**МИНОБРНАУКИ РОССИИ**

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ**

**ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)**

**Кафедра МО ЭВМ**

**ОТЧЕТ**

**по лабораторной работе №9**

**по дисциплине «Вычислительная математика»**

**Тема: Интерполяция функций для неравностоящих узлов**

|  |  |  |
| --- | --- | --- |
| Студент гр. 3341 |  | Ягудин Д.Р. |
| Преподаватель |  | Пуеров Г. Ю. |

Санкт-Петербург

2025

**Цель работы.**

Целью лабораторной работы являются изучение методов интерполяции для случая неравноотстоящих узлов, написание программы, реализующей вычисление по интерполяционной формуле Ньютона и интерполяционную схему Эйткена.

**Выполнение работы.**

Напишем програму реализующую вычисление по интерполяционной формуле Ньютона и интерполяционной схеме Эйткена.

Формула Ньютона:

Сначала рекурсивно вычисляется таблица разделённых разностей. Первый столбец матрицы заполнен значениями интерполируемой функции в узлах интерполяции. Каждый последующий столбец получается посредством вычисления разделённой разности двух соседних коэффициентов в предыдущем столбце, пока в столбце не останется один элемент.

Далее результат рассчитывается как значение интерполяционного многочлена Ньютона в заданной точке:

* – разделённые разности k-ого порядка.

Общий вид поиска разделённых разностей k-ого порядка:

Схема Эйткена:

Заполнение таблицы Эйткена начинается с того, что в первом столбце записываются значения функции в узлах интерполяции, то есть.

Далее, для каждого следующего столбца вычисляются интерполяционные многочлены, степень которых равна номеру столбца. Для этого используется рекурсивная формула Эйткена:

* — приближённое значение функции в точке x, вычисленное по узлам от до .
* — точка, в которой нужно интерполировать значение.
* , — значение узлов из твоего списка.
* , — предыдущие значения приближений.

Процесс продолжается до тех пор, пока не останется только один элемент в последнем столюце. Этот последний коэффициент () и является искомым значением интерполяции в точке x.

**Тестирование.**

Пример для случайной генерации (8 узлов):

X: [-9.946843777474419 -8.194451561396365 -7.667304430621307 -5.53348235509149 0.3327565341133525 1.2415500548736773 6.570607990294594 8.790776312665859]

Y: [-66.38731893045089 -50.720720820249255 -27.347669048681865 -6.335819640906351 -0.8934714691159087 33.258536076852465 55.5868981196181 79.64820682233352]

Точка интерполяции x = 6.262997191884088.

