МИНОБРНАУКИ РОССИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технологический университет «СТАНКИН»
(ФГБОУ ВО МГТУ «СТАНКИН»)

Кафедра прикладной математики

Учебный курс «Вычислительная математика»

Лабораторная работа №2.

Интерполирование кубическим сплайном дефекта 1

Вариант 15

Выполнил: студент Набойщиков А. А.
Группы ИДБ-22-15
Дата выполнения 15.04.2024

Оценка
Дата

Проверил:
преподаватель Москалёв П.В.

Москва 2024 г.

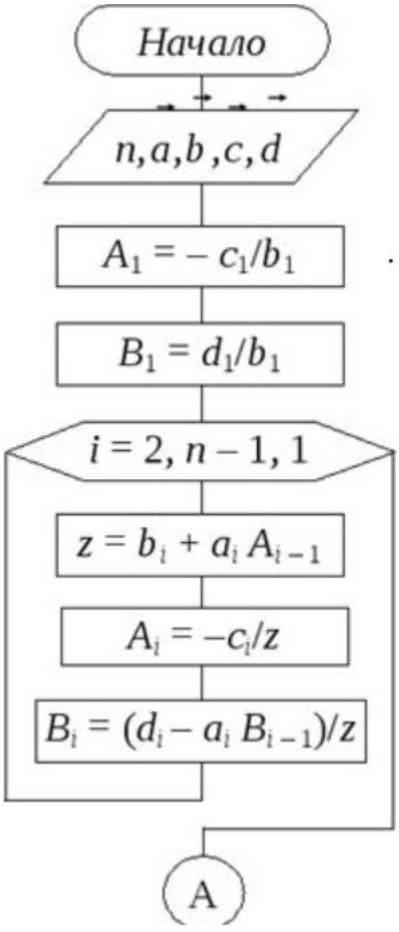
Цель работы: изучить метод интерполяции кубическим сплайном дефекта 1 и применить его на практике для получения сплайна функции f(x) на отрезке [a, b].

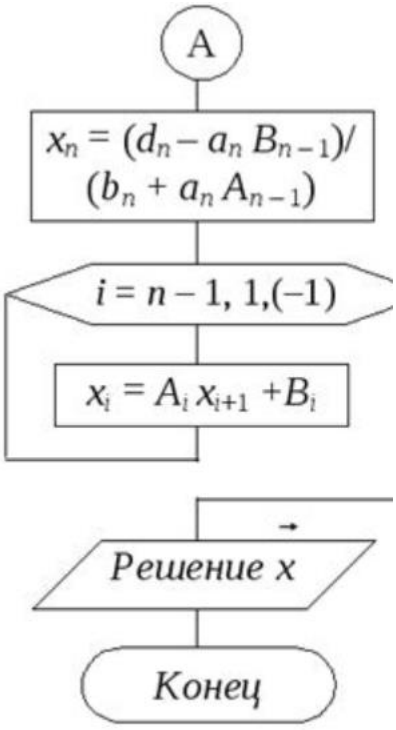
Для варианта 15: f(x) = $e^{-\frac{(x-\frac{15}{10})^2}{2}}$ на отрезке [-2, 2]

Значения исходной функции f(x) для 6 точек:

x	y
-2.0	0.10315887444431626
-1.2	-1.4128568960923964
0.3999999999999999	-0.02064938155235707
0.40000000000000036	1.4140628002466882
1.2	-0.061930534990095154
2.0	-1.4104461161715403

Блок-схема алгоритма метода прогонки





Код программы:

```
def cubic_spline_interpolation_my(xs: list[float], ys: list[float]):
    n = len(xs)

    # Вычисляем коэффициенты для кубических сплайнов
    h = [xs[i+1] - xs[i] for i in range(n-1)]
    alpha = [(3/h[i]) * (ys[i+1]-ys[i]) - (3/h[i-1]) * (ys[i]-ys[i-1]) for i in range(1, n-1)]

    l = [1] * n
    mu = [0] * n
    z = [0] * n

    # Прямой проход метода прогонки
    for i in range(1, n-1):
        l[i] = 2 * (xs[i+1] - xs[i-1]) - h[i-1] * mu[i-1]
        mu[i] = h[i] / l[i]
        z[i] = (alpha[i-1] - h[i-1] * z[i-1]) / l[i]

    # Обратный проход метода прогонки
    c = [0] * n
    b = [0] * n
    d = [0] * n
    for j in range(n-2, -1, -1):
        c[j] = z[j] - mu[j] * c[j+1]
        b[j] = (ys[j+1] - ys[j]) / h[j] - h[j] * (c[j+1] + 2 * c[j]) / 3
        d[j] = (c[j+1] - c[j]) / (3 * h[j])

    # Находим минимальное и максимальное значение x
    min_x = min(xs)
    max_x = max(xs)

    # Вычисляем значения для интерполяции в пределах min_x и max_x с шагом 0.1
    x_interp = [min_x + i * 0.1 for i in range(int((max_x - min_x) / 0.1) + 1)]

    # Интерполяция для новых значений
    y_interp = []

    def interpolated_function(x_val):
        for j in range(n-1):
            if xs[j] <= x_val <= xs[j+1]:
                return ys[j] + b[j] * (x_val - xs[j]) + c[j] * (x_val - xs[j]) ** 2 + d[j] * (x_val - xs[j]) ** 3

    return interpolated_function

cs_my = cubic_spline_interpolation_my(xs, ys)
x_interp_my = np.linspace(xs[0], xs[-1], 100)
y_interp_my = [cs_my(x) for x in x_interp]
```

