МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра ТЭВН

Лабораторная работа №7

“СОЗДАНИЕ ПОЛЬЗОВАТЕЛЬСКИХ ФУНКЦИЙ ДЛЯ ОБРАБОТКИ ДИСКРЕТНЫХ СИГНАЛОВ НА ЯЗЫКЕ PYTHON”

Вариант № 22

Факультет: ФЭН

Группа: Эн1-21

Студент: Кихаев Е.А.

Преподаватель: Лаптев О.И.

Новосибирск 2022

Цель: знакомство с основами программирования пользовательских функций на языке Python. Получение навыков решения прикладных задач по обработке дискретных сигналов.

Задание:

1. В соответствии с вариантом написать программу для обработки дискретных сигналов (варианты заданы в Приложении 1). Программа должна выполнять следующее:
   1.1. Создавать тестовый дискретный сигнал напряжения.
   1.2. Вызывать функцию по расчету интеграла дискретного тестового сигнала.
   1.3. Вызывать функцию по расчету производной дискретного тестового сигнала.
   1.4. Построить графики: исходного сигнала, производной сигнала.
   1.5. Вывести результат расчета интеграла сигнала (число)
2. Создать отчет по проделанной работе. Отчет должен содержать:
   2.1. Титульный лист, цель работы, задание. Все страницы, кроме первой, должны быть пронумерованы в верхнем правом углу. В верхнем колонтитуле разместить с выравниванием влево фамилию и инициалы студента и надпись “Лабораторная работа № 7”.
   2.2. Постановку задачи.
   2.3. Математическую модель.
   2.4. Блок-схемы написанных программ.
   2.5. Листинг написанных программ.
   2.6. Тестирование программ. Тестирование выполнить на **линейной** и **квадратичной** функциях. Привести результаты работы для тестовых функций.
   2.7. Текстовые пояснения, объясняющие ход выполнения работы по всем этапам.
   2.8. Результаты обработки сигналов (графики).
   2.9. Присвоить файлу имя “Лабораторная работа №7”. Сохранить файл в соответствующей папке Google Диска. Оповестить преподавателя через почту о созданном отчете и прикрепить отчет.
   2.10. Получить замечания по документу, в соответствии с замечаниями внести изменения в документ. Оповестить преподавателя ответом на его письмо.

Анализ задачи:

Что дано?

Функция U(t)=3·$sin(t$·314) + 0.5·$sin(t$·314·4), диапазон значений t, с (0…0.1), частота дискретизации, Гц 5000. Нужно создать тестовый сигнал напряжения, вызвать функцию по расчёту интеграла и производной дискретного тестового сигнала и построить графики исходного и производного сигнала.

Как решать задачу?

Для решения задачи, напишем две функции, которые находят производную и интеграл от данной функции. Проверим программу при помощи тестирования на линейной и квадратичной функциях. После с помощью уже написанной программы построим график основной и её производной функции при помощи библиотеки matplotlib и выведем значения определённого интеграла.

Что будет результатом?

Результатом данной задачи будет являться график функции, график производной функции и расчёт определённого интеграла.

Математическая модель.

Для нахождения производной в каждой точке дискретного сигнала используется формула:

$$dy[k] = \frac{y[k+1]-y[k]}{x[k+1]-x[k]} \quad (1)$$

Чтобы найти значение определённого нужно найти сумму площадей прямоугольников:

$$S[k] = y[k] * (x[k+1] - x[k]) \quad (2)$$

$$\sum_{k=0}^{n-1} S[k] \quad (3)$$

Алгоритм функций для вычисления производной и интеграла:

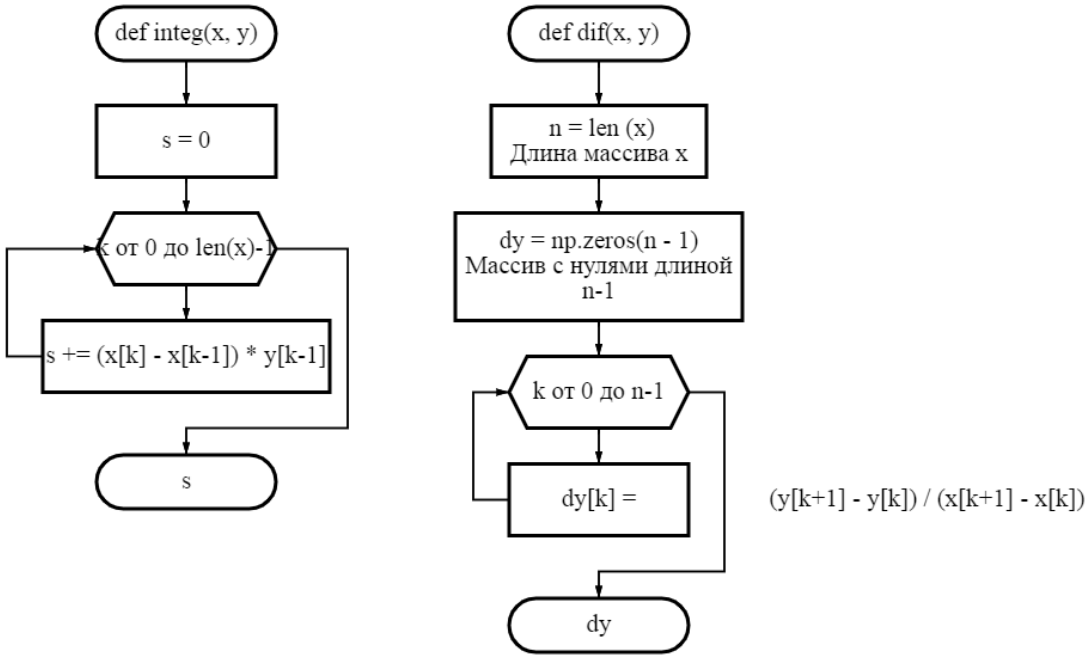

Рисунок №1-Блок-схема вычисление интеграла и производной

Алгоритм для первого тестового сигнала:

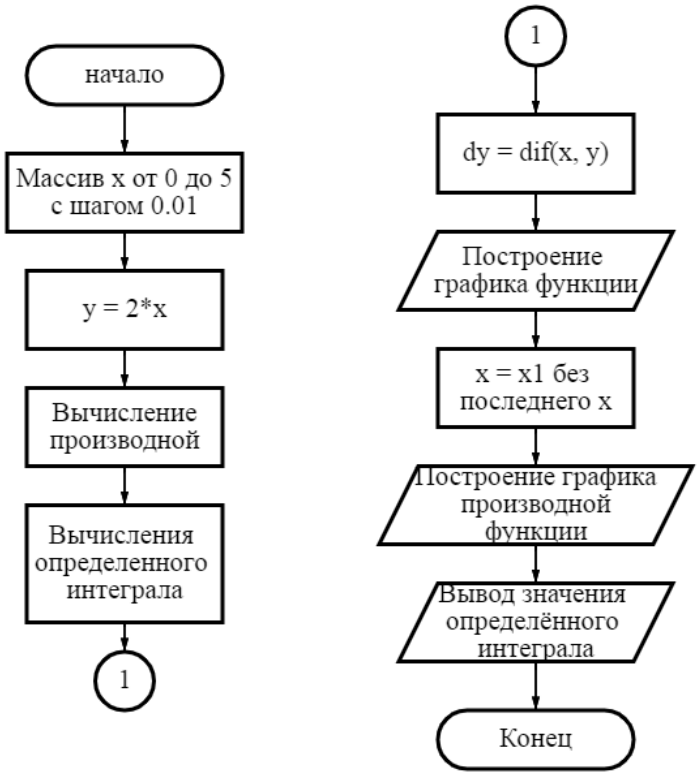

Рисунок №2 - Блок-схема алгоритма 1-го тестового сигнала

Программа:

```
import numpy as np
import matplotlib.pyplot as plt
x = np.arange(0, 5.01, 0.01) # Создание массива значений x
y = 2 * x # Вычисление значений линейной функции
# Функция для вычисления производной
def dif(x, y):
  n = len (x) # Число точек массива x
   dy = np.zeros(n - 1) # Создание массива длиной n - 1, заполненного нулями
  # Вычисление производной для каждой точки дискретного сигнала
  for k in range(n - 1):
    dy[k] = (y[k+1] - y[k]) / (x[k+1] - x[k])
  return dy
# Функция для вычисления определенного интеграла
def integ(x, y):
  s = 0
     # Вычисление определенного интеграла как суммы площадей прямоугольников
  for k in range(len (x) - 1):
    s += (x[k + 1] - x[k]) * y[k]
  return s
dy = dif(x, y)
plt.figure(1)
plt.plot(x, y)
plt.grid()
plt.title('График функции')
plt.figure(2)
x1 = x[0: len(x)-1]
plt.plot(x1, dy)
plt.grid()
plt.title('График производной функции')
print('Интеграл = {0:.2f}'.format(integ(x, y)))
```

Результат программы:

Интеграл = 24.95

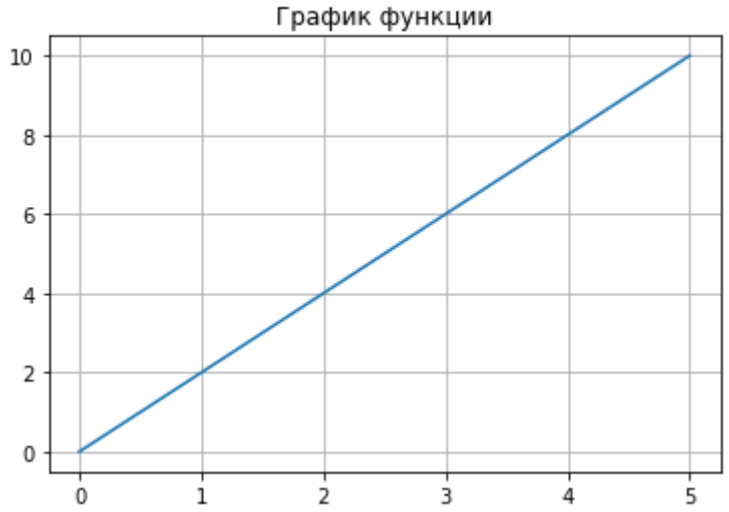

Рисунок №3 - График линейной функции

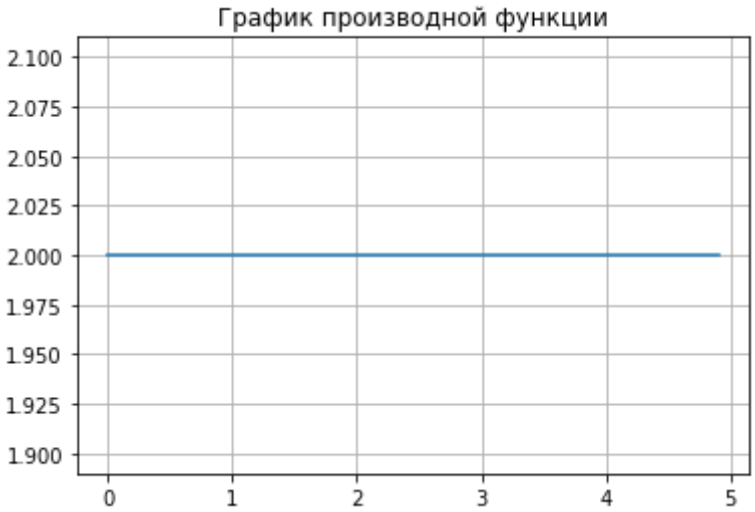

Рисунок №4-График производной от линейной функции

Тестирование:

График линейной функции y = 2 * x выглядит как прямая проходящая через (0;0) под некоторым углом, так же как и на нашем графике, а график производной этой функции выглядит как прямая параллельная оси x, проходящая через точку y = 2.

