Университет ИТМО

Факультет программной инженерии и компьютерной техники

Вычислительная математика

Лабораторная работа №4

Вариант – Метод Лагранжа

Выполнил:

Ким Даниил Кванхенович

Группа:

P3231

Преподаватель:

Перл Ольга Вячеславовна

2022

4 семестр

Описание метода, расчетные формулы:

Метод интерполяции с помощью интерполяционного полинома Лагранжа заключается в выведении полиномиальной функции. Формула для общего случая (точки – не обязательно равностоящие):

$$L(x) = \sum_{i=0}^{n} y_i * \ l_{i(x)}$$

$$, где \quad l_{i(x)} = \prod_{j=0, j\neq i}^{n} \frac{x - x_j}{x_i - x_j} \text{ - базисный полином}$$

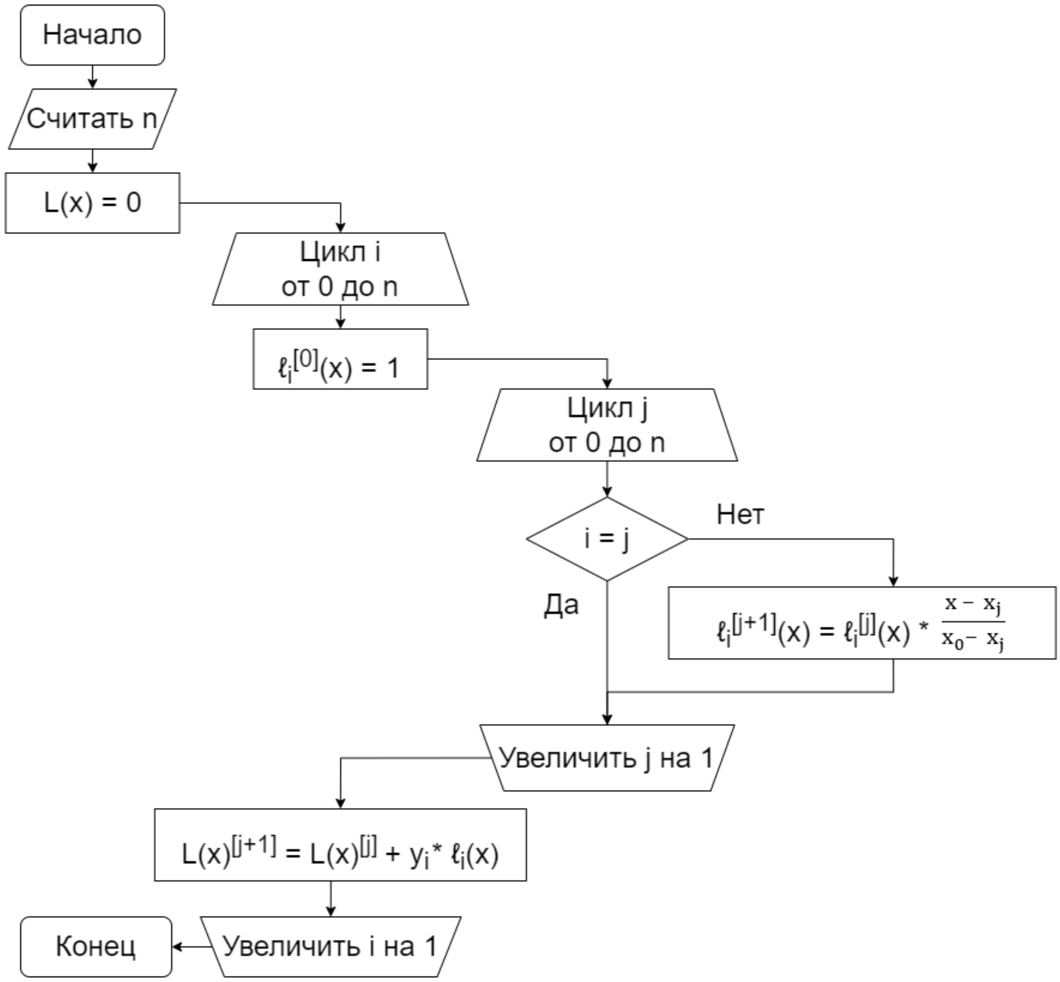
Как видно из формулы, она состоит из суммы базисных полиномов ℓ(x), множенных на значение исходной функции y от заданного набора значений x.

Каждых из базисных полиномов обращается в 1, когда аргумент равен x_i и в 0 – когда не равен. Благодаря этому все узлы интерполяции будут принадлежать полученной функции.

