Федеральное государственное автономное образовательное учреждение
высшего образования
Национальный исследовательский
университет ИТМО

Факультет программной инженерии и компьютерной техники

Направление подготовки:

Системное и Прикладное Программное Обеспечение

(09.03.04 Программная инженерия)

Дисциплина «Вычислительная математика»

Отчет

По лабораторной работе №3

Вариант №10

Студент

Карташев Владимир Сергеевич,
группа P3215

Преподаватель / Практик

Малышева Татьяна Алексеевна

г. Санкт-Петербург, 2024 г.

Цель работы

Найти приближенное значение определенного интеграла с требуемой точностью различными численными методами.

Описание метода, расчетные формулы

Формула Ньютона - Котеса

Простой прием построения квадратурных формул состоит в том, что подынтегральная функция $f(x)$ заменяется на отрезке $[a, b]$ **интерполяционным многочленом Лагранжа** $L_n(x)$, совпадающий с $f(x)$ в узлах интерполяции $x_0, x_1, \ldots, x_n \in [a, b]$

Полином Лагранжа имеет вид:

Ро́джер Котс (1682-1716) Английский математик, астроном и философ, помощник Исаака Ньютона. «По своим математическим способностям из его поколения в Англии он уступал только Ньютону».

$$L_n(x) = \sum_{i=0}^{n} f(x_i) L_n^i(x), \qquad n = 0, 1, 2 \ldots$$

где $L_n^i(x)$ - коэффициенты Лагранжа (полиномы степени *n*):

$$L_n^i(x) = \frac{(x - x_0)(x - x_1) \ldots (x - x_{i-1})(x - x_{i+1}) \ldots (x - x_n)}{(x_i - x_0)(x_i - x_1) \ldots (x_i - x_{i-1})(x_i - x_{i+1}) \ldots (x_i - x_n)}$$

Если полином Лагранжа «близок» к $f(x)$, то интегралы от них тоже должны быть «близки»:

$$\int_a^b f(x)dx \approx \int_a^b L_n(x)dx = \sum_{i=0}^{n} f(x_i) \int_a^b L_n^i(x)dx$$

Формула Ньютона - Котеса

Вводим коэффициенты Котеса: $c_n^i = \int_a^b L_n^i(x)dx$

Формула Ньютона-Котеса порядка n:

$$\int_a^b \boldsymbol{f(x)dx} \approx \int_a^b \boldsymbol{L_n(x)dx} = \sum_{i=0}^{n} \boldsymbol{f(x_i)c_n^i}$$

Пример: Вычислить коэффициенты Котеса $c_1^0 = c_1^1$

Пусть значения функции $f(x)$ заданы в двух узлах: $x_0 = a$, $x_1 = b$

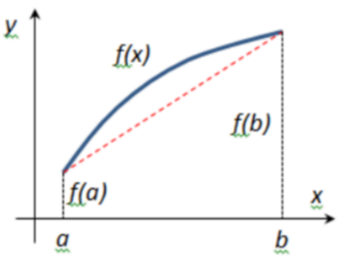
Аппроксимируем функцию полиномом Лагранжа первой степени:

$$f(x) \approx L_1(x) = f(x_0)L_1^0(x) + f(x_1)L_1^1(x) = f(x_0)\frac{x - x_1}{x_0 - x_1} + f(x_1)\frac{x - x_0}{x_1 - x_0}$$

$$= f(a)\frac{x - b}{a - b} + f(b)\frac{x - a}{b - a}$$

$$\int_a^b f(x)dx = \frac{f(a)}{a - b}\int_a^b (x - b)dx + \frac{f(b)}{b - a}\int_a^b (x - a)dx =$$

$$f(a)\frac{b - a}{2} + f(b)\frac{b - a}{2}$$



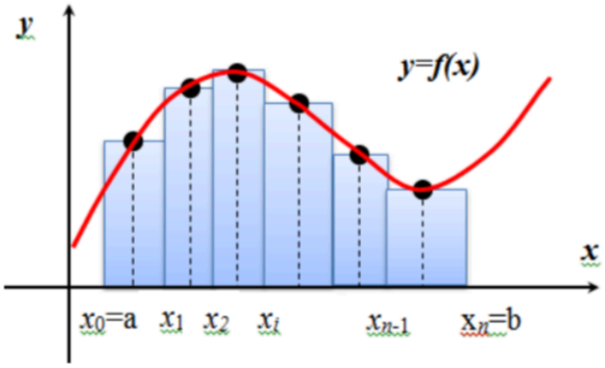
Тогда коэффициенты Котеса $c_1^0 = c_1^1 = \frac{b-a}{2}$

