Бахышов Вахид, 409 группа

Отчёт по задаче "Полиномиальная интерполяция".

Пусть задана дискретная функция $y_i = f(x_i), i = 0, 1, 2, \ldots, n-1$. Требуется построить алгебраичекий полиноном $P_{n-1}(x)$ степени $n-1$, удовлетворяющий условиям:

$$P_{n-1}(x_i) = y_i, i = 0, 1, 2, \ldots, n-1$$

. Такой полином называется интерполяционным. Его коэффициенты могут быть найдены из решения следующей системы линейных алгебраических уравнений относительно неизвестных $a_0, a_1, \ldots, a_{n-1}$:

$$\begin{cases} a_0 + a_1 x_0 + \ldots + a_{n-1} x_0^{n-1} = y_0 \\ a_0 + a_1 x_1 + \ldots + a_{n-1} x_1^{n-1} = y_1 \\ \ldots \\ a_0 + a_1 x_{n-1} + \ldots + a_{n-1} x_{n-1}^{n-1} = y_{n-1} \end{cases}$$

Система имеет единственное решение, т.к. ее определителем является определитель Ван дер Монда (отличен от нуля в случае $x_i \neq x_j, i \neq j$). Однако, построение полинома $P_{n-1}(x)$ через явное вычисление его коэффициентов приводит к катастрофической потере точности уже при $n \approx 20/50$. Поэтому обычно для расчетов используют запись интерполяционного полинома в форме Лагранжа:

$$P_{n-1}(x) = L_n(x) = \sum_{i=0}^{n-1} y_i \Phi_i(x), \Phi_i(x) = \prod_{j \neq i} \frac{x - x_j}{x_i - x_j}$$

Task 1

Реализовать построение интерполяционного полинонома в канонической форме $P_{n-1}(x)$ и в форме Лагранжа $L_n(x)$.

Task 2

Численно продемонстрировать: (i) правильность работы программы;

(ii) плохую вычислительную устойчивость при больших n методов построения как $P_{n-1}(x)$, так и $L_n(x)$;

(iii) отсутствие сходимости для бесконечно дифференцируемой функции Рунге по равноотстоящим узлам и экспоненциальную сходимость по узлам Чебышева;

(iv) отсутствие сходимости интерполяционного полинома для недифференцируемой функции $|x|$.

Описание решения и кода

В классе Ln реализована интерполяция с помощью полинома Лагранжа: класс хранит узлы и значения функций, при необходимости вычисления значения полинома в точке x рассчёты каждый раз ведутся с нуля, посредством формул, описанных выше.

В классе Pn реализован второй метод. В частности, класс поддерживает конструирование интерполяционного полинома на основе заданных узлов и значений. При этом, СЛУ решается методом Гаусса с выбором главного элемента по столбцам, метод реализован в файле "system.cpp":

Давайте рассмотрим функции и их шаги в "system.cpp":

Функция **solve**:

Реализует метод Гаусса для решения с выбором главного элемента по столбцу. (Входные данные: M - матрица коэффициентов в виде одномерного массива (размер n x n), b - вектор правой части, x - вектор-результат

, в который будет записан ответ, memory - вектор памяти для отслеживания перестановок столбцов.

Вычисляется параметр точности eps, зависящий от среднего значения элементов матрицы (для проверки вырожденности).

Шаги метода Гаусса:

(a) Прямой ход (обнуление нижнего треугольника матрицы) цикл по шагам step от 1 до n:

1. Выбор главного элемента: находит максимальный по модулю элемент в текущем столбце (от строки step до n) и если максимальный элемент меньше порога eps, система считается вырожденной.

2. Перестановка строк: если главный элемент не на текущей строке step, строки меняются местами в M и memory.

3. Обнуление элементов ниже главного:

Для каждой строки $i > step$: вычисляется коэффициент $v = \frac{M[i,step]}{max}$, элементы строки обновляются: $M[i,j] -= v \cdot M[step,j]$,вВектор обновляется: $b[i] -= v \cdot b[step]$.

(b) Обратный ход (приведение к диагональному виду):

Цикл по шагам step от n до 1:

1. Для каждой строки i < step: обновляет b[i] и Обнуляет элемент $M[i, step]$.

