Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)» (МГТУ
им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Лабораторная работа № 3

Тема Интерполяция функции кубическим сплайном

Студент Челядинов Илья

Группа ИУ7-43Б

Оценка (баллы) _______________

Преподаватель Градов Владимир Михайлович

Москва.
2020 г.

Интерполяция функции кубическим сплайном

Задание

Построить алгоритм и программу кубической сплайн интерполяции.

Решим СЛАУ

$$\begin{cases} \mathrm{C}_1 = 0 \\ h_{n-1}C_{n-1} + (h_{n-1} + h_n)C_n + h_nC_{n+1} = 3(\frac{y_n - y_{n-1}}{h_n} - \frac{y_{n-1} - y_{n-2}}{h_{n-1}}) \\ C_{N+1} = 0 \end{cases}$$

2$\leq$ n $\leq$ N

Выписанная система уравнений решается методом прогонки, которая состоит из двух этапов.

1 этап - прямой ход, 2 этап - обратный ход

(1) $y_n = \xi_{n+1}y_{n+1} + \eta_{n+1}$

(2) $\xi_{n+1} = \frac{D_n}{B_n - A_n\xi_n}$

$\eta_{n+1} = \frac{A_n\eta_n + F_n}{B_n - A_n\xi_n}$

(3)$\xi_1 = \frac{-M_0}{K_0}y_1$

$\eta_1 = \frac{P_0}{K_0}$

(4)$y_N = \frac{P_N - M_N\eta_n}{K_N + M_N\xi_n}$

Начальные условия

$\xi_2 = 0$

$\eta_2 = 0$

```python
from math import cos
import lab_01

def f(x):
    return x * x


def get_table(x_beg, step, amount):
    x_tbl = [x_beg + step * i for i in range(amount)]
    y_tbl = [f(x) for x in x_tbl]
    return x_tbl, y_tbl


def print_table(x, y):
    length = len(x)
    for i in range(length):
        print("%.4f %.4f" % (x[i], y[i]))
    print()
```

```python
def interpolate(x, y, x_value):
    n = len(x)
    i_near = min(range(n), key=lambda i: abs(x[i] - x_value))  # index of nearest value

    h = [0 if not i else x[i] - x[i - 1] for i in range(n)]  # step value

    A = [0 if i < 2 else h[i - 1] for i in range(n)]
    B = [0 if i < 2 else -2 * (h[i - 1] + h[i]) for i in range(n)]
    D = [0 if i < 2 else h[i] for i in range(n)]
    F = [0 if i < 2 else -3 * ((y[i] - y[i - 1]) / h[i] - (y[i - 1] - y[i - 2]) / h[i - 1]) for i in range(n)]

    # forward
    ksi = [0 for i in range(n + 1)]
    eta = [0 for i in range(n + 1)]
    for i in range(2, n):
        ksi[i + 1] = D[i] / (B[i] - A[i] * ksi[i])
        eta[i + 1] = (A[i] * eta[i] + F[i]) / (B[i] - A[i] * ksi[i])

    # backward
    c = [0 for i in range(n + 1)]
    for i in range(n - 2, -1, -1):
        c[i] = ksi[i + 1] * c[i + 1] + eta[i + 1]

    a = [0 if i < 1 else y[i - 1] for i in range(n)]
    b = [0 if i < 1 else (y[i] - y[i - 1]) / h[i] - h[i] / 3 * (c[i + 1] + 2 * c[i]) for i in range(n)]
    d = [0 if i < 1 else (c[i + 1] - c[i]) / (3 * h[i]) for i in range(n)]

    '''
    print(h, '\n', A, '\n', B, '\n', D, '\n', F)
    print()
    print(ksi, '\n', eta)
    print()
    print(a, '\n', b, '\n', c, '\n', c)'''

    return a[i_near] + b[i_near] * (x_value - x[i_near - 1]) + c[i_near] * ((x_value - x[i_near - 1]) ** 2) + d[
        i_near] * ((x_value - x[i_near - 1]) ** 3)
```

```
x_beg = float(input("Input beginning value of x: "))
x_step = float(input("Input step for x value: "))
x_amount = int(input("Input amount of dots: "))

x_tbl, y_tbl = get_table(x_beg, x_step, x_amount)
print("\nCreated table:")
print_table(x_tbl, y_tbl)

x = float(input("Input x: "))

# Results
found = interpolate(x_tbl, y_tbl, x)
print("\nInterpolated: ", found)
print("F(x)         : ", f(x))
print("Error        : ", abs(f(x) - found), "\n")
lab_1_get_int()
```

Pn

Для сравнения результатов, использовалась программа из ЛР1. Она вызывалась последней строчкой. Там было изменено только степень(3) полинома и точка X
Результаты работы:

```
-6.5000 42.2500
-6.0000 36.0000
-5.5000 30.2500
-5.0000 25.0000
-4.5000 20.2500
-4.0000 16.0000
-3.5000 12.2500
-3.0000 9.0000
-2.5000 6.2500
-2.0000 4.0000
-1.5000 2.2500
-1.0000 1.0000
-0.5000 0.2500
0.0000 0.0000
0.5000 0.2500
1.0000 1.0000
1.5000 2.2500
2.0000 4.0000
2.5000 6.2500
3.0000 9.0000
3.5000 12.2500
4.0000 16.0000
4.5000 20.2500
5.0000 25.0000
5.5000 30.2500
6.0000 36.0000
6.5000 42.2500
7.0000 49.0000
7.5000 56.2500
8.0000 64.0000
8.5000 72.2500
9.0000 81.0000
9.5000 90.2500

Input x: 3

Interpolated:  9.0
F(x)        :  9.0
Error       :  0.0


Pn[x] = 9.00
```

Ответы на контрольные вопросы:

1) Выписать значения коэффицентов сплайна, построенного на двух точках.

   Функция выродится в прямую, так как коэффиценты c и d будут 0.

2) Выписать все условия для определения коэффицентов сплайна, построенного на 3 точках.

   Для того, чтобы однозначно записать один полином 3 степени, необходимо 4 условия. Так как нужно построить сплайн на 3 точках, необходимо 4 * 2 = 8 условий.

Условия таковы:

Первый полином определен на первой и второй точках(это два условия).

Второй полином определен на второй и третьей точках(еще два условия).

S1’(x1) = k1

Sn-1’(xn) = k2

S1’(x) = S2’(x)

S1”(x) = S2’’(x)

Также нужно задать граничные условия вида S’’(a) = S’’(b) = 0

Условия непрерывности второй производной S’’(a) = S’’(b) = 0

Условие переодичности S’(a) = S’(b), S’’(a) = S’’(b)

3) Определить начальные значения прогоночных коэффицентов, если принять, что для сплайна справедливо C1 = C2.

№3 $C_1 = C_2$

$C_1 = \varepsilon 2 C_2 + \eta 2$

$\varepsilon_2 C_2 + \eta_2 \Rightarrow \underline{\varepsilon_2 = 1}$ $\quad \eta_2 = 0$

4) Написать формулу для определения последнего коэффицента сплайна Cn, чтобы можно было выполнить обратный ход метода прогонки, если задано kCn+1 + mCn = p, где k, m, p –-заданные числа.

N4

$K C_{N-1} + m C_N = p$

$C_{N-1} = \varepsilon_N C_N + \eta_N \Rightarrow K \varepsilon_N C_N + K \cdot \eta_N + m C_N = p$

$\Rightarrow C_N = (p - K\eta_N) / (K \varepsilon_N + m)$