**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

Факультет прикладной математики и информатики

Стаскевич Виталий Дмитриевич

Вариант 7

Отчет по лабораторной работе №2

**«Приближенное вычисление интегралов»**

студента 2 курса 13 группы

**Преподаватель**

Горбачёва Ю. Н.

Минск 2024

Постановка задачи

**Задание 1:**

Вычислить интеграл точностью , используя составные квадратурные формулы (КФ), указанные в варианте задания, и правило Рунге оценки погрешности. Сравнить полученные приближенные значения интеграла с точным значением *I*.

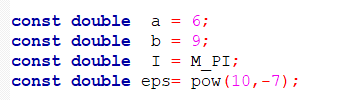
**Задание 2:**

Вычислить приближенное значение интеграла из задания 1, используя квадратурную формулу наивысшей алгебраической степени точности (НАСТ) с k узлами.

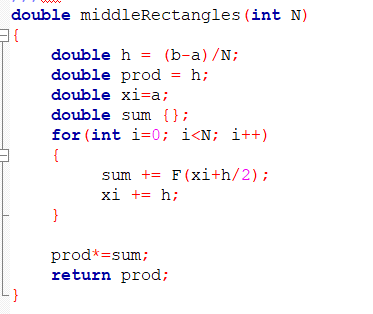
**Вариант 7:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Номер варианта | Определённый интеграл | Составные квадратурные формулы | k | Точное значение I |
| 7 |  | КФ средних прямоугольников, КФ Симпсона. | 5 |  |

Задание 1

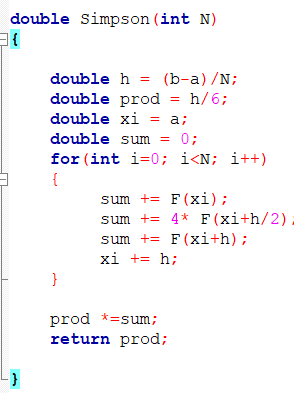


Для вычисления суммы по КФ средних прямоугольников мы используем функцию



, которая реализует формулу

И функция

,

Которая реализует формулу

Для достижения желаемой погрешности эпсилон используется правило Рунге.

Алгоритм правила Рунге:

1. Выбираем точность , первоначальный шаг , Квадратурную формулу Q, k=0;
2. Вычисляем и
3. Вычисляем
4. Если , иначе переходим на шаг 2 и увеличиваем k на 1

Данная функция реализует метод Рунге, а также выводит все данные в файл для дальнейшего построения таблицы:

double Runge(Func func)

{

double h = (b-a)/2;

if(func == middle)

{

ofstream fout("MiddleRect.txt");

fout << "Middle Rectangle Form; ; ; ; ;";

fout << "Divisions;" << "Step;" << "Approximation;" << "Error assesment;" << "Abs error;";

double m = 2;

double Qprev = middleRectangles(2);

fout << "N = 2;" << "h=(b-a)/2;";

fout << "Q\_h = " << Qprev << ";";

fout << "—;" << abs(I-Qprev)<<";";

double Qcurr = middleRectangles(4);

double denom = pow(2,m) - 1;

double R = abs((Qcurr-Qprev)/denom);

fout << "N = 4;" << "h=(b-a)/4;";

fout << "Q\_h/2 = " << Qcurr << ";";

fout << "R\_h/2 = " << R << ";" << abs(I-Qcurr)<<";";

double k = 8;

while(R >=eps)

{

Qprev = Qcurr;

Qcurr = middleRectangles(k);

k\*=2;

R = abs((Qcurr-Qprev)/denom);

fout << "N = " << k << ";" << "h=(b-a)/" << k << ";";

fout << "Q\_h/"<< k/2 << " = " << setprecision(precision) << Qcurr << ";";

fout << "R\_h/"<< k/2 << " = " << R << ";" << abs(I-Qcurr)<<";";

}

fout.close();

return Qcurr;

}

else

{

ofstream fout("Simpson.txt");

fout << "Simpson Form; ; ; ; ;";

fout << "Divisions;" << "Step;" << "Approximation;" << "Error assesment;" << "Abs error;";

double m = 4;

double Qprev = Simpson(2);

fout << "N = 2;" << "h=(b-a)/2;";

fout << "Q\_h = " << Qprev << ";";

fout << "—;" << abs(I-Qprev)<<";";

double Qcurr = Simpson(4);

double denom = pow(2,m) - 1;

double k = 8;

double R = abs((Qcurr-Qprev)/denom);

fout << "N = 4;" << "h=(b-a)/4;";

fout << "Q\_h/2 = " << Qcurr << ";";

fout << "R\_h/2 = " << R << ";" << abs(I-Qcurr)<<";";

while(R >= eps)

{

Qprev = Qcurr;

Qcurr = Simpson(k);

k\*=2;

R = abs((Qcurr-Qprev)/denom);

fout << "N = " << k << ";" << "h=(b-a)/" << k << ";";

fout << "Q\_h/"<< k/2 << " = " << setprecision(precision) << Qcurr << ";";

fout << "R\_h/"<< k/2 << " = " << R << ";" << abs(I-Qcurr)<<";";

}

fout.close();

return Qcurr;