}

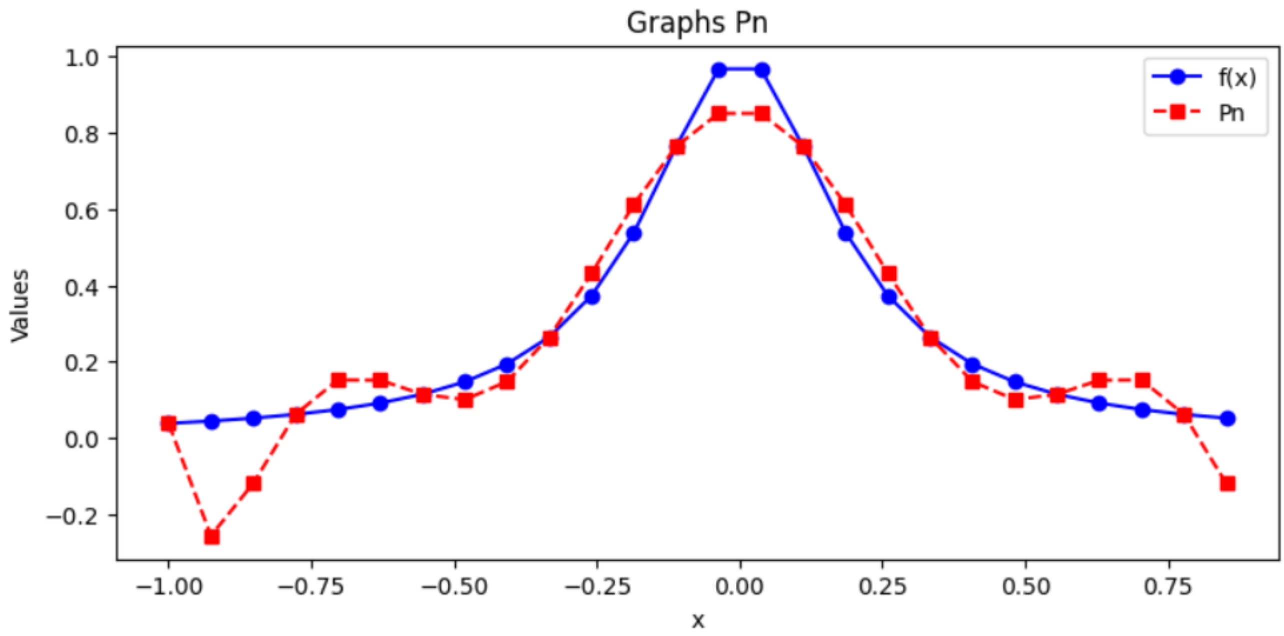


Рис. 1: Runge's function Pn, N=10, равноостоящие узлы

2. Нормирует диагональный элемент: делит $b[step]$ на $M[step, step]$. Диагональный элемент становится единичным.

(c) С помощью цикла по столбцам u от 0 до n - 1 записываем значения из b в x, учитывая порядок перестановок memory.

0.1 Работоспособность.

Рассматривались функции:

```
    double f(double x)
{
    return 1./(1.+25.*x*x); //Runge's function
    return std::fabs(x);
}
```