Каждый базисный полином имеет степень равную количеству точек из набора. Полином, полученный из суммы нескольких базисных, будет иметь ту же степень. Известно, что для того, чтобы однозначно задать полином степени n, необходимо n+1 точка. В итоге имеем многочлен минимальной степени = n, принимающий заданные значения в заданном наборе из n+1 точек.

Блок-схема численного метода:

1) Интерполирование функции по точкам с помощью полинома Лагранжа:



Листинг реализованного численного метода программы:

1) Получение базисного полинома:

```
def get_basic_polynomial(x_values, i):
    def basic_polynomial(x):
        numerator, denominator = 1, 1
        for j in range(len(x_values)):
            if j != i:
                numerator *= (x - x_values[j])
                denominator *= (x_values[i] - x_values[j])
        return numerator/denominator
    return basic_polynomial
```

2) Получение интерполяционного полинома Лагранжа:

```
def get_polynomial(x_values, y_values):
    basic_polynomials = [get_basic_polynomial(x_values, i)
                         for i in range(len(x_values))]

    def polynomial(x):
        sum = 0
        for i in range(len(x_values)):
            sum += y_values[i] * basic_polynomials[i](x)
        return sum
    return polynomial
```

Примеры работы программы:

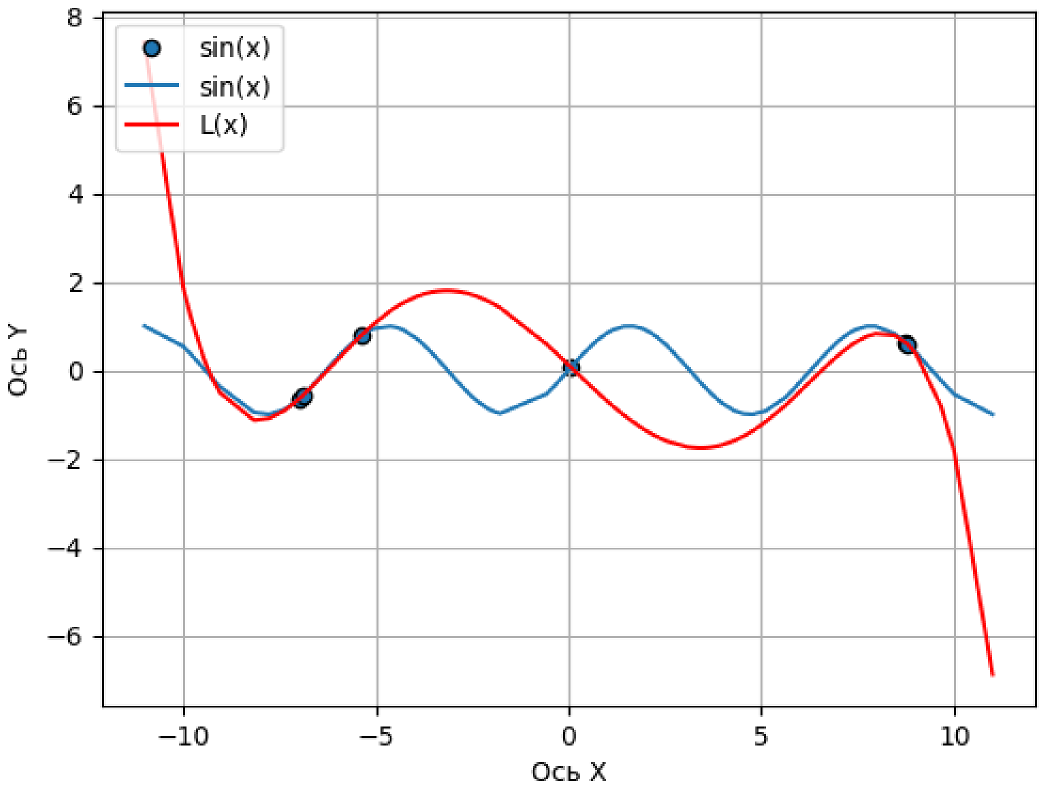
Пример 1 (периодичная функция при малом числе точке):

```
Выберите интегрируемую интерполируемую функцию:
    1) sin(x)
    2) 2x^3 - x^2 + 9
    3) 7*ln(x)
[Ввод] Номер уравнения: 1

[Ввод] Введите границу a: -10

[Ввод] Введите границу b: 10

[Ввод] Количество генерируемых точек: 6
```



