Дисциплина: Численные методы
Лабораторное задание №1

Отчет
Тема: Интерполирование функции с помощью многочленов Эрмита по m точкам, в которых заданы значения функции и производных.

Выполнила:
студентка 3 курса 62 группы
Пахомова П.В.

Проверила:
старший преподаватель Фролова О.А.

## 1. Постановка задачи

Входные параметры:

X – вектор значений аргументов в порядке возрастания (вектор узлов интерполяции);
Y – вектор значений функции в узлах интерполяции;
DY – вектор значений производной функции в узлах интерполяции;
N – количество узлов интерполяции, в которых заданы значения функций;
XX – значение аргумента, при котором будет вычисляться интерполяционное значение функции;
m – количество точек, по которым строится многочлен Эрмита.

Выходные параметры:

YY – вычисленное интерполяционное значение функции в точке XX ;

*Метод*. Вычисляется значение интерполяционного многочлена Эрмита в точке XX по значениям функции и её производных в точках, наименее
удалённых от точки XX .

## 2. Теоретическая часть

### 1.4. Многочлены Эрмита

Построим интерполяционный многочлен, принимающий в узлах интерполирования $x_i, i = 0,1,\ldots,n$ значения $f_i$ и имеющий в них производные, равные $f_i'$. Такие многочлены называются **многочленами Эрмита**. Ясно, что многочлен Эрмита имеет степень не меньше, чем $2n+1$, поскольку должны быть удовлетворены $2n+2$ условий. Аналогично коэффициентам Лагранжа $L_n^{(i)}(x)$ определим коэффициенты Эрмита $H_i(x)$ и $h_i(x)$. **Коэффициенты Эрмита** – это многочлены степени $2n+1$, удовлетворяющие условиям:

$$\begin{array}{ll} H_i(x_j) = \delta_{ij}; & h_i(x_j) = 0; \\ H_i'(x_j) = 0; & h_i'(x_j) = \delta_{ij}; \end{array} \quad i, j = 0,1,\ldots,n. \tag{13}$$

Построим многочлен $h_i(x)$. Если $h_i(x_j) = h_i'(x_j) = 0$ при $i \neq j$, то $h_i(x)$ должен содержать множители $(x - x_j)^2$. Так как $h_i(x_i) = 0$, то $h_i(x)$ должен содержать простой множитель $(x - x_i)$. Для того чтобы $h_i'(x_i) = 1$, нужно взять

$$h_i(x) = \frac{(x-x_0)^2(x-x_1)^2\ldots(x-x_{i-1})^2(x-x_i)(x-x_{i+1})^2\ldots(x-x_n)^2}{(x_i-x_0)^2(x_i-x_1)^2\ldots(x_i-x_{i-1})^2(x_i-x_{i+1})^2\ldots(x_i-x_n)^2}. \tag{14}$$

Построение многочлена $H_i(x)$ похоже на построение многочлена $h_i(x)$. Многочлен $H_i(x)$ должен иметь множители $(x - x_j)^2$ для $i \neq j$ и простой множитель $(a_i x + b_i)$. Будем искать $H_i(x)$ в виде:

$$H_i(x) = \frac{(x-x_0)^2(x-x_1)^2\ldots(x-x_{i-1})^2(a_i x + b_i)(x-x_{i+1})^2\ldots(x-x_n)^2}{(x_i-x_0)^2(x_i-x_1)^2\ldots(x_i-x_{i-1})^2(x_i-x_{i+1})^2\ldots(x_i-x_n)^2}. \tag{15}$$

Продифференцировав выражение $H_i(x)$ и подставив в производную $H_i'(x)$ значение $x = x_i$, получим

$$H_i'(x_i) = \frac{2}{x_i - x_1} + \frac{2}{x_i - x_2} + \ldots + \frac{2}{x_i - x_{i-1}} + a_i + \frac{2}{x_i - x_{i+1}} + \ldots + \frac{2}{x_i - x_n}.$$

По определению коэффициента Эрмита $H_i'(x_i) = 0$ и $H_i(x_i) = 1$. Следовательно, необходимо положить

$$a_i = -2 \sum_{k=0, k\neq i}^{n} \frac{1}{x_i - x_k}, \quad b_i = 1 - a_i x_i. \tag{16}$$

Итак, требуемая формула (15) для коэффициентов Эрмита $H_i(x)$ построена. Теперь записать многочлены Эрмита не представляет труда:

$$H(x) = \sum_{i=0}^{n} (H_i(x) f_i + h_i(x) f_i'). \tag{17}$$

Заметив, что коэффициенты Эрмита (14), (15) связаны соотношениями

$$H_i(x) = \frac{(a_i x + b_i) \cdot h_i(x)}{(x - x_i)}, \tag{18}$$

запишем многочлен Эрмита в иной, более удобной для вычислений форме:

$$H(x) = \sum_{i=0}^{n} \frac{[(a_i x + b_i) f_i + f_i')] \cdot h_i(x)}{(x - x_i)}. \tag{19}$$

Во время сдачи преподавателю программы студент на ряде тестовых примеров, которые подготавливает самостоятельно, должен показать, что его программа работает в соответствии с заданием. Построение тестовых примеров требует глубокого понимания программируемого метода и не должно сопровождаться большой вычислительной работой. Ниже приведён ряд общих замечаний относительно разработки тестовых примеров.

