<u>МГТУ им. Н.Э. Баумана</u>

ВЫЧИСЛИТЕЛЬНЫЕ АЛГОРИТМЫ

Лабораторный практикум №3

по теме: «*Построение и программная реализация алгоритма сплайн-интерполяции табличных функций*.»

Студент: Нгуен Фыок Санг
Группа: ИУ7И-46
Преподаватель: Градов В.М.

2020

Цель работы. Получение навыков владения методами интерполяции таблично заданных функций с помощью кубических сплайнов.

Исходные данные.

1. Таблица функции с количеством узлов N
2. Значение аргумента x.

Результат работы программы.

Значения y(x).

```
Input beginning value of x: -5
Input step for x: 1
Input number of x: 10

Table:
-5.0000 35.0000
-4.0000 24.0000
-3.0000 15.0000
-2.0000 8.0000
-1.0000 3.0000
0.0000 0.0000
1.0000 -1.0000
2.0000 0.0000
3.0000 3.0000
4.0000 8.0000

Input x: 1.75

F_inter:  -0.46117912371134007
F(x)        :  -0.4375
Error       :  0.023679123711340067
```

Алгоритм:

Кубический сплайн — это кривая, состоящая из „состыкованных“ полиномов третьей степени (y (IV) (x) = 0). В точках стыковки значения и производные двух соседних полиномов равны.

На каждом отрезке $[x_{i-1}, x_i]$, (i = 1..N) функция $\varphi(x)$ есть полином третьей степени $\varphi_i(x)$ в виде: $\varphi_i(x) = a_i + b_i(x - x_{i-1}) + c_i(x - x_{i-1})^2 + d_i(x - x_{i-1})^3$

- $y_{i-1} = a_i$ (1)
- $y_i = a_i + b_i h_i + c_i h_i^2 + d_i h_i^3$ (2)
- $\varphi'_i(x) = b_i + 2c_i(x - x_{i-1}) + 3d_i(x - x_{i-1})^2$
- $\varphi''_i(x) = 2c_i + 6d_i(x - x_{i-1})$
- $b_{i+1} = b_i + 2c_i h_i + 3d_i h_i^2$ (3)
- $c_{i+1} = c_i + 3d_i h_i$ (4)

- $\varphi'_1(x_0) = 0 \Rightarrow c_1 = 0$ (5)
- $\varphi''_N(x_N) = 0 \Rightarrow c_N + 3d_N = 0$ (6)

Решение системы:

$C_1 = 0$

$h_{i-1}c_{i-1} + 2(h_{i-1}+h_i)c_i + h_ic_{i+1} = 3(\frac{y_i-a_{i-1}}{h_i} - \frac{y_{i-1}-y_{i-2}}{h_{i-1}})$

$c_{N-1} = 0$

- $A_iy_{i-1} - B_iy_i + D_iy_{i+1} = -F_i$ (7)
- $K_0y_0 + M_0y_1 = P_0$ (7')
- $K_Ny_N + M_Ny_{N+1}\ P_N$ (7'') , $(i = 1..N-1)$
- $y_i = \xi_{i+1}y_{i+1} + \eta_{i+1}$ (8)
- $A_i\,\xi_iy_i + A_i\eta_i - B_iy_i + D_iy_{i+1} = -F_i$
- $Y_i = \frac{D_i}{B_i-A_i\xi_i}y_{i+1} + \frac{F_i+A_i\eta_i}{B_i-A_i\xi_i}$ (9)
- $\xi_{i+1} = \frac{D_i}{B_i-A_i\xi_i}$, $\eta_{i+1} = \frac{F_i+A_i\eta_i}{B_i-A_i\xi_i}$ (10)

Найдем начальное значение коэффициент:

Из (7') => $y_0 = -\frac{M_0}{K_0}y_1 + \frac{P_0}{K_0}$

Из (8) => $y_0 = \xi_1y_1 + \eta_1$

Отсюда $\xi_1 = -\frac{M_0}{K_0}$, $\eta_1 = \frac{P_0}{K_0}$

$c_i = \xi_{i+1}c_{i+1} + \eta_{i+1}$, $c_1 = 0$

$c_1 = \xi_2c_2 + \eta_2$, т.е $\eta_2 = 0$, $c_2 = 0$

$a_i = y_{i-1}$

$d_i = \frac{c_i-c_{i-1}}{3h_i}$

$b_i = \frac{a_i-a_{i-1}}{h_i} + \frac{2c_i+c_{i-1}}{3}\ h_i$

$y_i = a_i + b_i h_i + c_i h_i^2 + d_i h_i^3$

Ответы на вопросы:

1. **Выписать значения коэффициентов сплайна, построенного на двух точках .**
   Сплайн построенный на 2 точках и есть прямая линия. Соответственно коэффициенты c и d становятся равными нулю(c = 0, d = 0).
   Коэффициент a[i] = y[i — 1]. А коэффициент

   b = (y[i] — y[i-1]/(x[i] — x[i -1])).

2. **Выписать все условия для определения коэффициентов сплайна, построенного на 3-х точках**

   Для определения коэффициенты сплайна для 3-х точек, необходимо чтоб выполнялись следующие условия:

   - ξ2 ``(x0) = 0
   - ξ2 ``(x2) = 0
   - η1`(x1) = η2`(x1)
   - η1``(x1) = η2``(x1)
   - ξ1(x0) = y0
   - ξ1(x1) = y1
   - ξ2(x1) = y1
   - ξ2(x2) = y2

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

   Для того чтобы найти начальные прогоночные коэффициенты и приняв в условие то, что c1 = c2, имеем: Воспользуемся формулой y [i]= ξ [i+1] * y [i+1] + η [i+1].

Для поиска коэффициента сплайна, знаем что c[i] и y[i] совпадают, то есть прямо пропорциональны.

Отсюда и получаем выражение: $c[i]= \xi [i+1] * c [i+1] + \eta [i+1]$. Следовательно получаем $c1 = \xi 2 * c2 + \eta 2$. Зная, что $c1 = c2$, получаем что $c1 = \xi 2 * c1+ \eta 2$. Отсюда следует что $\xi = 1$ и $\eta = 0$.

4. **Написать формулу для определения последнего коэффициента сплайна C_N, чтобы можно было выполнить обратный ход метода прогонки, если задано $kC_{N-1}+mC_N=p$, где k,m и p - заданные числа.**

Преобразовав формулу указанной в пункте 3 получим, $c[n — 1] = \xi[i] * c[n] + \eta[i]$.

Зная начальное условие $k * c[n—1] = -m * c[n] +p$, получим что $k = 1$, $m = -1$, $p = 1$.

Подставив эти вычисленные значения в начальное условие получим, что $C[n] = C[n -1] — 1$.

Код программы:

```
from math import cos

def f(x):
   return x*x - 2 * 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()

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)]

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

    c = [0 for i in range(n + 1)]
    for i in range(n - 2, -1, -1):
        c[i] = si[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)]


    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: "))
x_amount = int(input("Input number of x: "))

x_tbl, y_tbl = get_table(x_beg, x_step, x_amount)
print("\nTable:")
print_table(x_tbl, y_tbl)
```

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

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