Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«КАЗАНСКИЙ (ПРИВОЛЖСКИЙ) ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

ИНСТИТУТ ВЫЧИСЛИТЕЛЬНОЙ МАТЕМАТИКИ
И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Кафедра прикладной математики и искусственого интеллекта

Направление подготовки: 01.03.04 – Прикладная математика

ОТЧЁТ

По дисциплине «Численные методы»
на тему:
«Вычисление интеграла с помощью квадратурных формул»

Выполнил:
студент группы 09-222
Фаррахова Л. Ф.
Проверил:
ассистент Глазырина О.В.

Казань, 2024 год

Содержание

1 Постановка задачи

Необходимо изучить и сравнить различные способы приближённого вычисления функции

$$Si(x) = \sum_{n=0}^{\infty} \frac{(-1)^n x^{2n+1}}{(2n+1)(2n+1)!} \tag{1}$$

1. Протабулировать Si(x) на отрезке [a,b] с шагом h и точностью ε, основываясь на ряде Тейллора, предварительно вычислив его. Получив таким образом таблицу из 11 точек вида:

   $x_0\ x_1\ x_2\ \ldots$
   $f_0\ f_1\ f_2\ \ldots$

   $f_i = \mathrm{Si}(x_i), \quad x_i = a + ih, \quad i = 0, \ldots, n.$

2. Вычислить Si(x) при помощи пяти составных квадратурных формул при

   $$h = (x_{i+1} - x_i) = \frac{b-a}{n} = \frac{\sin(t)}{t}$$

   2.1. Формула правых прямоугольников:

   $$J_N(x) = \sum_{i=1}^{n} hg(x_i) \tag{2}$$

   2.2. Формула центральных прямоугольников:

   $$J_N(x) = \sum_{i=1}^{n} hg\left(\frac{x_i + x_{i+1}}{2}\right) \tag{3}$$

   2.3. Формула трапеции:

   $$J_N(x) = \sum_{i=1}^{n} h\frac{g(x_i) + g(x_{i+1})}{2} \tag{4}$$

   2.4. Формула Симпсона:

   $$J_N(x) = \sum_{i=1}^{n} \frac{h}{6}\left[g(x_i) + 4g\left(\frac{x_i + x_{i+1}}{2}\right) + g(x_{i+1})\right] \tag{5}$$

   2.5. Формула Гаусса:

   $$J_N(x) = \sum_{i=1}^{n} \frac{h}{2}\left[g\left(x_i + \frac{h}{2}\left(1 - \frac{1}{\sqrt{3}}\right)\right) + g\left(x_i + \frac{h}{2}\left(1 + \frac{1}{\sqrt{3}}\right)\right)\right] \tag{6}$$

Вычисления проводятся от начала интегрирования до каждой из 11 точек, увеличивая количество разбиений между точками в 2 раза до тех пор, пока погрешность больше ε.

2 Ход работы

Для того чтобы найти значение функции в точке, необходимо протабулировать искомый интеграл на отрезке [a, b] с шагом $h = 0.4$ и точностью ε.

Для этого выделим два общих члена a_n, a_{n+1} из $Si(x)$ и найдём $q_n = \frac{a_{n+1}}{a_n}$:

$$a_n = \frac{(-1)^n x^{2n+1}}{(2n+1)(2n+1)!}, \quad a_{n+1} = \frac{(-1)^{n+1} x^{2n+3}}{(2n+3)(2n+3)!}. \tag{7}$$

$$q_n = \frac{-x^2(2n+1)}{(2n+2)(2n+3)^2}. \tag{8}$$

Вычислим по ней значения функции в 11 узлах интерполяции (Таблица 1).

x_i	$erf(x_i)$
0,0	0,0000000000
0,4	0,3964614570
0,8	0,7720957994
1,2	1,1080472469
1,6	1,3891806602
2,0	1,6054129601
2,4	1,7524855137
2,8	1,8320965767
3,2	1,8514009714
3,6	1,8219480515
4,0	1,7582031488

Таблица 1 - точки x_i и значения разложения функции $Si(x_i)$

Далее вычислим $Si(x)$ с помощью пяти составных квадратурных формул и составим для каждой формулы таблицу, в которой первый столбец - одиннадцать точек разбиения, второй - значение интеграла, третий - значение интеграла, вычисленного соответсвующим методом. В четвертом столбце находятся значения погрешности. В последнем - количество разбиений, необходимых для подсчета интеграла с заданной точностью.