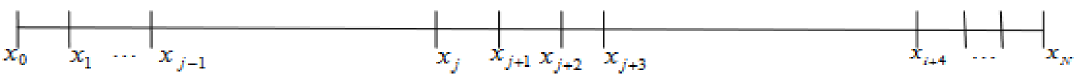
**Замечание 1** (об определении значения функции в точке $x^*$ с помощью интерполяционного многочлена степени $m$). Если дана таблица значений $x_i, y_i$, $i = 0,1,\ldots,n$ и требуется вычислить значение табулированной функции в точке $x^*$, построив многочлен степени $m$ по ближайшим к $x^*$ точкам, то нужно показать, что:

a) действительно строится многочлен степени $m$ и на выход подаётся значение этого многочлена в точке $x^*$;

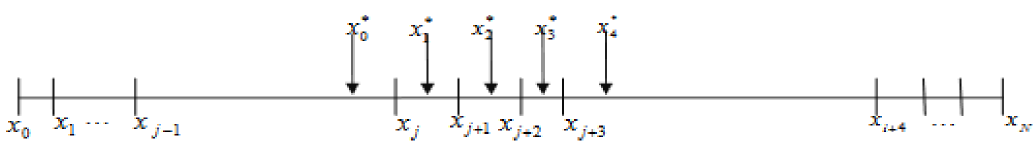
b) многочлен строится по ближайшим точкам;

c) в исходной таблице содержится достаточное количество точек для построения многочлен степени $m$, то есть точек больше, чем $m$. Последнее требование тестируется очевидным образом. Поэтому полагаем, что в исходной таблице достаточно много точек. Тестирование требований a, b рассмотрим на примере, взяв $m = 3$, то есть многочлен должен строиться по четырём ближайшим к $x^*$ точкам.

Выберем такой набор узлов $x_i$, $i = 0,1,\ldots,n$, в котором четыре последовательных узла $x_j, x_{j+1}, x_{j+2} x_{j+3}$ находятся на значительном расстоянии от остальных узлов.




Тогда в точках $x_k^*, k = 0,1,2,3,4$ приближённые значения функции должны определяться значениями многочлена $P_3(x)$, построенного по точкам $x_j, x_{j+1}, x_{j+2} x_{j+3}$:



Возьмём некоторый конкретный многочлен $\tilde{P}_3(x)$ и положим в узлах $x_k, k = j, j+1, j+2, j+3$ табличные значения $y_k$ равными значениям многочлена $\tilde{P}_3(x)$. В остальных узлах $x_i$, $i = 0,1,\ldots,n$, $i \neq j, j+1, j+2, j+3$ табличные значения не есть значения многочлена $\tilde{P}_3(x)$, а равны, к примеру, $1000 + \tilde{P}_3(x)$. Построив описанную выше входную таблицу, можно провести пять тестов: зададимся некоторым конкретным многочленом $\tilde{P}_3(x)$, программно вычислим значения в точке $x_k^*, k = 0,1,2,3,4$. Полученные значения должны совпадать со значениями $\tilde{P}_3(x_k^*)$. Этими тестами мы показываем, что в самом деле строится многочлен третьей степени по ближайшим к $x^*$ узлам таблицы при любом взаимном расположении узлов и точки $x^*$.

Теперь возьмём произвольную сетку узлов $x_i$, $i = 0,1,\ldots,n$, $n > 3$ и произвольное $x^*$, $x_0 < x^* < x_n$. Проведём три теста: во всех узлах сетки значениям $y_i$, $i = 0,1,\ldots,n$ присвоим значения $y_i = Q_0(x_i)$ в первом тесте, $y_i = Q_1(x_i)$ во втором, $y_i = Q_2(x_i)$ в третьем, где $Q_0(x)$, $Q_1(x)$, $Q_2(x)$ - многочлены нулевой, первой и второй степени соответственно. Поскольку многочлены $Q_0(x)$, $Q_1(x)$, $Q_2(x)$ формально являются многочленами третьей степени, вычисленные программные значения функции в точке $x^*$ должны быть равны $Q_0(x^*)$, $Q_1(x^*)$, $Q_2(x^*)$.

## 3. Алгоритм

1. Инициализация:
- Входные параметры: массивы
X , Y , DY ,
значение
XX
и количество ближайших точек m.
- Переменные: YY (результат интерполяции), closestPoints (ближайшие точки к XX ).

2. Нахождение ближайших точек:
- Для каждого элемента массива X находим разницу с XX и сортируем пары по этой разнице.
- Выбираем m пар с наименьшей разницей как ближайшие точки.

