3. Реализация быстрого преобразования Фурье

Андрей Валиков

1 Математическая модель сигнала

Константы имеют следующие значения:

$$f_1 = 1000$$

$$A_{f_1} = 0.5$$

$$\varphi_1 = 120$$

$$f_2 = 3000$$

$$A_{f_2} = 1$$

$$\varphi_2 = 180$$

$$f_3 = 6000$$

$$A_{f_3} = 5$$

$$\varphi_3 = 90$$

По ним вычисляется значение исходного сигнала:

$$x(t) = \sum_{k=1}^{3} A_{f_k} \sin(2\pi f_k t + \varphi_k)$$

```
def Func(t):
    A   = [     1,       2,      10]
    f   = [10000,   20000,  40000]
    Phi = [.5236,  2.0944,  .7854]
```

```
y = 0
for k in range(3):
  y += A[k] * np.sin(t * np.pi * f[k] + Phi[k])
return y
```

Рис. 1:

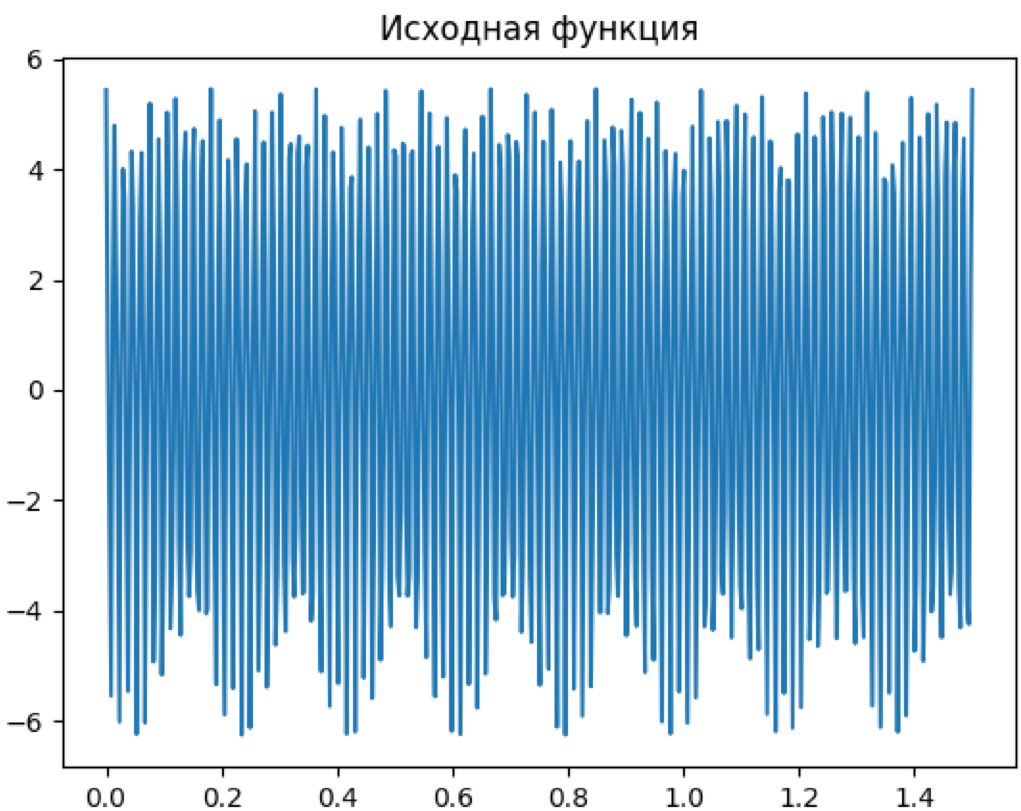

Рис. 2:

2 Вычисление дискретного преобразованния Фурье

Формула:

$$X(t) = \sum_{n=0}^{N-1} x(n) e^{-2\pi nki/N}$$

```
import numpy as np

def dft_exp(k, n, N):
  return np.exp((-1j * 2 * np.pi * n * k) / N)
```

