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 «Теоретическая информатика и компьютерные технологии»

Отчёт о лабораторной работе № 1

по курсу «Численные методы»

ИНТЕРПОЛЯЦИЯ B-СПЛАЙНАМИ

Студент: К. Лозовска

Группа: ИУ9И-64Б

Преподаватель: А.Б.Домрачева

Москва, 2023

СОДЕРЖАНИЕ

ЦЕЛЬ И ПОСТАНОВКА ЗАДАЧИ

Цель:

Интерполировать функцию $f(x)$ кубическими сплайнами и оценить погрешность интерполяции в серединах частичных отрезков интерполяции.

Постановка задачи:

Дано: Функция $y_i = \varphi(x_i)$, $i = \overline{1, n}$ задана таблично, исходные данные включают ошибки измерения.

x_1	x_2	…	x_{n-1}	x_n
y_1	y_2	…	y_{n-1}	y_n

Вычислить: Функцию (интерполянту) $f(x)$, совпадающую с значениями $y_i, i = \overline{1, n}$ в контрольных точках $x_i, i = \overline{1, n} : f(x_i) = y_i$.

Тестовый пример: Зададим функцию $\varphi(x)$ таблично:

1.0	1.5	2.0	2.5	3.0	3.5	4.0	4.5	5.0
0.95	2.07	1.96	2.62	3.75	4.12	3.98	3.63	4.70

ОСНОВНЫЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Интерполяция, интерполирование — в вычислительной математике способ нахождения промежуточных значений величины по имеющемуся дискретному набору известных значений.

Основная цель интерполяции — получить быстрый (экономичный) алгоритм вычисления значений $y(x)$ для значений x, не содержащихся в таблице данных. Интерполирующие функции $f(x)$, как правило строятся в виде линейных комбинаций некоторых элементарных функций: $f(x) = \sum_{k=0}^{N} c_k \Phi_k(x)$, где $\{\Phi_k(x)\}$ – фиксированный линейно независимые функции, $c_0, c_1, \dots, c_n$ – не определенные пока коэффициенты.

В случае задачи интерполяции исходной функции $f(x)$ при условии, что $f(x_i) = g(x_i), i = 1, \dots, n$ в узлах интерполяции. Очевидно, решение задачи интерполяции не единственно. Т.е. задача некорректно поставлена. В данной работе интерполяцию будем производить кубическими сплайнами.

Сплайн – кусочно-полиномиальная функция с постоянными коэффициентами (для каждого интервала свои). Как правило, на отдельных интервалах области определения степень полинома не меняется, меняются только коэффициенты:

$$S_i(x) = a_i + b_i(x - x_{i-1}) + c_i(x - x_{i-1})^2 + d_i(x - x_{i-1})^3, x \in [x_{i-1}, x_i]$$

$$h = \frac{b-a}{n}, x_i = a + ih, i = 0, \dots, n$$

На каждый частном отрезке $x \in [x_{i-1}, x_i]$, $i = 1, \dots, n$ определяются коэффициенты a_i, b_i, c_i, d_i, при этом выполняются условия гладкости, в том числе условия гладкости на краях отрезка:

$$S_i(x_i) = S_{i+1}(x_i)$$

$$S_i'(x_i) = S_{i+1}'(x_i)$$

$$S_i''(x_i) = S_{i+1}''(x_i)$$

$$S_0''(x_0) = S_1''(x_0)$$

$$S_{n-1}''(x_n) = S_n''(x_n)$$

Составим СЛАУ с разреженной матрицей для нахождения коэффициентов c_i:

$$\begin{cases} 4c_2 + c_3 = \frac{3}{h^2}(y_2 - 2y_1 + y_0) \\ \dots \\ c_{i-1} + 4c_i + c_{i+1} = \frac{3}{h^2}(y_{i+1} - 2y_i + y_{i-1}) \\ \dots \\ c_{n-1} + 4c_n = \frac{3}{h^2}(y_n - 2y_{n-1} + y_{n-2}) \end{cases}$$

Остальные коэффициенты сплайн-функции находятся по формулам:

$$a_i = y_{i-1}, i = 1, \dots, n$$

$$b_i = \frac{y_i - y_{i-1}}{h} - \frac{h}{3}(c_{i+1} + 2c_i)$$

$$c_1 = 0, c_{n+1} = 0$$

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

Для решения полученной СЛАУ используется метод прогонки, реализованный в лабораторной работе №1.

ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ

Данный код реализует интерполяцию кубическими сплайнами.