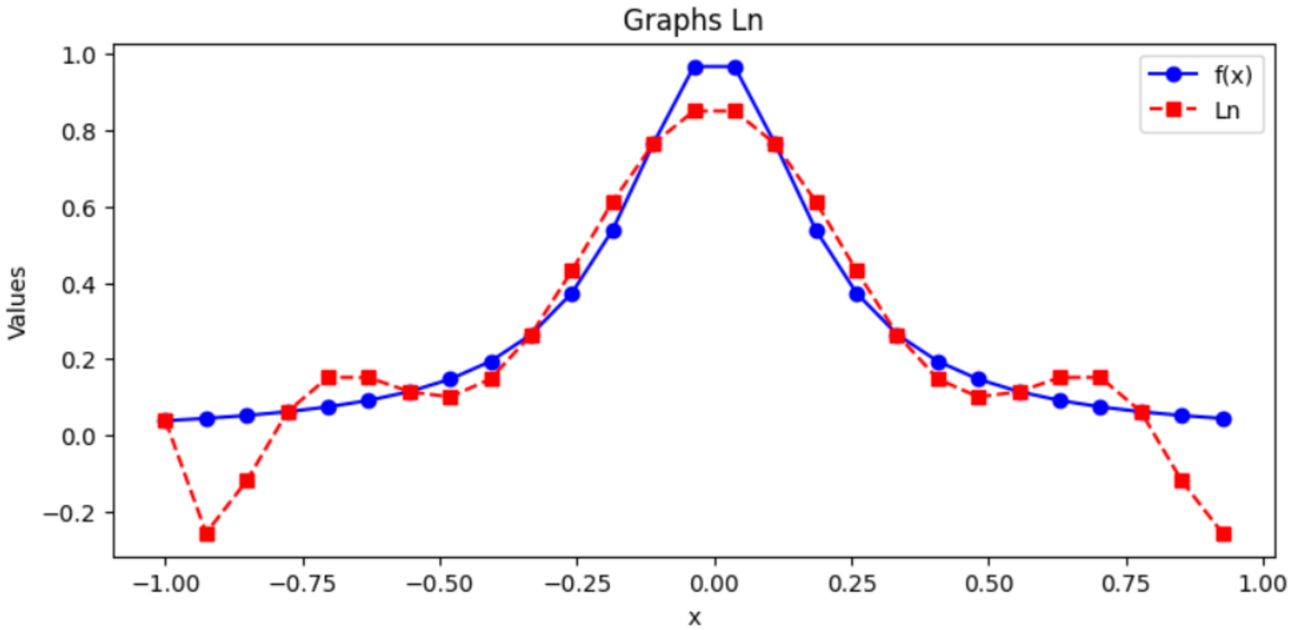


Рис. 2: Runge's function Ln, N=10, равноостоящие узлы

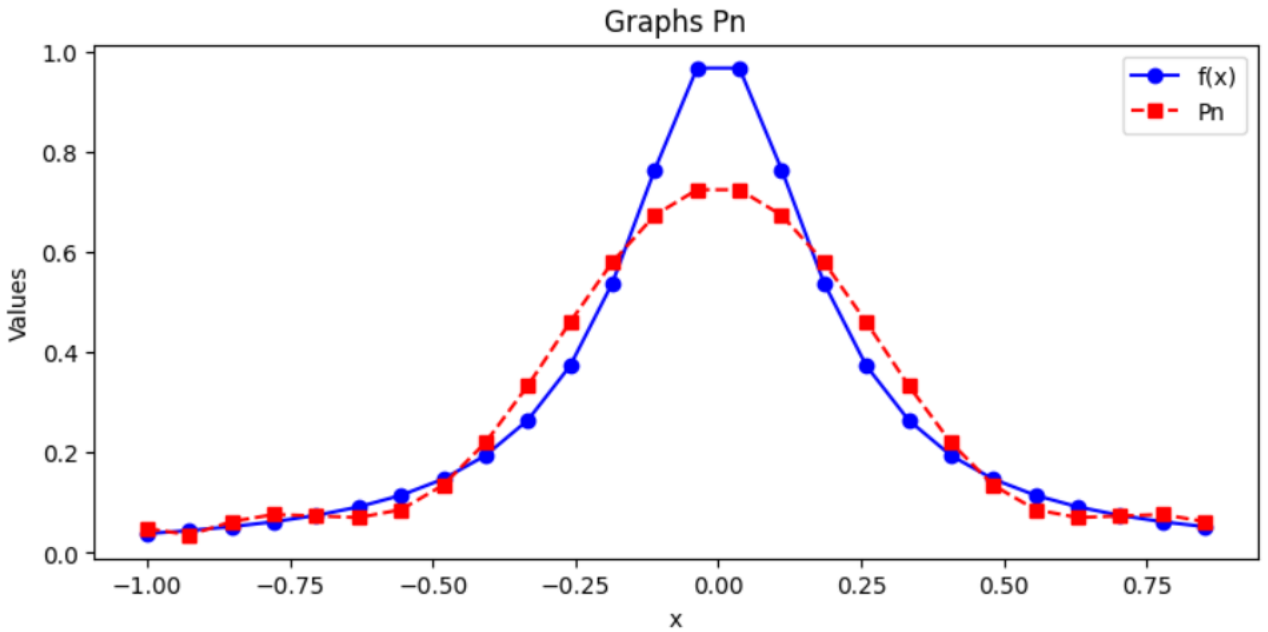


