Санкт-Петербургский Государственный Университет

Вычислительный практикум

Задача алгебраического интерполирования Интерполяционный многочлен в формах Ньютона и Лагранжа

Борис Мельников
b.melnikov17@gmail.com

под руководством Алцыбеева Глеба Олеговича
Преподаватель & Ассистент

май, 2023

Содержание

1 Введение

В повседневной жизни студенту математического факультета часто нужно знание о значении какой-либо функции во вполне определенной точке. Однако нахождение точных значений не всегда представляется возможным и, вообще говоря, требуется далеко не всегда – чаще всего в решении прикладной задачи предполагается нахождение ответа определенной степени точности, поэтому вполне разумным будет поставить задачу о нахождении приближенного значения данной функции в данной точке. Для решения подобной задачи существует несколько подходов, но именно в этой работе мы рассмотрим построение интерполирующих многочленов.

Целью работы ставилось знакомство с задачей интерполирования как таковой и видами многочленов, которые могут быть построены при решении данной задачи. В ходе работы были сделаны попытки реализовать удобный для использования интерфейс между пользователем и компьютером через встроенное в Windows 11 решение PowerShell.

2 Постановка задачи

Пусть на промежутке $[a, b]$ задана таблица значений вещественной функции $y = f(x)$:

x	$f(x)$
x_0	$f(x_0)$
x_1	$f(x_1)$
$\cdots$	$\cdots$
x_n	$f(x_n)$

узлы которой предполагаются попарно различными, т.е. $x_i \neq x_j, i \neq j$.

Требуется найти значение функции в точке $x = \bar{x}$, которая не совпадает ни с одним из узлов.

3 Необходимая теория

3.1 Подготовительный этап

Конечно, итогом работы должна стать программа, которая будет находить требуемое значение несколькими методами, которые далее будут описаны. Но для начала нужно сформировать входные данные. Поступим так: поскольку вычислительные мощности современных машин позволяют находить зачения сложных функций с небольшой погрешностью, будем напрямую в ходе исполнения программы составлять таблицу из условия. Создав двумерный вектор `vector <vector <double>> table`, заполним его $m+1$ значением точек из отрезка $[a, b]$, где шаг между самими точками укажем равным $h = (b-a)/m$. В данном случае значения m, a, b становятся переменными, значения которых задаются пользователем в ходе исполнения программы, а 1-ая точка (она же x_0) в массиве будет совпадать с a. Значения функции $f(x) = \cos(x) + 2 \cdot x$ (соответствует варианту #10), соответствующие точкам x_i в объявленном массиве, заполним значениями, которые будут автоматически вычислены при помощи функции `double f (double x)` (в теле функции просто задано выражение `cos(x) + 2 * x`).

Теперь, когда имеется подготовленный массив данных с узлами интерполирования, можем переходить непосредственно к нахождению приближенного значения функции в точке $\bar{x}$ при помощи интерполяционных многочленов (значение $\bar{x}$ будет храниться в переменной `double x`.

3.2 Интерполяционный многочлен Лагранжа $P_n^L(x)$

Суть подхода основана на том, что нам заранее известны все необходимые данные для построения многочлена Лагранжа – его степень n и значения интерполируемой функции в $n+1$ узле. Искомый многочлен дается следующей формулой:

$$P_n^L(x) = \sum_{k=0}^{n} \frac{\omega_{n+1}(x)}{(x-x_k)\cdot\omega'_{n+1}(x_k)} f(x_k) = \sum_{k=0}^{n} \frac{\prod\limits_{i\neq k}(x-x_i)}{\prod\limits_{i\neq k}(x_k-x_i)} f(x_k), \text{ т.е. } \omega_{n+1}(x) = \prod_{i=0}^{n}(x-x_i)$$

3.3 Интерполяционный многочлен Ньютона $P_n^N(x)$

Что же касается подхода Ньютона, то этот способ допускает построение интерполяционного многочлена, степень которого не задана изначально, т.е. степень уже известного интерполирующего многочлена можно повысить добавлением узлов интерполирования без необходимости перестраивать требуемый многочлен с нуля, что, конечно, является преимуществом перед многочленом Лагранжа с вычислительной точки зрения.

