МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
Кафедра математического обеспечения и применения ЭВМ

ОТЧЕТ

по практической работе №8

по дисциплине «Вычислительная математика»

Тема: Формула Гаусса

Студент гр. 8383 ____________________ Ларин А.

Преподаватель ____________________ Сучков А.И.

Санкт-Петербург
2019

Цель работы.

Изучение и сравнение различных методов численного интегрирования на примере квадратурной формулы Гаусса.

Основные теоретические положения.

В квадратурной формуле Гаусса

$$\int_{-1}^{1} f(x)dx \approx \sum_{\mathrm{j=1}}^{\mathrm{n}} w_j f(x_j)$$

узлы x_j и веса w_j подобраны так, чтобы формула была точна для всех многочленов степени 2_{n-1}. Для приближенного вычисления интеграла по конечному отрезку $[a, b]$ выполняется замена переменной: $t_j = \frac{a+b}{2} + \frac{b-a}{2} x_j$ тогда квадратурная формула Гаусса принимает вид

$$\int_{a}^{b} f(t)dt \approx \frac{b-a}{2} \sum_{j=1}^{n} w_j f(t_j).$$

Узлы квадратурной формулы Гаусса x_j являются корнями полинома Лежандра:

$$P_n(x) = \frac{1}{2^n n!} \frac{d^n}{dx^n} \ (x^2 - 1)^n.$$

Полином можно выразить рекуррентно:

$$P_{n+1}(x) = \frac{2n+1}{n+1} x P_n(x) - \frac{n}{n+1} P_{n-1}(x),$$

причём первые две функции имеют вид $P_0(x) = 1$ и $P_1(x) = x$.

Производную полинома можно найти как:

$$P_n'(x) = \frac{n}{1-x^2} \left(P_{n-1}(x) - x P_n(x)\right).$$

Корни полином Лежандра можно вычислить итеративно, используя метод Ньютона:

$$x_j^{(k+1)} = x_j^{(k)} - \frac{P_n\left(x_j^{(k)}\right)}{P_n'\left(x_j^{(k)}\right)},$$

причём начальное приближение для j-го корня ($j = \overline{1,n}$) берется по формуле:

$$x_j^{(0)} = cos\left(\pi\ \frac{4j-1}{4n+2}\right).$$

Гауссовы веса вычисляются по формуле:

$$w_j = \frac{2}{\left(1-x_j^2\right)\left(P_n'(x_j)\right)^2}.$$

Если подынтегральная функция достаточно гладкая, то квадратурная формула Гаусса обеспечивает очень высокую точность при небольшом числе узлов, так как для погрешности R_n формулы Гаусса с n узлами справедлива оценка:

$$|R_n| \approx \frac{b-a}{2.5\sqrt{n}}\left(\frac{b-a}{3n}\right)^{2n} \max_{x\in[a,b]}\left|f^{(2n)}(x)\right|.$$

Постановка задачи.

Используя квадратурную формулу Гаусса $(2n-1)$-го порядка точности вычислить приближенное значение заданного интеграла. Порядок выполнения работы следующий:

1. Составить подпрограмму-функцию `F` для вычисления значений подынтегральной функции.
2. Составить подпрограммы-функции `P` и `P1` для вычисления значения полинома Лежандра n-ой степени и его производной соответственно. Данные функции необходимы для вычисления узлов, а также весов в формуле Гаусса. Корни полинома Лежандра можно вычислить используя функцию `NEWTON` из файла methods.cpp.
3. Составить подпрограмму-функцию `GAUSS` для вычисления интеграла по формуле Гаусса.
4. Составить головную программу, содержащую обращение к вычислительным процедурам и осуществляющую печать результатов.
5. Провести вычисления по программе при различных порядках точности.

Выполнение работы.

Вычислим интеграл:

$$\int_0^1 \frac{\sin(x)}{1+x}dx$$

Для нахождения значения определенного интеграла численными методами была составлена функция, вычисляющая подынтегральное выражение, а также три функции, вычисляющие интеграл методом численного интегрирования. График подынтегральной функции $f(x)$ с отмеченными пределами интегрирования представлен на рис. 1.

