МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное автономное
образовательное учреждение высшего образования
"Национальный исследовательский университет ИТМО"

ФАКУЛЬТЕТ БЕЗОПАСНОСТИ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ
Кафедра проектирования и безопасности компьютерных систем

ЛАБОРАТОРНАЯ РАБОТА №9
по дисциплине
"ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА"

по теме:
«Приближенное интегрирование»

Выполнил: Нгуен Ле Минь
Группа: N3251
Преподаватель: Гришенцев А.Ю.

УНИВЕРСИТЕТ ИТМО

Санкт-Петербург
2021

Задание 9.1: Функции для интегрирования

1: Задание

Функции для интегрирования :

$$1.f(x) = x^2, x \in [-5; 5]$$
$$2.f(x) = sin^2(x), x \in [-\pi; \pi]$$
$$3.f(x) = sin(2x) + cos(7x) + 8, x \in [-\pi; \pi]$$
$$4.f(x) = 2x^4 + x^3 + 2x^2 + 3x + 24, x \in [-1; 3]$$
$$5.f(x) = ln(x^2 + 1) + sin(\frac{x}{3}) + 17, x \in [-100; 100]$$
$$6.f(x) = 5^x + sinx + x + 1, x \in [-\pi; \pi]$$
$$7.f(x) = x^5 + 2x^4 + 3x^3 + 4x^2 + 5x + 6, x \in [-7; 7]$$

2: Теория

Функции для интегрирования :

$$1.f(x) = x^2, x \in [-5; 5]$$
$$2.f(x) = sin^2(x), x \in [-\pi; \pi]$$
$$3.f(x) = sin(2x) + cos(7x) + 8, x \in [-\pi; \pi]$$
$$4.f(x) = 2x^4 + x^3 + 2x^2 + 3x + 24, x \in [-1; 3]$$
$$5.f(x) = ln(x^2 + 1) + sin(\frac{x}{3}) + 17, x \in [-100; 100]$$
$$6.f(x) = 5^x + sinx + x + 1, x \in [-\pi; \pi]$$
$$7.f(x) = x^5 + 2x^4 + 3x^3 + 4x^2 + 5x + 6, x \in [-7; 7]$$

3: Код

```cpp
/*
 * Author : Nguyen Le Minh
 * Group : N3251
 * Lab : 9.1
 */
#include<iostream>
#include<cmath>
using namespace std;
float f1(float x)
{
    return x * x;
}
float f2(float x)
{
    return sin(x) * sin(x);
}
float f3(float x)
{
    return sin(2 * x) + cos(7 * x) + 8;
}
float f4(float x)
{
    return 2 * pow(x, 4) + pow(x, 3) + 2 * pow(x, 2) + 3 * x + 24;
}

float f5(float x)
{
    return log(x * x + 1) + sin(x / 3) + 17;
}
```

```cpp
float f6(float x)
{
    return pow(5, x) + sin(x) + x + 11;
}

float f7(float x)
{
    return pow(x, 5) + 2 * pow(x, 4) + 3 * pow(x, 3) + 4 * pow(x, 2) + 5 * x +6;
}

float integral(float f(float), float a, float b, int n)
{
    float s = (f(a) + f(b)) / 2;
    float dx = (b - a) / n;
    for (int i = 0; i < n - 1; i++)
    {
        s += f(a + dx * (i + 1));
    }
    return dx * s;
}

void solver(){
    int n = 100000;
    cout << "F1 = " << integral(f1, -5, 5, n) << endl;
    cout << "F2 = " << integral(f2, -3.14, 3.14, n) << endl;
    cout << "F3 = " << integral(f3, -3.14, 3.14, n) << endl;
    cout << "F4 = " << integral(f4, -1, 3, n) << endl;
    cout << "F5 = " << integral(f5, -100, 100, n) << endl;
    cout << "F6 = " << integral(f6, -3.14, 3.14, n) << endl;
    cout << "F7 = " << integral(f7, -7, 7, n) << endl;
}

int main()
{
    solver();
    return 0;
}
```

4: Выводы программы

F1 = 83.3327
F2 = 3.1416
F3 = 50.2436
F4 = 244.267
F5 = 4848.51
F6 = 166.372
F7 = 14444.1

Задание 9.2: Метод прямоугольников

1: Задание

Наименование задачи: произвести интегрирование функций на заданном интервале методом прямоугольников.