Многочлен Ньютона дается формулой:

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

где $f(x_0;\ldots;x_n)$ есть разделенная разность, вычисляемая в общем случае по формуле $f(x_0;\ldots;x_n) = \sum\limits_{k=0}^{n} \dfrac{f(x_k)}{\prod\limits_{\substack{i=0\\ i\neq k}}^{n}(x_k-x_i)}$. Однако ее также можно вычислять рекуррентно – покажем:

По определению $f(x_j;x_{j+1};\ldots;x_{j+k-1};x_{j+k}) = \frac{f(x_{j+1};\ldots;x_{j+k-1};x_{j+k}) - f(x_j;x_{j+1};\ldots;x_{j+k-1})}{x_{j+k}-x_j}$;

Следовательно, $f(x_0;x_1;\ldots;x_{n-1};x_n) = \frac{f(x_1;\ldots;x_{n-1};x_n) - f(x_0;x_1;\ldots;x_{n-1})}{x_n-x_0}$, ..., $f(x_0;x_1;x_2) = \frac{f(x_1;x_2)-f(x_0;x_1)}{x_2-x_0} = \frac{\frac{f(x_2)-f(x_1)}{x_2-x_1} - \frac{f(x_1)-f(x_0)}{x_1-x_0}}{x_2-x_0}$.

Замечаем, что разделенные разности высокого порядка можно вычислять через разделенные разности малого порядка (рекурсия), что удобно на практике, поэтому именно такой подход выберем для их вычисления компьютером.

3.4 Дополнения

Конечно, во многих изданиях уже не раз доказывалось, что описанные выше многочлены действительно решают задачу интерполяции. Остается лишь заметить, что точность искомого приближенного значения $f(\bar{x})$ зависит от того, как именно строился многочлен. Поскольку нам заранее известна точка $\bar{x}$, можем применить некоторую хитрость, чтобы вычисленное значение было максимально близким к истинному, – мы отсортируем точки в таблице узлов по удаленности от $\bar{x}$, т.е. чем ближе точка x_i известного узла интерполирования к $\bar{x}$, тем выше она будет расположена в таблице (иначе говоря, тем меньше значение ее индекса i).

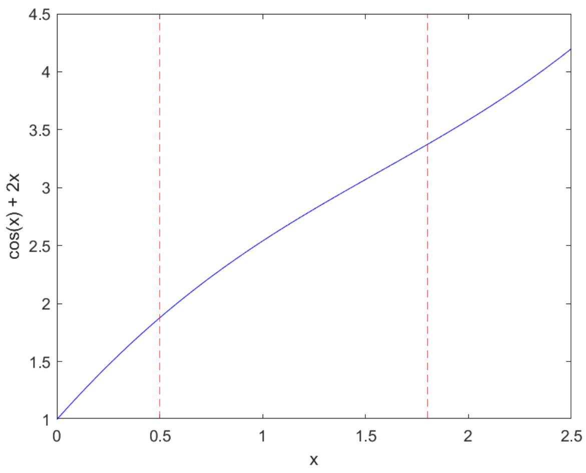

Рис. 1: График $f(x)$ на $[a,b]$

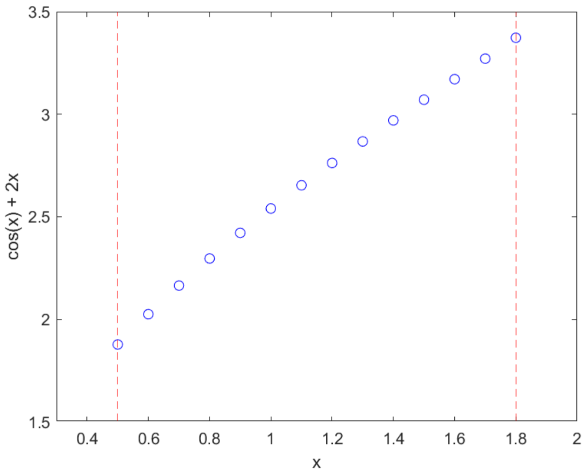

Рис. 2: Узлы интерполирования

4 Практическая реализация

Итак, теперь опишем данные для задачи, которую предстоит решить. Будем решать задачу из варианта #10 со следующими начальными данными:
$f(x) = cos(x) + 2 \cdot x$;
$a = 0.5$;
$b = 1.8$;
$m = 13$;

