Разработка графических интерфейсов

Лабораторная работа 2

Построение вычислений и визуализация их результатов.

Пакеты numpy и matplotlib.

Модуль numpy


```
#!/usr/bin/python

import numpy as np

data = np.array([1, 2, 3, 4, 5])

print(data)
```

```
[1 2 3 4 5]
```

Массивы в numpy

```
#!/usr/bin/python

import numpy as np

data = np.array([1, 2, 3, 4, 5])

print(data)                                                    [1 2 3 4 5]



data = np.array([1, 2, 3, 4, 5], dtype=np.float32)    [1. 2. 3. 4. 5.]



data = np.array([1, 2, 3, 4, 5], dtype=complex)       [1.+0.j 2.+0.j 3.+0.j 4.+0.j 5.+0.j]
```

Варианты генерации массивов (1)

```
#!/usr/bin/python

import numpy as np

data = np.array([1, 2, 3, 4, 5])

print(data)


data = np.arange(10)

data = np.arange(1, 10)

data = np.arange(1, 10, 2)
```

```
[1 2 3 4 5]

[0 1 2 3 4 5 6 7 8 9]

[1 2 3 4 5 6 7 8 9]

[1 3 5 7 9]
```

Варианты генерации массивов (2)

```
#!/usr/bin/python

import numpy as np

data = np.arange(1, 10, 2)

print(data)
```

```
[1 3 5 7 9]
```

```
data = np.zeros(10)
```

```
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
```

```
data = np.ones(10)
```

```
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
```

```
data = np.empty(5)
```

```
[4.9406565e-324 3.9525252e-323
 3.9525252e-323 3.9525252e-323
 3.7236325e-317]
```

Варианты генерации массивов (3)

```
#!/usr/bin/python

import numpy as np

data = np.arange(1, 10, 2)

print(data)
```

```
[1 3 5 7 9]
```

```
data = np.linspace(1, 3, 7)
```

```
[1.         1.33333333
 1.66666667 2.
 2.33333333 2.66666667
 3.        ]
```

```
data = np.stack(2 * i for i in range(10))
```

```
[0 2 4 6 8 10 12 14 16 18]
```

Обращение к элементам массива

```
#!/usr/bin/python

import numpy as np

data = np.array([1, 2, 3, 4, 5])

print(data)                                    [1 2 3 4 5]


print(data[3])                                 4


print(data[:3])                                [1 2 3]


print(data[3:])                                [4 5]


print(data[1:4])                               [2 3 4]
```

Изменение элементов массива

```
#!/usr/bin/python

import numpy as np

data = np.array([1, 2, 3, 4, 5])

print(data)                                    [1 2 3 4 5]

data[3] = -1                                   [1 2 3 -1 5]

data[:3] = [6, 7, 8]                           [6 7 8 4 5]

data[3:] = [-4, -5]                            [1 2 3 -4 -5]

data[1:4] = [-2, -3, -4]                       [1 -2 -3 -4 5]
```

Операции с массивами

```
#!/usr/bin/python

import numpy as np

a = np.array([1, 2, 3, 4, 5])
b = np.array([1, 2, 3, 4, 5])
c = a + b

print(c)                                    [2 4 6 8 10]

c = a - b                                   [0 0 0 0 0]


c = a*2                                     [2 4 6 8 10]


c = a + 3*b                                 [ 4  8 12 16 20]
```

Матрицы в numpy

```
#!/usr/bin/python

import numpy as np

data = np.array([[1, 2, 3], [4, 5, 6]])

print(data)
```

```
[[1 2 3]
 [4 5 6]]
```

```
data = np.array([[1, 2, 3], [4, 5, 6]], dtype=np.float32)
```

```
[[1. 2. 3.]
 [4. 5. 6.]]
```

Варианты генерации матриц (1)

```
#!/usr/bin/python

import numpy as np

data = np.array([[1, 2, 3], [4, 5, 6]])

print(data)
```

```
[[1 2 3]
 [4 5 6]]
```

```
data = np.zeros((3, 3))
```

```
[[0. 0. 0.]
 [0. 0. 0.]
 [0. 0. 0.]]
```

```
data = np.ones((3, 3))
```

```
[[1. 1. 1.]
 [1. 1. 1.]
 [1. 1. 1.]]
```

Варианты генерации матриц (2)

```
#!/usr/bin/python

import numpy as np

data = np.ones((3, 3))

print(data)
```

```
[[1. 1. 1.]
 [1. 1. 1.]
 [1. 1. 1.]]
```

```
data = np.eye(3)
```

```
[[1. 0. 0.]
 [0. 1. 0.]
 [0. 0. 1.]]
```

```
data = np.empty((3, 3))
```

```
[[0.00000000e+000 0.00000000e+000 0.00000000e+000]
 [0.00000000e+000 0.00000000e+000 2.66795449e-321]
 [4.10463701e-288 2.29812144e-311 0.00000000e+000]]
```

Варианты генерации матриц (3)

```
#!/usr/bin/python

import numpy as np

data = np.linspace(1, 9, 9)


print(data)
```

```
[1. 2. 3. 4. 5. 6. 7. 8. 9.]
```

```
data = data.reshape((3, 3))
```

```
[[1. 2. 3.]
 [4. 5. 6.]
 [7. 8. 9.]]
```

```
data = np.linspace(1, 9, 9).reshape((3, 3))
```

```
[[1. 2. 3.]
 [4. 5. 6.]
 [7. 8. 9.]]
```

Обращение к элементам матриц (1)

```
#!/usr/bin/python

import numpy as np

data = np.linspace(1, 9, 9).reshape((3, 3))

print(data)
```

```
[[1. 2. 3.]
 [4. 5. 6.]
 [7. 8. 9.]]
```

```
print(data[1:])
```

```
[[4. 5. 6.]
 [7. 8. 9.]]
```

```
print(data[:1])
```

```
[[1. 2. 3.]]
```

```
print(data[:,2])
```

```
[3. 6. 9.]
```

Обращение к элементам матриц (2)

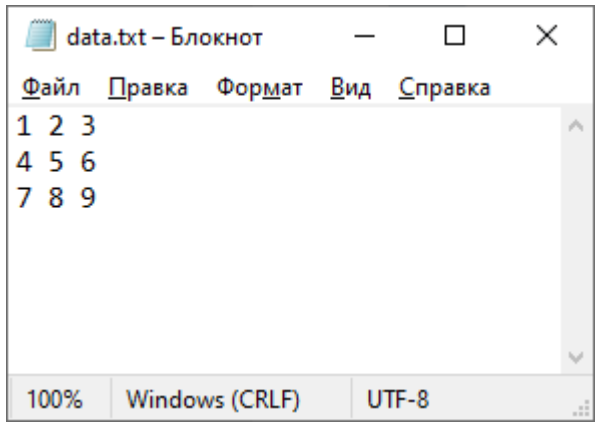
```
#!/usr/bin/python

import numpy as np

data = np.linspace(1, 9, 9).reshape((3, 3))

print(data)
```

```
[[1. 2. 3.]
 [4. 5. 6.]
 [7. 8. 9.]]
```

```
print(data[1:,[0, 2]])
```

```
[[4. 6.]
 [7. 9.]]
```

Чтение данных из файла




```
#!/usr/bin/python

import numpy as np

data = np.loadtxt("data.txt", np.float32)

print(data)
```

```
[[1. 2. 3.]
 [4. 5. 6.]
 [7. 8. 9.]]
```

Функции работы с массивами и матрицами

```
#!/usr/bin/python

import numpy as np

data = np.array([1, 6, 7, 2, 1, 5]).reshape(2, 3)
```

Для одномерных массивов

```
print(max(data))          7

print(data.max())         7

print(sum(data))          22

print(data.sum())         22
```

Для матриц

```
print(data.max())         7

print(data.sum())         22

print(sum(data))          [ 3  7 12]
```

Методы линейной алгебры

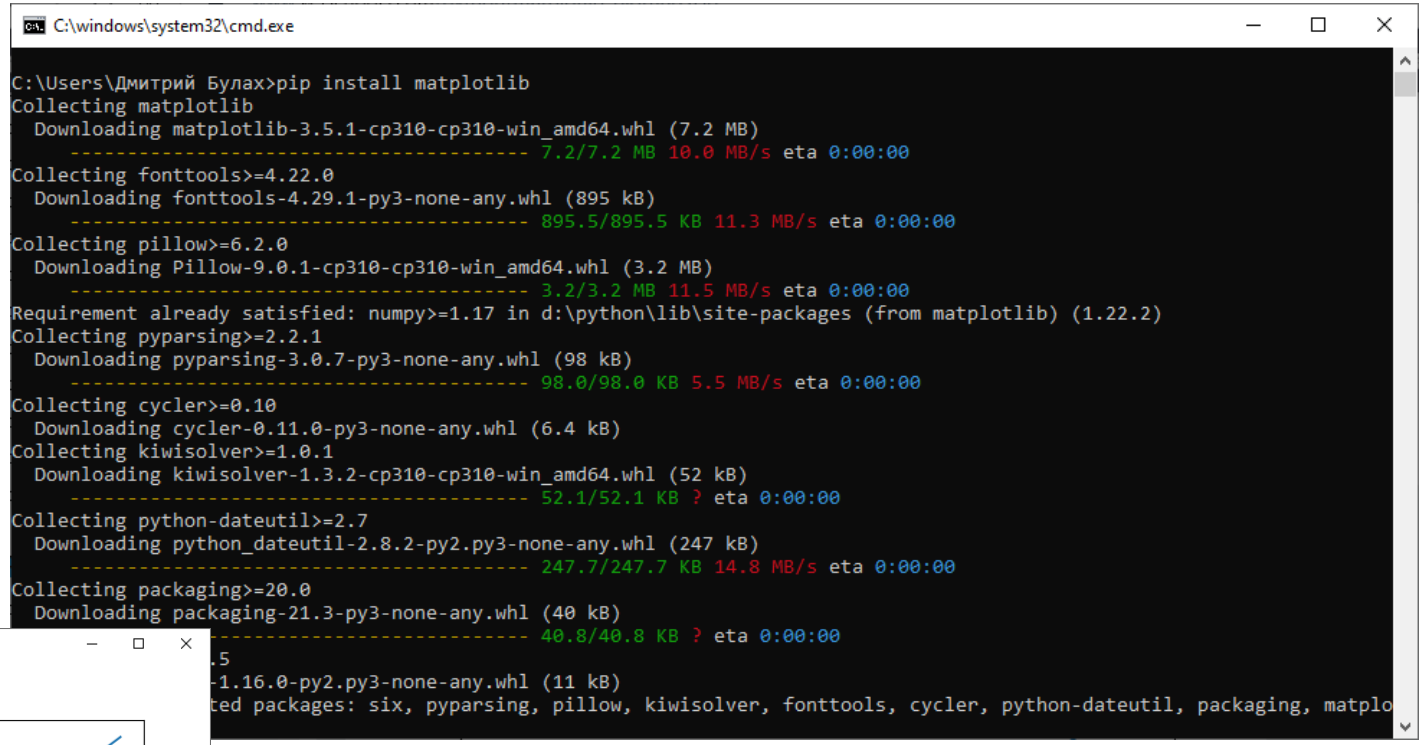
$$\begin{cases} 1x_1 + 2x_2 = 1 \\ 3x_1 + 5x_2 = 2 \end{cases}$$

