Университет ИТМО
Мегафакультет компьютерных технологий и управления
Факультет программной инженерии и компьютерной техники

ЛАБОРАТОРНАЯ РАБОТА №5

ИНТЕРПОЛЯЦИЯ ФУНКЦИИ

Вариант №10

Группа: P3211
Студент: Орчиков Даниил Валерьевич
Преподаватель: Малышева Татьяна Алексеевна

г. Санкт-Петербург
2024

Оглавление

Цель работы

Решить задачу интерполяции, найти значения функции при заданных значениях аргумента, отличных от узловых точек.

Вычислительная реализация задачи:

Рабочие формулы

Конечные разности

- Первого порядка $\Delta y_i = y_{i+1} - y_i$
- Второго порядка $\Delta^2 y_i = \Delta y_{i+1} - \Delta y_i$
- ...
- k-го порядка $\Delta^k y_i = \Delta^{k-1} y_{i+1} - \Delta^{k-1} y_i$

Первая интерполяционная формула Ньютона

$$t = \frac{x - x_i}{h}, \qquad i = \left[\frac{x - a}{h}\right]$$

$$N_n(x) = y_i + t\Delta y_i + \frac{t(t-1)}{2!}\Delta^2 y_i + \cdots + \frac{t(t-1)\ldots(t-n+1)}{n!}\Delta^n y_i$$

Вторая интерполяционная формула Ньютона

$$t = \frac{x - x_i}{h}, \qquad i = \left[\frac{x - a}{h}\right] + 1$$

$$N_n(x) = y_i + t\Delta y_{i-1} + \frac{t(t+1)}{2!}\Delta^2 y_{i-2} + \cdots + \frac{t(t+1)\ldots(t+n-1)}{n!}\Delta^n y_0$$

Интерполяционные формулы Гаусса

$$t = \frac{x - a}{h}$$

Первая интерполяционная формула Гаусса $(x > a)$

$$P_n(x) = y_0 + t\Delta y_0 + \frac{t(t-1)}{2!}\Delta^2 y_{-1} + \frac{(t+1)t(t-1)}{3!}\Delta^3 y_{-1} + \frac{(t+1)t(t-1)(t-2)}{4!}\Delta^4 y_{-2} + \frac{(t+2)(t+1)t(t-1)(t-2)}{5!}\Delta^5 y_{-2} + \cdots + \frac{(t+n-1)\ldots(t-n+1)}{(2n-1)!}\Delta^{2n-1} y_{-(n-1)} + \frac{(t+n-1)\ldots(t-n)}{(2n)!}\Delta^{2n} y_{-n}$$

Вторая интерполяционная формула Гаусса $(x < a)$

$$P_n(x) = y_0 + t\Delta y_{-1} + \frac{t(t+1)}{2!}\Delta^2 y_{-1} + \frac{(t+1)t(t-1)}{3!}\Delta^3 y_{-2} + \frac{(t+1)(t+1)t(t-1)}{4!}\Delta^4 y_{-2} + \cdots + \frac{(t+n-1)\ldots(t-n+1)}{(2n-1)!}\Delta^{2n-1} y_{-n} + \frac{(t+n)(t+n-1)\ldots(t-n+1)}{(2n)!}\Delta^{2n} y_{-n}$$

Решение

x	y
2,10	3,7578
2,15	4,1861
2,20	4,9218
2,25	5,3487
2,30	5,9275
2,35	6,4193
2,40	7,0839

$X_1 = 2.355, \quad X_2 = 2.254, \quad a = 2.25$

Конечные разности:

	x_i	y_i	Δy_i	$\Delta^2 y_i$	$\Delta^3 y_i$	$\Delta^4 y_i$	$\Delta^5 y_i$	$\Delta^6 y_i$
0	2.1	3.7578	0.4283	0.3074	-0.6162	1.0769	-1.7765	2.9748
1	2.15	4.1861	0.7357	-0.3088	0.4607	-0.6996	1.1983	
2	2.2	4.9218	0.4269	0.1519	-0.2389	0.4987		
3	2.25	5.3487	0.5788	-0.087	0.2598			
4	2.3	5.9275	0.4918	0.1728				
5	2.35	6.4193	0.6646					
6	2.4	7.0839						

