Университет ИТМО, кафедра ВТ

Лабораторная работа №3 по “Вычислительной математике”

"Интерполирование многочленом Ньютона"

Работу выполнил

студент группы Р3200

Рогов Я. С.

Преподаватель:

Исаев И.В.

Санкт-Петербург, 2016

Описание метода:
Основная идея интерполяции в большинстве методов, в том числе и Ньютона, заключается в подборе такого многочлена, что значения полученной интерполированной функции находились как можно ближе к оригиналу. Отличительной особенностью же метода Ньютона аргументы – точки, по которым строится интерполяция – является равноудалёнными (по оси OX).
Сам многочлен ищется в виде:

$$N(x)=a_0+a_1(x-x_0)+a_2(x-x_0)(x-x_1)+\dots a_n(x-x_0)(x-x_1)\dots(x-x_{n-1}) \quad (1)$$

Т.к. график должен проходить через заданные узлы (аргументы), то можно выразить коэффициенты из них:

$$\begin{cases} N(x_0)=a_0=y_0 \\ N(x_1)=a_0+a_1(x_1-x_0)=a_0+a_1 h=y_1 \\ N(x_2)=a_0++a_1(x_2-x_0)+a_2(x_2-x_0)(x_2-x_1)=a_0+2a_1 h+2a_2 h^2 \\ \dots \end{cases} \quad (2)$$

Введём понятие **конечной разности:**

$$\Delta^n y_k=\Delta^{n-1} y_{k+1}-\Delta^{n-1} y_k; \Delta^0 y_k=y_k \quad (3)$$

Тогда, выразив коэффициенты a из выражения (2) и подставив выражения (3) мы получим:

$$\begin{cases} a_0=y_0 \\ a_1=\dfrac{\Delta y_0}{h} \\ a_2=\dfrac{\Delta^2 y_0}{2h^2} \\ \dots \end{cases}$$

Тогда выражение (1) можно записать в виде:

$$N(x)=y_0+\frac{\Delta y_0}{h}(x-x_0)+\frac{\Delta^2 y_0}{2!h^2}(x-x_0)(x-x_1)+\dots+\frac{\Delta^n y_0}{n!h^n}(x-x_0)(x-x_1)\dots(x-x_{n-1})$$

Или, введя коэффициент *b:*

$$b^n=\frac{\Delta^n y_0}{n!h^n}$$

Формулу можно записать:

$$N(x)=\sum_{k=0}^{n} b^k \prod_{m=0}^{k-1}(x-x_m)$$

Именно алгоритм в этой форме и будет реализован.

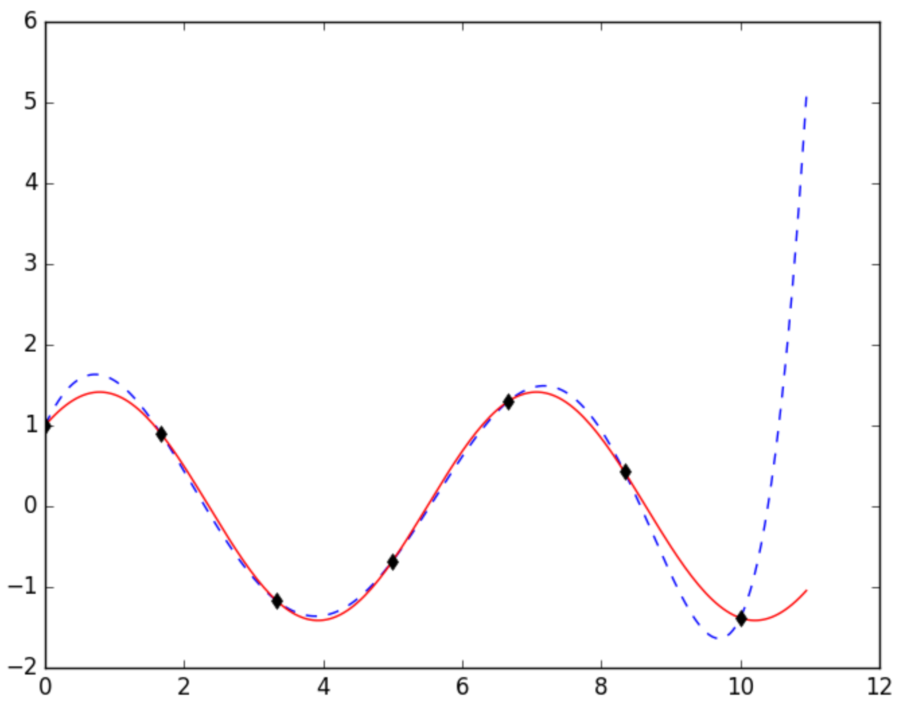
Реализация: сам алгоритм разбит на две части: вычисление коэффициентов *b* и вычисление значения функции по этим коэффициентам. Реализация алгоритма написана на C, функции вызываются как составляющие динамической библиотеки **lib_poly_newton.so** из программы-обёртки для UI и вывода графиков **newtoninterp.py**, написанной на Python.