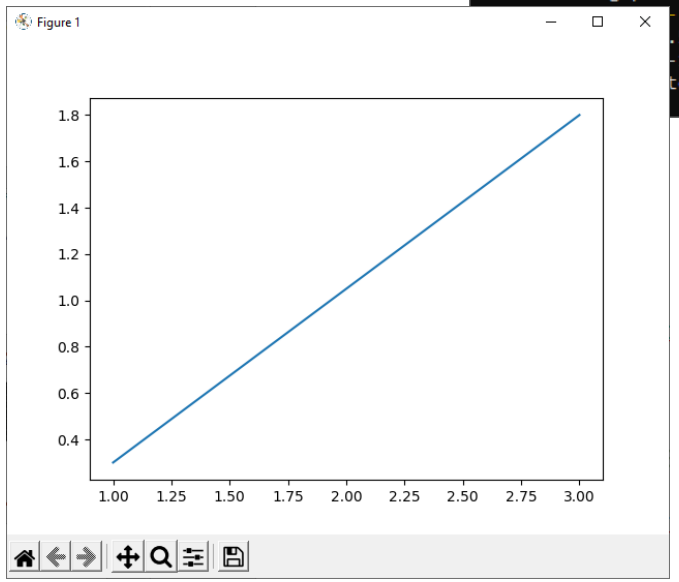
```python
#!/usr/bin/python

import numpy as np

a = np.array([[1, 2], [3, 5]])
b = np.array([1, 2])


x = np.linalg.solve(a, b)

print(x)
```

```
[-1.  1.]
```

Модуль matplotlib

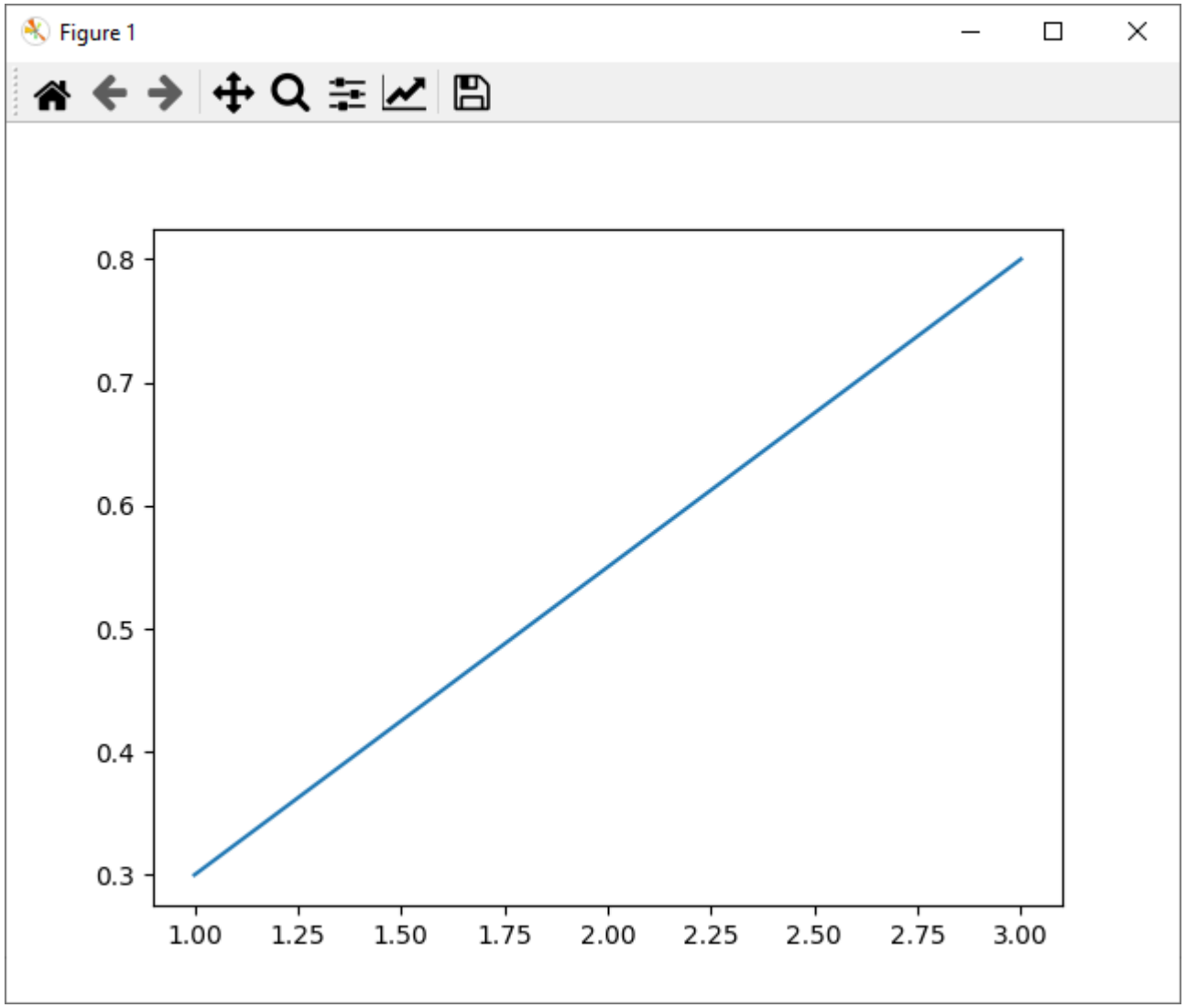

```
#!/usr/bin/python

import matplotlib.pyplot as plt
import numpy as np

plt.plot([1, 3], [0.3, 1.8])

plt.show()
```





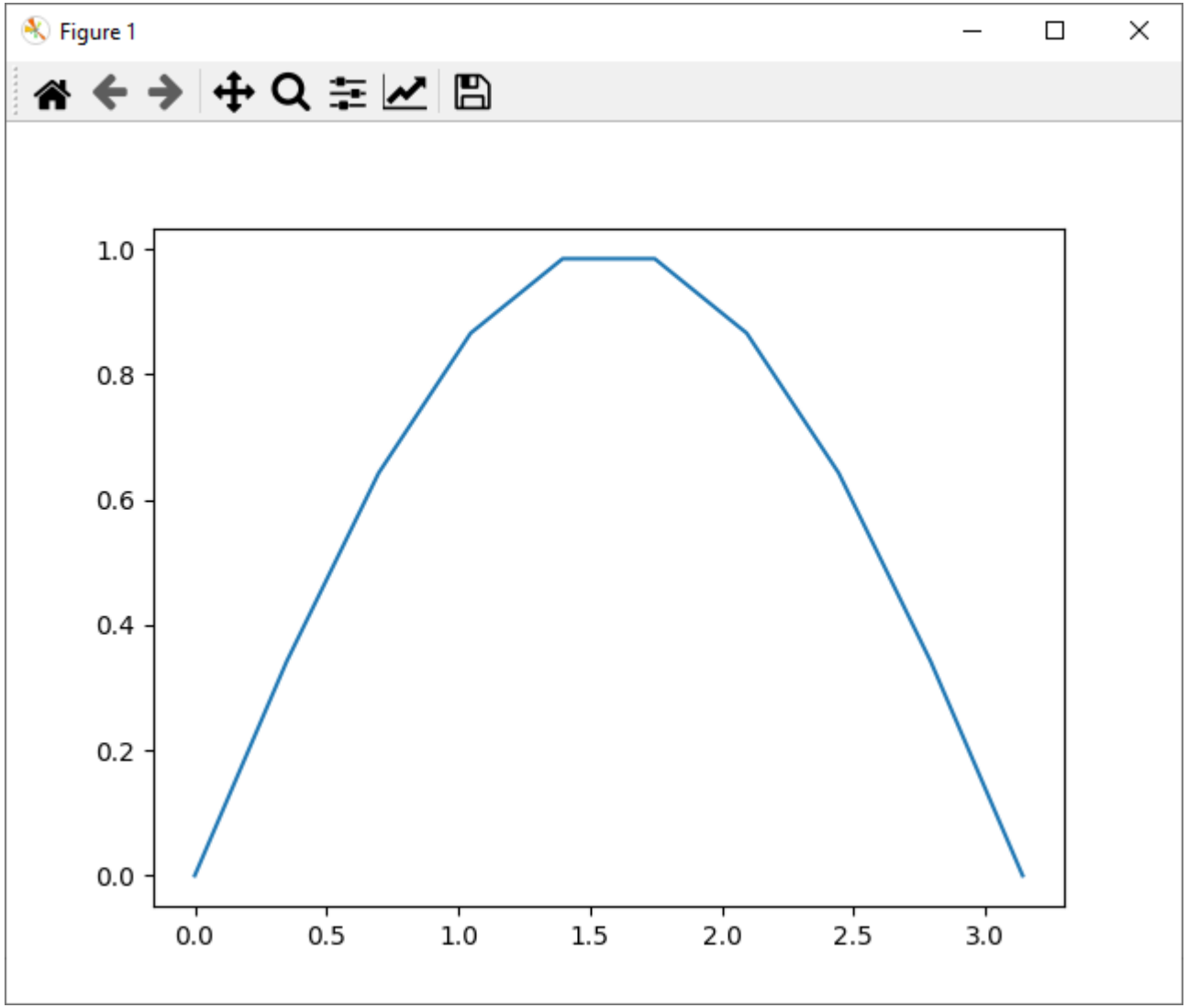
Разделение данных по осям

```
#!/usr/bin/python

import matplotlib.pyplot as plt
import numpy as np

xdata = np.array([1, 3])
ydata = np.array([0.3, 0.8])

plt.plot(xdata, ydata)
plt.show()
```