Для удобства будем использовать различные математические пакеты, чтобы точно представлять картинку того, что именно происходит. На рис. 1 изображен график функции $f(x)$ для $x \in [a,b]$. Поскольку в условии зафиксировано, что $m = 13$, можем графически построить дискретное множество узлов интерполирования - рис. 2:

```
Output:
@@ Table of values:
          x        |       f(x)
   0.50000000      |    1.87758256
   0.60000000      |    2.02533561
   0.70000000      |    2.16484219
   0.80000000      |    2.29670671
   0.90000000      |    2.42160997
   1.00000000      |    2.54030231
   1.10000000      |    2.65359612
   1.20000000      |    2.76235775
   1.30000000      |    2.86749883
   1.40000000      |    2.96996714
   1.50000000      |    3.07073720
   1.60000000      |    3.17080048
   1.70000000      |    3.27115551
   1.80000000      |    3.37279791
```

Теперь нужно выбрать 2 точки, приближенное значение в которых мы будем вычислять. $f(x)$ достаточно "хорошая"функция на отрезке $[a,b]$ из условия, потому нет особой разницы, какую точку на этом отрезке выбрать, но для определенности положим, что $\tilde{x}_0 = 0.84$. Куда более интересен вопрос, какую точку выбрать вне этого отрезка: на рис. 3 видно, что $f(x)$ имеет "холм"в окрестности -6 – ее и возьмем в качестве $\tilde{x}_1$.

К сожалению, поскольку мы не строим явно многочлены P_n^L и P_n^N, а лишь считаем по алгоритму их построения значение в конкретной точке, у нас не получится изобразить кривые

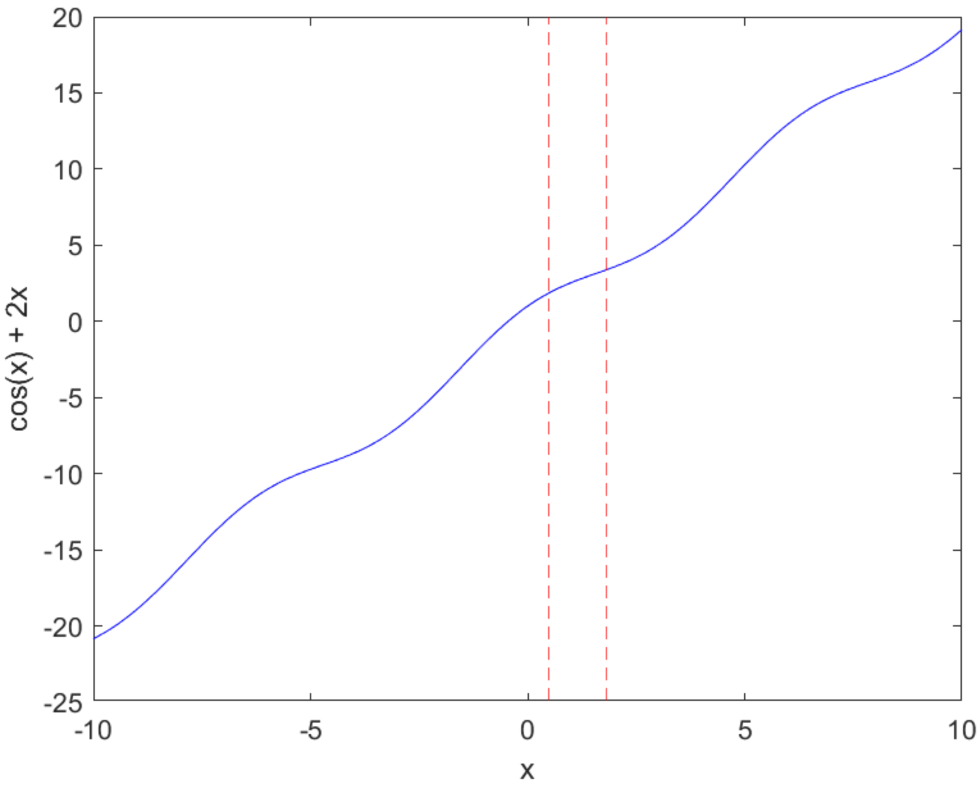

Рис. 3: График $f(x)$ на большем промежутке

для P_n^L и P_n^N, однако для различных степеней n вполне можно отметить точки, ординаты которых считаются по указанным схемам, что и сделаем ниже. Скажем несколько слов о коде программы:

1) После ввода точки, для которой ищется приближенное значение, сортировку узлов производим посредством определенной в `iostream` функции `sort`, а в качестве 3-го параметра (грубо, условия сортировки) используем лямбда выражение `[](const auto& v1, const auto& v2) { return abs(v1[0] - x) < abs(v2[0] - x); };`