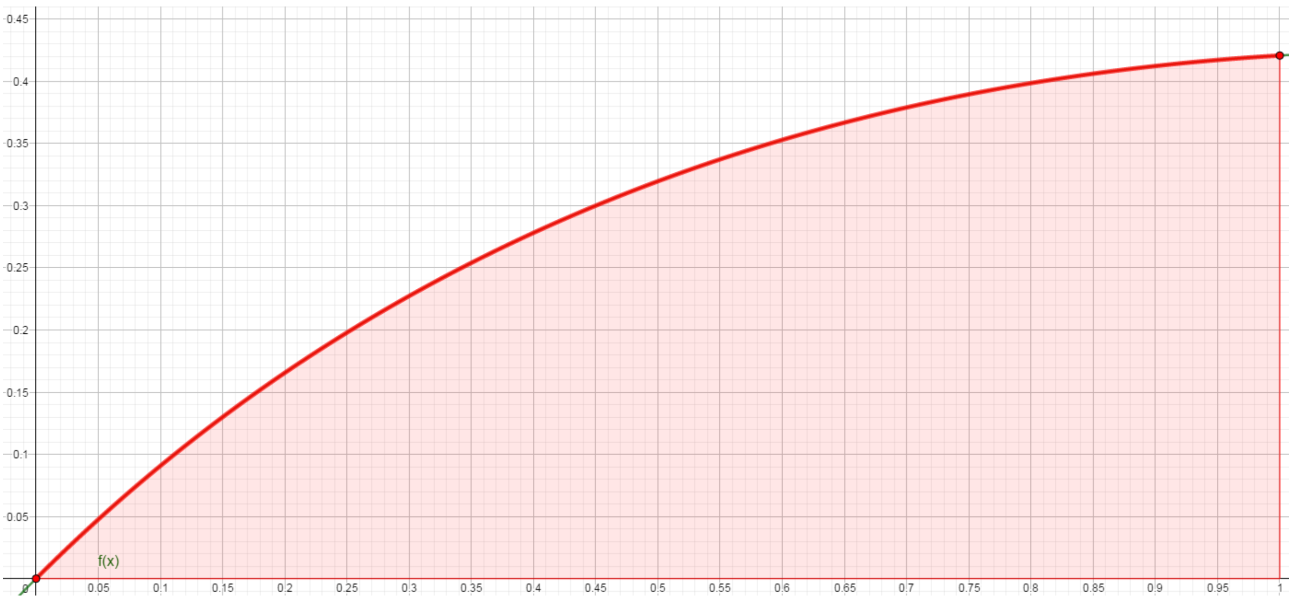

Рисунок 1 – График подынтегральной функции к отмеченными пределами интегрирования

Головная функция и функции для вычисления интеграла по формуле Гаусса предоставлена в приложении А. Была написана функция `p(n,x)`, вычисляющая значение полинома Лежандра степени n в точке x, и функция `pp(n,x)`, вычисляющая ее производную.

Была написана функция `proot(n)`, вычисляющая корни x_j полинома Лежандра n-ной степени методом Ньютона, и соответствующие им веса $w_j = \frac{2}{\left(1-x_j^2\right)\left(P_n'(x_j)\right)^2}$, и возвращающая их в виде списка. Для непосредственнй оценки значения определенного интеграла написана функция `gauss(a,b,e,f,n)`. На вход функция принимает следующие параметры: `a,b` – предел интегрирования $[a, b]$, `f(x)` – интегрируемая функция, `n` – порядок точности; возвращает численное значение определенного интеграла.

Требуется провести вычисления по формуле Гаусса при различных порядках точности n. Оценим абсолютную погрешность результата d при помощи сравнения с эталонным значением I^*, вычисленным по методу Симпсона с большой точностью($\varepsilon = 0.000000000000001$): $I^* = 0.284226985512411$. Результаты вычислений приведены в табл. 1.

Таблица 1 – Вычисление значения интеграла по формуле Гаусса с варьированием значения порядка точности $(2n - 1)$

Значение n	Значение интеграла	Значение d
3	0.284227004619646	1.0e-7
5	0.284226985512425	1.0e-13
8	0.284226985512411	1.0e-15
10	0.284226985512411	1.0e-16
12	0.284226985512411	1.0e-16

Оценим зависимость погрешности результата от порядка точности n. Результаты оценки зависимости точности d расчета значения интеграла I от порядка n приведены в табл. 2 на графике на рис. 2.

Выводы.

Проанализировав результаты расчетов, мы можем сделать вывод, что вычисление определенного интеграла по формуле Гаусса дает неплохую точность результата. По результатам оценки зависимости погрешности от порядка точности видим также, что увеличение порядка на единицу приводит к уменьшению погрешности примерно на полтора порядка. Быстрая сходимость дает данному методу преимущество над рассмотренными ранее методами вычисления интеграла по формулам прямоугольника, трапеции и Симпсона. Из недостатков можно выделить сложность вычисления в связи с большим объемом расчетов, требуемым для вычисления значения, а также большую сложность оценки теоретической погрешности результата $|R_n|$

Таблица 2 – Оценка зависимости погрешности от порядка точности

Значение n	Значение I	Значение d
0	0	2.84E-01
1	0.319617026	3.54E-02
2	0.284889847	6.63E-04
3	0.284248499	2.15E-05
4	0.284227628	6.42E-07
5	0.284227005	1.91E-08
6	0.284226986	5.67E-10
7	0.284226986	1.68E-11
8	0.284226986	4.96E-13
9	0.284226986	1.43E-14
10	0.284226986	1.11E-16
11	0.284226986	5.55E-17
12	0.284226986	5.55E-17