Вид решения: программа и отчёт.

Реализация решения: язык C или C++.

Разработать алгоритм и написать программу реализующую: интегрирование функций на заданном интервале методом прямоугольников, расчёт остаточного члена. При равном шаге сетки, сравнить полученный результат с результатами полученными иными методами численного интегрирования. Оценить вычислительную сложность.

2: Теория

Метод прямоугольников определяют как метод численного интегрирования функции одной переменной, заключающийся в замене подынтегральной функции на многочлен нулевой степени, то есть константу, на каждом элементарном отрезке. Если рассмотреть график подынтегральной функции, то метод будет заключаться в приближенном вычислении площади под графиком суммированием площадей конечного числа прямоугольников, ширина которых будет определяться расстоянием между соответствующими соседними узлами интегрирования, а высота — значением подынтегральной функции в этих узлах. Алгебраический порядок точности равен 0. (Для формулы средних прямоугольников равен 1).

Если отрезок $[a; b]$ является элементарным и не подвергается дальнейшему разбиению, значение интеграла можно найти по:

1) Формуле левых прямоугольников:

$$\int_a^b f(x)\,dx \approx f(a)(b-a).$$

2) Формуле правых прямоугольников:

$$\int_a^b f(x)\,dx \approx f(b)(b-a).$$

3) Формуле прямоугольников (средних):

$$\int_a^b f(x)\,dx \approx f\left(\frac{a+b}{2}\right)(b-a).$$

3: Код

```cpp
/*
 * Author : Nguyen Le Minh
 * Group : N3251
 * Lab : 9.2
 */
#include <iostream>
#include <cmath>

using namespace std;

double testFunction(double x){
    return pow(x,5) + 2*pow(x,4) + 3*pow(x,3) + 4*pow(x,2) + 5*x + 6;
}

int main() {
    double a, b, steps;
    cout << "Введите границы a, b и количество разбиений через пробел:\n";
    cin >> a >> b >> steps;
    double step = (b - a) / steps;
    double integral = 0;
```

```
    for (int i = 0; i < steps; i++) {
        integral += testFunction(a + i * step) * step;
    }
    cout << "Интеграл: " << integral;
    return 0;
}
```

4: Выводы программы

Введите границы a, b и количество разбиений через пробел:
-7 7 100000
Интеграл: 14441.8
Сравниваем с значением, которое было получено по методу численного интегрирования : 14444.1

Мы научились средствами языка C++ реализовывать интегрирование функций методом прямоугольников.

Задание 9.3: Метод трапеций

1: Задание

Наименование задачи: произвести интегрирование функций на заданном интервале методом трапеций.

Вид решения: программа и отчет.

Реализация решения: язык C или C++.

Разработать алгоритм и написать программу реализующую: интегрирование функций на заданном интервале методом трапеций, расчет остаточного члена. При равном шаге сетки, сравнить полученный результат с результатами полученными иными методами численного интегрирования. Оценить вычислительную сложность.

2: Теория

Метод трапеций представляет собой метод численного интегрирования функции одной переменной, заключающийся в замене на каждом элементарном отрезке подынтегральной функции на многочлен первой степени, то есть линейную функцию. Площадь под графиком функции аппроксимируется прямоугольными трапециями. Алгебраический порядок точности равен 1.