Метод прямоугольников. Метод средних

Для аналитически заданных функций более точным является использование значений в средних точках элементарных отрезков (полуцелых узлах):

$$\int_a^b f(x)dx \approx \sum_{i=1}^n h_i f(x_{i-1/2})$$

$$x_{i-1/2} = \frac{x_{i-1} + x_i}{2} = x_{i-1} + \frac{h_i}{2}, i = 1, 2, \dots n$$



При $h_i = h = \frac{b-a}{n} = const$:

$$\int_a^b f(x)dx = h\sum_{i=1}^n f(x_{i-1/2})$$

Метод трапеций

Подынтегральную функцию на каждом отрезке $[x_i; x_{i+1}]$ заменяют интерполяционным многочленом первой степени:

$$f(x) \approx \varphi_i(x) = a_i x + b$$

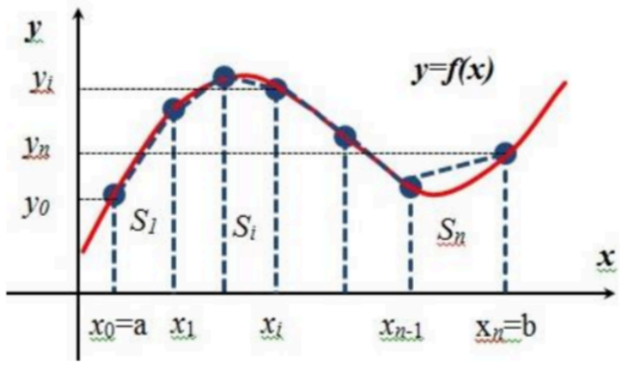
Используют линейную интерполяцию, т.е. график функции $y = f(x)$ представляется в виде ломаной, соединяющий точки (x_i, y_i) . Площадь всей фигуры (криволинейной трапеции):

$$S_{\text{общ}} = S_1 + S_2 + \dots + S_n = \frac{y_0 + y_1}{2}h_1 + \frac{y_1 + y_2}{2}h_2 + \dots + \frac{y_{n-1} + y_n}{2}h_n$$

$$y_0 = f(a), \quad y_n = f(b), \quad y_i = f(x_i), \quad h_i = x_i - x_{i-1}$$

Складывая все эти равенства, получаем формулу трапеций для численного интегрирования:

$$\int_a^b f(x)dx = \frac{1}{2}\sum_{i=1}^n h_i(y_{i-1} + y_i)$$



При $h_i = h = \frac{b-a}{n} = const$ формула трапеций:

$$\int_a^b f(x)dx = h \cdot \left(\frac{y_0 + y_n}{2} + \sum_{i=1}^{n-1} y_i\right)$$

или $$\int_a^b f(x)dx = \frac{h}{2} \cdot \left(y_0 + y_n + 2\sum_{i=1}^{n-1} y_i\right)$$

Метод Симпсона (**Симпсон Томас**(20.08.1710–14.05.1751) – английский математик)

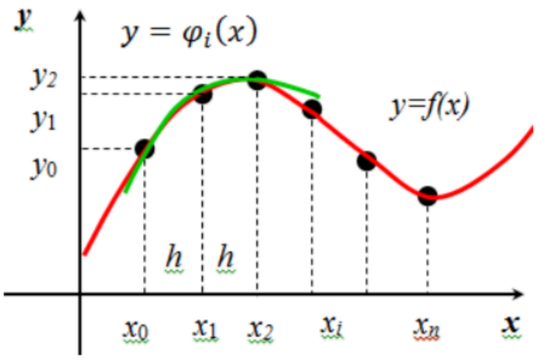
Разобьем отрезок интегрирования [*a*, *b*] на четное число *n* равных частей с шагом *h*.