Интерполяционная формула Ньютона

Используем вторую формулу, так как X_1 смещен относительно середины отрезка вправо

$$i = \left[\frac{2.355 - 2.1}{0.05}\right] + 1 = 6$$

$$t = \frac{2.355 - 2.4}{0.05} = -0.9$$

$$N_7(x) = 7.0839 - 0.9 * 0.6646 + \frac{-0.9 * 0.1}{2} 0.1728 + \frac{-0.9 * 0.1 * 1.1}{6} + \frac{-0.9 * 0.1 * 1.1 * 2.1}{24} * 0.4987 + \frac{-0.9 * 0.1 * 1.1 * 2.1 * 3.1}{120} * 1.1983 + \frac{-0.9 * 0.1 * 1.1 * 2.1 * 3.1 * 4.1}{720} * 2.9748 = 6.4431$$

Интерполяционная формула Гаусса

	x_i	y_i	Δy_i	$\Delta^2 y_i$	$\Delta^3 y_i$	$\Delta^4 y_i$	$\Delta^5 y_i$	$\Delta^6 y_i$
-3	2.1	3.7578	0.4283	0.3074	-0.6162	1.0769	-1.7765	2.9748
-2	2.15	4.1861	0.7357	-0.3088	0.4607	-0.6996	1.1983	
-1	2.2	4.9218	0.4269	0.1519	-0.2389	0.4987		
0	2.25	5.3487	0.5788	-0.087	0.2598			
1	2.3	5.9275	0.4918	0.1728				
2	2.35	6.4193	0.6646					
3	2.4	7.0839						

$$t = \frac{2.254 - 2.25}{0.05} = 0.08$$

Используем первую формулу, так как $x > a$

$$P_7(x) = 5.3487 + 0.08 * 0.5788 + \frac{0.08(0.08-1)}{2} * 0.1519 + \frac{(0.08+1)0.08(0.08-1)}{6} * (-0.2389) + \frac{(0.08+1)0.08(0.08-1)(0.08-2)}{24} * (-0.6996) + \frac{(0.08+2)(0.08+1)0.08(0.08-1)(0.08-2)}{120} * 1.1983 + \frac{(0.08+2)(0.08+1)0.08(0.08-1)(0.08-2)(0.08-3)}{720} * 2.9748 = 5.3875$$