Интеграл от данной функции можно вычислить по формуле:

$\int_0^5 2 * x = 2 * \frac{x^2}{2}|_0^5 = 2 * \frac{25}{2} - 0 = 25$, так как значения интеграла отличаются не сильно результат программы можно считать верным.

Алгоритм второго тестового сигнала:

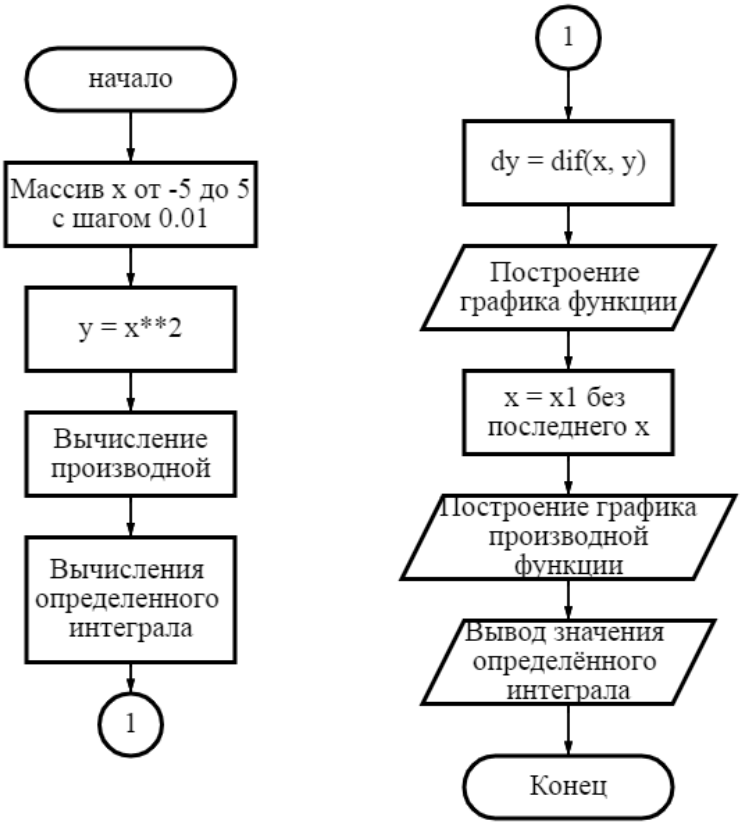

Рисунок №5-Блок-схема алгоритма 2-го тестового сигнала

Программа:

```
import numpy as np
import matplotlib.pyplot as plt
x = np.arange(-5, 5.01, 0.01) # Создание массива значений x
y = x**2 # Вычисление значений линейной функции
# Функция для вычисления производной
def dif(x, y):
  n = len (x) # Число точек массива x
   dy = np.zeros(n - 1) # Создание массива длиной n - 1, заполненного
нулями
  # Вычисление производной для каждой точки дискретного сигнала
  for k in range(n - 1):
    dy[k] = (y[k+1] - y[k]) / (x[k+1] - x[k])
  return dy
# Функция для вычисления определенного интеграла
def integ(x, y):
  s = 0
```

```
    # Вычисление определенного интеграла как суммы площадей прямоугольников
  for k in range(len (x) - 1):
    s += (x[k + 1] - x[k]) * y[k]
  return s
dy = dif(x, y)
plt.figure(1)
plt.plot(x, y)
plt.grid()
plt.title('График функции')
plt.figure(2)
x1 = x[0: len(x)-1]
plt.plot(x1, dy)
plt.grid()
plt.title('График производной функции')
print('Интеграл = {0:.2f}'.format(integ(x, y)))
```