Рис. 3: Runge's function on Chebyshev nodes Pn, N=10

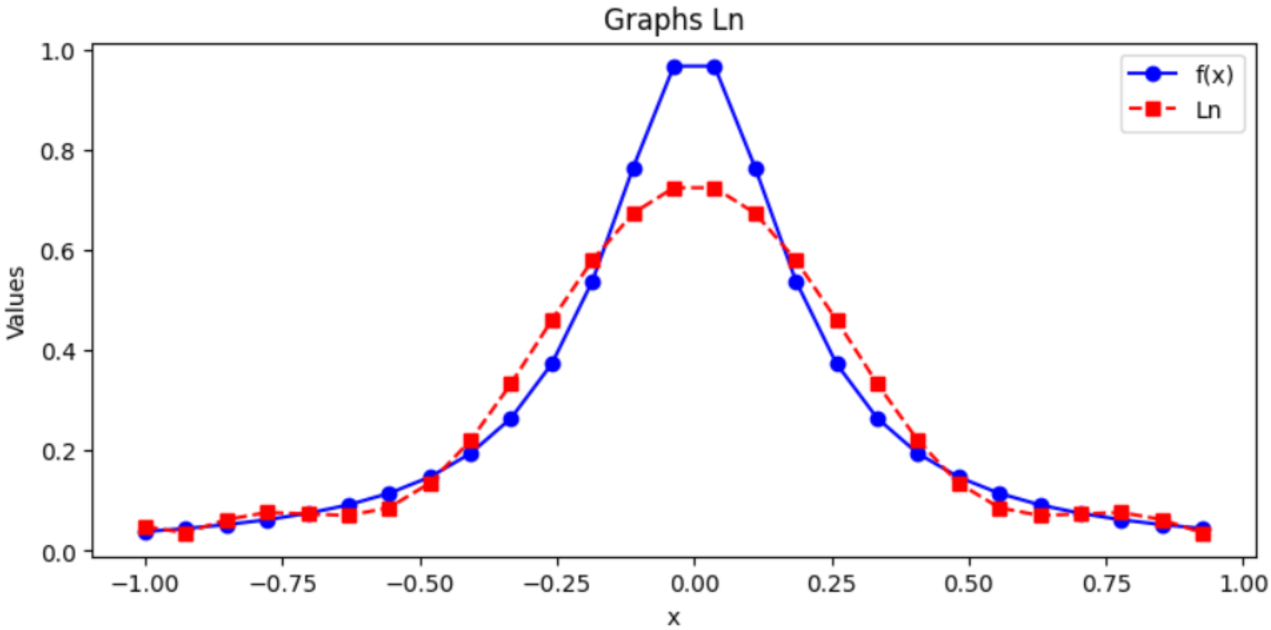


Рис. 4: Runge's function on Chebyshev nodes Ln, N=10