Программная реализация задачи

Листинг программы

Представлен только код, непосредственно выполняющий вычисления

Весь код можно посмотреть [тут (GitHub)]

```
function Lagrange(x) {
  let Ln = 0
  let q = []
  for (let i = 0; i < XY.length; i++) {
    let li = XY[i][1]
    q.push(XY[i][1])
    for (let j = 0; j < XY.length; j++)
      if (i !== j) {
        li *= (x - XY[j][0]) / (XY[i][0] - XY[j][0])
        q[q.length - 1] += `(x - ${XY[j][0]}) / (${XY[i][0]} - ${XY[j][0]})`
      }
    Ln += li
```

```
    }
    calculator.setExpression({id: 'graph4', latex: q.join("+")})
    return Ln
}

function separated_differences(x) {
    let f = []
    f.push([])
    for (let i = 0; i < XY.length; i++)
        f[0].push(XY[i][1])
    for (let i = 1; i < XY.length; i++) {
        let last = f[f.length - 1]
        let current = []
        for (let k = 0; k < last.length - 1; k++)
            current.push((last[k + 1] - last[k]) / (XY[i + k][0] - XY[k][0]))
        f.push(current)
    }
    let N = f[0][0]
    let q = [getting_rid_of_scientific_notation_no_arr(f[0][0])]
    for (let k = 1; k < f.length; k++) {
        let n = f[k][0]
        q.push(`(${getting_rid_of_scientific_notation_no_arr(f[k][0])})`)
        for (let j = 0; j < k; j++) {
            n *= (x - XY[j][0])
            q[q.length - 1] += `(x-(${XY[j][0]}))`
        }
        N += n
    }
    calculator.setExpression({id: 'graph3', latex: q.join("+")})
    return N
}

function finite_differences(x) {
    let f = []
    f.push([])
    for (let i = 0; i < XY.length; i++)
        f[0].push(XY[i][1])
    for (let i = 1; i < XY.length; i++) {
        let last = f[f.length - 1]
        let current = []
        for (let k = 0; k < last.length - 1; k++)
            current.push((last[k + 1] - last[k]))
        f.push(current)
    }
    {
        let table_h = document.querySelector(`#finite_differences thead`)
        let table_b = document.querySelector(`#finite_differences tbody`)
        let t_h = "<tr><th></th><th>xi</th><th>yi</th>"
        for (let j = 1; j < XY.length; j++)
            t_h += `<th>△${j}yi</th>`
        t_h += "</tr>"
        let t_b = "<tr>"
        for (let i = 0; i < XY.length; i++) {
            t_b += `<td>${i}</td><td>${XY[i][0]}</td>`
            for (let j = 0; j < f[i].length; j++) {
                t_b += `<th>${Math.round(f[j][i] * 10000) / 10000}</th>`
            }
            t_b += "</tr>"
        }
        table_h.innerHTML = t_h
        table_b.innerHTML = t_b
    }

    let mn = Infinity
    let mx = -Infinity
    XY.forEach(xy => {
        mn = Math.min(mn, xy[0])
        mx = Math.max(mx, xy[0])
    })
```

```
  let h = XY[1][0] - XY[0][0]
  let Nn = 0
  let Q = []
  if ((mn + mx) / 2 > x) {
    let i = Math.floor((x - mn) / h)
    let t = (x - XY[i][0]) / h
    let t_str = `(x-${getting_rid_of_scientific_notation_no_arr(XY[i][0])})/${h}`
    for (let j = 0; j < f.length - i; j++) {
      let s = f[j][i] / factorial(j)
      console.log(f[j][i])
      let q1 = `${getting_rid_of_scientific_notation_no_arr(f[j][i])}/${getting_rid_of_scientific_notation_no_arr(factorial(j))}`
      for (let q = 0; q < j; q++) {
        s *= (t - q)
        q1 += `*(${t_str} - ${q})`
      }
      Nn += s
      Q.push(q1)
    }
    calculator.setExpression({id: 'graph5', latex: Q.join("+")})
    return [Nn, 1]
  } else {
    let i = Math.floor((x - mn) / h) + 1
    let t = (x - XY[i][0]) / h
    let t_str = `(x-${getting_rid_of_scientific_notation_no_arr(XY[i][0])})/${h}`
    for (let j = 0; j <= i; j++) {
      let s = f[j][i - j] / factorial(j)
      console.log(f[j][i - j])
      let q1 = `${getting_rid_of_scientific_notation_no_arr(f[j][i-j])}/${getting_rid_of_scientific_notation_no_arr(factorial(j))}`
      for (let q = 0; q < j; q++) {
        s *= (t + q)
        q1 += `*(${t_str} + ${q})`
      }
      Nn += s
      Q.push(q1)
    }
    calculator.setExpression({id: 'graph5', latex: Q.join("+")})
    return [Nn, 2]
  }
}

function getting_rid_of_scientific_notation_no_arr(x) {
  let w = x.toString()
  let parts = w.split('e')
  if (parts.length > 1)
    return `${parts[0]}*10^{${parts[1]}}`
  return `${parts[0]}`
}

function factorial(n) {
  if (n <= 0)
    return 1
  return n * factorial(n - 1)
}
```

