Лабораторная работа 1. Введение в анализ данных с Python

Имя: Борисов Максим

Номер в ИСУ: 225169

Группа: R41341C

Задание

Есть набор данных, который содержит данные о напряжении, токе и времени измерения. Период измерения 0,001 с. Период тестового сигнала 0,1 с. Количество периодов тестового сигнала - 1000. Тип двигателя - двигатель постоянного тока.

Необходимо

1. Импортировать библиотеки
2. Загрузить и подготовить данные
3. Отобразить графики тока и напряжения
4. Рассчитать значения параметров L и R
5. Рассчитать средние значения и стандартное отклонение параметров L и R

1. Импортирование библиотек

```
1  from typing import Tuple
2
3  import numpy as np
4  import matplotlib.pyplot as plt
```

Библиотека `typing` необходима для указания сигнатур функций для удобства работы в IDE.

2. Загрузка и подготовка данных

```
17 ## Given
18 T = 0.1  # Period
19 Td = 1e-3  # sampling period
20
21 img_dir = '../img/'
22 data_path = '../data/testLab1Var6.csv'
23 columns = ['time', 'current', 'voltage']
24
25 data = np.genfromtxt(data_path, names=columns, delimiter=',')
```

Значение напряжения и времени 11-го элемента массива: t = 0.011; U = 1

3. Графики тока и напряжения

```
42 ## Display only N periods
43 N = 2
44 t_final = N * T
45 time = data['time'][data['time'] < t_final]
46 current = data['current'][data['time'] < t_final]
47 voltage = data['voltage'][data['time'] < t_final]
48
49 fig, (cur_ax, vol_ax) = plt.subplots(2, 1, sharex=True)
50 cur_ax.plot(time, current)
51 cur_ax.set(xlabel='Время t, с',
52            ylabel='Сила тока, А',
53            title='Сила тока и напряжение от времени')
54 cur_ax.grid()
55
56 vol_ax.plot(time, voltage)
57 vol_ax.set(xlabel='Время t, с',
58            ylabel='Напряжение, В')
59 vol_ax.grid()
60 fig.savefig(img_dir + f'current-voltage-{N}T.png')
```