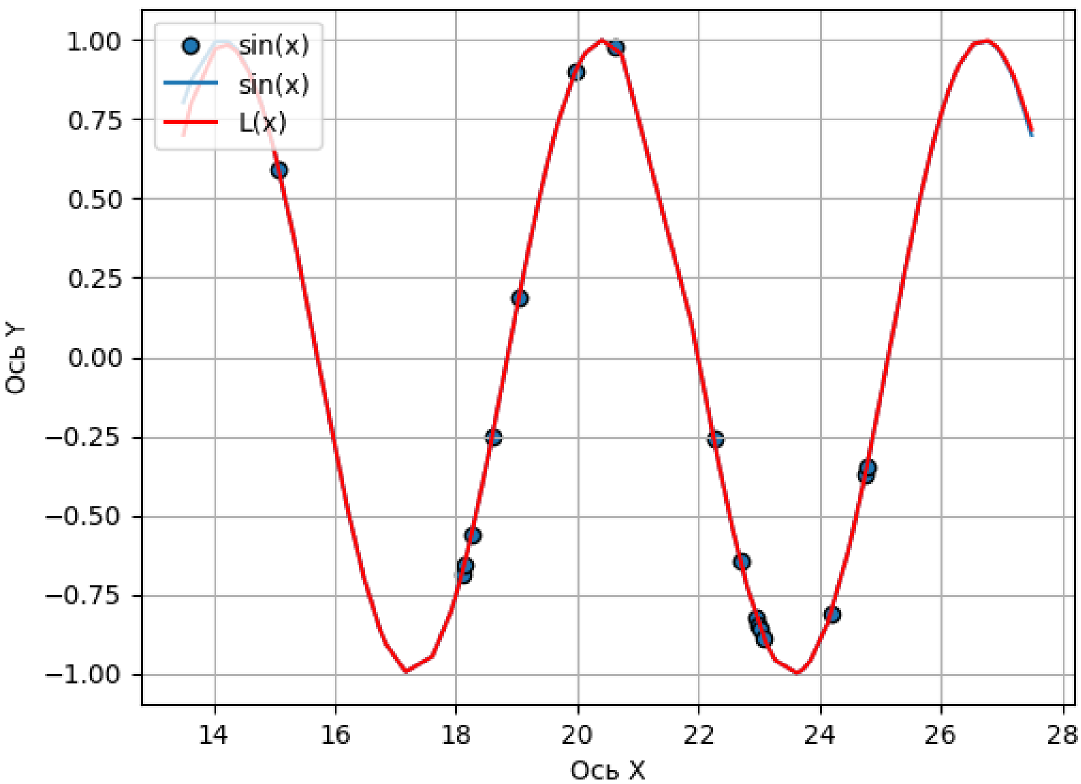
Пример 2 (периодичная функция при большем числе точек):

```
Выберите интегрируемую интерполируемую функцию:
    1) sin(x)
    2) 2x^3 - x^2 + 9
    3) 7*ln(x)
[Ввод] Номер уравнения: 1

[Ввод] Введите границу a: 15

[Ввод] Введите границу b: 25

[Ввод] Количество генерируемых точек: 17
```