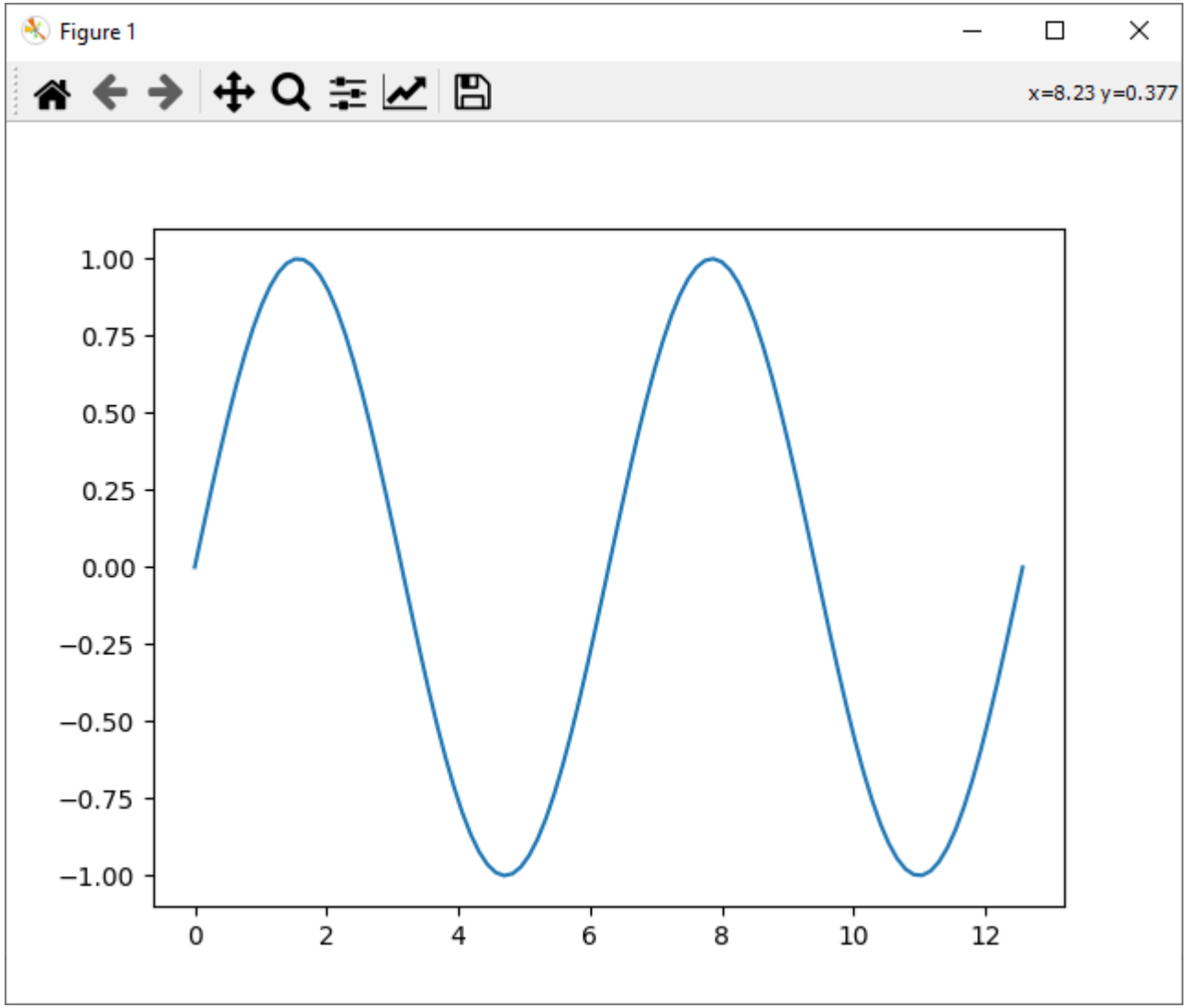
Визуализация синусоиды (1)

```
#!/usr/bin/python

import matplotlib.pyplot as plt
import numpy as np

x = np.linspace(0, np.pi, 10)
y = np.sin(x)

plt.plot(x, y)
plt.show()
```

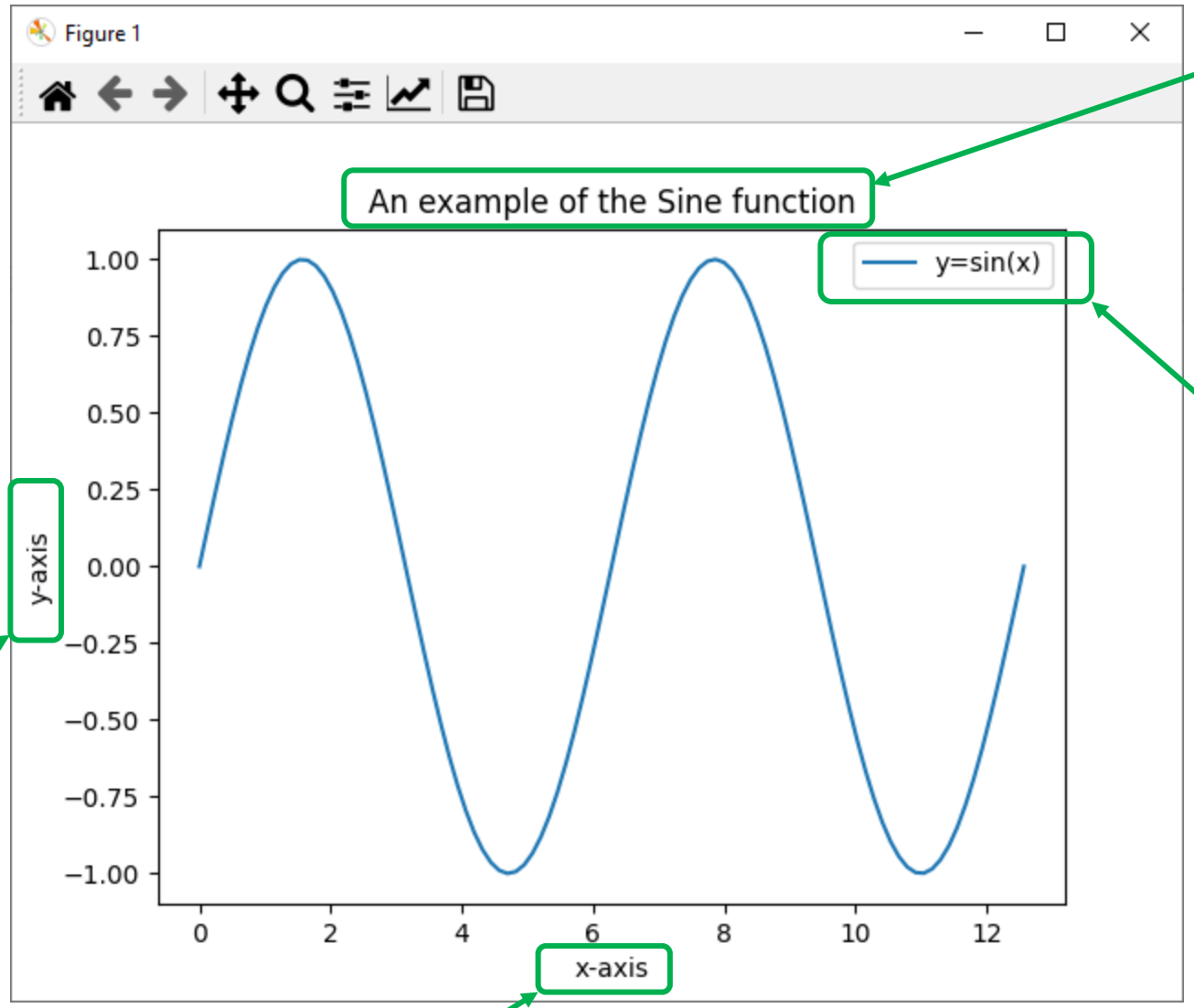


Визуализация синусоиды (2)

```
#!/usr/bin/python

import matplotlib.pyplot as plt
import numpy as np

x = np.linspace(0, 4*np.pi, 100)
y = np.sin(x)

plt.plot(x, y)
plt.show()
```