График исходной функции:

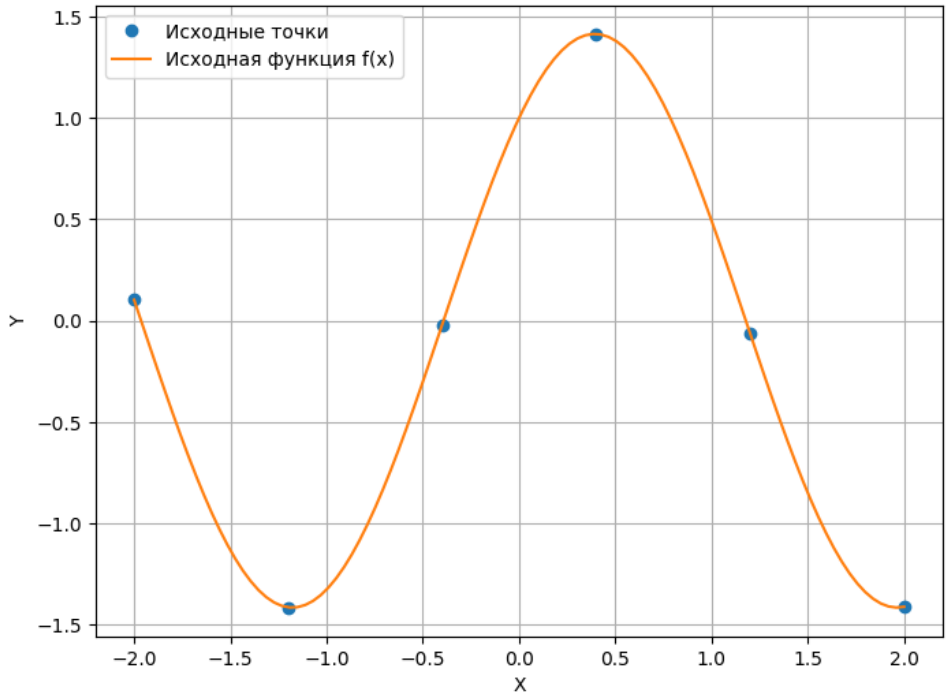
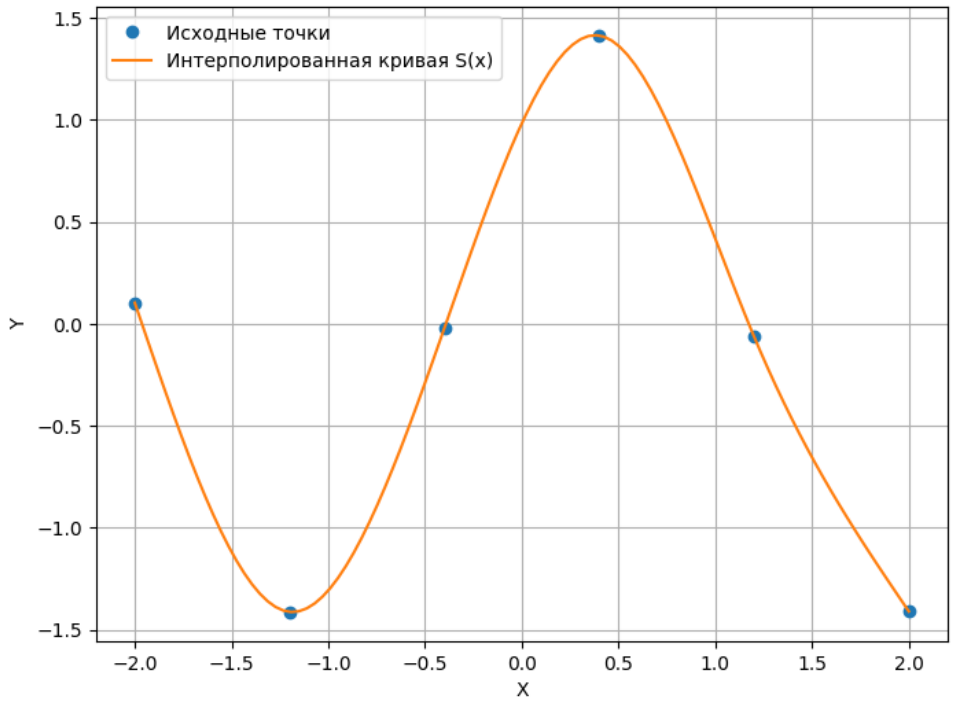


График полученного сплайна:



Среднеквадратическое отклонение:

$\sigma = 4.3140830754274083 * e^{-32}$

Вывод: был изучен метод интерполяции кубическим сплайном и применён на

практике для получения сплайна функции f(x). Полученное среднеквадратичное отклонение позволяет сказать, что сплайн обладает высокой степенью точности.