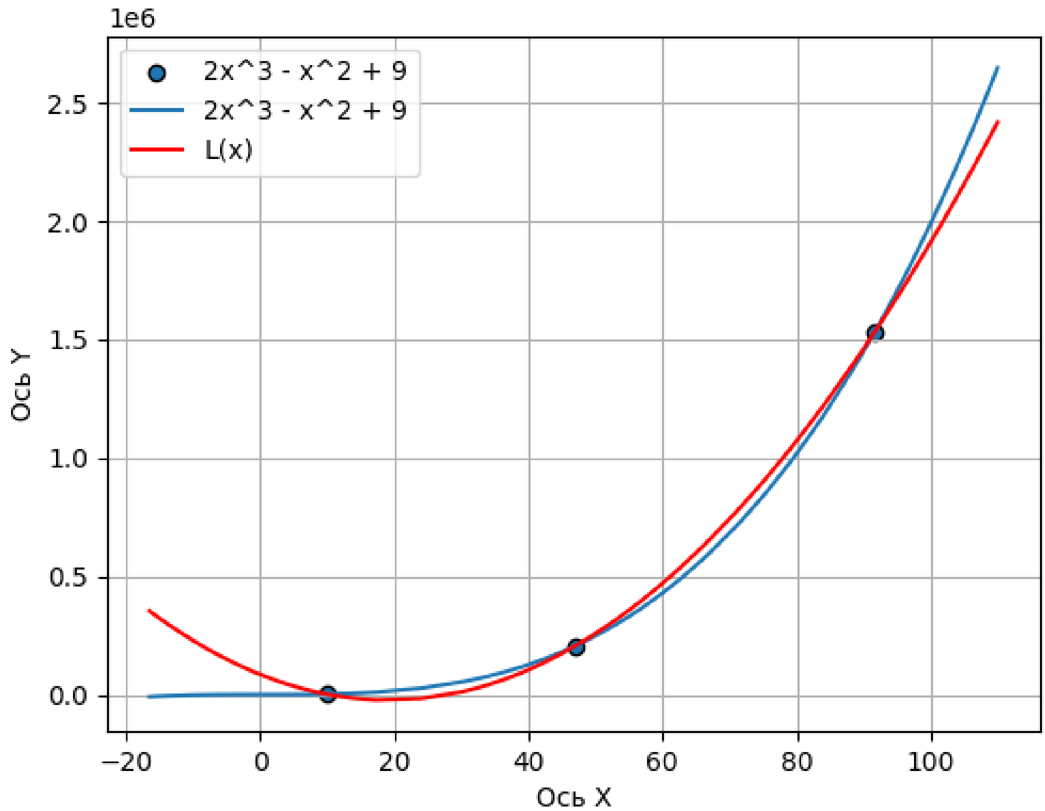
Пример 3 (полином степени n и n точка):

```
Выберите интегрируемую интерполируемую функцию:
    1) sin(x)
    2) 2x^3 - x^2 + 9
    3) 7*ln(x)
[Ввод] Номер уравнения: 2

[Ввод] Введите границу a: -15

[Ввод] Введите границу b: 100

[Ввод] Количество генерируемых точек: 3
```



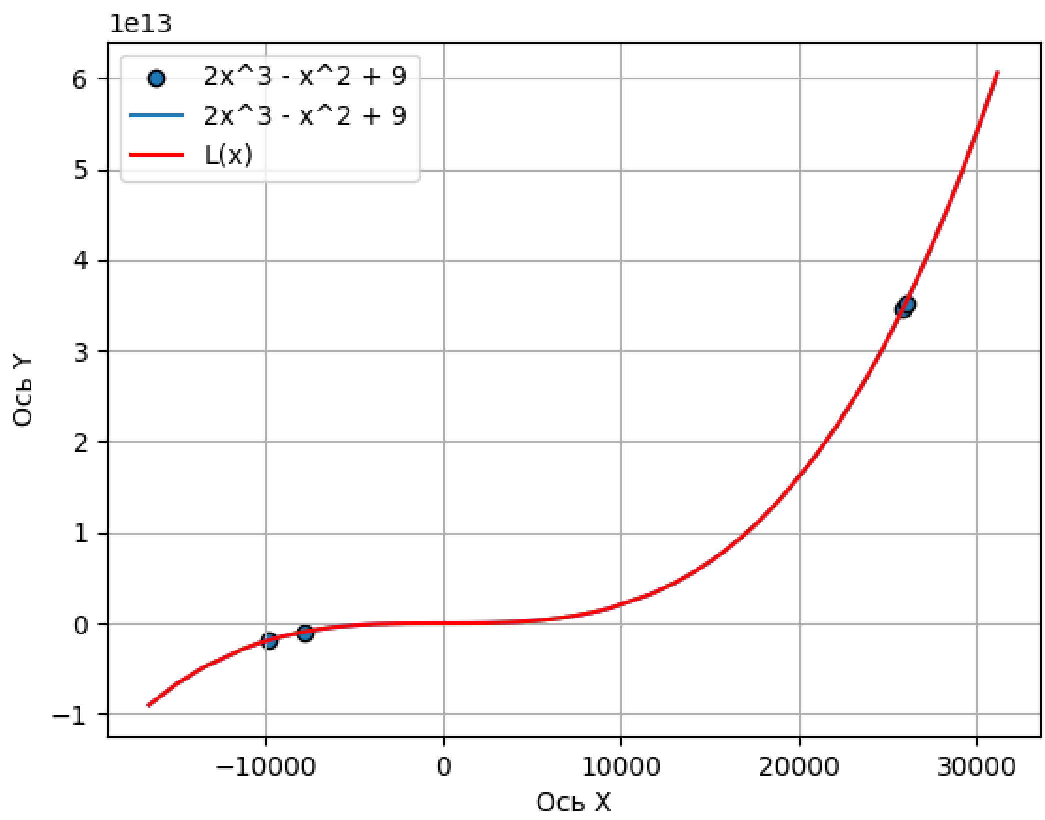
Пример 4 (полином степени n и n+1 точка):

```
Выберите интегрируемую интерполируемую функцию:
    1) sin(x)
    2) 2x^3 - x^2 + 9
    3) 7*ln(x)
[Ввод] Номер уравнения: 2

[Ввод] Введите границу a: -15000

[Ввод] Введите границу b: 28341

[Ввод] Количество генерируемых точек: 4
```