2) Вычисление значения методом Лагранжа происходит в функции `double Lagrange();`

3) Разделенные разности для многочлена Ньютона вычисляются в функции `double divDiff (int index)`, где `index` есть индекс слагаемого из записи P_n^N, для которого требуется найти значение разделенной разности. Само же искомое значение в точке вычисляется в функции `double Newton();`

Для точки $\tilde{x}_0$ получилось, что значения, расчитанные по методам Лагранжа и Ньютона, не отличаются, но с повышением степени интерполирующего многочлена точность вычисленного значения растет и в конечном счете становится зависима от декларации типа `double`, что не позволяет делать более точный анализ, хотя результаты будут вполне ожидаемыми – точность повысится еще больше.

Для точки $\tilde{x}_1$, не лежащей внутри $[a, b]$ результаты получились непредсказуемыми: с повышением степени интерполирующего многочлена точность может как увеличиваться, так и наоборот уменьшаться (ярко выражено для $n = 8$ и $n = 13$). При этом значения P_n^L и P_n^N не совпали только для $n = 13$, что так же подтверждает идею о непредсказуемости поведения интерполяционного многочлена вне промежутка интерполирования, по точкам которого он строился.

Ссылка на репозиторий: `https://github.com/BoriskaCat/-CM-Projects/tree/main/Lab%202`

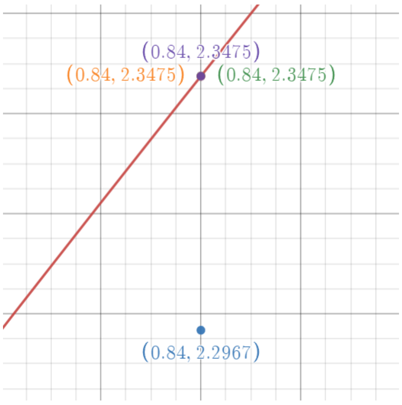

Рис. 4: Точки, вычисленные для $\tilde{x}_0$ при различных n (цвета: n = 0 - син., n = 3 - зел., n = 8 - оранж., n = 13 - фиол.)

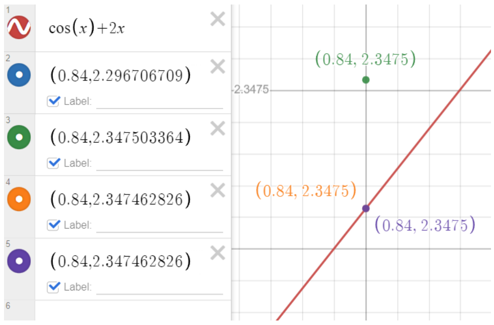

Рис. 5: Точки, вычисленные для $\tilde{x}_0$ при различных n (цвета: n = 0 - син., n = 3 - зел., n = 8 - оранж., n = 13 - фиол.)