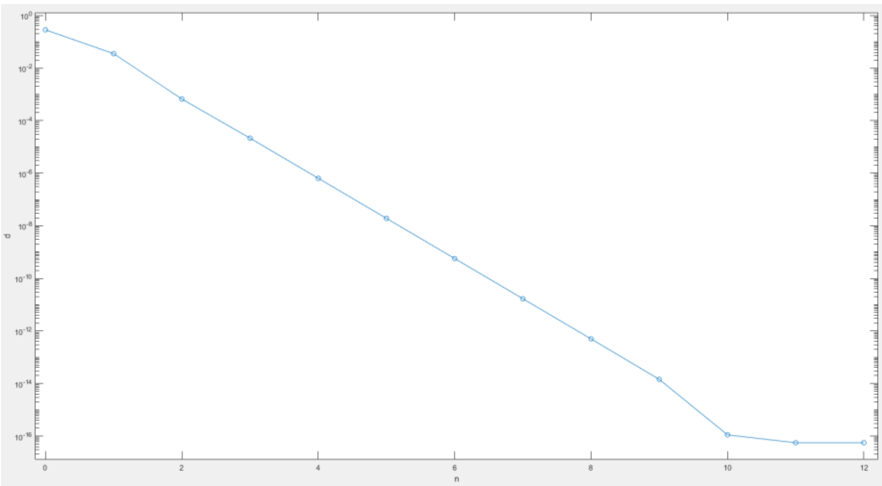

Рисунок 2 – График зависимости погрешности от порядка точности

ПРИЛОЖЕНИЕ А
ИСХОДНЫЙ КОД ПРОГРАММЫ

```
import math
    #math.sin()
    #math.cos()
    #math.acos()
    #math.tan()
    #math.atan()
    #math.exp()
    #math.pi
    #x**2
    #x**(1/2)
    #math.sqrt()

def func(x):                    # Given function
    return math.sin(x)/(1+x)

def p(n,x):
    nn=n-1
    if(n==0):return 1
    if(n==1):return x
    return ((2*nn+1)/(nn+1))*x*(p(n-1,x)) - (nn/(nn+1))*p(n-2,x)

def pn(n):
    return lambda x:p(n,x)

def pp(n,x):
    return n/(1-x**2)*(p(n-1,x)-x*p(n,x))

def ppn(n):
    return lambda x: pp(n, x)

def proot(n):
    lst=[]
    for i in range(1,n+1):
        x0=math.cos(math.pi * (4*i-1)/(4*n+2))
        lst.append(newton(pn(n),ppn(n),x0,1,1.87777,0.01,100,0,0))
    w = [2/((1-x**2)*(pp(n,x))**2) for x in lst]

    return [lst,w]

def _simps(h,n,x0,f):
    m=int(n/2)
    s=0
```

```
    for i in range (m):
        s+=f((x0+(2 * i)*h)) + 4 * f((x0+(2 * i + 1)*h)) +
f((x0+(2 * i + 2)*h))
    s*=h/3
    return s

def simps(a,b,e,f):
    h,h2,Ih,Ih2=(0,0,0,0)
    k = 4
    e0 = e * ((2**k) - 1)
    d = e0 + 1
    n = 1

    Ih2 = _simps(((b - a) / n),n, a, f)
    while d>=e0:
        n*=2
        Ih = Ih2;
        Ih2 = _simps(((b - a) / n), n, a, f);
        d = abs(Ih2 - Ih);
    return [(Ih2 - (Ih2 - Ih) / 15),n]


def gauss(a,b,f,nn):
    def R(n):
        #m = f(i for i in range(a,b,e))
        #m1=max(m)
        r = ((b-a)/(2.5*math.sqrt(n)))*((b-a)/(3*n))**(2*n)
        return n

    x,w=proot(nn)
    t = [(a+b)/2 + ((b-a)/2)*i for i in x]
    s = sum([w[i]*f(t[i]) for i in range(len(t))])
    l1 = len(x)
    l2 = len(w)
    I = ((b-a)/2 * s)
    return I

a=0
b=1
e=0.00000000000000001
s=simps(a,b,e,func)
print(s)

d=[]
gg=[]
nn=[]
```

```
for n in range(22):
    g = gauss(a, b, func, n)
    nn.append(n)
    gg.append(g)
    d.append(abs(g-s[0]))
    print(str(n)+"\t"+str(g)+"\t" +str(abs(g-s[0])))
print(nn)
print(gg)
print(d)
```