Если отрезок $[a, b]$ является элементарным и не подвергается дальнейшему разбиению, значение интеграла можно найти по формуле :

$$\int_a^b f(x)\,dx = \frac{f(a)+f(b)}{2}(b-a) + E(f), \qquad E(f) = -\frac{f''(\xi)}{12}\,(b-a)^3\,.$$

Это простое применение формулы для площади трапеции — произведение полусуммы оснований, которыми в данном случае являются значения функции в крайних точках отрезка, на высоту (длину отрезка интегрирования). Погрешность аппроксимации можно оценить через максимум второй производной

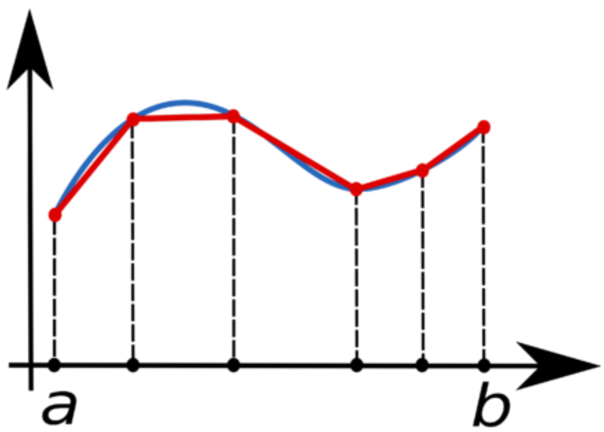
$$|E(f)| \leqslant \frac{(b-a)^3}{12n^2} \max_{x\in[a,b]} |f''(x)|\,, \frac{(b-a)^3}{12n^2} = \frac{nh^3}{12}.$$

Если отрезок $[a, b]$ разбивается узлами интегрирования и на каждом из элементарных отрезков применяется формула трапеций, то суммирование даст составную формулу трапеций :

$$\int_a^b f(x)\,dx \approx \sum_{i=0}^{n-1} \frac{f(x_i)+f(x_{i+1})}{2}(x_{i+1}-x_i) =$$

$$= \frac{f(a)}{2}(x_1 - a) + \sum_{i=1}^{n-2} \frac{f(x_i)}{2}(x_{i+1}-x_{i-1}) + \frac{f(b)}{2}(b - x_{n-1}).$$

$x_j = a + jh, h = (b-a)/N, N-$ четное



3: Код

```cpp
/*
 * Author : Nguyen Le Minh
 * Group : N3251
 * Lab : 9.3
 */
#include <iostream>
#include <cmath>

using namespace std;

double f(double x){
    return pow(x,5) + 2*pow(x,4) + 3*pow(x,3) + 4*pow(x,2) + 5*x + 6;
}

void solver(){
    double a, b, steps;
    cout << "Введите границы a, b и количество разбиений через пробел:\n";
    cin >> a >> b >> steps;
    double step = (b-a)/steps;
    double integral = 0;
    double left;
    for(int i = 0; i < steps - 1; i++) {
        left = f(a + step * i);
        integral += (left * step + (f(a + step * (i + 1)) - left) * step / 2);
    }
    cout << "Интеграл: " << integral;
}

int main(){
    solver();
    return 0;
}
```

4: Выводы программы

Введите границы a, b и количество разбиений через пробел:
-7 7 100000
Интеграл: 14441.1
Сравниваем с значением, которое было получено по методу численного интегрирования : 14444.1
Мы научились средствами языка C++ реализовывать интегрирование функций методом трапеций.

Задание 9.4: Метод Симпсона

1: Задание

Наименование задачи: произвести интегрирование функций на заданном интервале методом Симпсона.

Вид решения: программа и отчёт.

Реализация решения: язык C или C++.

Разработать алгоритм и написать программу реализующую: интегрирование функций на заданном интервале методом Симпсона, расчёт остаточного члена. При равном шаге сетки, сравнить полученный результат с результатами полученными иными методами численного интегрирования. Оценить вычислительную сложность.

2: Теория

Формула Симпсона (также Ньютона-Симпсона) относится к приёмам численного интегрирования.

Суть метода заключается в приближении подынтегральной функции на отрезке $[a, b]$ интерполяционным многочленом второй степени $p_2(x)$, то есть приближение графика функции на отрезке параболой. Метод Симпсона имеет порядок погрешности 4 и алгебраический порядок точности 3.