Примеры и результаты работы программы

Пример 1

Выберите файл Файл не выбран

Введите точки

```
-10 2.5440211108893696
-8.947368421052632
1.5405200963877446
-7.894736842105264
1.0008303785480153
-6.842105263157895 1.469729184531725
-5.7894736842105265
2.4738975258547855
-4.736842105263158 2.999701037239482
-3.6842105263157894
2.5163795995000284
-2.6315789473684212
1.5118107911335246
-1.578947368421053 1.000033219555871
-0.526315789473685
1.4976488453964867
0.5263157894736832
2.502351154635118
```

Исходная функция: sin(x)+2

Количество точек = 20

Диапазон: a = -10 , b = 10

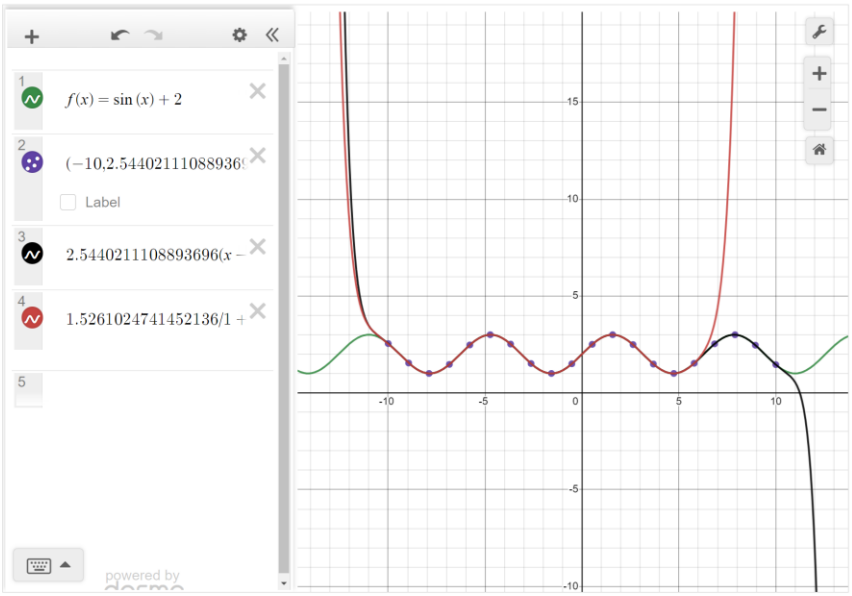
Показывать функцию

Случайные промежутки

Загрузить точки по функции

Введите x: 5

Рассчитать



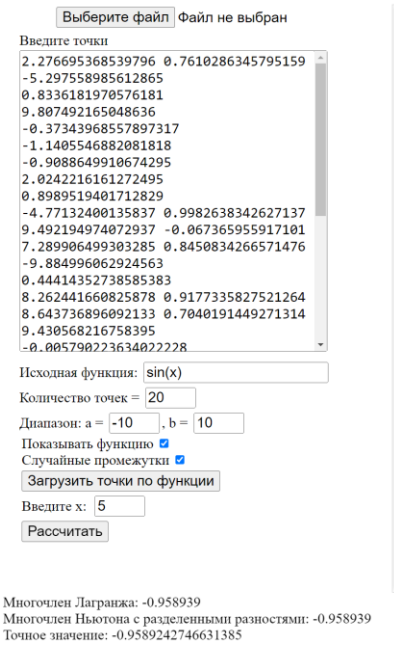
Многочлен Лагранжа: 1.041076

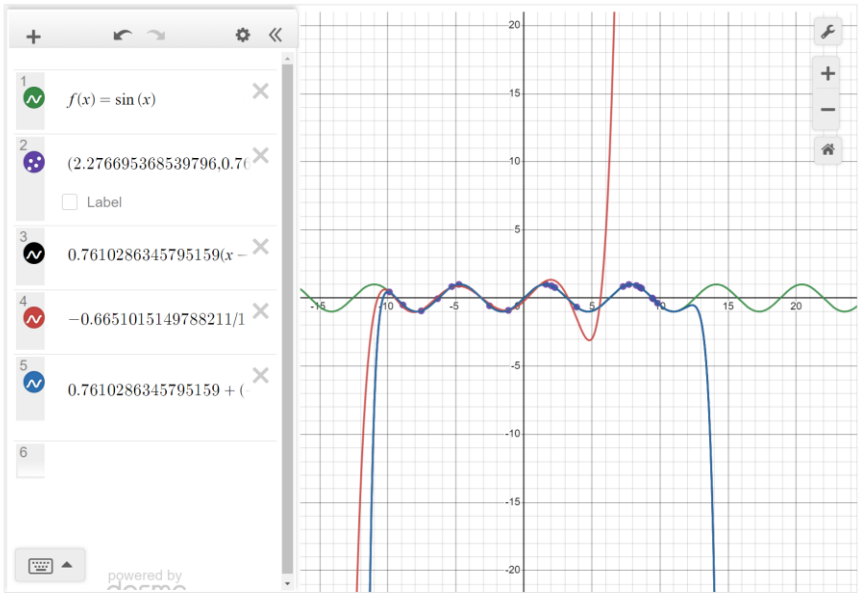
Формула Ньютона для интерполирования назад: 1.039186

Точное значение: 1.0410757253368614

	xi	yi	△1yi	△2yi	△3yi	△4yi	△5yi	△6yi	△7yi	△8yi	△9yi	△10yi	△11yi	△12yi	△13yi	△14yi	△15yi	△16yi	△17yi	△18yi	△19yi
0	-10	2.544	-1.0035	0.4638	0.5448	-1.0181	0.4778	0.5454	-1.0328	0.492	0.5459	-1.0477	0.5065	0.5463	-1.0627	0.5213	0.5465	-1.0779	0.5364	0.5466	-1.0932
1	-8.947368421052632	1.5405	-0.5397	1.0086	-0.4733	-0.5403	1.0232	-0.4874	-0.5408	1.0379	-0.5018	-0.5412	1.0528	-0.5165	-0.5414	1.0678	-0.5314	-0.5415	1.083	-0.5466	
2	-7.894736842105264	1.0008	0.4689	0.5353	-1.0136	0.4829	0.5358	-1.0282	0.4971	0.5361	-1.043	0.5116	0.5364	-1.0579	0.5264	0.5365	-1.0729	0.5415	0.5364		