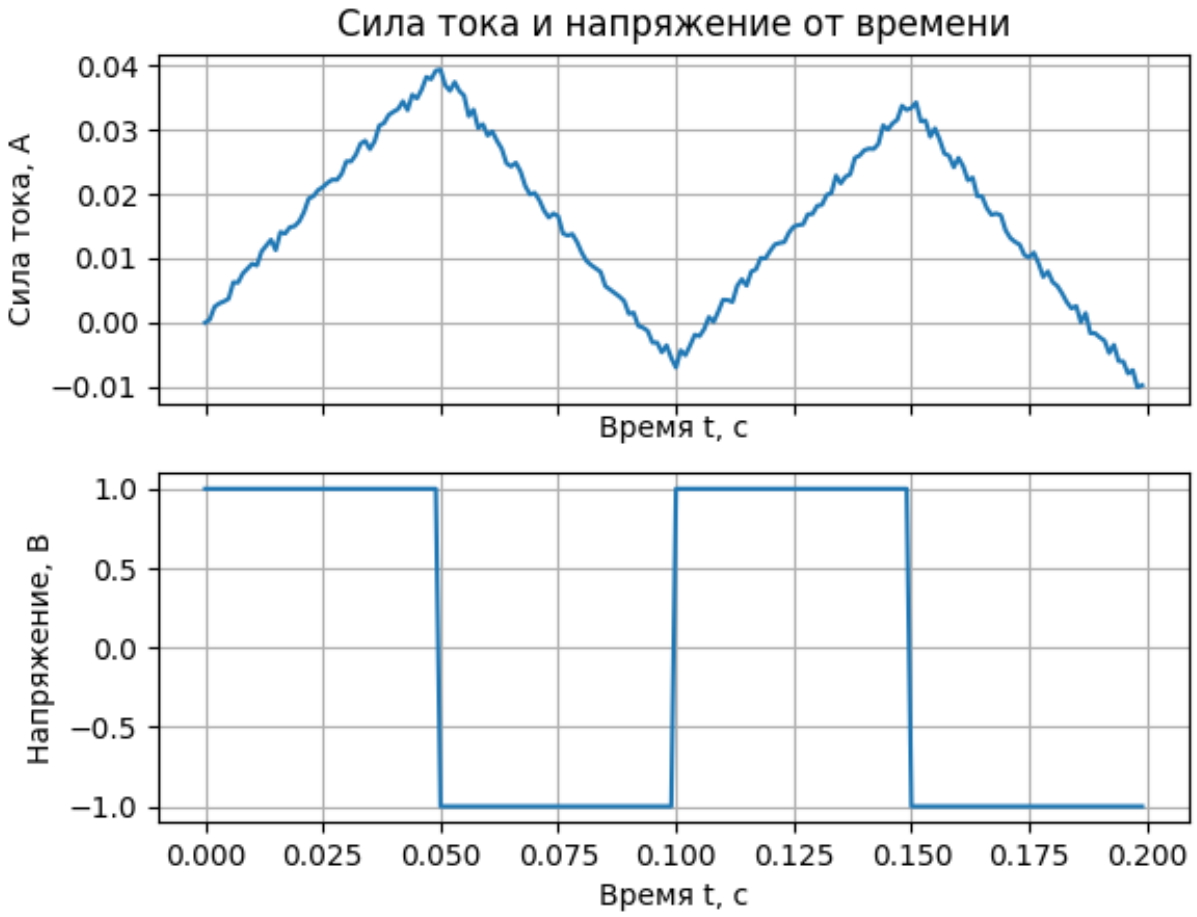

Рис. 1: График тока и напряжения от времени

4. Рассчёт параметров L и R

Функция расчёта параметров L и R. Проверки коэффициентов k (например на равенство нулю) не требуется, так как для numpy это вызовет лишь warning, а не exception и значения в дальнейшем легко отфильтровать.

```
 7  def estimate_params(X: np.ndarray, Y: np.ndarray, Td: float) ->
    ↪  Tuple[float, float, float, np.ndarray]:
 8      k = np.linalg.inv(X.T @ X) @ X.T @ Y
 9      R = (1 - k[1]) / k[0]
10      Te = -Td / np.log(k[1])
11      L = Te * R
12
13      return (R, L, Te, k)
```

Расчёт параметров на всей выборке и отображение графиков.

```
62  ## Estimate parameters
63  X = np.c_[data['voltage'][:-1], data['current'][:-1]]
64  Y = data['current'][1:]
65  R, L, Te, k = estimate_params(X, Y, Td)
66  print(f'Estimated R = {R} Ohm\n'
67        f'Estimated L = {L} Hn')
68
69  ## Compare model and given data
70  current_est = (X @ k)[:time.size]
71
72  fig, ax = plt.subplots()
73  ax.plot(time, current, label='Исходные данные')
74  ax.plot(time, current_est, label='Аппроксимированные данные')
75  ax.set(xlabel='Время, с',
76         ylabel='Сила тока, А',
77         title='Сравнение модели и данных')
78  ax.grid()
79  ax.legend()
80  fig.savefig(img_dir + 'current_estimation_comparison.png')
```