Формулой Симпсона называется интеграл от интерполяционного многочлена второй степени на отрезке $[a, b]$

$$\int\limits_a^b f(x)dx \approx \int\limits_a^b p_2(x)dx = \frac{b-a}{6}\left(f(a) + 4f\left(\frac{a+b}{2}\right) + f(b)\right),$$

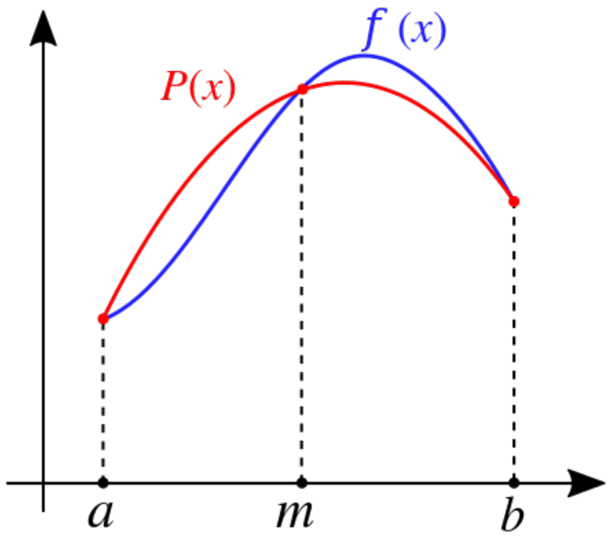
где $f(a), f((a+b)/2)$ и $f(b)$– значения функции в соответствующих точках (на концах отрезка и в его середине).

При условии, что у функции $f(x)$ на отрезке $[a, b]$ существует четвёртая производная, погрешность $E(f)$, согласно найденной Джузеппе Пеано формуле, равна:

$$E(f) = -\frac{(b-a)^5}{2880} f^{(4)}(\zeta), \quad \zeta \in [a, b].$$

В связи с тем, что значение ζ зачастую неизвестно, для оценки погрешности используется следующее неравенство:

$$|E(f)| \leqslant \frac{(b-a)^5}{2880} \max_{x\in[a,b]} \left|f^{(4)}(x)\right|.$$



3: Код

```
/*
 * Author : Nguyen Le Minh
 * Group : N3251
 * Lab : 9.4
 */
#include <iostream>
#include <cmath>

using namespace std;

double f(double x){
    return pow(x,5) + 2*pow(x,4) + 3*pow(x,3) + 4*pow(x,2) + 5*x + 6;
}

void solver(){
    double a, b, eps, I, I1 = 0;
    cout << "Введите границы a, b и желаемую точность через пробел:\n";
    cin >> a >> b >> eps;
    I = eps + 1;
    for (int N = 2; (N <= 4) || (fabs(I1 - I) > eps); N *= 2) {
        double h, sum2 = 0, summ = 0, sum = 0; h = (b - a)/(2*N);
        for (int i = 1; i < 2 * N; i += 2) {
            summ += f(a + h*i);
            sum2 += f(a + h*(i + 1)); }
        sum = f(a) + 4*summ + 2*sum2 - f(b); I = I1;
        I1 = (h / 3) * sum;
    }
    cout << "Интеграл: " << I1 << endl;
}

int main(){
    solver();
    return 0;
}
```

4: Выводы программы

Введите границы a, b и желаемую точность через пробел:
-7 7 100000
Интеграл: 14479.3
Сравниваем с значением, которое было получено по методу численного интегрирования : 14444.1
Мы научились средствами языка C++ реализовывать интегрирование методом Симпсона.

Задание 9.5: Метод Ньютона-Кортеса

1: Задание

Наименование задачи: произвести интегрирование функций на заданном интервале методом Ньютона-Кортеса 3-го и 4-го порядков.

Вид решения: программа и отчёт.

Реализация решения: язык C или C++.

Разработать алгоритм и написать программу реализующую: интегрирование функций на заданном интервале методом Ньютона-Кортеса 3-го и 4-го порядков, расчёт остаточного члена. При равном шаге сетки, сравнить полученный результат с результатами полученными иными методами численного интегрирования. Оценить вычислительную сложность.

2: Теория

Выше были рассмотрены три схожих метода интегрирования функций – метод прямоугольников, метод трапеций, метод Симпсона. Их объединяет общая идея: интегрируемая функция интерполируется на отрезке интегрирования по равноотстоящим узлам многочленом Лагранжа, для которого аналитически вычисляется значение интеграла. Семейство методов, основанных на таком подходе, называется методами Ньютона-Котеса.