Результат программы:

Интеграл = 83.33

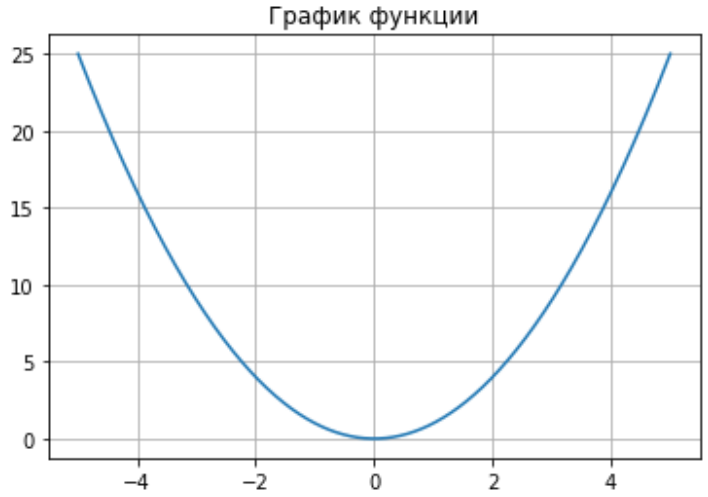

Рисунок № 6 - График квадратичной функции

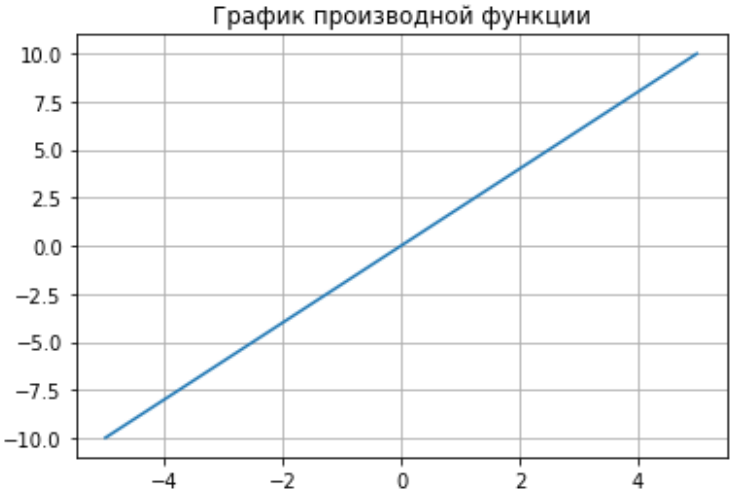

Рисунок №7 - График производной квадратичной функции

Тестирование:

Графиком квадратичной функции является парабола, так как и вывела программа, а графиком производной от нашей параболы будет являться прямая y = 2* x проходящая через точку (0;0).

Чтобы найти интеграл воспользуемся формулой:

$\int_{-5}^{5} x^2 = \frac{x^3}{3}|_{-5}^{5} = \frac{125}{3} - \frac{-125}{3} \approx 83,33$, так как значения вычислений и результат программы идентичен результат программы можно считать верным.

Алгоритм основного сигнала:

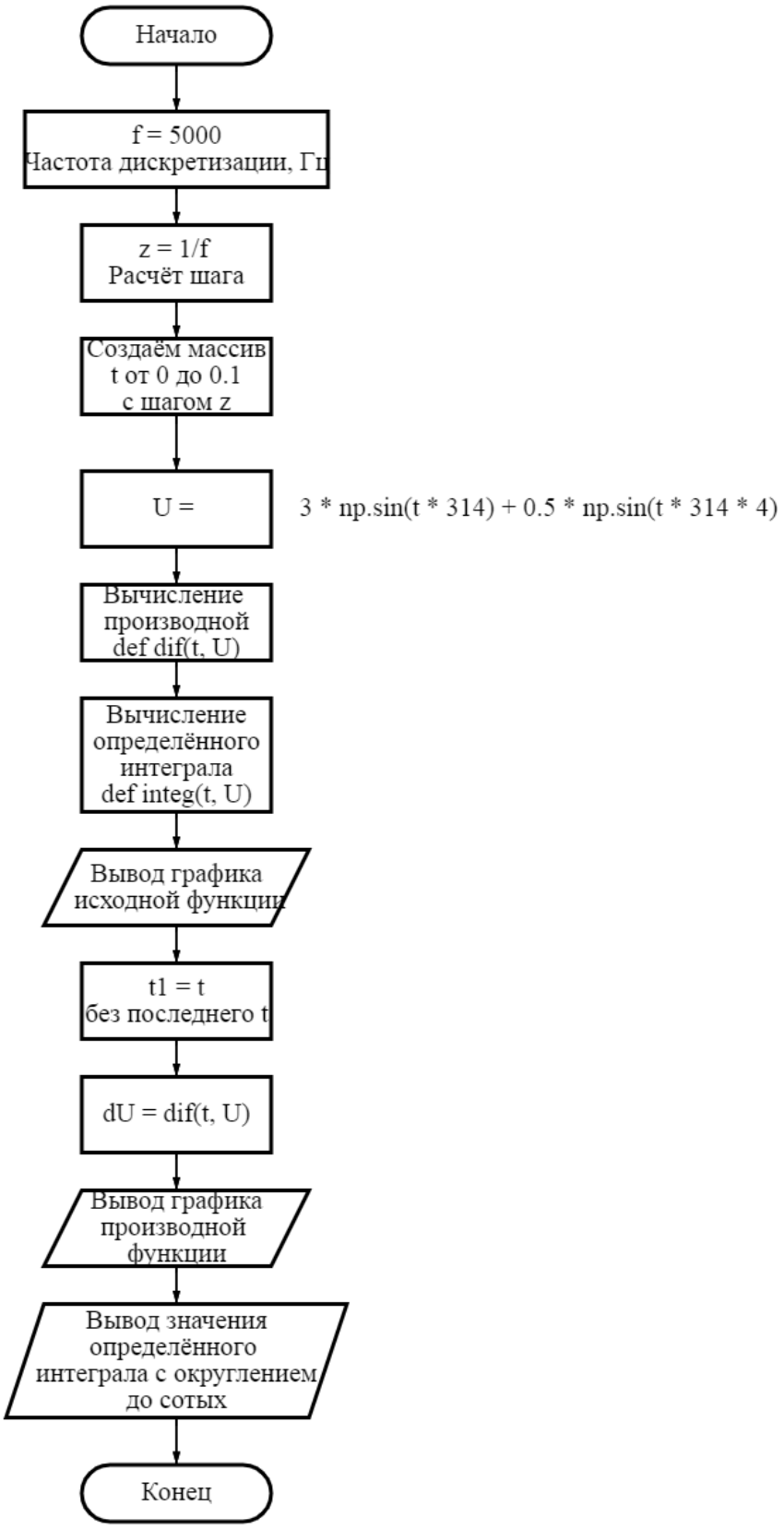

Рисунок №8 - Блок-схема алгоритма основного сигнала

Программа:

```
import numpy as np
import matplotlib.pyplot as plt
f = 5000 # Частота дискретизации, Гц
z = 1/f # Расчёт шага дискретизации
t = np.arange(0, 0.1 + z, z) # Создание массива значений t
# Вычисление массива значений U
U = 3 * np.sin(t * 314) + 0.5 * np.sin(t * 314 * 4)
# Функция для вычисления производной
def dif(t, U):
  n = len (t) # Длина массива x
  dU = np.zeros(n - 1)
  # Вычисление производной для каждой точки дискретного сигнала
  for k in range(n - 1):
    dU[k] = (U[k+1] - U[k]) / (t[k+1] - t[k])
  return dU
# Функция для вычисления определенного интеграла
def integ(t, U):
  s = 0