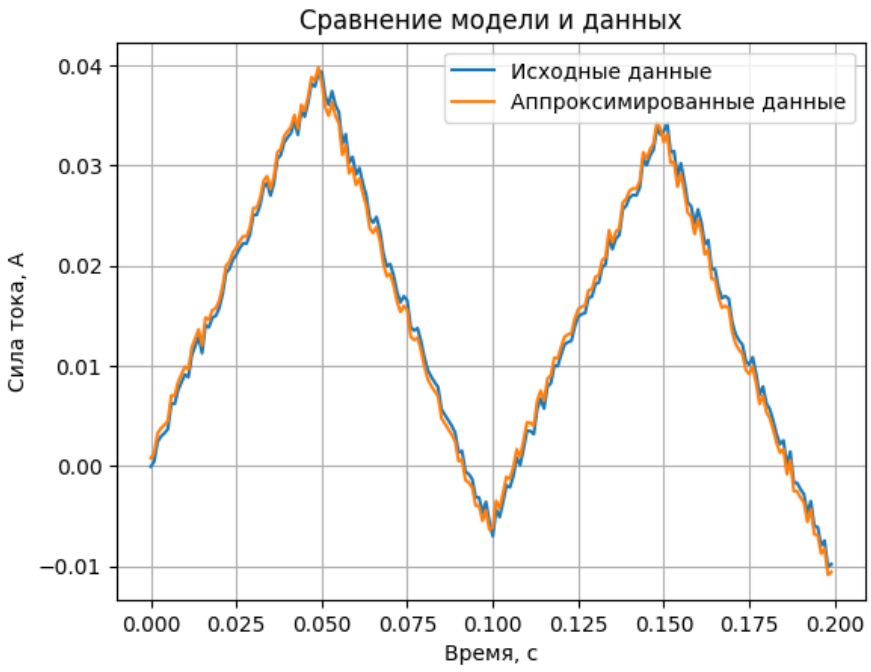

Рис. 2: Сравнение реальных данных и модели

Значения параметров: $R = 8$ Ом, $L = 1.17$ Гн

5. Рассчёт среднего значения и стандартного отклонения параметров L и R

```
82  ## Mean and std of parameters
83  R_est = np.array([])
84  L_est = np.array([])
85  n = 1000
86  for i in range(n):
87      indices = (i * T <= data['time']) & (data['time'] <= (i + 1)
        ↪  * T)
88      curr = data['current'][indices]
89      volt = data['voltage'][indices]
90
91      X = np.c_[volt[:-1], curr[:-1]]
92      Y = curr[1:]
93
94      R, L, Te, k = estimate_params(X, Y, Td)
95
96      R_est = np.append(R_est, R)
97      L_est = np.append(L_est, L)
98
99  print(
100         f'Mean R: {np.mean(R_est[np.isfinite(R_est)])}, Ohm\n'
101         f'Standard deviation R:
            ↪  {np.std(R_est[np.isfinite(R_est)])}\n'
102         f'Mean L: {np.mean(L_est[np.isfinite(L_est)])}, Hn\n'
103         f'Standard deviation L:
            ↪  {np.std(L_est[np.isfinite(L_est)])}\n'
104         )
```

	R, Ом	L, Гн
Медиана	8.04	1.17
СКО	1.82	0.02

Полный код

```python
from typing import Tuple

import numpy as np
import matplotlib.pyplot as plt


def estimate_params(X: np.ndarray, Y: np.ndarray, Td: float) -> Tuple[float, float, float, np.ndarray]:
    k = np.linalg.inv(X.T @ X) @ X.T @ Y
    R = (1 - k[1]) / k[0]
    Te = -Td / np.log(k[1])
    L = Te * R

    return (R, L, Te, k)


if __name__ == '__main__':
    ## Given
    T = 0.1  # Period
    Td = 1e-3  # sampling period

    img_dir = '../img/'
    data_path = '../data/testLab1Var6.csv'
    columns = ['time', 'current', 'voltage']

    data = np.genfromtxt(data_path, names=columns, delimiter=',')

    ## Display all data
    fig, (cur_ax, vol_ax) = plt.subplots(2, 1, sharex=True)
    cur_ax.plot(data['time'], data['current'])
    cur_ax.set(xlabel='Время t, с',
               ylabel='Сила тока, А',
               title='Сила тока и напряжение от времени')
    cur_ax.grid()

    vol_ax.plot(data['time'], data['voltage'])
    vol_ax.set(xlabel='Время t, с',
               ylabel='Напряжение, В')
    vol_ax.grid()

    fig.savefig(img_dir + 'current-voltage-all.png')

    ## Display only N periods
    N = 2
    t_final = N * T
```

```
45      time = data['time'][data['time'] < t_final]
46      current = data['current'][data['time'] < t_final]
47      voltage = data['voltage'][data['time'] < t_final]
48
49      fig, (cur_ax, vol_ax) = plt.subplots(2, 1, sharex=True)
50      cur_ax.plot(time, current)
51      cur_ax.set(xlabel='Время t, с',
52                 ylabel='Сила тока, А',
53                 title='Сила тока и напряжение от времени')
54      cur_ax.grid()
55
56      vol_ax.plot(time, voltage)
57      vol_ax.set(xlabel='Время t, с',
58                 ylabel='Напряжение, В')
59      vol_ax.grid()
60      fig.savefig(img_dir + f'current-voltage-{N}T.png')
61
62      ## Estimate parameters
63      X = np.c_[data['voltage'][:-1], data['current'][:-1]]
64      Y = data['current'][1:]
65      R, L, Te, k = estimate_params(X, Y, Td)
66      print(f'Estimated R = {R} Ohm\n'
67            f'Estimated L = {L} Hn')
68
69      ## Compare model and given data
70      current_est = (X @ k)[:time.size]
71
72      fig, ax = plt.subplots()
73      ax.plot(time, current, label='Исходные данные')
74      ax.plot(time, current_est, label='Аппроксимированные
        ↪  данные')
75      ax.set(xlabel='Время, с',
76             ylabel='Сила тока, А',
77             title='Сравнение модели и данных')
78      ax.grid()
79      ax.legend()
80      fig.savefig(img_dir + 'current_estimation_comparison.png')
81
82      ## Mean and std of parameters
83      R_est = np.array([])
84      L_est = np.array([])
85      n = 1000
86      for i in range(n):
87          indices = (i * T <= data['time']) & (data['time'] <= (i
            ↪  + 1) * T)
88          curr = data['current'][indices]
89          volt = data['voltage'][indices]
90
```

```
91          X = np.c_[volt[:-1], curr[:-1]]
92          Y = curr[1:]
93
94          R, L, Te, k = estimate_params(X, Y, Td)
95
96          R_est = np.append(R_est, R)
97          L_est = np.append(L_est, L)
98
99      print(
100             f'Mean R: {np.mean(R_est[np.isfinite(R_est)])},
                ↪  Ohm\n'
101             f'Standard deviation R:
                ↪  {np.std(R_est[np.isfinite(R_est)])}\n'
102            f'Mean L: {np.mean(L_est[np.isfinite(L_est)])}, Hn\n'
103             f'Standard deviation L:
                ↪  {np.std(L_est[np.isfinite(L_est)])}\n'
104             )
```