На каждом отрезке $[x_0, x_2]$, $[x_2, x_4]$, …, $[x_{i-1}, x_{i+1}]$, … , $[x_{n-2}, x_n]$ подынтегральную функцию заменим интерполяционным многочленом второй степени:

$$f(x) \approx \varphi_i(x) = a_i x^2 + b_i x + c_i, \quad x_{i-1} \le x \le x_{i+1}$$

Коэффициенты этих квадратных трехчленов могут быть найдены из условий равенства многочлена и подынтегральной функции в узловых точках.

В качестве $\varphi_i(x)$ можно принять интерполяционный многочлен Лагранжа второй степени, проходящий через точки (x_{i-1}, y_{i-1}), (x_i, y_i), (x_{i+1}, y_{i+1}).



Метод Симпсона

Для точек x_0, x_1, x_2:

$$\varphi_1(x) = \frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)} y_0 + \frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)} y_1 + \frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)} y_2.$$

При $x_0 = 0; x_1 = h;\ x_2 = 2h$, получим:

$$\varphi_1(x) = \frac{(x-h)(x-2h)}{h \cdot 2h} y_0 + \frac{x(x-2h)}{-h \cdot h} y_1 + \frac{x(x-h)}{2h \cdot h} y_2 = \frac{x^2 - x \cdot 3h + 2h^2}{2h^2} y_0 + \frac{x^2 - 2h \cdot x}{-h^2} y_1 + \frac{x^2 - h \cdot x}{2h^2} y_2$$

$$S_1 = \int_{x_0}^{x_0+2h} \varphi_1(x)dx = \int_0^{2h} \left(\frac{x^2 - x \cdot 3h + 2h^2}{2h^2} y_0 + \frac{x^2 - 2h \cdot x}{-h^2} y_1 + \frac{x^2 - h \cdot x}{2h^2} y_2 \right) dx =$$

$$= \frac{y_0}{2h^2}\left(\frac{x^3}{3} - 3h\frac{x^2}{2} + 2h^2 x\right)\Big|_0^{2h} - \frac{y_1}{h^2}\left(\frac{x^3}{3} - 2h\frac{x^2}{2}\right)\Big|_0^{2h} + \frac{y_2}{2h^2}\left(\frac{x^3}{3} - h\frac{x^2}{2}\right)\Big|_0^{2h} = \frac{y_0 h}{3} + \frac{4y_1 h}{3} + \frac{y_2 h}{3}$$
$$= \frac{h}{3}(y_0 + 4y_1 + y_2)$$

Для каждого элементарного отрезка $[x_{i-1}, x_{i+1}]$: $S_i = \frac{h}{3}(y_{i-1} + 4y_i + y_{i+1})$

$$S_{общ} = S_1 + S_2 + \dots + S_n = \frac{h}{3}(y_0 + 4y_1 + 2y_2 + 4y_3 + 2y_4 + \dots + 2y_{n-2} + 4y_{n-1} + 2y_n)$$

Вычислительная реализация задачи

$$\int_2^4 (x^3 - 3x^2 + 7x - 10)dx = (-\frac{1}{4}x^4 - x^3 + \frac{7}{2}x^2 - 10x)|_2^4 = 256/4 - 64 + 7 * 16/2 - 40 - (4 - 8 + 14 - 20) = 26$$

Вычисление по формуле Ньютона-Котеса

n = 6

h = $\frac{b-a}{2}$ = $\frac{4-2}{6}$ = 0.33

x0 = 2

x1 = 2. 33

x2 = 2. 66

x3 = 3

x4 = 3.33

x5 = 3. 66

x6 = 4 - равноотстоящие узлы

$$c_6^0 = c_6^6 = \frac{41(b-a)}{840} \quad c_6^1 = c_6^5 = \frac{216(b-a)}{840} \quad c_6^2 = c_6^4 = \frac{27(b-a)}{840} \quad c_6^3 = \frac{272(b-a)}{840}$$

$$\int_2^4 f(x)dx \approx \sum_{i=0}^{6} f(x_i)c_n^i =$$

f(2)*(41*2/840) + f(2.333)*(216*2/840) + f(2.666)*(27*2/840) + f(3)*(272*2/840) + f(3.333)*(27*2/840) + f(3.666)*(216*2/840) + f(4)*(41*2/840) = 25.988752249807142