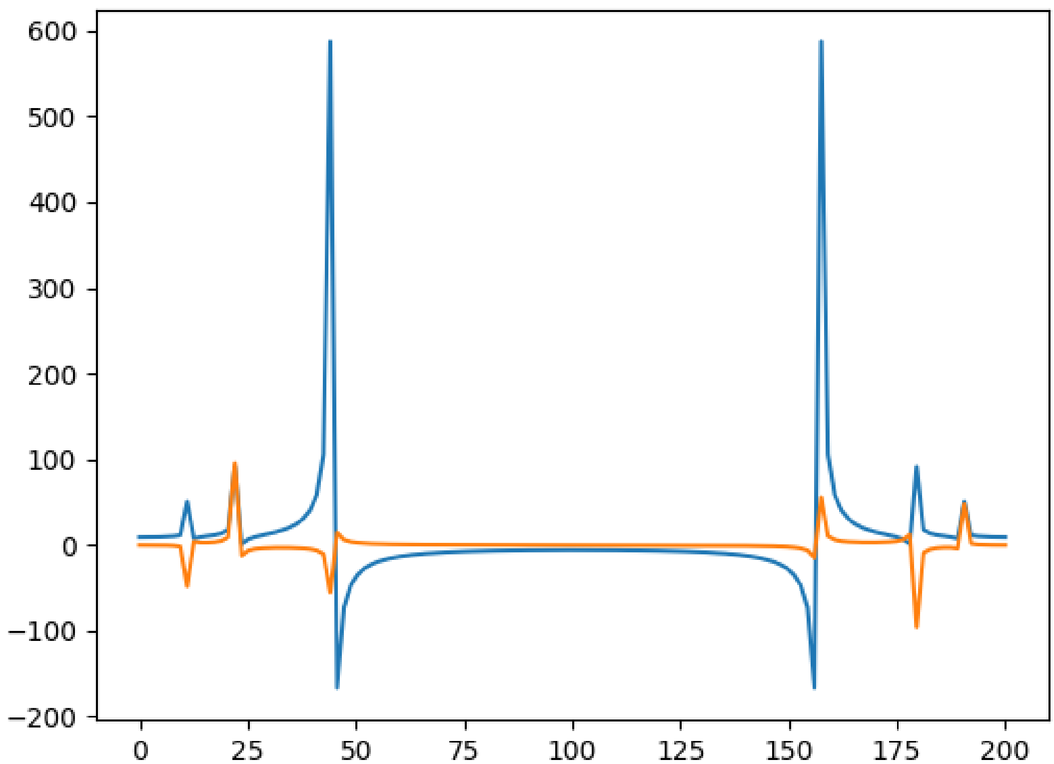

Рис. 3:

3 Вычисление быстрого преобразованния Фурье

```
import numpy as np
from dft_part import dft_exp


def FFT(X):

    N = len(X)
    V = np.log2(N)
```

```
if V % 2 != 0:
  V = np.ceil(V)
  AddN = 2 ** V - N
  X = np.concatenate([X, np.zeros(int(AddN))])
  N = len(X)

y = []

if N == 2:
  y.append(X[0] + X[1])
  y.append(X[0] - X[1])
  return y

else:
  E  = np.zeros(N // 2)
  O  = np.zeros(N // 2)

  Ek = 0
  Ok = 0

  for k in range(N):

    if k % 2 == 0:
      E[Ek] = X[k]
      Ek += 1

    else:
      O[Ok] = X[k]
      Ok += 1

  X1 = FFT(E)
  X2 = FFT(O)

  for k in range(N):
    if k < N / 2:
      y.append(X1[k] + dft_exp(k, 1, N) * X2[k])

    else:
```

```
        y.append(X1[k - N // 2] - dft_exp(k - N // 2, 1, N) * X2[k - N //
    return y
```

