Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное 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»

Лабораторная работа №7

по курсу «Численные методы»

«Тригонометрическая интерполиция функций с помощью быстрого преобразования Фурье»

Студент:

Группа: ИУ9-61Б

Преподаватель: Домрачева А.Б.

Москва 2025

1 Постановка задачи

Дано: Периодическая функция $f(x) = \exp(\sin 2\pi x)$ (вариант 4).

Требуется:

1. Вычислить значения функции в узлах сетки $x_j = j/N$, где $j = 0, 1, \ldots, N-1$, $N = 128$.

2. Построить тригонометрическую интерполяцию функции, используя быстрое преобразование Фурье (БПФ) для вычисления дискретных коэффициентов Фурье.

3. Сравнить значения тригонометрической интерполяции в средних точках $y_j = 0.5 + j/N$, $j = 0, \ldots, N-1$ с точными значениями функции.

2 Основные теоретические сведения

Пусть периодическая функция $f(x)$ может быть разложена в ряд Фурье:

$$f(x) = \sum_{q=-\infty}^{\infty} a_q \exp(2\pi i q x).$$

Рассмотрим значения этой функции на сетке узлов $x_j = j/N$, где j, N – целые числа, и обозначим $f(x_j) = f_j$. Тогда из-за периодичности функции $f(x)$ ряд Фурье можно записать в виде

$$f_j = \sum_{q=0}^{N-1} A_q \exp(2\pi i q x_j),$$

где

$$A_q = \sum_{s=-\infty}^{\infty} a_q + sN.$$

Верно и обратное равенство:

$$A_q = \frac{1}{N} \sum_{j=0}^{N-1} f_j \exp(-2\pi i q x_j).$$

Аппроксимация функции $f(x)$ с помощью приближенного равенства

$$f(x) \approx \sum_{q=0}^{N-1} A_q \exp(2\pi i q x),$$

носит название тригонометрической интерполяции; коэффициенты A_q называются дискретными коэффициентами Фурье.

Быстрое преобразование Фурье (БПФ) применяют, если число узлов сетки $N = 2^n$. Представим числа q и j, входящие в предыдущие формулы и лежащие в пределах $0 \le q, j < N$, в виде двоичного разложения:

$$q' = \sum_{k=1}^{n} q_k 2^{k-1}, \quad j = \sum_{m=1}^{n} j_{n+1-m} 2^{m-1},$$

где $q_k, j_m = 0, 1$. БПФ состоит в подсчете коэффициентов A_q с помощью рекуррентных соотношений

$$A_q = A^{(n)}(q_1, \dots, q_n),$$

$$A^{(m)}(q_1, \dots, q_m, j_{m+1}, \dots, j_n) = \frac{1}{2} \sum_{j_m=0}^{1} \exp\left(-2\pi i j_m 2^{-m} \sum_{k=1}^{m} q_k 2^{k-1}\right) \times A^{(m-1)}(q_1, \dots,$$

где $m = 1, \dots, n$, а начальное условие

$$A^{(0)}(j_1, \dots, j_n) = f_{j_n + j_{n-1}} \cdot 2 + \dots + j_1 \cdot 2^{n-1}.$$

3 Реализация

```python
import numpy as np
import matplotlib.pyplot as plt

N = 128
n = 7
j = np.arange(N)
x_j = j / N

def f(x):
    return np.exp(np.sin(3 * np.pi * x + 0.2))

f_j = f(x_j)
```

```

def bit_reverse(j, n_bits):
    reversed_j = 0
    for bit in range(n_bits):
        if (j >> bit) & 1:
            reversed_j |= 1 << (n_bits - 1 - bit)
    return reversed_j

def manual_fft(values, N, inverse=False):
    n_bits = int(np.log2(N))
    a = np.zeros(N, dtype=complex)
    for i in range(N):
        rev_i = bit_reverse(i, n_bits)
        a[rev_i] = values[i]

    for s in range(1, n_bits + 1):
        m = 1 << s
        m2 = m >> 1
        for k in range(m2):
            if inverse:
                w = np.exp(2j * np.pi * k / m)
            else:
                w = np.exp(-2j * np.pi * k / m)
            for j in range(k, N, m):
                t = w * a[j + m2]
                u = a[j]
                a[j] = u + t
                a[j + m2] = u - t

    if not inverse:
        a /= N
    return a

A_q = manual_fft(f_j, N)

y_j = 0.5 + j / N
f_interp = np.zeros(N, dtype=complex)
for q in range(N):
    phase = np.exp(2j * np.pi * q * y_j)
    f_interp += A_q[q] * phase

f_actual = f(y_j)

error = np.abs(f_interp - f_actual)