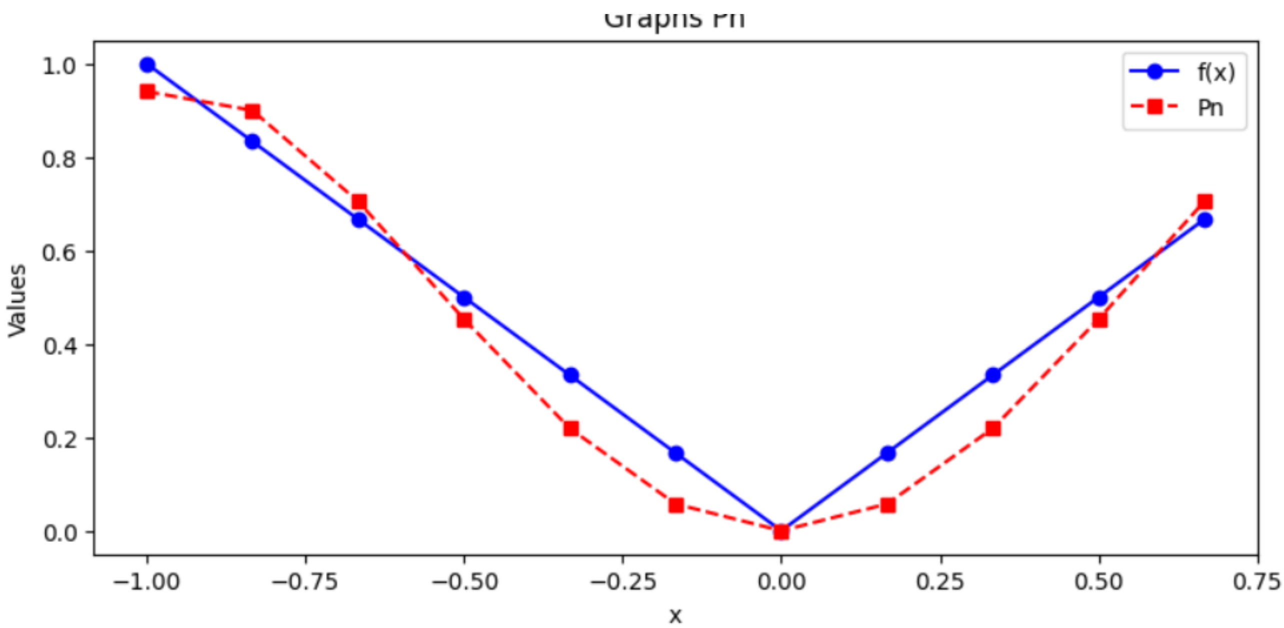


Рис. 5: f(x) = |x| Pn on Chebyshev nodes, N=5

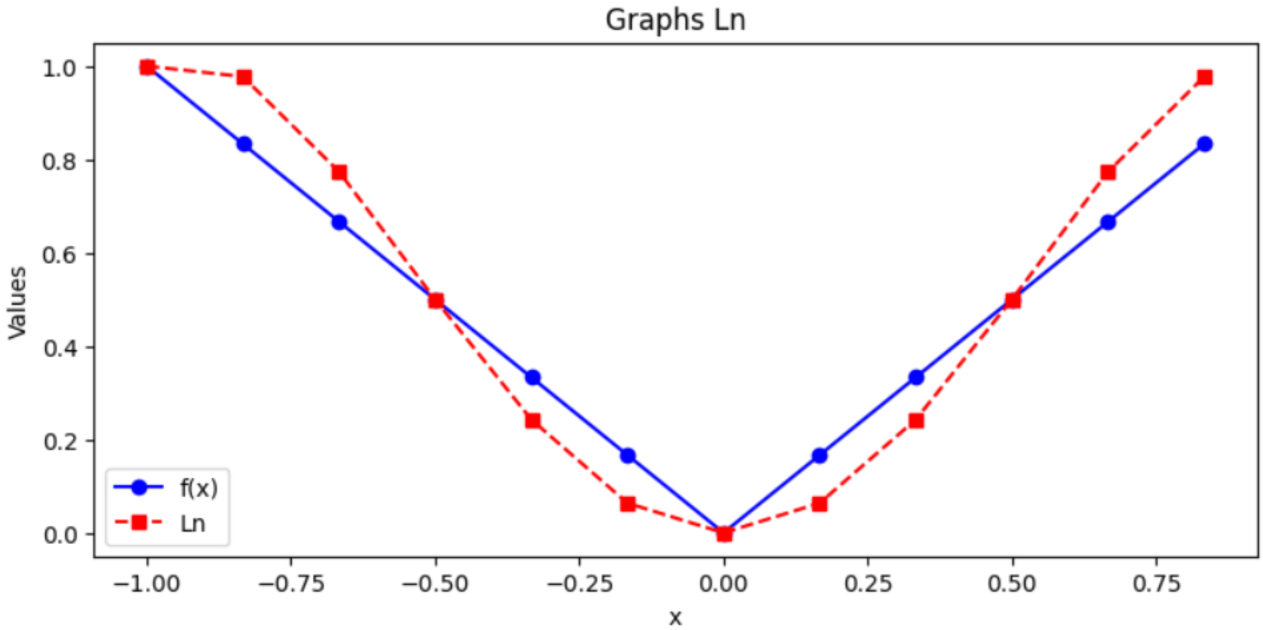