  # Вычисление определенного интеграла как суммы площадей
прямоугольников
  for k in range(len (t) - 1):
    s += (t[k] - t[k-1]) * U[k-1]
  return s
# График исходной функции
plt.figure(1)
plt.title('График функции')
plt.plot(t, U)
plt.grid()
plt.xlabel('t, c')
plt.ylabel('U, B')
t1 = t[0: len(t)-1]
dU = dif(t, U) # Вычисление производной
# График производной функции
plt.figure(2)
plt.title('График производной функции')
plt.plot(t1, dU)
plt.grid()
plt.xlabel('t, c')
plt.ylabel('U, B')
print('Интеграл функции = {0:.2f}'.format(integ(t, U)))
```

Результат программы:

Интеграл функции = -0.02

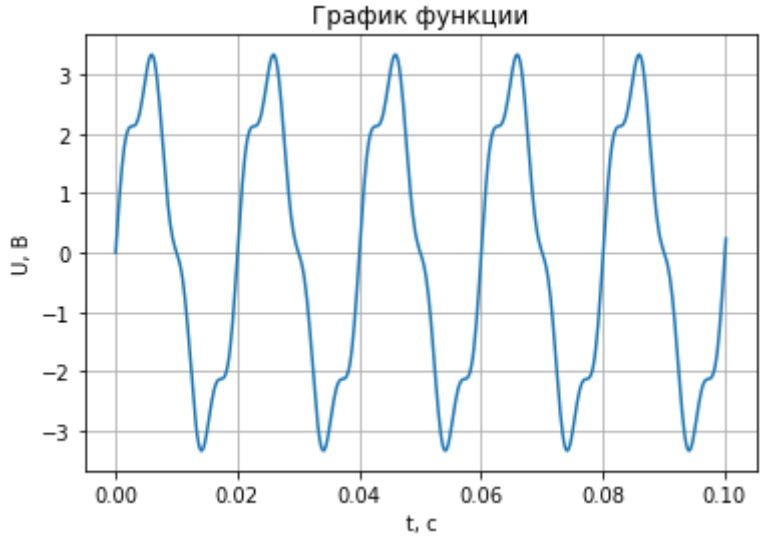

Рисунок №9 - График основной функции

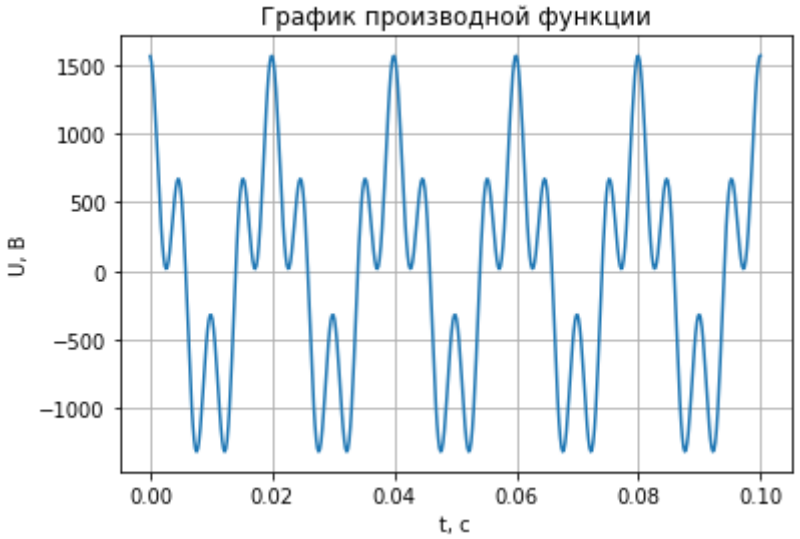

График №10 - График производной основной функции