Код библиотеки **lib_poly_newton.c**

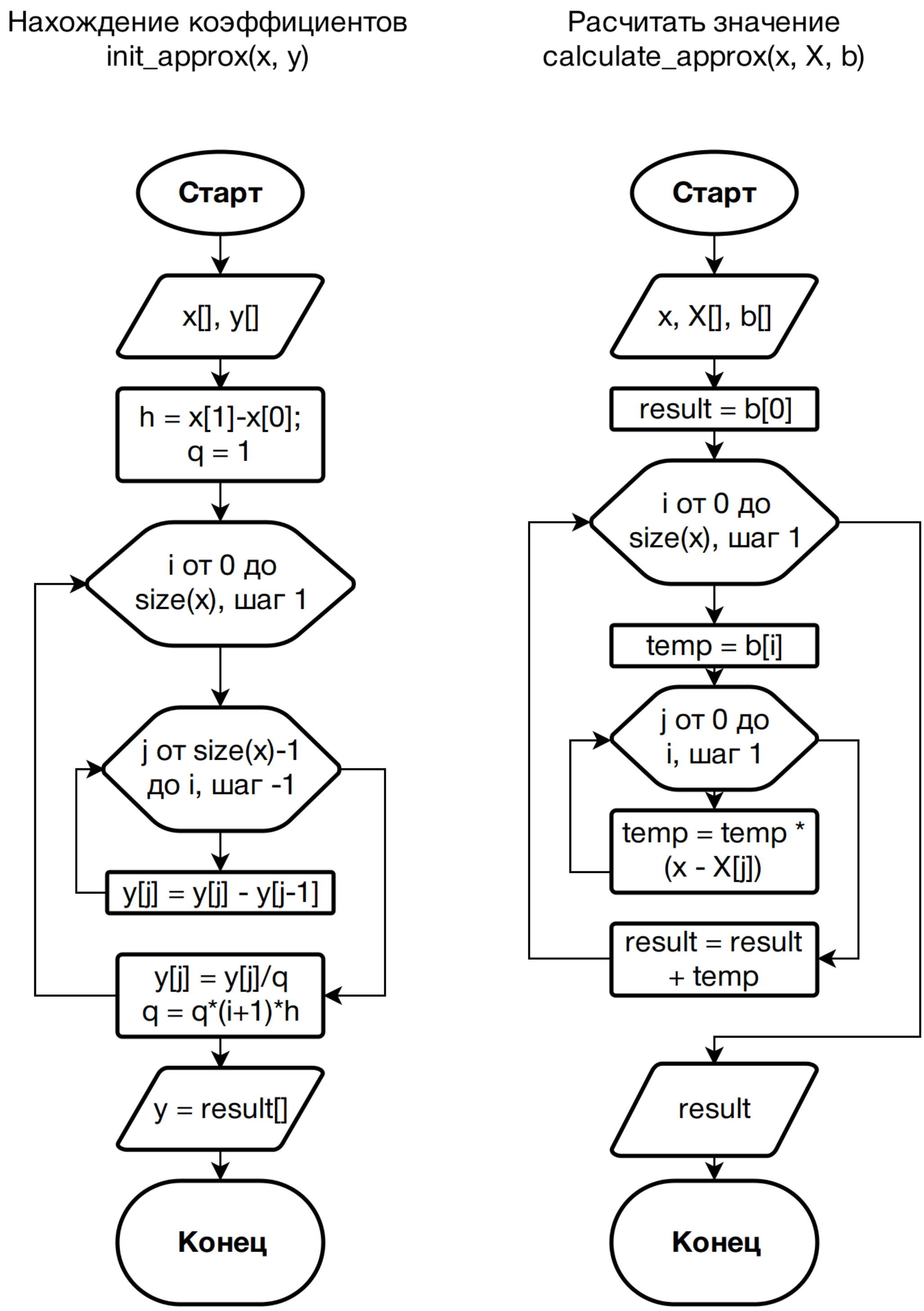
```
#include <stdio.h>

int init_approx(double* x, double* y, int size){
	int i, j;
	double q;
	double h = x[1]-x[0];
	q=1;
	for(i=0; i<size; i++){
		for(j=size-1; j>i; j--)
			y[j]-=y[j-1];
		y[j]/=q;
		q*=(i+1)*h;
	}
	return 0;
}

int calculate_approx(double x, double* X, double* b, int size, double* result){

	int i,j;
	double temp;
	*result=b[0];
	for( i=1; i<size; i++){
		temp = b[i];
		for (j=0; j<i; j++)
			temp *= (x-X[j]);
		*result += temp;
	}
	return 0;
}
```

Пример работы программы:

```
./newtoninterp.py
Введите функцию f(x):
sin(x)+cos(x)
Введите значение начального x:
0
Введите значение конечного x:
10
Введите количество точек:
7
```



Блок-схема метода:



Вывод: в ходе выполнения данной лабораторной работы я изучил варианты интерполяции функций по точкам, и написал реализацию интерполяции многочленом Ньютона.