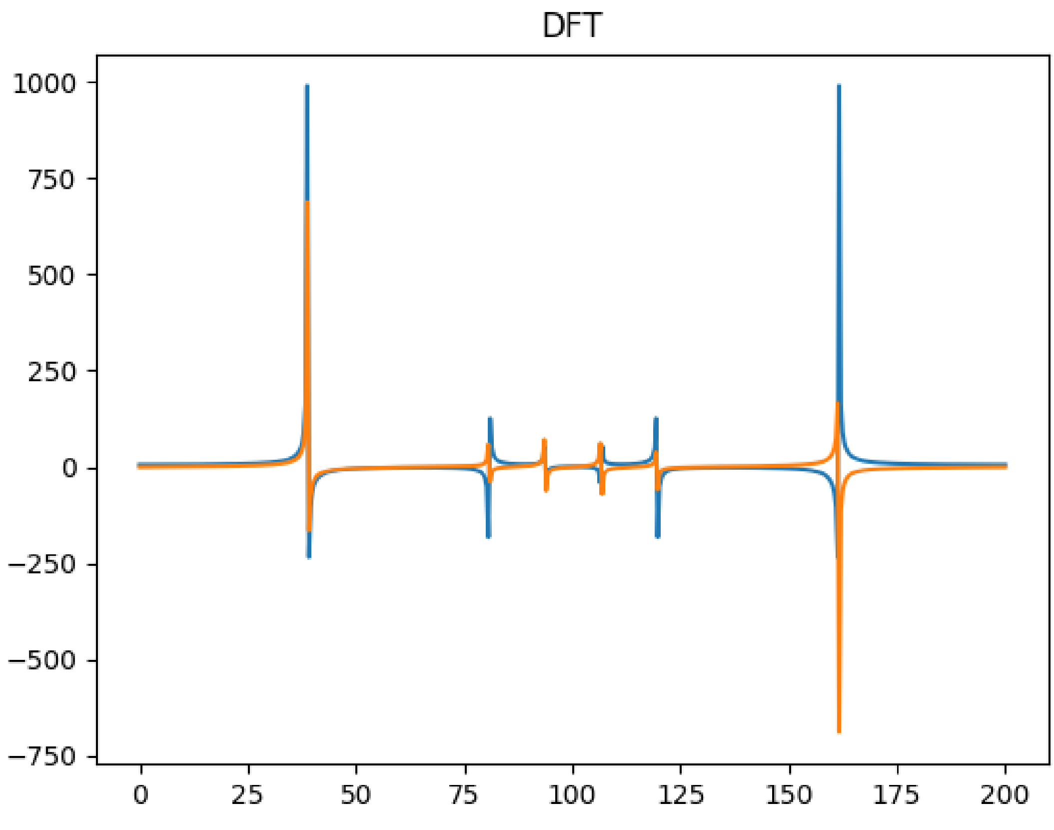

Рис. 4:

4 Функция ошибки

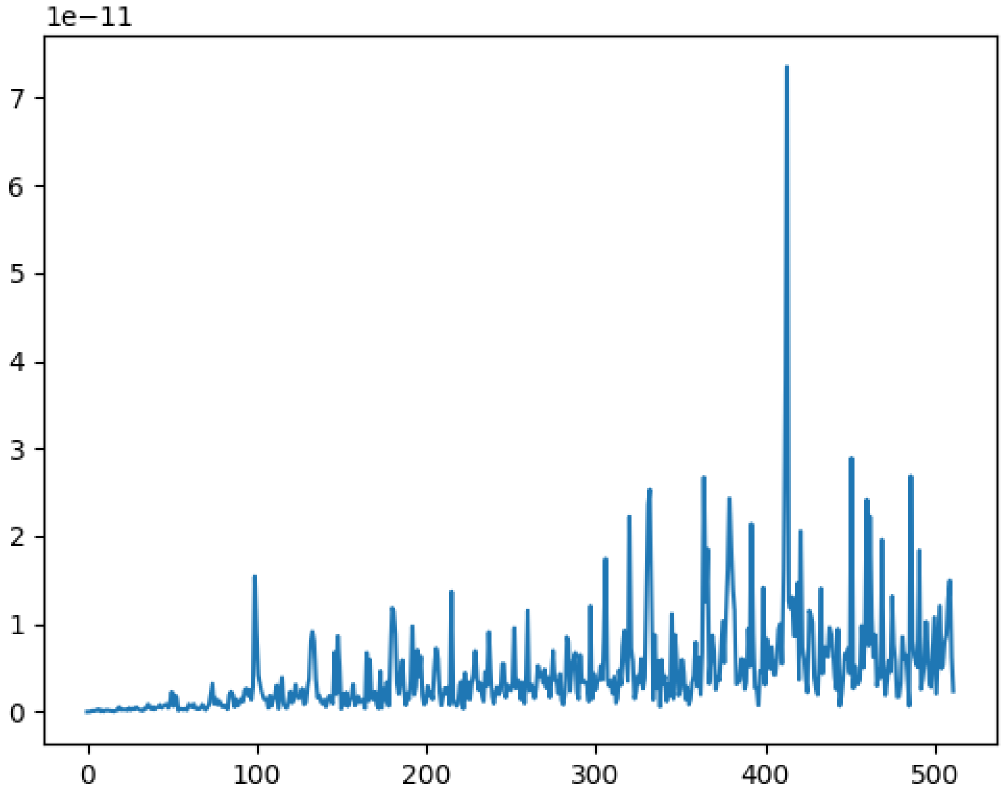

Рис. 5:

5 Вывод

Имеется исходная полигармоническая функция. В работе сравнивается скорость и точность дискретного и быстрого преобразований Фурье. По формуле дискретного преобразования Фурье (исходное значение умноженное на поворотный множитель). В быстром преобразовании Фурье значительно сокращается количество вычислительно сложной операции умножения. Функция ошибки показывает максимальное отклонение около 10^{-11}, что говорит о высокой точности. Скорость также намного выше.