1. Правые прямоугольники:

x_i	$J_0(x_i)$	$J_(x_i)$	$\|J_0(x_i) - J_N(x_i)\|$	N
0,0	0,0000000000	0,0000000000	0,0000000000	2
0,4	0,3964614570	0,3964561820	0,0000052750	1024
0,8	0,7720957994	0,7720555663	0,0000402331	1024
1,2	1,1080472469	1,1079159975	0,0001312494	1024
1,6	1,3891806602	1,3888883591	0,0002923012	1024
2,0	1,6054129601	1,6048804522	0,0005325079	1024
2,4	1,7524855137	1,7516450882	0,0008404255	1024
2,8	1,8320965767	1,8308923244	0,0012042522	1024
3,2	1,8514009714	1,8498086929	0,0015922785	1024
3,6	1,8219480515	1,8199745417	0,0019735098	1024
4,0	1,7582031488	1,7558794022	0,0023237467	1024

Таблица 2 - таблица значений для формулы правых прямоугольников

2. Центральные прямоугольники:

x_i	$J_0(x_i)$	$J_(x_i)$	$\|J_0(x_i) - J_N(x_i)\|$	N
0,0	0,0000000000	0,0000000000	0,0000000000	2
0,4	0,3964614570	0,3964616656	0,0000002086	64
0,8	0,7720957994	0,7720960975	0,0000002980	256
1,2	1,1080472469	1,1080478430	0,0000005960	512
1,6	1,3891806602	1,3891806602	0,0000000000	1024
2,0	1,6054129601	1,6054127216	0,0000002384	1024
2,4	1,7524855137	1,7524878979	0,0000023842	1024
2,8	1,8320965767	1,8320971727	0,0000005960	1024
3,2	1,8514009714	1,8514013290	0,0000003576	512
3,6	1,8219480515	1,8219479322	0,0000001192	1024
4,0	1,7582031488	1,7582037449	0,0000005960	512

Таблица 3 - таблица значений для формулы центральных прямоугольников

3. Формула трапеций:

x_i	$J_0(x_i)$	$J_(x_i)$	$\lvert J_0(x_i) - J_N(x_i)\rvert$	N
0,0	0,0000000000	0,0000000000	0,0000000000	2
0,4	0,3964614570	0,3964614570	0,0000000000	128
0,8	0,7720957994	0,7720955014	0,0000002980	256
1,2	1,1080472469	1,1080472469	0,0000000000	512
1,6	1,3891806602	1,3891803026	0,0000003576	1024
2,0	1,6054129601	1,6054120064	0,0000009537	512
2,4	1,7524855137	1,7524851561	0,0000003576	1024
2,8	1,8320965767	1,8320960999	0,0000004768	1024
3,2	1,8514009714	1,8514009714	0,0000000000	1024
3,6	1,8219480515	1,8219457865	0,0000022650	1024
4,0	1,7582031488	1,7582020760	0,0000010729	1024

Таблица 4 - таблица значений для формулы трапеций

4. Формула Симпсона

4.1. Вывод формулы Симпсона через интегральный полином Лагранжа:
Формула для полинома Лагранжа:

$$L_n(x) = \sum_{i=0}^{n} f(x_i) \prod_{i \neq j, j=0}^{n} \frac{x - x_j}{x_i - x_j} \tag{9}$$

По трём узлам $(x_1 = a, x_2 = \dfrac{a+b}{2}, x_3 = b) : L_2 = f(a) \left(\dfrac{x - \dfrac{a+b}{2}}{a - \dfrac{a+b}{2}} \right) \left(\dfrac{x-b}{a-b} \right) +$

