Университет ИТМО

Факультет программной инженерии и компьютерной техники

Лабораторная работа №5 по "Вычислительной математике"

Интерполяция функции

Вариант 4

Выполнил: Дьяконов Михаил Павлович
Группа: P3211
Преподаватель: Малышева Татьяна Алексеевна

Университет ИТМО, 2021

Цель работы

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

Рабочие формулы

Метод Лагранжа:

$$Ln(x) = \sum_{i=0}^{n} y_i \frac{(x-x_0)(x-x_1)\ldots(x-x_{i-1})(x-x_{i+1})\ldots(x-x_n)}{(x_i-x_0)(x_i-x_1)\ldots(x_i-x_{i-1})(x_i-x_{i+1})\ldots(x_i-x_n)}$$

Метод Ньютона с разделенными разностями:

$$f(x_i, x_{i+1}, \ldots, x_{i+k}) = \frac{f(x_{i+1}, \ldots, x_{i+k}) - f(x_i, x_{i+k-1})}{x_{i+k} - x_i}$$

$$N_n(x) = f(x_0) + f(x_0,\ x_1)\cdot(x - x_0) + f(x_0,\ x_1, x_2)\cdot(x - x_0)\cdot(x - x_1) + \cdots + f(x_0,\ x_1, \ldots, x_n)\cdot(x - x_0)\cdot(x - x_1)\ldots(x - x_{n-1})$$

Оценка погрещности:

$$R_n(x) \le \frac{M^{(n+1)}(x)}{(n+1)!}|(x - x_0)(x - x_1)\ldots.(x - x_n)|$$

$$M^{(n+1)}(x) = \max_{x\in[x_0;x_n]} f^{n+1}(x)$$

Листинг программы

LagrangeMethod.kt:

```
class LagrangeMethod : InterpolationMethod {

    override fun calculate(points: Array<Point>, x: Double): Double {
        val lCfs: DoubleArray = points.mapIndexed {i: Int, p: Point ->
            val filteredPoints: Array<Point> = points.filterIndexed { index, _ -> index != i
                }.toTypedArray()
            p.y * filteredPoints.map { x - it.x }.reduce {a, b -> a * b } /
                    filteredPoints.map { p.x - it.x }.reduce {a, b -> a * b}
        }.toDoubleArray()
        return lCfs.sum()
    }
}
```

NewtonMethod.kt:

```
class NewtonMethod : InterpolationMethod {

    override fun calculate(points: Array<Point>, x: Double): Double {
        fun f(i1: Int, i2: Int): Double {
            if (i1 == i2) return 0.0
            return if (i2 - i1 == 1) (points[i2].y - points[i1].y) / (points[i2].x - points[i1].x)
            else (f(i1 + 1, i2) - f(i1, i2 - 1)) / (points[i2].x - points[i1].x)
        }

        return points[0].y + points.mapIndexed { i: Int, _: Point ->
            f(0, i) * points.mapIndexed { index, point -> if (index < i) x - point.x else 1.0 }.reduce
                { a, b -> a * b }
        }.toDoubleArray().sum()
    }
}
```

FaultCalculator.kt:

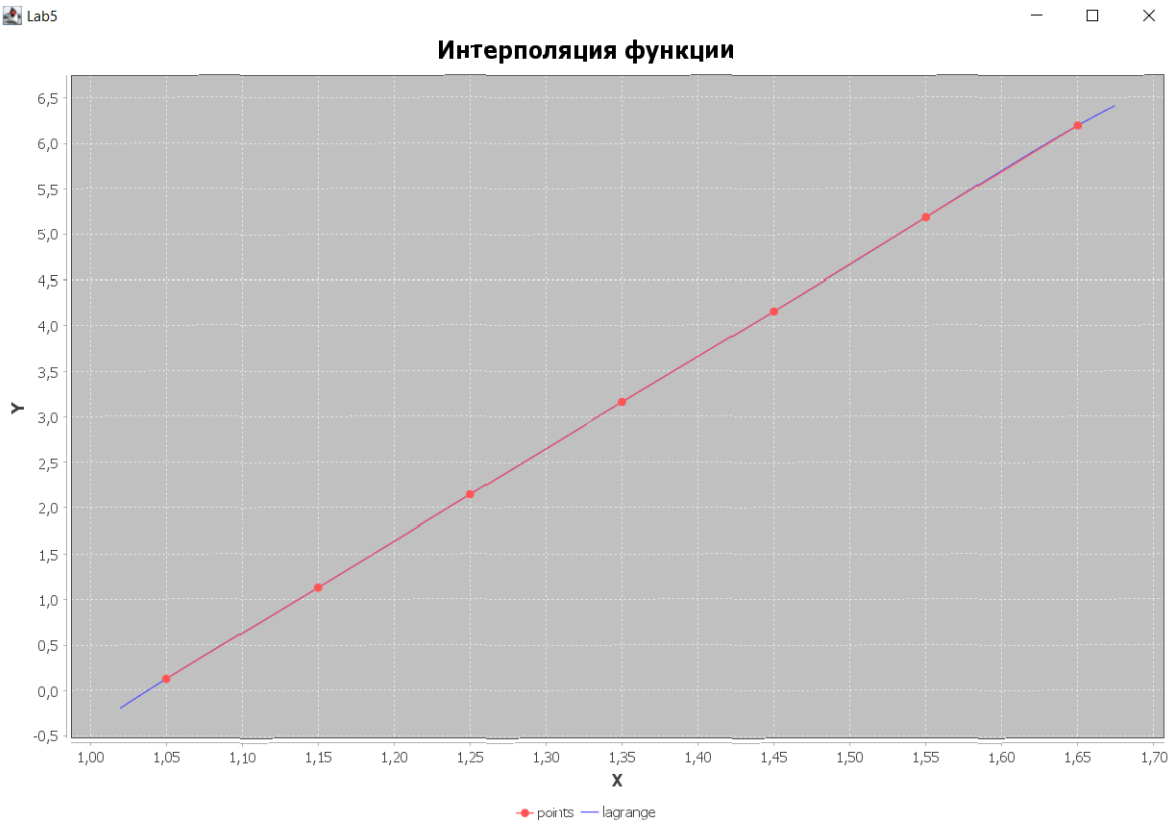
```
class FaultCalculator {

    fun calculate(input: Input): Double {
        val maxFValue: Double = input.points
            .map { functions[input.funcNumber - 1].derivative(input.pointsCount + 1, it.x)
                }.toDoubleArray().maxOrNull() ?: 0.0
        return maxFValue / (1..input.pointsCount+1).reduce {a, b -> a*b} *
                abs(input.points.map { input.xToSolve - it.x }.reduce {a, b -> a*b})
    }
}
```

Вычисление значения функции

```
Выберите формат ввода:
1 - Таблица значений
2 - На основе функции
1
Введите кол-во точек([5 - 20]):
7
Введите значения X:
1.05 1.15 1.25 1.35 1.45 1.55 1.65
Введите значения Y:
0.1213 1.1316 2.1459 3.1565 4.1571 5.1819 6.1969
Выберите метод:
1 - Многочлен Лагранжа
2 - Многочлен Ньютона с разделенными разностями
1
Введите X для расчета:
1.051
============================
Результат: 0.132
```

```
Выберите формат ввода:
1 - Таблица значений
2 - На основе функции
1
Введите кол-во точек([5 - 20]):
7
Введите значения X:
1.05 1.15 1.25 1.35 1.45 1.55 1.65
Введите значения Y:
0.1213 1.1316 2.1459 3.1565 4.1571 5.1819 6.1969
Выберите метод:
1 - Многочлен Лагранжа
2 - Многочлен Ньютона с разделенными разностями
1
Введите X для расчета:
1.557
============================
Результат: 5.255
```



Вывод

В результате выполнения данной лабораторной работы я узнал, какие существуют методы интерполяции функции, как они работают и реализовал два из них.