Пример 5 (логарифм и ОДЗ):

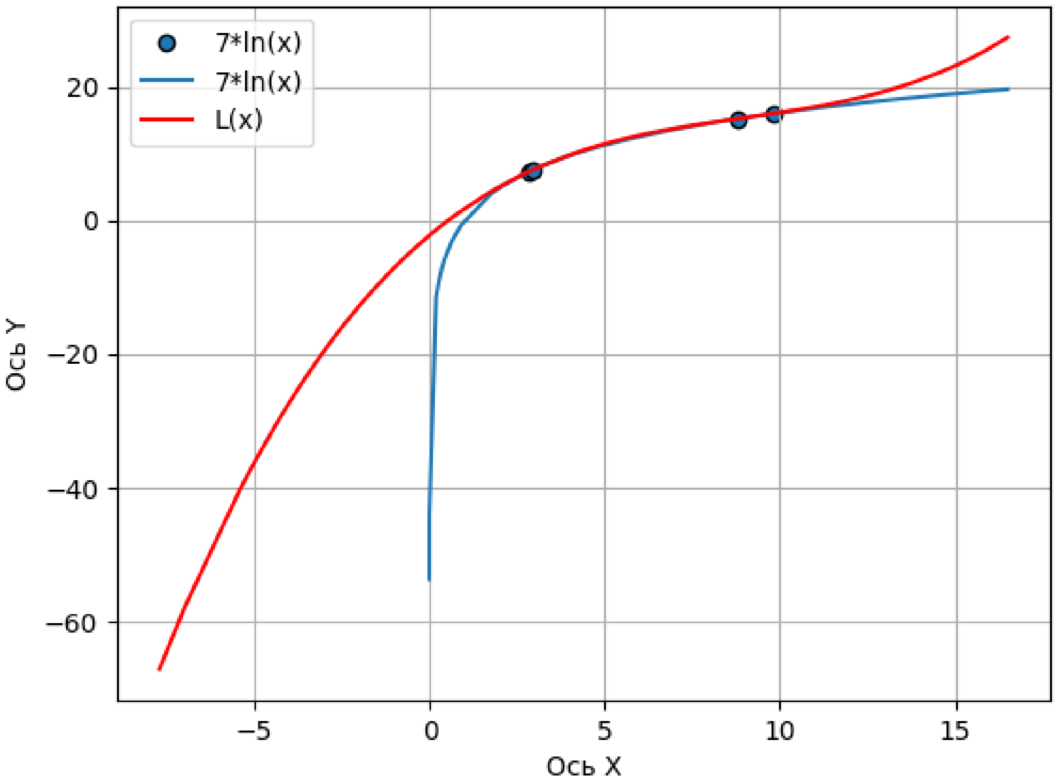
```
Выберите интегрируемую интерполируемую функцию:
    1) sin(x)
    2) 2x^3 - x^2 + 9
    3) 7*ln(x)
[Ввод] Номер уравнения: 3

[Ввод] Введите границу a: -7

[Ввод] Введите границу b: 15

[Ввод] Количество генерируемых точек: 6

[Ошибка] Выбранный интервал вышел за область допустимых значений функции.
Часть сгенерированных точек будет потеряна!
```



Вывод: метод интерполяции с помощью полинома Лагранжа, как и с помощью полинома Ньютона, наиболее эффективен, когда исходная функция является полиномиальной. Как видно из примера 4 при количестве точек, на единицу превышающем степень исходного полинома, функции получаются абсолютно идентичны. Однако, недостаток полиномиальных методов – они плохо себя показывают, при большом числе точек и когда исходная функция является периодической (пример 1).

При интерполяции полиномом Лагранжа интерполяционная функция рассчитывается на всем наборе узлов интерполяции, поэтому следует использовать полином Ньютона, если задача предполагает увеличение набора точек со временем. В данной лабораторной работе подразумевается фиксированный набор точек, поэтому выбранный метод можно считать оптимальным.