В выражении

$$\int_a^b f(x) \approx \sum_{j=1}^{N} c_j f(x_j)$$

Коэффициенты c_j правильнее называть весовыми коэффициентами. Величину

$$\Psi_N = \int_a^b f(x) - \sum_{j=1}^{N} c_j f(x_j)$$

, определяющую погрешность численного интегрирования, называют остатком.

Для семейства методов Ньютона-Котеса можно записать общее выражение:

$$\int_a^b f(x) \approx \frac{n * h}{C_n} \sum_{j=1}^{N} \sum_{i=0}^{n} c_{in} f(x_i)(1)$$

где n – порядок метода Ньютона-Котеса, N – количество частичных отрезков, $h = \frac{x_j - x_{j-1}}{n}$, $c_n = \sum_{j=1}^{N} c_{in}, x_i = x_j + ih$

Из выражения (1) легко можно получить формулу прямоугольников для $n = 0$, , формулу трапеций для $n = 1$, и формулу Симпсона для $n = 2$.

Коэффициенты c_{in} могут быть заданы в табличной форме :

n	C_n	c_{0n}	c_{1n}	c_{2n}	c_{3n}	c_{4n}	c_{5n}
0	**1**	1					
1	**2**	1	1				
2	**6**	1	4	1			
3	**8**	1	3	3	1		
4	**90**	7	32	12	32	7	
5	**288**	19	75	50	50	75	19

3: Код

```
/*
 * Author : Nguyen Le Minh
 * Group : N3251
 * Lab : 9.5
 */
#include <iostream>
#include <cmath>

using namespace std;

double f(double x){
    return pow(x,5) + 2*pow(x,4) + 3*pow(x,3) + 4*pow(x,2) + 5*x + 6;
}

double NewtonCotes(double a, double b, int Degree, int Ndivisions){
    int koef[10][10] = { 1,0,0,0,0,0,0,0,0,0,
                         1,1,0,0,0,0,0,0,0,0,
                         1,4,1,0,0,0,0,0,0,0,
                         1,3,3,1,0,0,0,0,0,0,
                         7,32,12,32,7,0,0,0,0,0,
                         19,75,50,50,75,19,0,0,0,0,
                         41,216,27,272,27,216,41,0,0,0,
                         751,3577,1323,2989,2989,1323,3577,751,0,0,
                         989,5888,-928,10496,-4540,10496,-928,5888,989,0,
                         2857,15741,1080,19344,5778,5778,19344,1080,15741,2857};
    double mltp[10] = { 1,1.0 / 2,1.0 / 3,3.0 / 8,2.0 / 45,5.0 / 288,1.0 /

    140,7.0 / 17280,4.0 / 14175,9.0 / 89600 };
    if ((Degree < 0) || (Degree > 9)) puts("Wrong degree");
    if (a >= b)
        puts("Wrong segments");
    if (Ndivisions < 1) Ndivisions = 1;
    double Sum, PartSum;
    double h = (b - a) / (Degree * Ndivisions); Sum = 0;
    for (int j = 0; j < Ndivisions; j++){
        PartSum = 0;
        for (int i = 0; i <= Degree; i++)
            PartSum += koef[Degree][i] * f(a + (i + j * Degree) * h); Sum += mltp[Degree] * PartSu
    }
    return Sum;
}

void solver(){
    double a, b;
    int Degree, Ndivisions;
    cout << "Введите границы a, b, степень полинома и кол-во отрезков для

    разбиения через пробел: \n";
    cin >> a >> b >> Degree >> Ndivisions;
    cout << NewtonCotes(a, b, Degree, Ndivisions);
}

int main(){
    solver();
    return 0;
}
```

4: Выводы программы

Введите границы a, b, степень полинома и кол-во отрезков для разбиения через пробел:
-7 7 3 100000
14444.3
Сравниваем с значением, которое было получено по методу численного интегрирования : 14444.1

Мы научились средствами языка C++ реализовывать интегрирование функций методом Ньютона-Кортеса.