Аннотирование графиков



Название графика

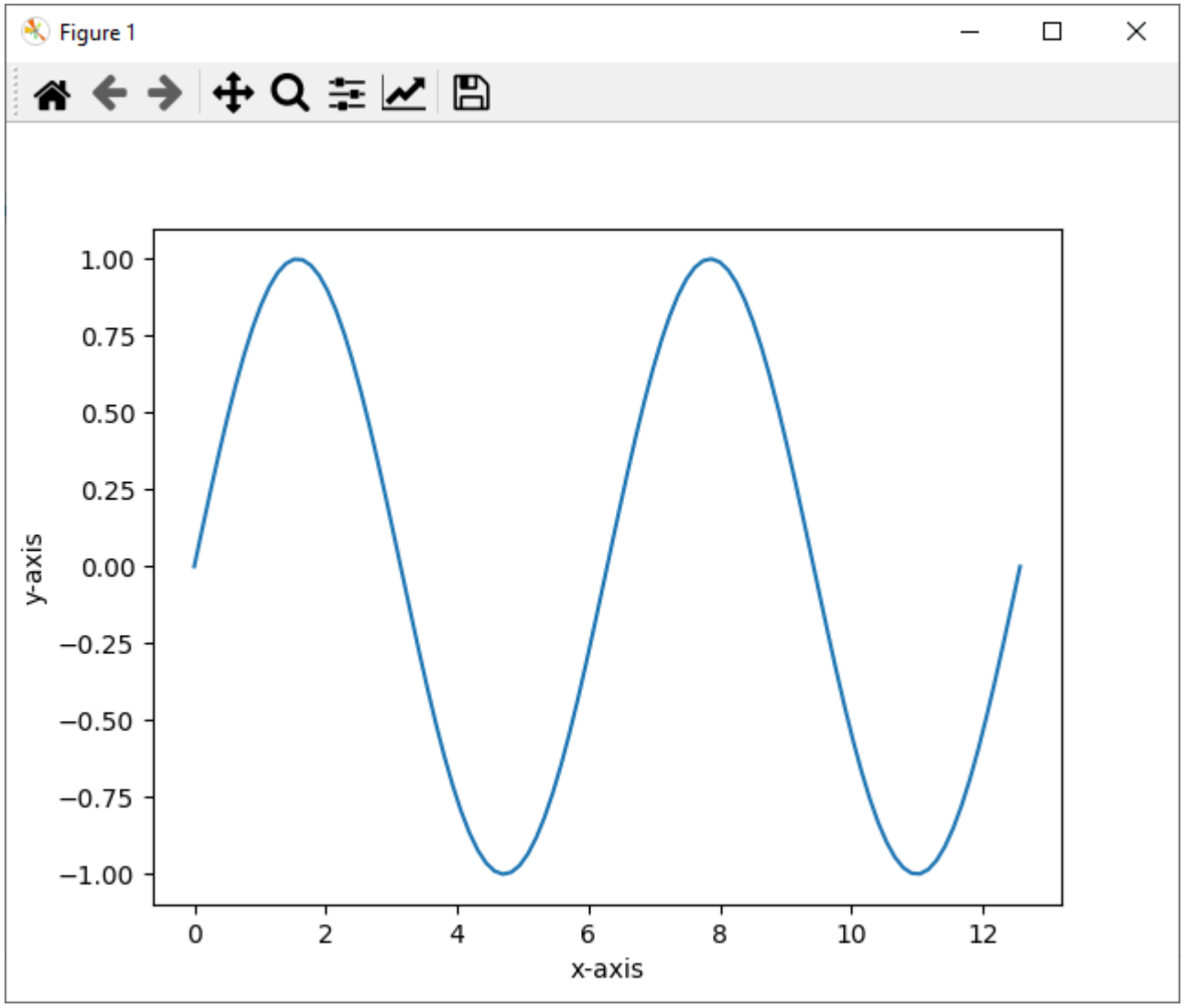
Легенда (обозначения функций)

Подпись оси Y

Подпись оси X

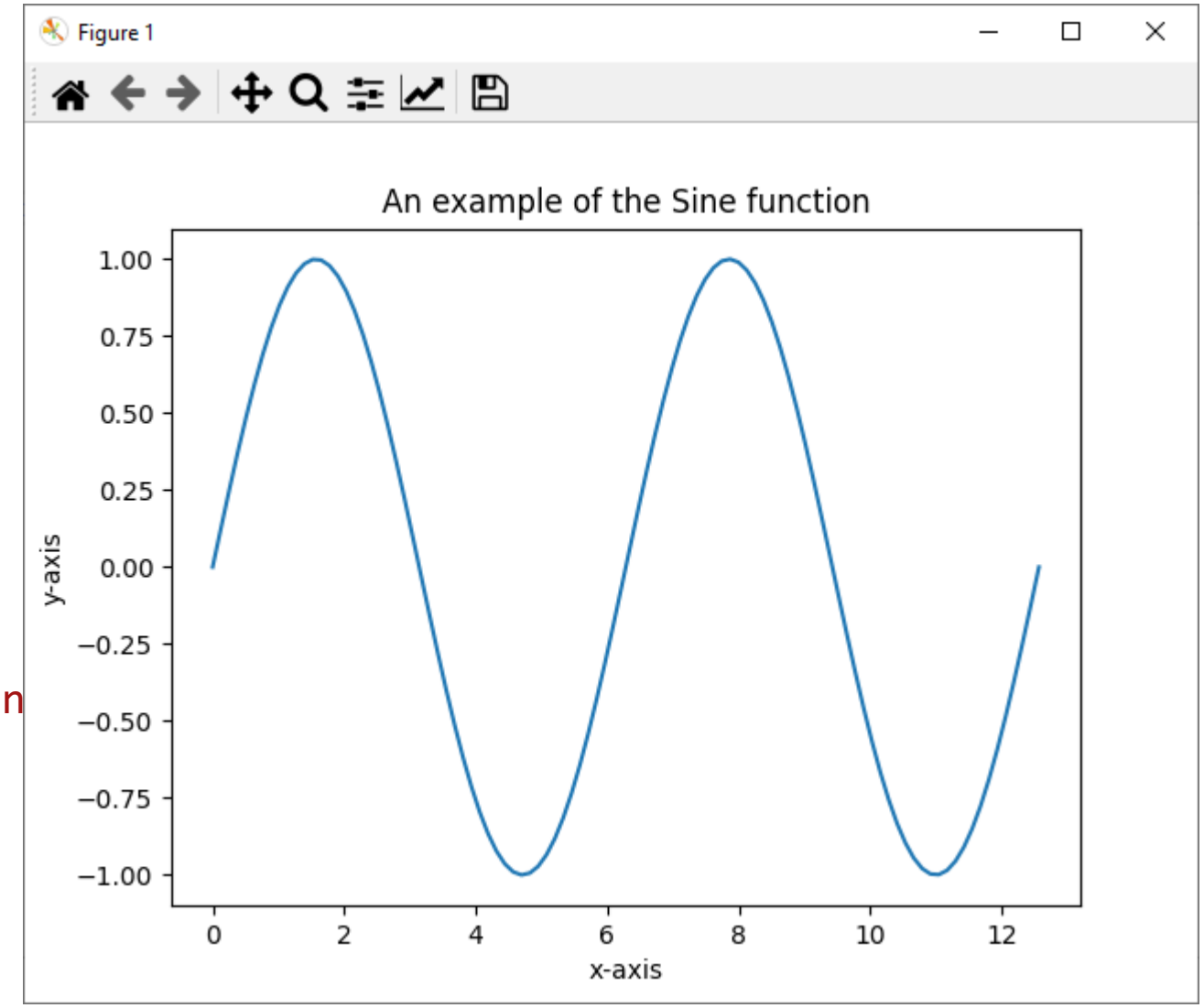
Аннотирование графиков: подписи по осям

```
#!/usr/bin/python

import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(0, 4*np.pi, 100)
y = np.sin(x)

plt.plot(x, y)

plt.xlabel('x-axis')
plt.ylabel('y-axis')

plt.show()
```



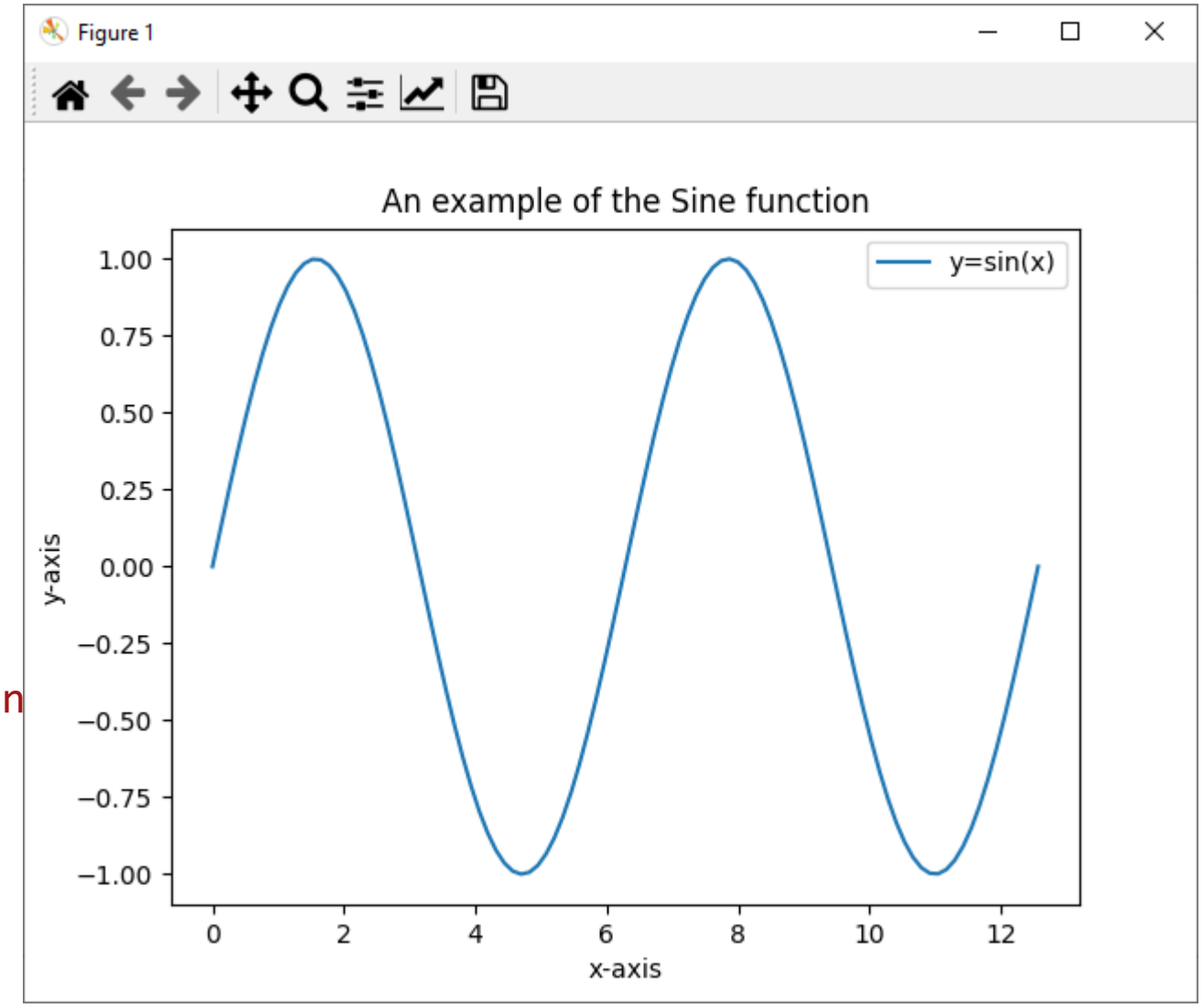
Аннотирование графиков: название графика

```
#!/usr/bin/python

import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(0, 4*np.pi, 100)
y = np.sin(x)

plt.plot(x, y)

plt.xlabel('x-axis')
plt.ylabel('y-axis')
plt.title('An example of the Sine function')

plt.show()
```