3. Вычисление интерполяционного значения YY :
- Для каждой ближайшей точки:
  1. Находим индекс выбранной точки idx .
  2. Вычисляем многочлен L :

$$L = \prod_{j=0}^{m-1} \left( \frac{(XX - X[closestPoints[j].second])^2}{(X[idx] - X[closestPoints[j].second])^2} \right)$$

3. Вычисляем значение H :

$$H = (XX - X[idx]) \cdot L$$

4. Вычисляем значение a , учитывая условие :

$$a = -4 \cdot \sum_{j=0}^{m-1} \frac{1}{X[idx] - X[closestPoints[j].second]}$$

5. Вычисляем значение b :

$$b = 1 - a \cdot X[idx]$$

6. Обновляем YY с учетом текущей точки:

$$YY+ = \left( \frac{(a \cdot XX + b) \cdot Y[idx]}{XX - X[idx]} + DY[idx] \right) \cdot H$$

4. Возвращение результата:

- Возвращаем значение YY как результат интерполяции в точке XX .

## Тестирование

| № теста | Функция | X | Y | DY | XX | m | Ожидаемый результат | Полученный результат |
|---|---|---|---|---|---|---|---|---|
| 1 | x^2 | [1.0, 2.0, 3.0, 4.0, 5.0] | [1.0, 4.0, 9.0, 16.0, 25.0] | [2.0, 4.0, 6.0, 8.0, 10.0] | 2.5 | 1 | 6.25 | 6.0 |
| 2 | x^2 | [1.0, 2.0, 3.0] | [0.0, 1.0, 4.0, 9.0] | [2.0, 4.0, 6.0, 8.0] | 1.5 | 2 | 2.25 | 2.25 |
| 3 | x^2 | [2.0, 4.0, 6.0, 8.0, 10.0] | [4.0, 16.0, 36.0, 64.0, 100.0] | [4.0, 8.0, 12.0, 16.0, 20.0] | 5.0 | 3 | 25.0 | 25..0 |
| 4 | x^4 + 2x^3 + 3x^2 + 4x | [1.0, 2.0, 3.0, 4.0, 5.0] | [10.0, 52.0, 174.0, 448.0, 970.0] | [20.0, 72.0, 184.0, 380.0, 684.0] | 2.5 | 1 | 99.0625 | 88.0 |
| 5 | x^4 + 2x^3 + 3x^2 + 4x | [1.0, 2.0, 3.0, 4.0, 5.0] | [10.0, 52.0, 174.0, 448.0, 970.0] | [20.0, 72.0, 184.0, 380.0, 684.0] | 2.5 | 2 | 99.0625 | 99.0 |
| 6 | x^4 + 2x^3 + 3x^2 + 4x | [1.0, 2.0, 3.0, 4.0, 5.0] | [10.0, 52.0, 174.0, 448.0, 970.0] | [20.0, 72.0, 184.0, 380.0, 684.0] | 2.5 | 3 | 99.0625 | 99.0625 |
| 7 | x^4 + 2x^3 + 3x^2 + 4x | [1.0, 2.0, 3.0, 4.0, 5.0] | [10.0, 52.0, 174.0, 448.0, 970.0] | [20.0, 72.0, 184.0, 380.0, 684.0] | 2.5 | 4 | 99.0625 | 99.0625 |
| 8 | 1000 | [-1000 -999 1 2 3 4 5 1000 ] | [1000 1000 1000....] | [1000 1000 2 4 6 8 10 1000 ] | 3.1 | 3 | 1000 | 1000.57519 |
| 9 | 1000 | [-1000 -999 1 2 3 4 5 1000 ] | [1000 1000 1000....] | [1000 1000 2 4 6 8 10 1000 ] | 1.1 | 3 | 1000 | 999.99316 |
| 10 | 1000 | [-1000 -999 1 2 3 4 5 1000 ] | [1000 1000 1000....] | [1000 1000 2 4 6 8 10 1000 ] | -500 | 3 | 1000 | 6.249971981265587E10 |
| 11 | x+1 | [1.0, 2.0, 3.0, 4.0, 5.0] | [2.0, 3.0, 4.0, 5.0, 6.0] | [1.0, 1.0, 1.0, 1.0, 1.0] | 2.5 | 2 | 3.5 | 3.5 |
| 12 | x^2 | [1.0, 2.0, 3.0, 4.0, 5.0] | [1.0, 4.0, 9.0, 16.0, 25.0] | [2.0, 4.0, 6.0, 8.0, 10.0] | 2.5 | 3 | 6.25 | 6.25 |
| 13 | x^3 | [1.0, 2.0, 3.0, 4.0, 5.0] | [1.0, 8.0, 27.0, 64.0, 125.0] | [3.0, 12.0, 27.0, 48.0, 75.0] | 2.5 | 4 | 15.625 | 15.625 |

Примеры 1-7 показывают, что при использовании функции hermiteInterpolation с заданными значениями X, Y, DY, XX и m, программа для многочлена степени 2 и 4 вычисляет значение этого многочлена в точке XX даже если m, количество точек, меньше степени+1 многочлена.

Примеры 8-10 демонстрируют работу функции hermiteInterpolation и её точность в зависимости от удалённости точек, этими тестами мы так же показываем, что строится многочлен по ближайшим к XX узлам таблицы при любом взаимном расположении узлов и точки XX

Примеры 11-13 показывают, что действительно строится многочлен степени n и на выход подаётся значение этого многочлена в точке XX