Рис. 6: f(x) = |x| Ln, N=5, равноостоящие узлы

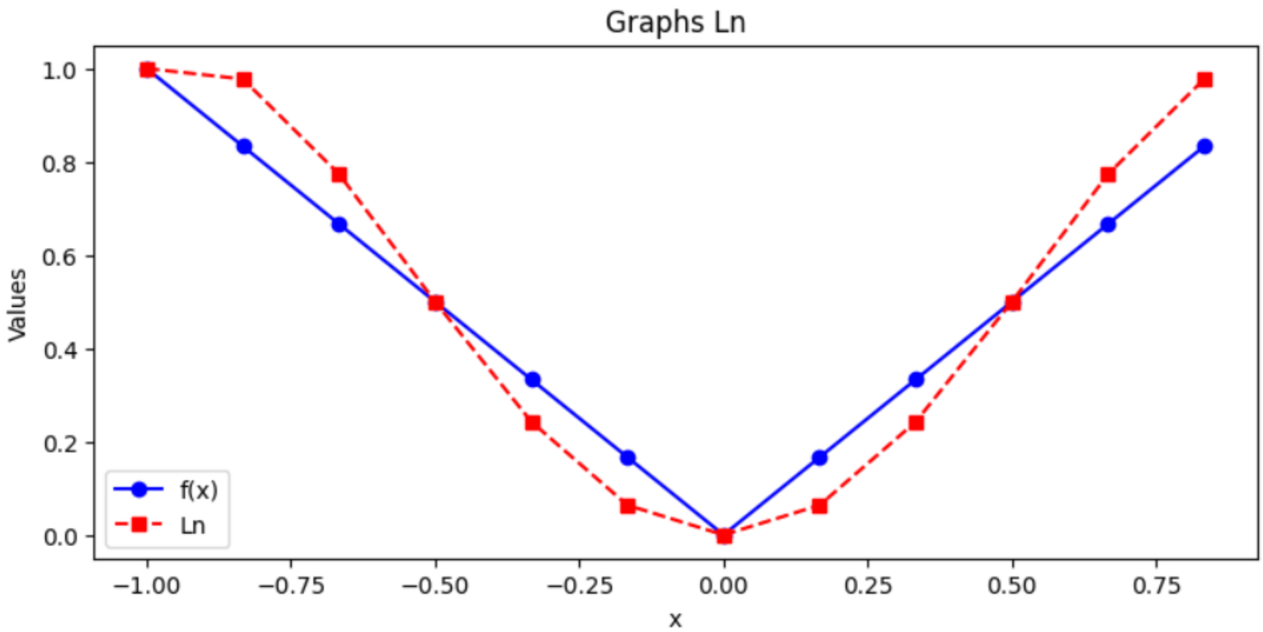
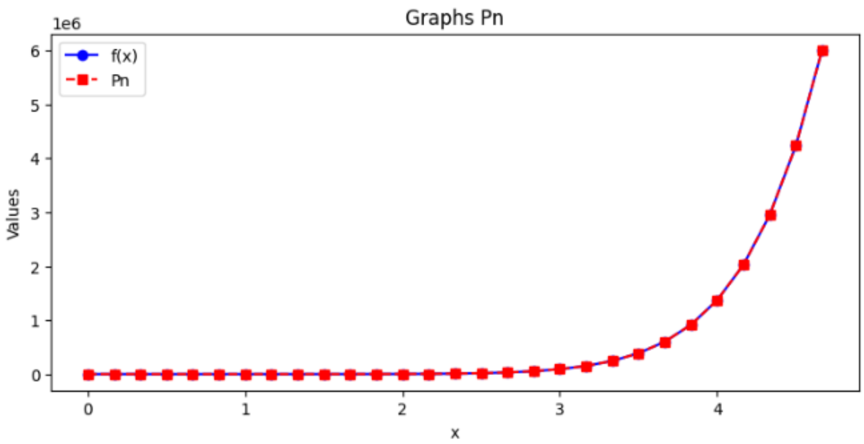