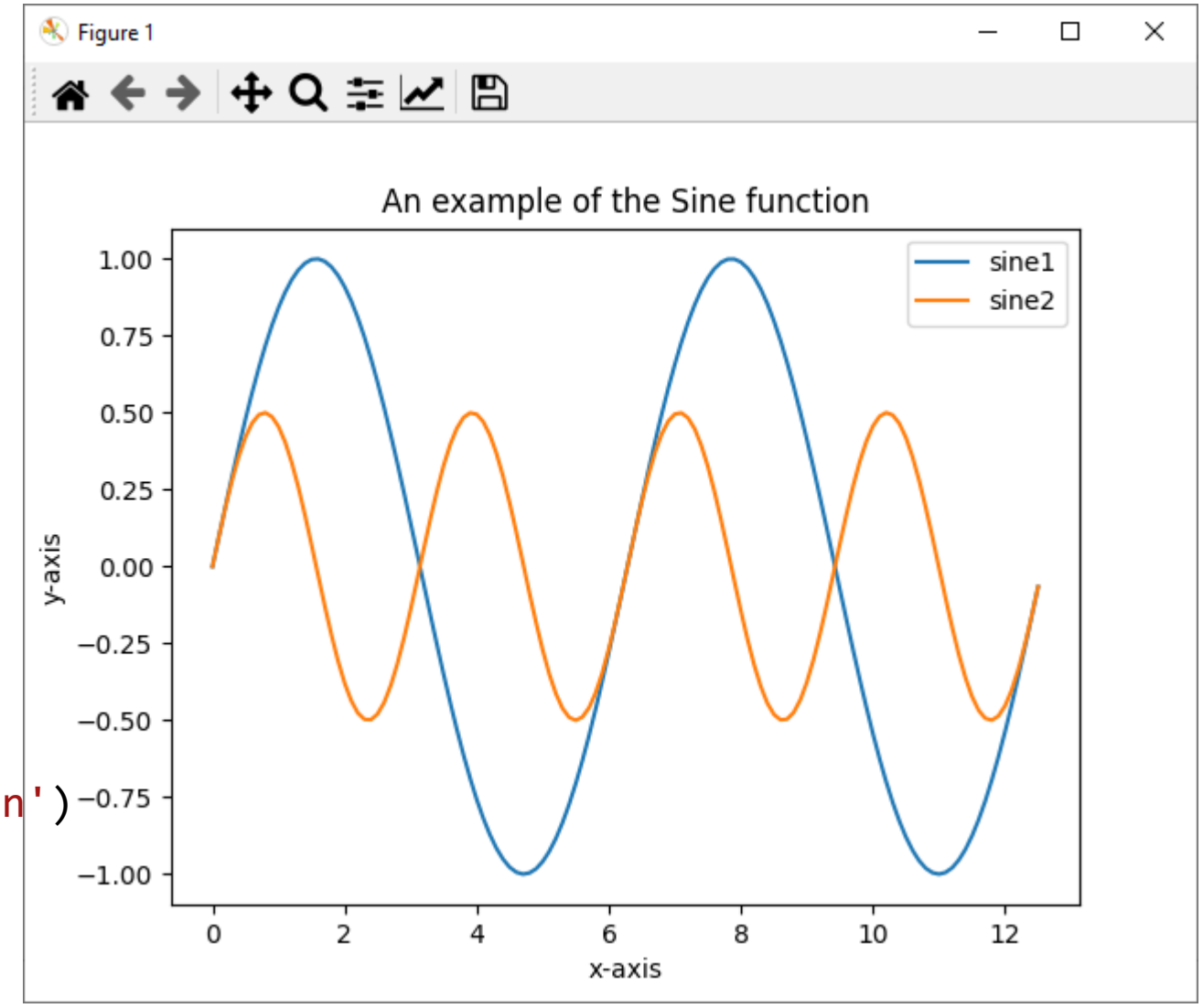
Аннотирование графиков: легенда

```
#!/usr/bin/python

import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(0, 4*np.pi, 100)
y = np.sin(x)

plt.plot(x, y, label='y=sin(x)')

plt.xlabel('x-axis')
plt.ylabel('y-axis')
plt.title('An example of the Sine function')
plt.legend()

plt.show()
```



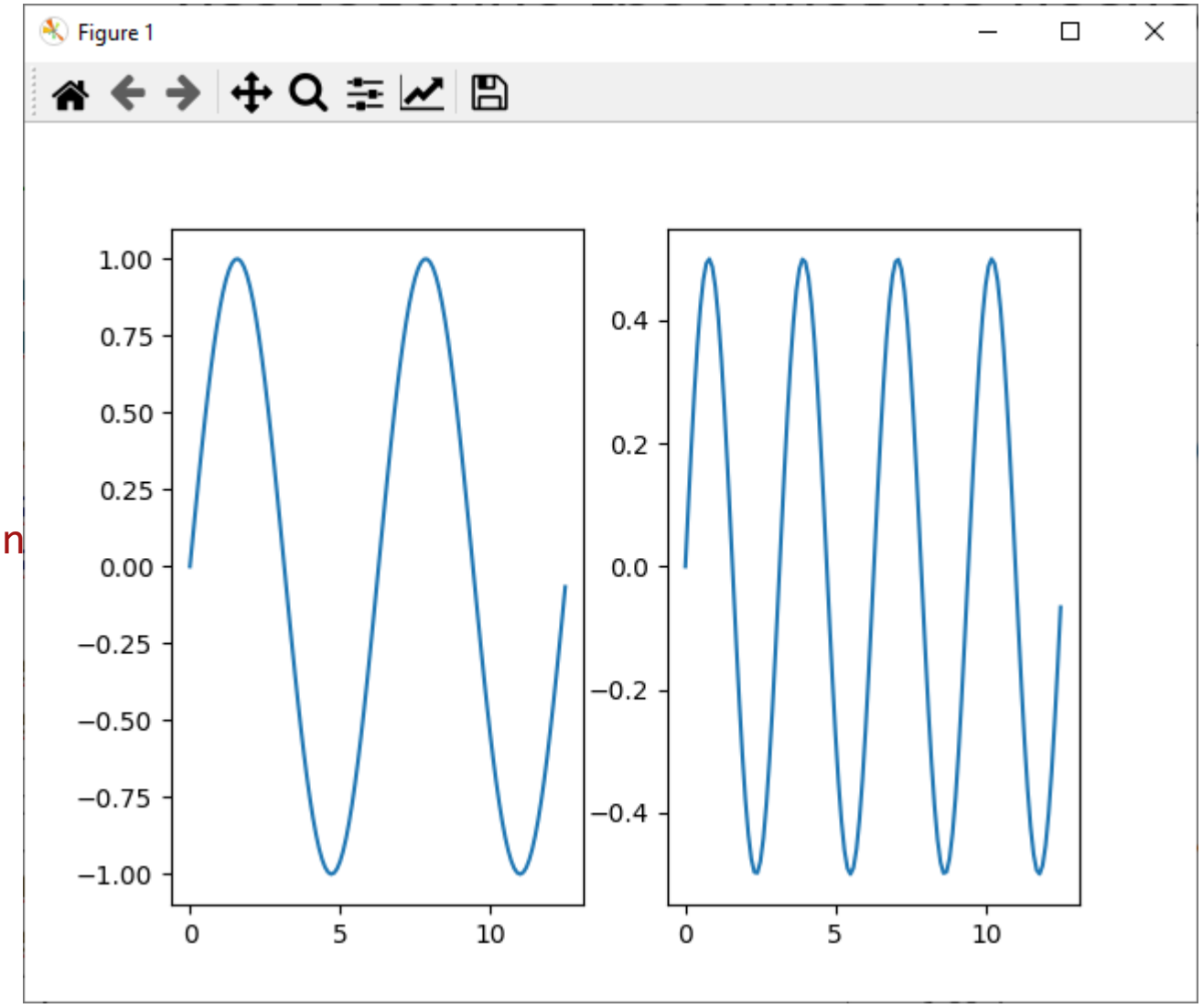
Визуализация нескольких графиков

```
#!/usr/bin/python

import matplotlib.pyplot as plt
import numpy as np

x = np.arange(0, 4*np.pi, 0.1)
y1 = np.sin(x)
y2 = np.sin(x*2)/2

plt.xlabel('x-axis')
plt.ylabel('y-axis')
plt.plot(x, y1, label='sine1')
plt.plot(x, y2, label='sine2')
plt.legend()
plt.title('An example of the Sine function')
plt.show()
```