$f\left(\dfrac{a+b}{2} \right) \left(\dfrac{x-a}{\dfrac{a+b}{2} - a} \right) \left(\dfrac{x-b}{\dfrac{a+b}{2} - b} \right) + f(b) \left(\dfrac{x - \dfrac{a+b}{2}}{b - \dfrac{a+b}{2}} \right) \left(\dfrac{x-b}{b-a} \right).$

Проинтегрируем выражение по интервалу [a,b]:

$$\int_a^b L_2(x) \mathrm{d}x = f(a)c_1 + f\left(\frac{a+b}{2} \right) c_2 + f(b)c_3 \tag{10}$$

где $c_1 = \dfrac{b-a}{6}, c_2 = \dfrac{2}{3}(b-a), c_3 = \dfrac{b-a}{6}.$

Тогда:

$$\int_a^b L_2(x) \mathrm{d}x = \frac{b-a}{6} \left(f(a) + 4f\left(\frac{a+b}{2} \right) + f(b) \right) \tag{11}$$

4.2. Значения, полученные для формулы Симпсона:

x_i	$J_0(x_i)$	$J_(x_i)$	$\lvert J_0(x_i) - J_N(x_i)\rvert$	N
0,0	0,0000000000	0,0000000000	0,0000000000	2
0,4	0,3964614570	0,3964614868	0,0000000298	4
0,8	0,7720957994	0,7720957398	0,0000000596	8
1,2	1,1080472469	1,1080472469	0,0000000000	8
1,6	1,3891806602	1,3891803026	0,0000003576	16
2,0	1,6054129601	1,6054130793	0,0000001192	16
2,4	1,7524855137	1,7524855137	0,0000000000	16
2,8	1,8320965767	1,8320968151	0,0000002384	16
3,2	1,8514009714	1,8514008522	0,0000001192	32
3,6	1,8219480515	1,8219482899	0,0000002384	16
4,0	1,7582031488	1,7582032681	0,0000001192	16

Таблица 5 - таблица значений для формулы Симпсона

5. Формула Гаусса:

x_i	$J_0(x_i)$	$J_(x_i)$	$\lvert J_0(x_i) - J_N(x_i)\rvert$	N
0,0	0,0000000000	0,0000000000	0,0000000000	2
0,4	0,3964614570	0,3964614868	0,0000000298	4
0,8	0,7720957994	0,7720956802	0,0000001192	4
1,2	1,1080472469	1,1080472469	0,0000000000	8
1,6	1,3891806602	1,3891805410	0,0000001192	16
2,0	1,6054129601	1,6054128408	0,0000001192	16
2,4	1,7524855137	1,7524856329	0,0000001192	16
2,8	1,8320965767	1,8320965767	0,0000000000	16
3,2	1,8514009714	1,8514007330	0,0000002384	16
3,6	1,8219480515	1,8219481707	0,0000001192	16
4,0	1,7582031488	1,7582032681	0,0000001192	16

Таблица 6 - таблица значений для формулы Гаусса

3 Выводы

В ходе работы были изучены численные методы вычисления интегралов с применением пяти различных квадратурных формул. Анализ результатов показал, что методы Гаусса и Симпсона являются наиболее эффективными. Они обеспечивает высокую точность при минимальном числе разбиений благодаря специально подобранным узлам для приближенного вычисления интеграла.