Метод Ньютона: 56.19505399167666

Метод Эйткена: 56.19505399167476

Scipy : 56.1950539916748

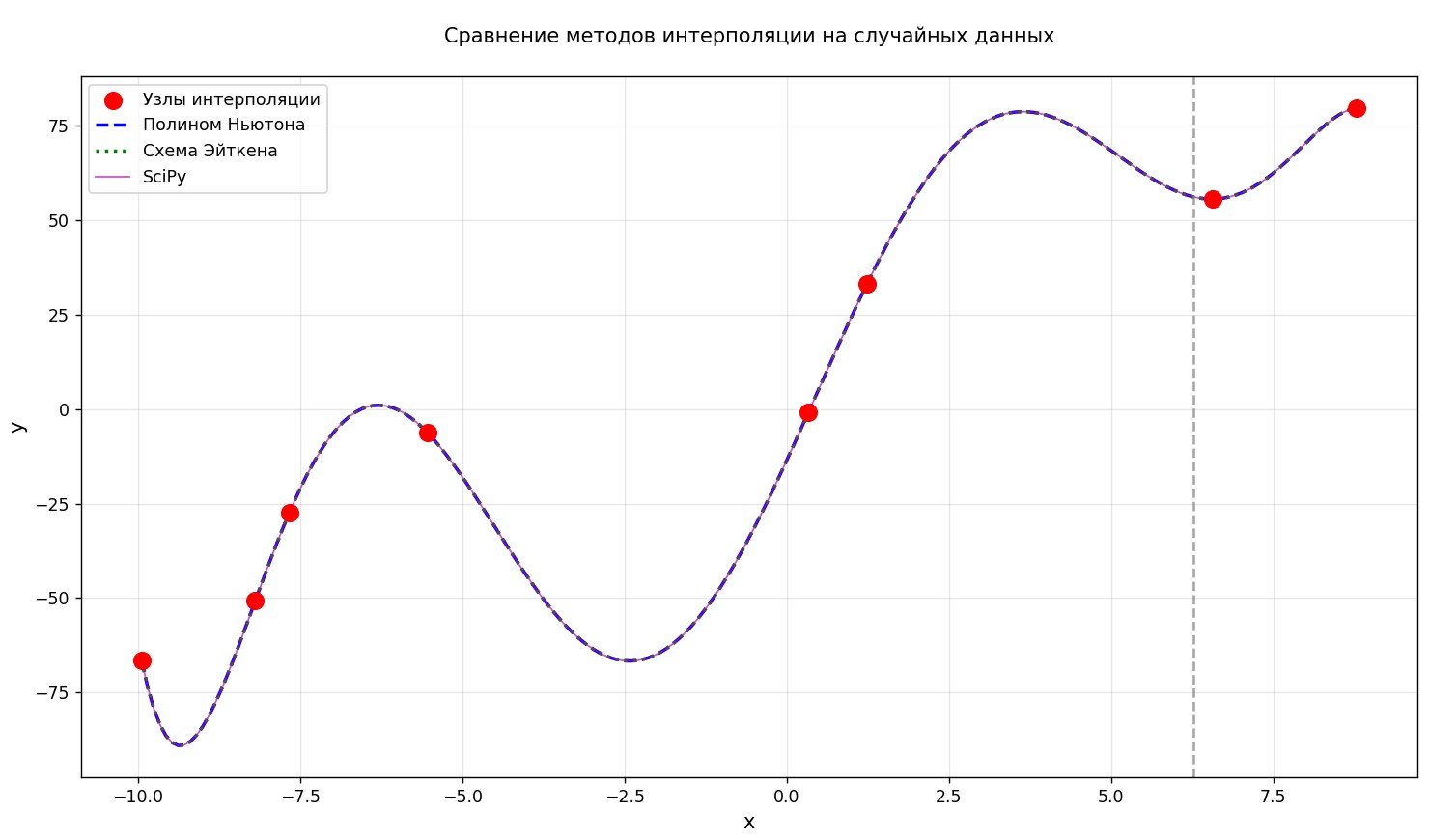


Рисунок № 1 — Сравнение методов интерполяции на случайных данных.

**Выводы.**

В результате выполнения данной лабораторной работы были изучены методы интерполяции для случая неравноотстоящих узлов, были написаны программы, реализующие вычисления по интерполяционной формуле Ньютона и по схеме Эйткена, а также визуализировано графически сравнение методов интерполяци на случайных данных.

**ПРИЛОЖЕНИЕ А**

**ИСХОДНЫЙ КОД ПРОГРАММЫ**

Название файла: main.py

import numpy as np

import matplotlib.pyplot as plt

import random

from scipy.interpolate import BarycentricInterpolator

def divided\_difference(x, y):

coefficients = y.copy()

for j in range(1, len(x)):

for i in range(len(x) - 1, j - 1, - 1):

coefficients[i] = (coefficients[i] - coefficients[i - 1]) / (x[i] - x[i - j])

return coefficients

def newton(coefficients, x\_nodes, x\_target):

result = coefficients[0]

term = 1

for i in range(1, len(coefficients)):

term \*= (x\_target - x\_nodes[i - 1])

result += coefficients[i] \* term

return result

def aitken(x, y, x\_target):

n = len(x)

P = [[0.0 for \_ in range(n)] for \_ in range(n)]

for i in range(n):

P[i][0] = y[i]

for j in range(1, n):

for i in range(n - j):

xi, xj = x[i], x[i + j]

Pi0 = P[i][j - 1]

Pi1 = P[i + 1][j - 1]

P[i][j] = ((x\_target - xj) \* Pi0 - (x\_target - xi) \* Pi1) / (xi - xj)

return P[0][n - 1]

n\_nodes = np.random.randint(5, 9)

x\_nodes = sorted([random.uniform(-10, 10) for \_ in range(n\_nodes)])

y\_values = sorted([random.uniform(-100, 100) for \_ in range(n\_nodes)])

x\_target = random.uniform(min(x\_nodes), max(x\_nodes))

coefficients = divided\_difference(x\_nodes, y\_values)

result = [

newton(coefficients, x\_nodes, x\_target),

aitken(x\_nodes, y\_values, x\_target),

BarycentricInterpolator(x\_nodes, y\_values)(x\_target)

]

print(f"Сгенерировано {n\_nodes} случайных узлов:"

f"\nX: {x\_nodes}"

f"\nY: {y\_values}"

f"\nТочка интерполяции x = {x\_target}")

print("\nРезультаты интерполяции:"

f"\nМетод Ньютона: {result[0]}"

f"\nМетод Эйткена: {result[1]}"

f"\n{'Scipy':<13}: {result[2]}")

plt.figure(figsize=(12, 7))

plt.scatter(x\_nodes, y\_values, c='red', s=100, label='Узлы интерполяции', zorder=5)

plot\_x = np.linspace(min(x\_nodes), max(x\_nodes), 200)

newton\_y = [newton(coefficients, x\_nodes, x) for x in plot\_x]

aitken\_y = [aitken(x\_nodes, y\_values, x) for x in plot\_x]

scipy\_y = BarycentricInterpolator(x\_nodes, y\_values)(plot\_x)

plt.plot(plot\_x, newton\_y, 'b--', label='Полином Ньютона', linewidth=2)

plt.plot(plot\_x, aitken\_y, 'g:', label='Схема Эйткена', linewidth=2)

plt.plot(plot\_x, scipy\_y, 'm-', label='SciPy', linewidth=1, alpha=0.7)

plt.axvline(x=x\_target, color='gray', linestyle='--', alpha=0.7)

plt.title('Сравнение методов интерполяции на случайных данных', pad=20)

plt.xlabel('x', fontsize=12)

plt.ylabel('y', fontsize=12)

plt.legend(fontsize=10)

plt.grid(True, alpha=0.3)

plt.tight\_layout()

plt.show()