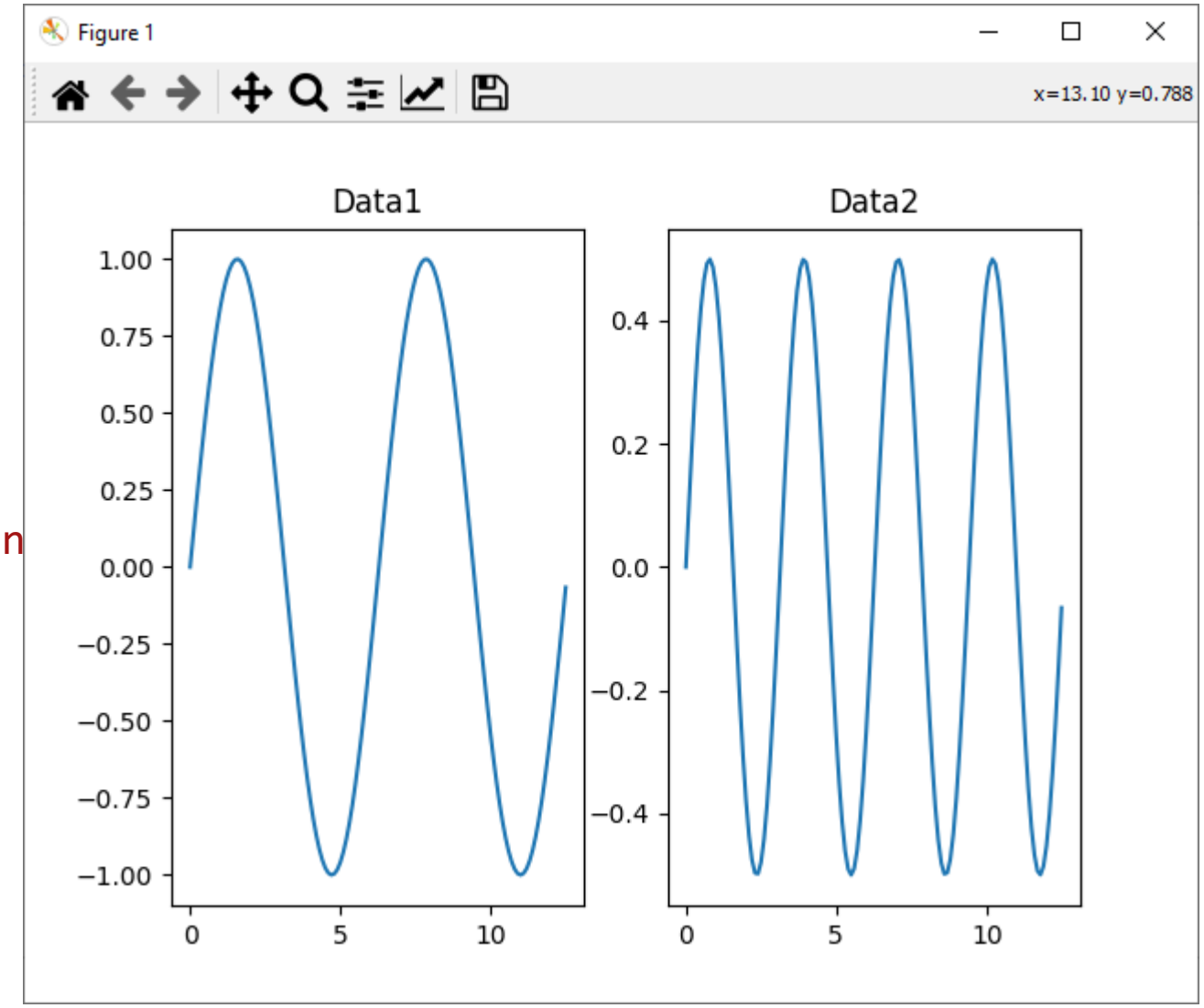
Разделение графиков на несколько областей (1)

```
#!/usr/bin/python

import matplotlib.pyplot as plt
import numpy as np

x = np.arange(0, 4*np.pi, 0.1)
y1 = np.sin(x)
y2 = np.sin(x*2)/2

plt.title('An example of the Sine function
plt.xlabel('x-axis')
plt.ylabel('y-axis')

plt.subplot(1, 2, 1)
plt.plot(x, y1)

plt.subplot(1, 2, 2)
plt.plot(x, y2)

plt.show()
```



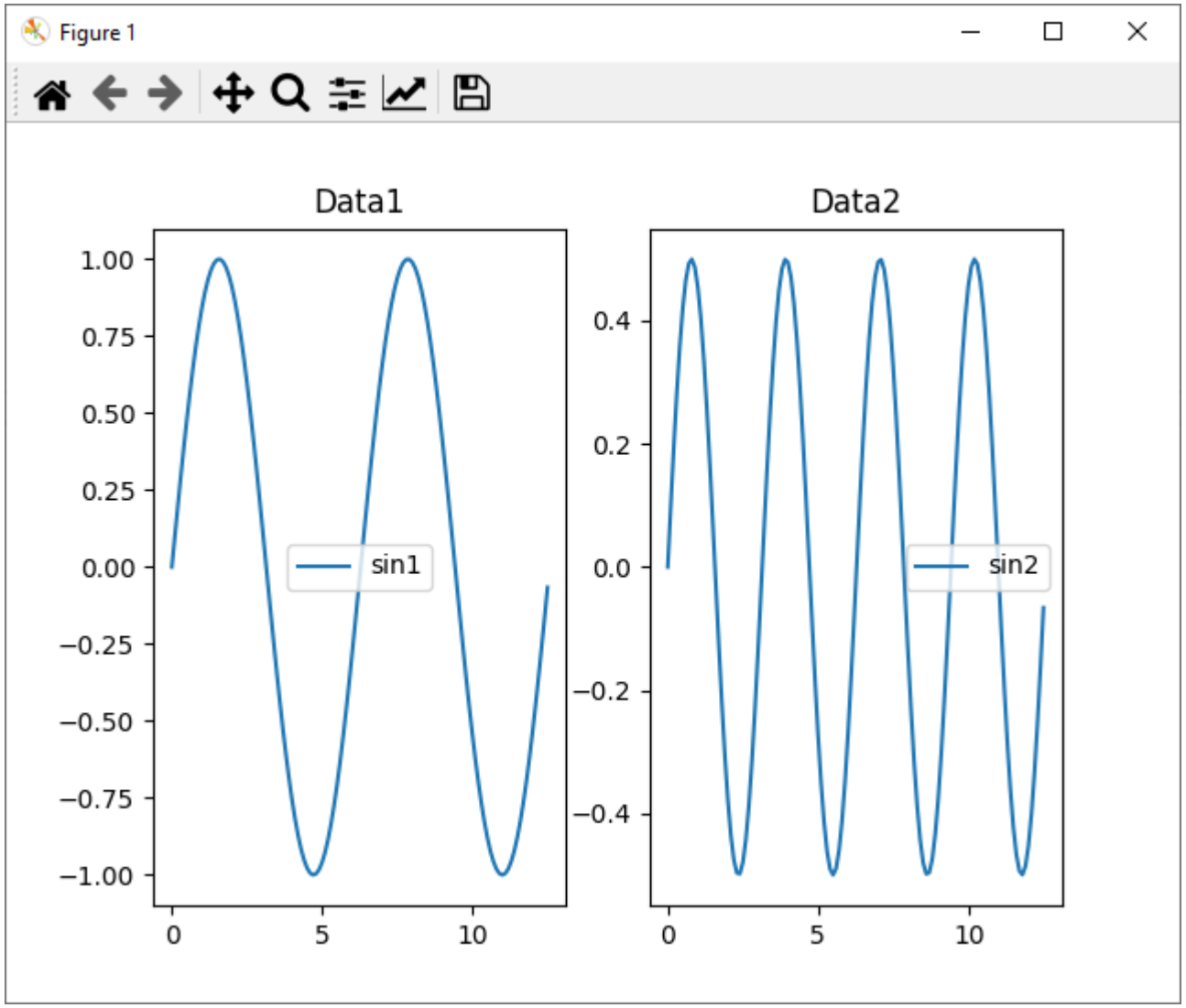
Разделение графиков на несколько областей (2)

```
#!/usr/bin/python

import matplotlib.pyplot as plt
import numpy as np

x = np.arange(0, 4*np.pi, 0.1)
y1 = np.sin(x)
y2 = np.sin(x*2)/2

plt.title('An example of the Sine function

plt.subplot(1, 2, 1)
plt.title('Data1')
plt.plot(x, y1, label='sin1')

plt.subplot(1, 2, 2)
plt.title('Data2')
plt.plot(x, y2, label='sin2')

plt.show()
```



