой.
    if r == INF:
        finish = False
        cur_x = max(l, 0)
        while not finish:
            iters += 1
            diff = integration_method(f, cur_x, cur_x + h
+ eps, h)
            cur_x += h
            if abs(diff) < eps:
                finish = True
            result += diff
        result += f(cur_x - h) * (f(cur_x - h) * h /
(f(cur_x - h) - f(cur_x))) # Правый треугольник
```

```
    else:
        result += integration_method(f, 0, r, h)
    if l == -INF:
        finish = False
        cur_x = min(0, r)
        # Если левая граница равна минус бесконечности (-
INF), аналогично предыдущему шагу, но
        # теперь идем влево от нуля до тех пор, пока
разница между текущими значениями интеграла
        # не станет достаточно малой. Затем добавляем
площадь левого треугольника.
        while not finish:
            iters += 1
            diff = integration_method(f, cur_x - h - eps,
cur_x, h)
            cur_x -= h
            if abs(diff) < eps:
                finish = True
            result += diff
        result += f(cur_x + h) * (f(cur_x + h) * h /
(f(cur_x + h) - f(cur_x))) # Левый треугольник
    else:
        result += integration_method(f, l, 0, h)
    return result, iters
```

Тест программы:

```
l = 1
r = INF
h = 0.01
eps = 1e-6

CP(l, r, h, eps)
```

Метод центральных прямоугольников:
Преобразование в определённый интеграл
Интеграл = 0.36418411129155914

Метод предельного перехода
Интеграл = 0.3678794051167605
Количество итераций: 822

Метод трапеций:
Преобразование в определённый интеграл
Интеграл = 0.36417946705145965

Метод предельного перехода
Интеграл = 0.3678840023622135
Количество итераций: 822

Метод парабол:
Преобразование в определённый интеграл
Интеграл = 0.36541498988285165

Метод предельного перехода
Интеграл = 0.37016067560979987
Количество итераций: 822

```
l = -INF
r = -10
h = 0.01
eps = 1e-8

CP(l, r, h, eps)
```

Метод центральных прямоугольников:
Преобразование в определённый интеграл
Интеграл = 4.389409762226072e-05

Метод предельного перехода
Интеграл = 4.54146357540127e-05
Количество итераций: 383

Метод трапеций:

Преобразование в определённый интеграл
Интеграл = 4.841792800395624e-05

Метод предельного перехода
Интеграл = 4.5415190931784 67e-05
Количество итераций: 383

Метод парабол:
Преобразование в определённый интеграл
Интеграл = 4.540204108282589e-05

Метод предельного перехода
Интеграл = 0.0017121571541039377
Количество итераций: 383

```
l = -INF
r = 10
h = 0.01
eps = 1e-4

CP(l, r, h, eps)
```

Метод центральных прямоугольников:
Преобразование в определённый интеграл
Интеграл = 1.9999504386470157

Метод предельного перехода
Интеграл = 1.9999961629161738
Количество итераций: 462

Метод трапеций:
Преобразование в определённый интеграл

Интеграл = 1.9999619393034518

Метод предельного перехода
Интеграл = 2.000021037899277
Количество итераций: 462

Метод парабол:
Преобразование в определённый интеграл
Интеграл = 1.9999542721915933

Метод предельного перехода
Интеграл = 2.0000210388511688
Количество итераций: 462

```
l = -INF
r = INF
h = 0.01
eps = 1e-5

CP(l, r, h, eps)
```

Метод центральных прямоугольников:
Преобразование в определённый интеграл
Интеграл = 1.9999800000990708

Метод предельного перехода
Интеграл = 2.000011534811885
Количество итераций: 1384

Метод трапеций:
Преобразование в определённый интеграл
Интеграл = 1.999980000098995

Метод предельного перехода
Интеграл = 2.0000365099392248
Количество итераций: 1384

Метод парабол:
Преобразование в определённый интеграл
Интеграл = 2.00331330009951

Метод предельного перехода
Интеграл = 2.000019876504453
Количество итераций: 1384

4 Выводы

Численные методы позволяют с высокой точностью вычислять несобственные интегралы. Однако их эффективность и точность зависят от выбранного алгоритма и способа обработки бесконечных пределов. В случае рассматриваемого интеграла наилучшую точность показал метод Симпсона.

5 Список используемой литературы

1. Раздел 5. Численные методы решения дифференциальных уравнений с частными производными – https://mainfo.ru/mietodichieskiie-matierialy