Вычисление по формуле средних прямоугольников

n = 10

h = (b - a)/n = 0.2

i	0	1	2	3	4	5	6	7	8	9	10
x_i	2	2.2	2.4	2.6	2.8	3	3.2	3.4	3.6	3.8	4
$x_{i-\frac{1}{2}}$		2.1	2.3	2.5	2.7	2.9	3.1	3.3	3.5	3.7	3.9
$f(x_{i-\frac{1}{2}})$		0.73	2.4	4.38	6.71	9.46	12.66	16.36	20.63	25.48	31

$$\int_2^4 f(x)dx \approx h\sum_{i=0}^{10} f(x_{i-1/2}) =$$

=0.2*(0.73+2.4+4.38+6.71+9.46+12.66+16.36+20.63+25.48+31)=25.962

Вычисление по формуле трапеций

n = 10
h = (b - a)/n = 0.2

i	0	1	2	3	4	5	6	7	8	9	10
x_i	2	2.2	2.4	2.6	2.8	3	3.2	3.4	3.6	3.8	4
y_i	0	1.53	3.34	5.5	8	11	14.45	18.42	22.98	28.15	34

$$\int_2^4 f(x)dx \approx h/2\,(y_0 + y_{10} + 2\sum_{i=0}^{6} y_i) =$$

=0.4/2 * (22 - 6 + 2*(6.52 - 3.05 - 7.9 - 9.16 - 8 - 5.55 - 2.97 - 1.42 - 2.05 - 6))=26.02

Вычисление по формуле Симпсона

i	0	1	2	3	4	5	6	7	8	9	10
x_i	2	2.2	2.4	2.6	2.8	3	3.2	3.4	3.6	3.8	4
y_i	0	1.53	3.34	5.5	8	11	14.45	18.42	22.98	28.15	34

$$\int_2^4 f(x)dx \approx h/3\,(y_0 + 4(y_1 + y_3 + \ \dots\ + y_{n-1}) \ + 2(y_2 + y_4 + \dots + y_{n-2}) + y_n) =$$

=(0.4/3) * (22 + 4*(6.52 -7.9 - 8 - 2.97 - 2.05) + 2*(-3.05 -9.16 - 5.55 - 1.42)- 6)=25.9959

Погрешность использованных методов

Метод	Относительная погрешность
Ньютона-Котеса	0.04%
Средних прямоугольников	0.15%
Трапеций	0.08%
Симпсона	0.02%

Листинг программы

Метод прямоугольников:

```
func RectangleMethod(f func(float64) float64, a, b, e float64, n int, h float64) (int,
func(f func(float64) float64, a float64, b float64, e float64, n int, h float64) float64)
{
   fmt.Println("Выберите модификацию для интегрирования:")
   fmt.Println("1. Левые")
   fmt.Println("2. Средние")
   fmt.Println("3. Правые")

   var integral float64

   var selectionMode string
   for {
      fmt.Println("Выберите модификацию (1-3):")
      fmt.Fscan(os.Stdin, &selectionMode)
      switch selectionMode {
      case "1":
         fmt.Println("Выбрана модификация - левые")
         return 1, LeftMethod
      case "2":
         fmt.Println("Выбрана модификация - средние")
         return 2, MiddleMethod
         fmt.Println(integral)
      case "3":
         fmt.Println("Выбрана модификация - правые")
         return 1, RightMethod
         fmt.Println(integral)
      default:
         fmt.Println("Укажите число, соответствующее предложенному варианту!")
         continue
      }
      break
   }
   return 0, nil
}

func LeftMethod(f func(float64) float64, a, b, e float64, n int, h float64) float64 {
   var sumY float64

   sumY = 0
   x := a
   for i := 0; i < int(n); i++ {
      sumY = f(x) + sumY
      x = x + h
   }
   sumY = sumY * h
   return sumY
}

func RightMethod(f func(float64) float64, a, b, e float64, n int, h float64) float64 {
   var sumY float64

   sumY = 0
   x := a + h
   for i := 1; i <= int(n); i++ {
      sumY = f(x) + sumY
      x = x + h
   }
   return sumY * h
}

func MiddleMethod(f func(float64) float64, a, b, e float64, n int, h float64) float64 {
   var sumY float64

```

```
    sumY = 0
    x := a + (h / 2)
    for i := 0; i < int(n); i++ {
        sumY = f(x) + sumY
        x = x + h
    }
    return sumY * h
}
```