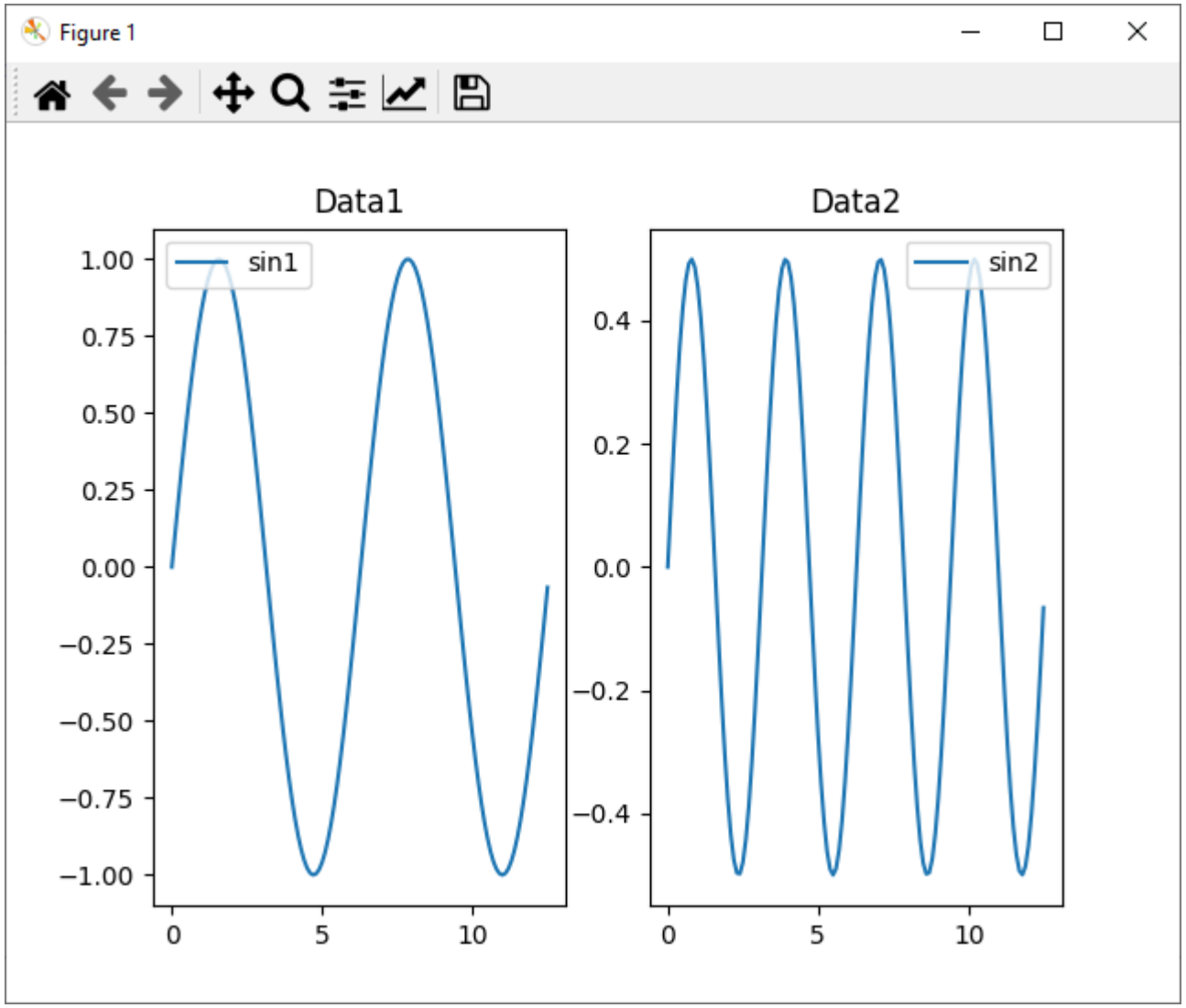
Разделение графиков на несколько областей (2)

```
#!/usr/bin/python

import matplotlib.pyplot as plt
import numpy as np

x = np.arange(0, 4*np.pi, 0.1)
y1 = np.sin(x)
y2 = np.sin(x*2)/2

plt.subplot(1, 2, 1)
plt.title('Data1')
plt.plot(x, y1, label='sin1')
plt.legend()

plt.subplot(1, 2, 2)
plt.title('Data2')
plt.plot(x, y2, label='sin2')
plt.legend()

plt.show()
```



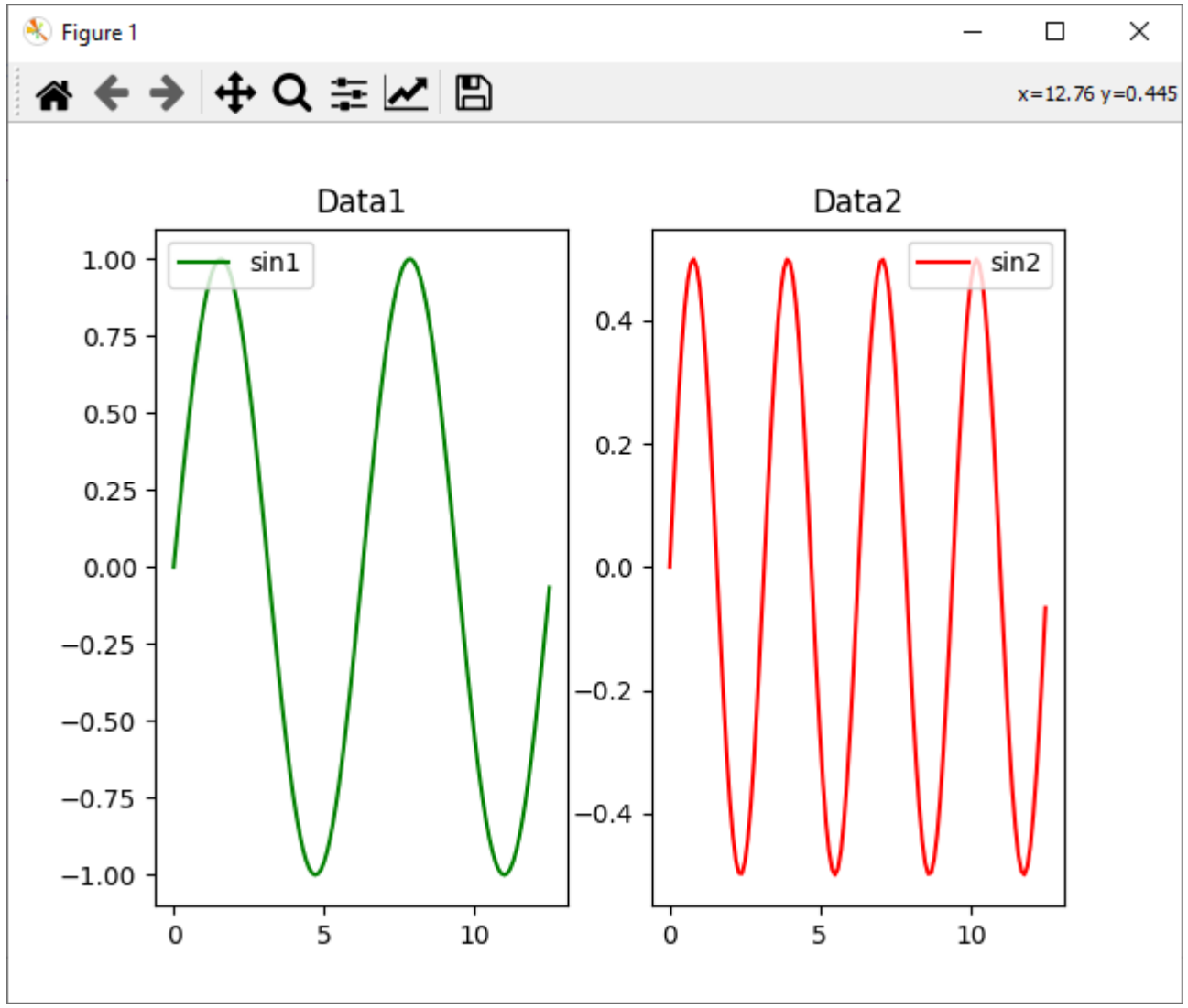
Позиционирование легенд

```
#!/usr/bin/python

import matplotlib.pyplot as plt
import numpy as np

x = np.arange(0, 4*np.pi, 0.1)
y1 = np.sin(x)
y2 = np.sin(x*2)/2

plt.subplot(1, 2, 1)
plt.title('Data1')
plt.plot(x, y1, label='sin1')
plt.legend(loc='upper left')

plt.subplot(1, 2, 2)
plt.title('Data2')
plt.plot(x, y2, label='sin2')
plt.legend(loc='upper right')

plt.show()
```



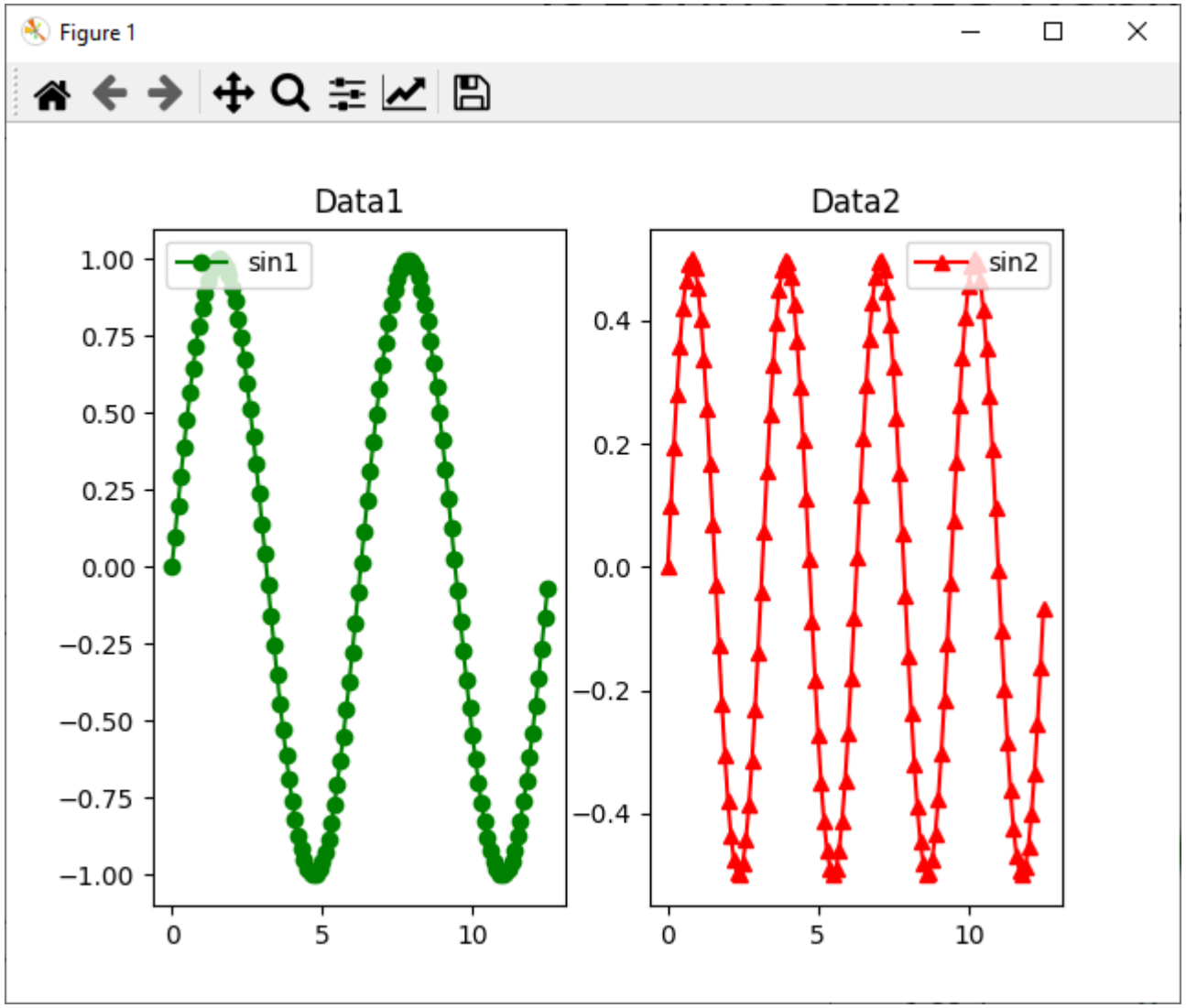
Раскраска графиков

```
...

plt.plot(x, y1, color='g', label='sin1')

...

plt.plot(x, y2, color='#f00', label='sin2'

...

plt.show()
```