4 Листинг программы

```cpp
#include <algorithm>
#include <cmath>
#include <iostream>
#include <vector>

using namespace std;

namespace constans {
const int STEPS = 1024;
const float LEFT_BORDER = 0;
const float EPSILON = 1e-6;
const float E = 2.71828182846;
}  // namespace constans

float Tfunc(float x) {
  int n = 0;
  float node_0 = x;
  float ans = x;
  while (fabs(node_0) > 1e-6) {
    float q = ((-1) * x * x * (2 * n + 1)) / ((2 * n + 2) * (2 * n +
       3) * (2 * n + 3));
    node_0 *= q;
    ans += node_0;
    n++;
  }
  return ans;
}

float func(float t) {
    if (t != 0) {
        return sin(t) / t;
    }
    return 1;
}

float leftRectangles(float (*func)(float), const float &a, float b,
                     int steps) {
  float result = 0.0;
  float stepSize = (b - a) / steps;
  for (int i = 0; i < steps; i++) {
    float x_i = a + stepSize * i;
```

```
    result += stepSize * func(x_i);
  }
  return result;
}

float rightRectangles(float (*func)(float), const float &a, float b,
                      int steps) {
  float result = 0.0;
  float stepSize = (b - a) / steps;
  for (int i = 1; i <= steps; i++) {
    float x_i_1 = a + stepSize * i;
    result += stepSize * func(x_i_1);
  }
  return result;
}

float middleRectangles(float (*func)(float), const float &a, float b,
                       int steps) {
  float result = 0.0;
  float stepSize = (b - a) / steps;
  for (int i = 0; i < steps; i++) {
    float x_i = a + stepSize * i;
    float x_i_1 = a + stepSize * (i + 1);
    result += stepSize * func((x_i + x_i_1) / 2);
  }
  return result;
}

float trapezeFormula(float (*func)(float), const float &a, float b,
                     int steps) {
  float result = func(a) + func(b);
  float stepSize = (b - a) / steps;
  for (int i = 1; i < steps; i++) {
    float x_i_1 = a + stepSize * i;
    result += 2 * func(x_i_1);
  }
  result *= stepSize / 2;
  return result;
}

float SypmsonsFormula(float (*func)(float), const float &a, float b,
                      int steps) {
  float stepSize = (b - a) / steps;
```

```
84    float result = 0;
85    float x = a;
86    for (int i = 0; i < steps; i++)
87    {
88        result += (func(x) + 4 * func(x + stepSize / 2) + func(x +
              stepSize)) * stepSize / 6;
89        x += stepSize;
90    }
91    return result;
92 }
93
94 float GaussFormula(float (*func)(float), const float &a, float b, int
       steps) {
95    float stepSize = (b - a) / steps;
96    float ad1 = (1 - 1.0 / sqrt(3)) * stepSize / 2;
97    float ad2 = (1 + 1.0 / sqrt(3)) * stepSize / 2;
98    float result = 0;
99    float x = a;
100   for (int i = 0; i < steps; i++)
101   {
102       result += (func(x + ad1) + func(x + ad2)) * stepSize / 2;
103       x += stepSize;
104   }
105   return result;
106 }
107
108 void CalculateFunc(vector<float> points,
109                    float (*function)(float (*)(float), const float &,
110                                      float, int)) {
111   for (auto point : points) {
112     int steps = 1;
113     float lastResult = 0.0;
114     float currentResult = 0.0;
115     do {
116       steps *= 2;
117       lastResult = currentResult;
118       currentResult = function(func, constans::LEFT_BORDER, point,
              steps);
119     } while (abs(lastResult - currentResult) > constans::EPSILON &&
           steps < constans::STEPS);
120
121     float difference = abs(Tfunc(point) - currentResult);
122
```

```
123     printf(
124         "x_i = %.1lf | J_o = %.10lf | J_n = %.10lf | |J_o - J_n| =
              %.10lf | N "
125         "= %d\n",
126         point, Tfunc(point), currentResult, difference, steps);
127   }
128 }
129
130 int main() {
131   vector<float> points = {0.0, 0.4, 0.8, 1.2, 1.6,
132                           2.0, 2.4, 2.8, 3.2, 3.6, 4.0};
133   cout << "Правые прямоугольники\n";
134   CalculateFunc(points, rightRectangles);
135   cout << "Левые прямоугольники\n";
136   CalculateFunc(points, leftRectangles);
137   cout << "Центральные прямоугольники\n";
138   CalculateFunc(points, middleRectangles);
139   cout << "Трапеции\n";
140   CalculateFunc(points, trapezeFormula);
141   cout << "Симпсон\n";
142   CalculateFunc(points, SypmsonsFormula);
143   cout << "Гаус\n";
144   CalculateFunc(points, GaussFormula);
145   return 0;
146 }
```