}

}

Таблица для КФ средних прямоугольников:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Middle Rectangle Form | | | | |
| Divisions | Step | Approximation | Error assesment | Abs error |
| N = 2 | h=(b-a)/2 | Q\_h = 3.09839 | — | 0.043206 |
| N = 4 | h=(b-a)/4 | Q\_h/2 = 3.12994 | R\_h/2 = 0.010517 | 0.0116551 |
| N = 8 | h=(b-a)/8 | Q\_h/4 = 3.13861055505 | R\_h/4 = 0.00289099770148 | 0.00298209853666 |
| N = 16 | h=(b-a)/16 | Q\_h/8 = 3.14084248371 | R\_h/8 = 0.000743976217615 | 0.000750169883813 |
| N = 32 | h=(b-a)/32 | Q\_h/16 = 3.14140481381 | R\_h/16 = 0.000187443367173 | 0.000187839782293 |
| N = 64 | h=(b-a)/64 | Q\_h/32 = 3.14154567495 | R\_h/32 = 4.69537132615e-05 | 4.6978642509e-05 |
| N = 128 | h=(b-a)/128 | Q\_h/64 = 3.14158090776 | R\_h/64 = 1.17442705008e-05 | 1.17458310065e-05 |
| N = 256 | h=(b-a)/256 | Q\_h/128 = 3.14158971706 | R\_h/128 = 2.93643335869e-06 | 2.93653093042e-06 |
| N = 512 | h=(b-a)/512 | Q\_h/256 = 3.14159191945 | R\_h/256 = 7.34131208417e-07 | 7.34137305169e-07 |
| N = 1024 | h=(b-a)/1024 | Q\_h/512 = 3.14159247006 | R\_h/512 = 1.83534230924e-07 | 1.83534612397e-07 |
| N = 2048 | h=(b-a)/2048 | Q\_h/1024 = 3.14159260771 | R\_h/1024 = 4.58836468079e-08 | 4.5883671973e-08 |

Таблица для КФ Симпсона:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Simpson Form | | | | |
| Divisions | Step | Approximation | Error assesment | Abs error |
| N = 2 | h=(b-a)/2 | Q\_h = 3.14294 | — | 0.00134873 |
| N = 4 | h=(b-a)/4 | Q\_h/2 = 3.1417 | R\_h/2 = 8.28955e-05 | 0.000105302 |
| N = 8 | h=(b-a)/8 | Q\_h/4 = 3.14159975407 | R\_h/4 = 6.54675445485e-06 | 7.10048164576e-06 |
| N = 16 | h=(b-a)/16 | Q\_h/8 = 3.141593107 | R\_h/8 = 4.43138261463e-07 | 4.53407723811e-07 |
| N = 32 | h=(b-a)/32 | Q\_h/16 = 3.14159268209 | R\_h/16 = 2.83274281365e-08 | 2.84963017627e-08 |

Как можно заметить, КФ Симпсона достигла желаемой погрешности намного быстрее, тем самым можно сделать вывод, что КФ Симпсона является более эффективным методом вычисления приближённого значения интегралов за счёт большей точности и большей алгебраической степени точности КФ.

Задание 2

Для построения КФ наивысшей алгебраической степени точности необходимо знать коэффициенты и узлы . Для их вычисления можно использовать алгоритм, который требует решения системы уравнений

Алгоритм:

1. Решаем СЛАУ, получаем
2. Находим корни уравнения
3. Находим по формуле
4. Строим КФНАСТ по формуле

АСТ формулы равно 2n+1

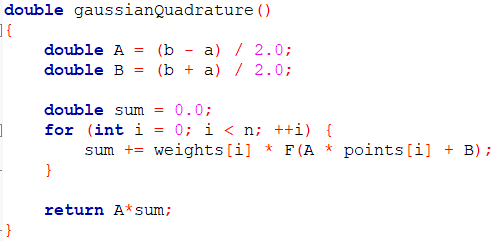
Для n=5 коэффициенты и корни известны и были взяты с сайта <https://pomax.github.io/bezierinfo/legendre-gauss.html>

И отсортированы в порядке возрастания для удобства

const vector <double> points {-0.9061798459386640, -0.5384693101056831, 0.0, 0.0.5384693101056831, 0.9061798459386640};

const vector <double> weights { 0.2369268850561891, 0.4786286704993665, 0.5688888888888889, 0.4786286704993665, 0.2369268850561891};

Эти коэффициенты и корни правильны для промежутка [-1;1]. Но так как наш промежуток [6;9], нам необходимо провести масштабирование с помощью замены Чебышева



В результате выполнения получаем , примерное значение интеграла и разность с действительным значением интеграла.

Вывод

В результате выполнения лабораторной работы мы научились высчитывать приближённые значения интеграла тремя разными квадратурными формулами, научились использовать правило Рунге для оценки погрешности. Научились строить КФНАСТ.

В результате наших вычислений мы можем сделать вывод, что АСТ значительно влияет на точность приближения. В КФ средних прямоугольников АСТ=1, а в КФ Симпсона АСТ=3, и можно увидеть, что погрешность уменьшается значительно быстрее.

КФНАСТ предоставляет также большую точность приближений, однако, она требует б**о**льших вычислительных затрат, так как необходимо решать систему линейных алгебраических уравнений для нахождения коэффициентов, а также требуется решить задачу нахождения корней полинома степени n. Однако, плюсом является то, что эти коэффициенты и корни являются постоянными и не меняются при смене функции или пределов интегрирования (за счёт замены Чебышева), и это значит, что можно вычислить их один раз, и дальнейшие вычисления будут простыми.