Рис. 7: f(x) = |x| Pn, N=5, равноостоящие узлы



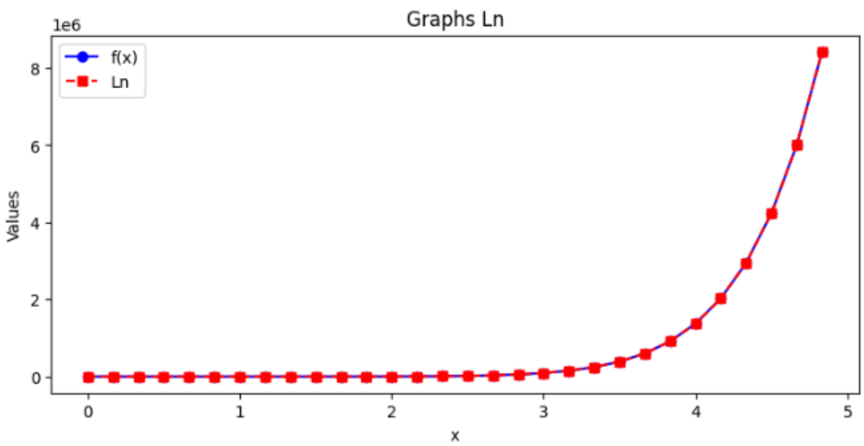
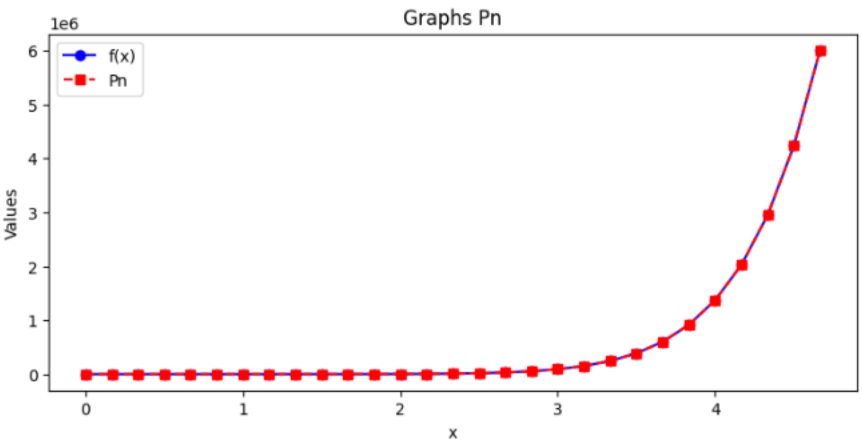


Рис. 8: f(x) = pow (x , 10) + 5 * pow(x , 8) -2 * pow(x , 6) + 3 * pow (x , 5) + 2 * pow (x , 3) + x * x + 11 Pn, N=11, равноостоящие узлы



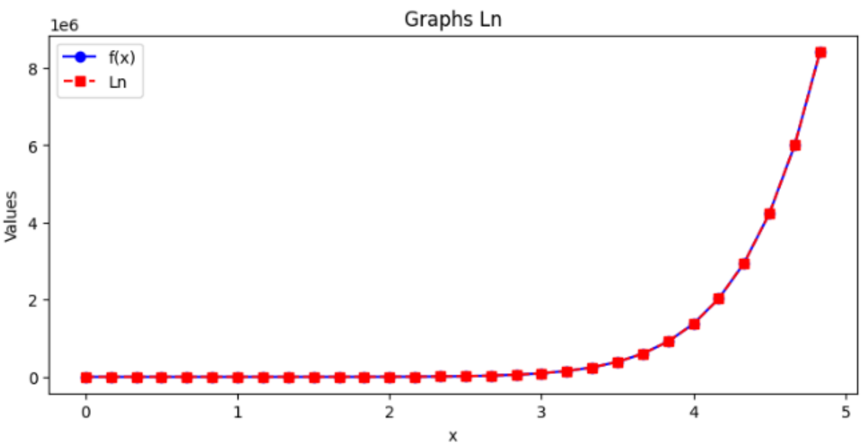


Рис. 9: f(x) = pow (x , 10) + 5 * pow(x , 8) -2 * pow(x , 6) + 3 * pow (x , 5) + 2 * pow (x , 3) + x * x + 11 Pn on Chebyshev nodes, N=11