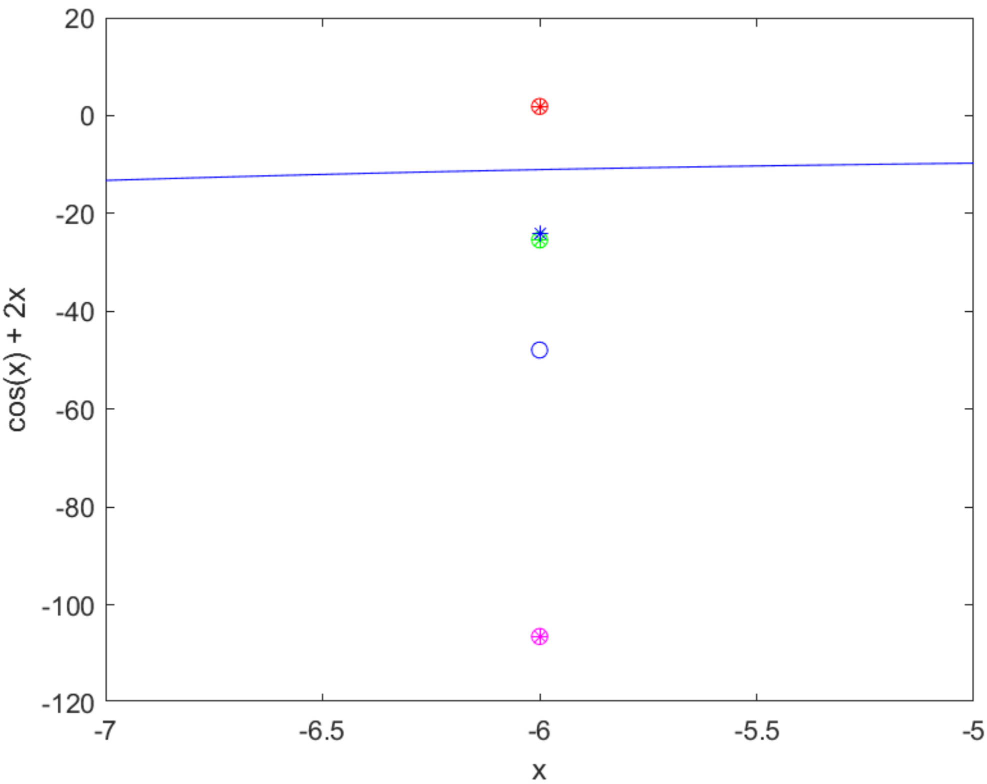

Рис. 6: Точки, вычисленные для $\tilde{x}_1$ при различных n (цвета: n = 0 - кр., n = 3 - зел., n = 8 - роз., n = 13 - син.; $\circ$ - знач. по Лагранжу, $*$ - знач. по Ньютону)

```
Output:
@@ #2. Problem of algebraic interpolation.

@@ #ID: 10

@@ Parameters:
   f(x) = cos(x) + 2x
   m = 13
   a (> -1) = 0.5
   b (> -1) = 1.8

@@ Table of values:
          x       |       f(x)
   0.50000000     |   1.87758256
   0.60000000     |   2.02533561
   0.70000000     |   2.16484219
   0.80000000     |   2.29670671
   0.90000000     |   2.42160997
   1.00000000     |   2.54030231
   1.10000000     |   2.65359612
   1.20000000     |   2.76235775
   1.30000000     |   2.86749883
   1.40000000     |   2.96996714
   1.50000000     |   3.07073720
   1.60000000     |   3.17080048
   1.70000000     |   3.27115551
   1.80000000     |   3.37279791

@@ Interpolation point:
   x = 0.84

@@ Degree of interpolation polynomial
   Hint: n <= 13
   n = 0

@@ Sorted table:
     x      |    f(x)
   0.800    |   2.297
   0.900    |   2.422
   0.700    |   2.165
   1.000    |   2.540
   0.600    |   2.025
   1.100    |   2.654
   0.500    |   1.878
   1.200    |   2.762
   1.300    |   2.867
   1.400    |   2.970
   1.500    |   3.071
   1.600    |   3.171
   1.700    |   3.271
   1.800    |   3.373

@@ Lagrange value: 2.296706709
  |L_res - f(x)|: 0.05075611649
```

```
@@ Newton value: 2.296706709
  |N_res - f(x)|: 0.05075611649

@@ True value: 2.347462826

?? Change the value of x or n? [y / n]
y

@@ Interpolation point:
   x = 0.84

@@ Degree of interpolation polynomial
   Hint: n <= 13
   n = 3

@@ Sorted table:
     x      |    f(x)
   0.800    |   2.297
   0.900    |   2.422
   0.700    |   2.165
   1.000    |   2.540
   0.600    |   2.025
   1.100    |   2.654
   0.500    |   1.878
   1.200    |   2.762
   1.300    |   2.867
   1.400    |   2.970
   1.500    |   3.071
   1.600    |   3.171
   1.700    |   3.271
   1.800    |   3.373

@@ Lagrange value: 2.347503364
  |L_res - f(x)|: 4.053865196e-05

@@ Newton value: 2.347503364
  |N_res - f(x)|: 4.053865196e-05

@@ True value: 2.347462826

?? Change the value of x or n? [y / n]
y

@@ Interpolation point:
   x = 0.84

@@ Degree of interpolation polynomial
   Hint: n <= 13
   n = 8

@@ Sorted table:
     x      |    f(x)
   0.800    |   2.297
   0.900    |   2.422
   0.700    |   2.165
```

```
   1.000    |   2.540
   0.600    |   2.025
   1.100    |   2.654
   0.500    |   1.878
   1.200    |   2.762
   1.300    |   2.867
   1.400    |   2.970
   1.500    |   3.071
   1.600    |   3.171
   1.700    |   3.271
   1.800    |   3.373

@@ Lagrange value: 2.347462826
  |L_res - f(x)|: 6.714184764e-12

@@ Newton value: 2.347462826
  |N_res - f(x)|: 6.713296585e-12

@@ True value: 2.347462826

?? Change the value of x or n? [y / n]
y

@@ Interpolation point:
   x = 0.84

@@ Degree of interpolation polynomial
   Hint: n <= 13
   n = 13

@@ Sorted table:
     x      |    f(x)
   0.800    |   2.297
   0.900    |   2.422
   0.700    |   2.165
   1.000    |   2.540
   0.600    |   2.025