3	-6.842105263157895	1.4697	1.0042	-0.4784	-0.5308	1.0186	-0.4925	-0.5311	1.0332	-0.5069	-0.5313	1.048	-0.5215	-0.5314	1.0629	-0.5365	-0.5314	1.0779			
4	-5.7894736842105265	2.4739	0.5258	-1.0091	0.4879	0.5262	-1.0236	0.5021	0.5264	-1.0382	0.5166	0.5265	-1.053	0.5314	0.5264	-1.0678	0.5465				
5	-4.736842105263158	2.9997	-0.4833	-0.5212	1.014	-0.4974	-0.5215	1.0285	-0.5118	-0.5216	1.0431	-0.5265	-0.5215	1.0579	-0.5414	-0.5213					
6	-3.6842105263157894	2.5164	-1.0046	0.4928	0.5166	-1.0189	0.507	0.5167	-1.0334	0.5216	0.5166	-1.048	0.5364	0.5165	-1.0627						
7	-2.6315789473684212	1.5118	-0.5118	1.0094	-0.5023	-0.5119	1.0237	-0.5167	-0.5118	1.0382	-0.5313	-0.5116	1.0528	-0.5463							
8	-1.578947368421053	1	0.4976	0.5071	-1.0142	0.5119	0.507	-1.0285	0.5264	0.5069	-1.043	0.5412	0.5065								
9	-0.526315789473685	1.4976	1.0047	-0.5071	-0.5023	1.0189	-0.5215	-0.5021	1.0332	-0.5361	-0.5018	1.0477									
10	0.5263157894736832	2.5024	0.4976	-1.0094	0.5166	0.4974	-1.0236	0.5311	0.4971	-1.0379	0.5459										
11	1.5789473684210513	3	-0.5118	-0.4928	1.014	-0.5262	-0.4925	1.0282	-0.5408	-0.492											
12	2.6315789473684212	2.4882	-1.0046	0.5212	0.4879	-1.0186	0.5358	0.4874	-1.0328												
13	3.6842105263157894	1.4836	-0.4833	1.0091	-0.5308	-0.4829	1.0232	-0.5454													
14	4.7368421052631575	1.0003	0.5258	0.4784	-1.0136	0.5403	0.4778														
15	5.789473684210526	1.5261	1.0042	-0.5353	-0.4733	1.0181															
16	6.842105263157894	2.5303	0.4689	-1.0086	0.5448																
17	7.894736842105264	2.9992	-0.5397	-0.4638																	
18	8.94736842105263	2.4595	-1.0035																		
19	10	1.456																			