Метод трапеций:

```
func TrapezoidMethod(f func(float64) float64, a, b, e float64, n int, h float64) float64 {
    var result float64
    var x float64

    result = (f(a) + f(b)) / 2
    x = a + h
    for i := 1; i < int(n); i++ {
        result = f(x) + result
        x = x + h
    }

    return result * h
}
```

Метод Симпсона:

```
func SimpsonMethod(f func(float64) float64, a, b, e float64, n int, h float64) float64 {
    var integral float64
    integral += f(a) + f(b) // начальные и конечные точки

    for i := 1; i < n; i++ {
        x := a + float64(i)*h
        if i%2 == 0 {
            integral += 2 * f(x)
        } else {
            integral += 4 * f(x)
        }
    }

    integral *= h / 3.0

    return integral
}
```

Функция вычисления:

```
func compute(k int, method func(f func(float64) float64, a float64, b float64, e float64,
n int, h float64) float64, f func(float64) float64, a, b, e float64, n int, h float64) {
    var (
        I_h   float64
        I_h_2 float64
    )
    maxLimit := 250
    for i := n; i <= maxLimit; i++ {
        I_h = method(f, a, b, e, n, h)
        h = h / 2
        n = n * 2
        I_h_2 = method(f, a, b, e, n, h)

        if math.Abs(utils.RungeRule(I_h, I_h_2, k)) <= e {
            fmt.Println()
            fmt.Println("e :", e)
            fmt.Println("(I_h_2 - I_h) / (2^k - 1) :", utils.RungeRule(I_h, I_h_2, k))
            fmt.Println("(I_h_2 - I_h) / (2^k - 1) <= e")
```

```
            //n = i
            break
        }

        if i == maxLimit {
            fmt.Println("Число итераций превысило лимит:", maxLimit)
            break
        }
    }

    fmt.Println("Вычисленный интеграл:")
    fmt.Println(I_h)

    //fmt.Println(I_h_2)

    fmt.Println("Число разбиения интервала интегрирования для достижения требуемой
точности:")
    fmt.Println(n)
}
```

Правило Рунге:

```
func RungeRule(I_h, I_h_2 float64, k int) float64 {
    if k <= 0 {
        return math.Abs(I_h_2 - I_h)
    } else {
        return math.Abs(I_h_2-I_h) / (math.Pow(2, float64(k)) - 1)
    }
}
```

Примеры и результаты работы программы

```
Выберите уравнение для интегрирования:
1. 5x^3 - 2x^2 + 3x - 15
2. 2x^3 - 9x^2 - 7x + 11
3. 1x^3 + 2x^2 - 3x - 12
4. 4x^3 - 5x^2 + 6x - 7
5. 3x^3 + 5x^2 + 3x - 6
6. 2x^3 - 3x^2 - 5x + 27
7. 1 / sqrt(x)
8. 1 / (1 - x)
Выберите уравнение (1-8):
1
Выбрано уравнение:
Введите a:
1
a = 1
Введите b:
3
b = 3
Введите e:
0.01
e = 0.01
n = 4
Выберите метод для интегрирования:
1. Метод прямоугольников
2. Метод трапеции
3. Метод Симпсона
Выберите метод (1-3):
1
Выбран метод прямоугольников
Выберите модификацию для интегрирования:
1. Левые
2. Средние
3. Правые
Выберите модификацию (1-3):
1
Выбрана модификация - левые

e : 0.01
(I_h_2 - I_h) / (2^k - 1) : 0.0073238015169749815
(I_h_2 - I_h) / (2^k - 1) <= e
Вычисленный интеграл:
64.65201878547668
Число разбиения интервала интегрирования для достижения требуемой точности:
16384
```

```
Выберите уравнение (1-8):
2
Введите a:
1
a = 1
Введите b:
3
b = 3
Введите e:
0.01
e = 0.01
n = 4
Выберите метод для интегрирования:
1. Метод прямоугольников
2. Метод трапеции
3. Метод Симпсона
Выберите метод (1-3):
```

```
3
Выбран метод Симпсона

e : 0.01
(I_h_2 - I_h) / (2^k - 1) : 0
(I_h_2 - I_h) / (2^k - 1) <= e
Вычисленный интеграл:
-44
Число разбиения интервала интегрирования для достижения требуемой точности:
8
```

Выводы

В результате выполнения данной лабораторной работы были изучены методы для приближённого вычисления значений определенного интеграла.