   1.100    |   2.654
   0.500    |   1.878
   1.200    |   2.762
   1.300    |   2.867
   1.400    |   2.970
   1.500    |   3.071
   1.600    |   3.171
   1.700    |   3.271
   1.800    |   3.373

@@ Lagrange value: 2.347462826
  |L_res - f(x)|: 4.440892099e-16

@@ Newton value: 2.347462826
  |N_res - f(x)|: 8.881784197e-16

@@ True value: 2.347462826
```

```
?? Change the value of x or n? [y / n]
y

@@ Interpolation point:
   x = -6

@@ Degree of interpolation polynomial
   Hint: n <= 13
   n = 0

@@ Sorted table:
     x      |    f(x)
   0.500    |   1.878
   0.600    |   2.025
   0.700    |   2.165
   0.800    |   2.297
   0.900    |   2.422
   1.000    |   2.540
   1.100    |   2.654
   1.200    |   2.762
   1.300    |   2.867
   1.400    |   2.970
   1.500    |   3.071
   1.600    |   3.171
   1.700    |   3.271
   1.800    |   3.373

@@ Lagrange value: 1.877582562
  |L_res - f(x)|: 12.91741228

@@ Newton value: 1.877582562
  |N_res - f(x)|: 12.91741228

@@ True value: -11.03982971

?? Change the value of x or n? [y / n]
y

@@ Interpolation point:
   x = -6

@@ Degree of interpolation polynomial
   Hint: n <= 13
   n = 3

@@ Sorted table:
     x      |    f(x)
   0.500    |   1.878
   0.600    |   2.025
   0.700    |   2.165
   0.800    |   2.297
   0.900    |   2.422
   1.000    |   2.540
   1.100    |   2.654
   1.200    |   2.762
```

```
   1.300    |   2.867
   1.400    |   2.970
   1.500    |   3.071
   1.600    |   3.171
   1.700    |   3.271
   1.800    |   3.373

@@ Lagrange value: -25.41506687
  |L_res - f(x)|: 14.37523715

@@ Newton value: -25.41506687
  |N_res - f(x)|: 14.37523715

@@ True value: -11.03982971

?? Change the value of x or n? [y / n]
y

@@ Interpolation point:
   x = -6

@@ Degree of interpolation polynomial
   Hint: n <= 13
   n = 8

@@ Sorted table:
     x      |    f(x)
   0.500    |   1.878
   0.600    |   2.025
   0.700    |   2.165
   0.800    |   2.297
   0.900    |   2.422
   1.000    |   2.540
   1.100    |   2.654
   1.200    |   2.762
   1.300    |   2.867
   1.400    |   2.970
   1.500    |   3.071
   1.600    |   3.171
   1.700    |   3.271
   1.800    |   3.373

@@ Lagrange value: -106.4822206
  |L_res - f(x)|: 95.44239094

@@ Newton value: -106.4821921
  |N_res - f(x)|: 95.4423624

@@ True value: -11.03982971

?? Change the value of x or n? [y / n]
y

@@ Interpolation point:
   x = -6
```

```
@@ Degree of interpolation polynomial
   Hint: n <= 13
   n = 13

@@ Sorted table:
     x      |    f(x)
   0.500    |   1.878
   0.600    |   2.025
   0.700    |   2.165
   0.800    |   2.297
   0.900    |   2.422
   1.000    |   2.540
   1.100    |   2.654
   1.200    |   2.762
   1.300    |   2.867
   1.400    |   2.970
   1.500    |   3.071
   1.600    |   3.171
   1.700    |   3.271
   1.800    |   3.373

@@ Lagrange value: -47.921875
  |L_res - f(x)|: 36.88204529

@@ Newton value: -24.16863689
  |N_res - f(x)|: 13.12880718

@@ True value: -11.03982971

?? Change the value of x or n? [y / n]
n
```

Список литературы

[1] Лебедева А.В., Пакулина А.Н. Практикум по методам вычислений. Часть 1. СПб., СПбГУ. 2021. 156 - стр.