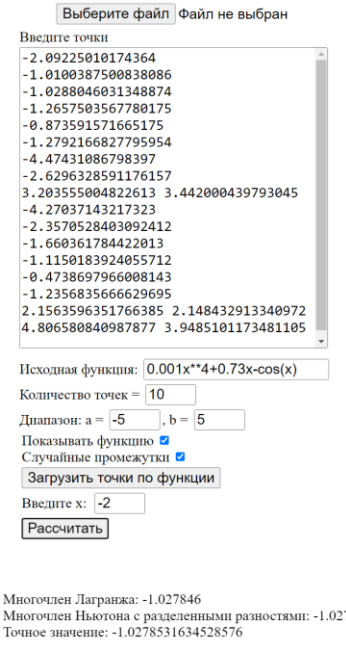
Пример 2

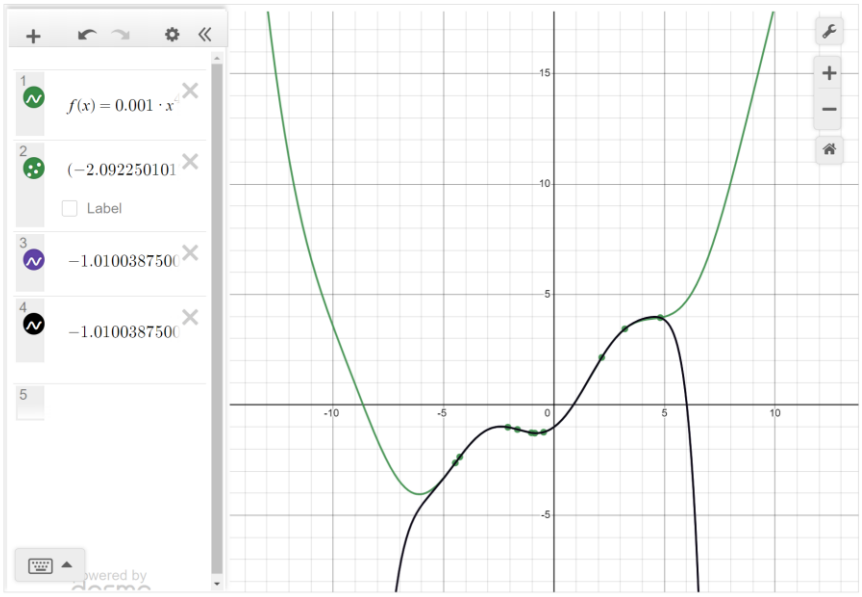




Многочлен Лагранжа: -0.958939
Многочлен Ньютона с разделенными разностями: -0.958939
Точное значение: -0.9589242746631385

Пример 3





Многочлен Лагранжа: -1.027846
Многочлен Ньютона с разделенными разностями: -1.027846
Точное значение: -1.0278531634528576

Вывод

Во время выполнения данной лабораторной работы я познакомился с разными методами интерполяции и реализовал некоторые из них на языке JavaScript.