Функция ***f(x)*** вычисляет значение экспоненты в точке x.

```
import math

SIZE = 8

def f(x):
    return math.exp(x)
```

Функция ***direct(b, a, c, d, size)*** решает систему уравнений методом прогонки для трехдиагональной матрицы, где b, a, c, d — это коэффициенты матрицы, а size - размер матрицы.

```
def direct(b, a, c, d, size):
    alpha = [-c[0] / b[0]]
    beta = [d[0] / b[0]]
    y = 0.0
    for i in range(1, size - 1):
        y = a[i - 1] * alpha[i - 1] + b[i]
        alpha.append(-c[i] / y)
        beta.append((d[i] - a[i - 1] * beta[i - 1]) / y)
    y = a[size - 2] * alpha[size - 2] + b[size - 1]
    beta.append((d[size - 1] - a[size - 2] * beta[size - 2]) / y)
    return alpha, beta
```

Функция ***reverse(alpha, beta, size)*** решает систему уравнений методом обратной прогонки.

```python
def reverse(alpha, beta, size):
    x = [0.0] * size
    x[size - 1] = beta[size - 1]
    for i in range(size - 2, -1, -1):
        x[i] = alpha[i] * x[i + 1] + beta[i]
    return x
```

Функция ***main()*** вычисляет значения коэффициентов сплайнов и выводит результаты интерполяции для узлов и середин отрезков.

```python
def main():
    l, r = 1, 5
    h = (r - l) / SIZE
    xs = [1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0]
    ys = [0.95, 2.07, 1.96, 2.62, 3.75, 4.12, 3.98, 3.63, 4.70]

    d = [3 * (ys[i + 1] - 2 * ys[i] + ys[i - 1]) / (h ** 2) for i in range(1, SIZE)]
    b = [4] * (SIZE - 1)
    a = [1] * (SIZE - 2)
    c = [1] * (SIZE - 2)
```

```python
    alpha, beta = direct(b, a, c, d, SIZE - 1)
    coefC = reverse(alpha, beta, SIZE - 1)
    coefC.insert(0, 0)
    coefC.append(0)

    coefA = ys[:-1]
    coefB = [(ys[i + 1] - ys[i]) / h - (h / 3) * (coefC[i + 1] + 2 * coefC[i]) for i in range(SIZE)]
    coefD = [(coefC[i + 1] - coefC[i]) / (3 * h) for i in range(SIZE)]
```

Вывод программы содержит значения x, y, y*, |y-y*|, где x - значение аргумента, y - значение функции в точке x, y* - значение интерполяционного многочлена в точке x, |y-y*| - модуль разности между y и y*.

```python
    print("\nInterpolation nodes and middles of it:")
    for i in range(SIZE):
        varX = l + i * h
        varY = f(varX)
        s = coefA[i] + coefB[i] * (varX - xs[i]) + coefC[i] * (varX - xs[i]) ** 2 + coefD[i] * (varX - xs[i]) ** 3
        print(f"x: {varX:.1f}, y: {varY:.16f}, y*: {s:.16f}, |y-y*|: {abs(varY - s):.16f}")

        varX1 = l + (i + 0.5) * h
        varY1 = f(varX1)
        s = coefA[i] + coefB[i] * (varX1 - xs[i]) + coefC[i] * (varX1 - xs[i]) ** 2 + coefD[i] * (varX1 - xs[i]) ** 3
        print(f"x: {varX1:.2f}, y: {varY1:.16f}, y*: {s:.16f}, |y-y*|: {abs(varY1 - s):.16f}")

main()
```

ТЕСТИРОВАНИЕ

Для тестирования полученной программы была задана табличная функция:

1.0	1.5	2.0	2.5	3.0	3.5	4.0	4.5	5.0
0.95	2.07	1.96	2.62	3.75	4.12	3.98	3.63	4.70

Для заданных узлов интерполяции (x_i, y_i) построен кубический сплайн с коэффициентами:

a: [0.95, 2.07, 1.96, 2.62, 3.75, 4.12, 3.98, 3.63]

b: [2.984795655375552, 0.7504086892488955, 0.07356958762886645, 2.2553129602356408, 1.6451785714285712, 0.16397275405007328, -0.9210695876288663, 0.5803055964653905]

c: [0, -4.468773932253313, 3.1150957290132553, 1.2483910162002934, -2.468659793814432, -0.49375184094256214, -1.676332842415317, 4.679083210603831, 0]

d: [-2.979182621502209, 5.055913107511046, -1.2444698085419745, -2.47803387334315, 1.31660530191458, -0.7883873343151699, 4.236944035346099, -3.1193888070692206].

Значения функции в узлах интерполяции и в серединах частичных отрезков между узлами интерполяции представлены в таблице 1:

Таблица 1 – Результаты программы

x_i	y_i
1	0.95
1.25	1.6496491853829161
1.5	2.0699999999999998
1.75	2.0573024438512517
2	1.96
2.25	2.1536410392120766
2.5	2.6200000000000001
2.75	3.2231333993004418

3	3.75
3.25	4.0275753635861555
3.5	4.1200000000000001
3.75	4.1178151463549337
4	3.98
4.25	3.7111640509941086
4.5	3.6299999999999999
4.75	4.0187786496686302
5	4.23

ВЫВОДЫ

В результате тестирования был сделан вывод о точности данного метода приближения функции. Таким образом, интерполяция кубическими сплайнами является одним из наиболее точных методов приближения функции. Она позволяет получить гладкий график, который проходит через все заданные узлы интерполяции.

Однако при использовании данного метода необходимо учитывать, что приближение функции в серединах частичных отрезков между узлами интерполяции может содержать погрешность вследствие ошибок, возникающих при реализации арифметических операций на ЦВМ.

Тем не менее, катастрофического роста погрешности не наблюдается, что делает метод интерполяции кубическими сплайнами очень привлекательным для использования в задачах приближения функции.