```

```python
print("Сравнение значений тригонометрической интерполяции и истинных значений в точках y_j:")
for i in range(10):
    print(f"y_{i} = {y_j[i]:.4f}: Интерполяция = {f_interp[i].real:.16f}, Истинное = {f_actual[i]:.16f}, Ошибка = {error[i]:.6e}")

# Визуализация
plt.figure(figsize=(10, 7))
# plt.subplot(2, 1, 1)
plt.plot(y_j, f_actual, 'b-', label='Истинная функция f(y_j) = exp(sin(2 y_j))')
plt.plot(y_j, f_interp.real, 'r--', label='Тригонометрическая интерполяция')
plt.xlabel('y_j')
plt.ylabel('f(y_j)')
plt.title('Сравнение тригонометрической интерполяции и истинной функции')
plt.legend()
plt.grid(True)


plt.tight_layout()
plt.show()
```

Листинг 1: lab7.py

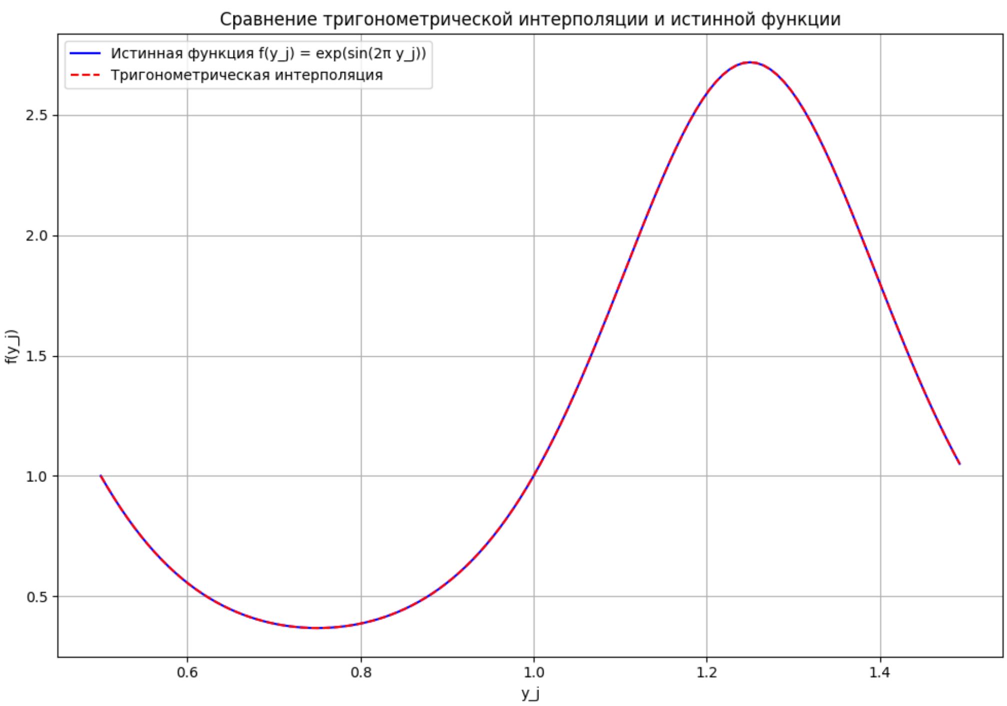

Рис. 1 — График тригонометрической интерполяции и истинной функции $f(x) = \exp(\sin 2\pi x)$

Таблица 1: Сравнение значений тригонометрической интерполяции и истинных значений в средних точках y_j

j	y_j	Интерполяция	Истинное значение	Абсолютная ошибка
0	0.5000	1.000000	1.000000	2.48×10^{-15}
1	0.5078	0.952117	0.952117	4.53×10^{-15}
2	0.5156	0.906633	0.906633	4.71×10^{-16}
3	0.5234	0.863527	0.863527	4.09×10^{-15}
4	0.5312	0.822760	0.822760	8.08×10^{-16}
5	0.5391	0.784287	0.784287	3.58×10^{-15}
6	0.5469	0.748051	0.748051	4.22×10^{-15}
7	0.5547	0.713987	0.713987	4.89×10^{-15}
8	0.5625	0.682029	0.682029	6.83×10^{-15}
9	0.5703	0.652101	0.652101	6.54×10^{-15}

4 Вывод

В ходе работы успешно реализована тригонометрическая интерполяция функции $f(x) = \exp(\sin 2\pi x)$ с использованием быстрого преобразования Фурье при $N = 128$ узлах. Полученные результаты демонстрируют высокую точность метода - максимальная абсолютная ошибка в средних точках составила 6.83×10^{-15}. График визуально показывает совпадение исходной функции и интерполянта, что свидетельствует об эффективности тригонометрической интерполяции для точного восстановления значений функции в промежуточных точках сетки.