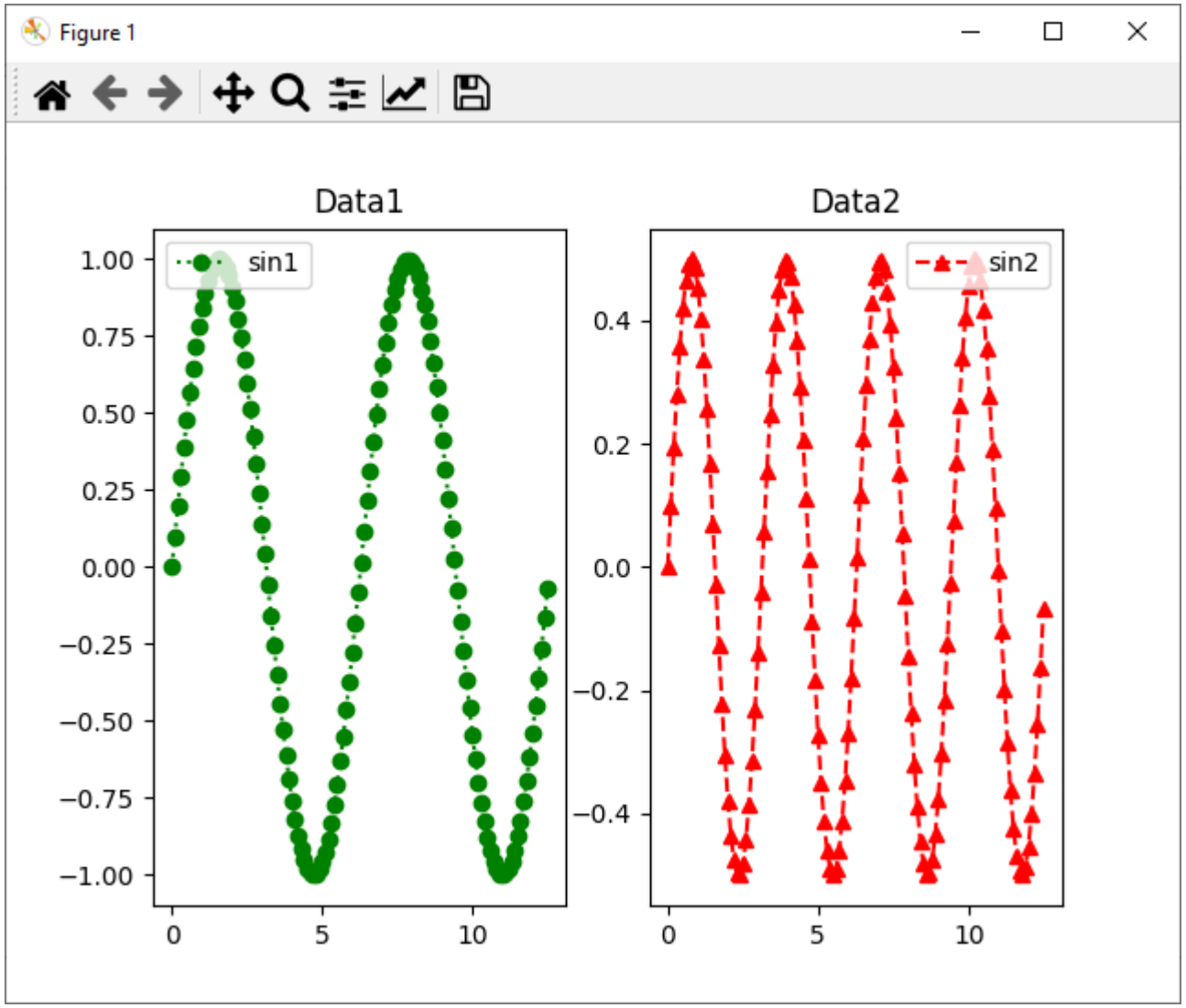
Задание стиля маркера

```
...

plt.plot(x, y1, color='g',
                marker='o',
                label='sin1')

...

plt.plot(x, y2, color='#f00',
                marker='^',
                label='sin2')

...

plt.show()
```



Задание стиля графика

```
...

plt.plot(x, y1, color='g',
                marker='o',
                linestyle=':',
                label='sin1')

...

plt.plot(x, y2, color='#f00',
                marker='^',
                linestyle='--',
                label='sin2')

...

plt.show()
```



Задание стилей графика

```
plt.plot(x, y1, color='g',
                marker='o',
                linestyle=':',
                label='sin1’)
```

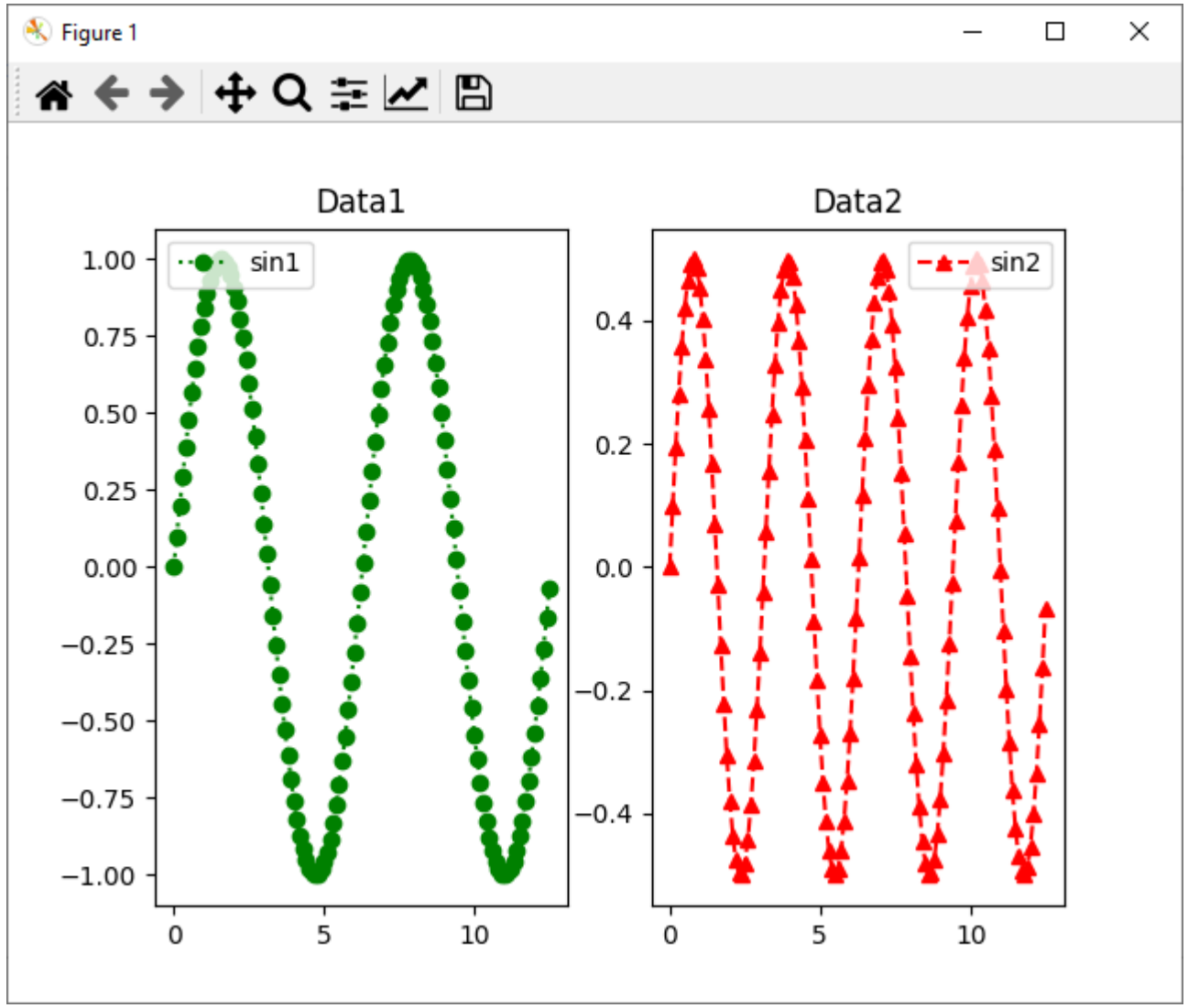


```
plt.plot(x, y1, 'go:', label='sin1')
```

```
plt.plot(x, y2, color='#f00',
                marker='^',
                linestyle='--',
                label='sin2’)
```

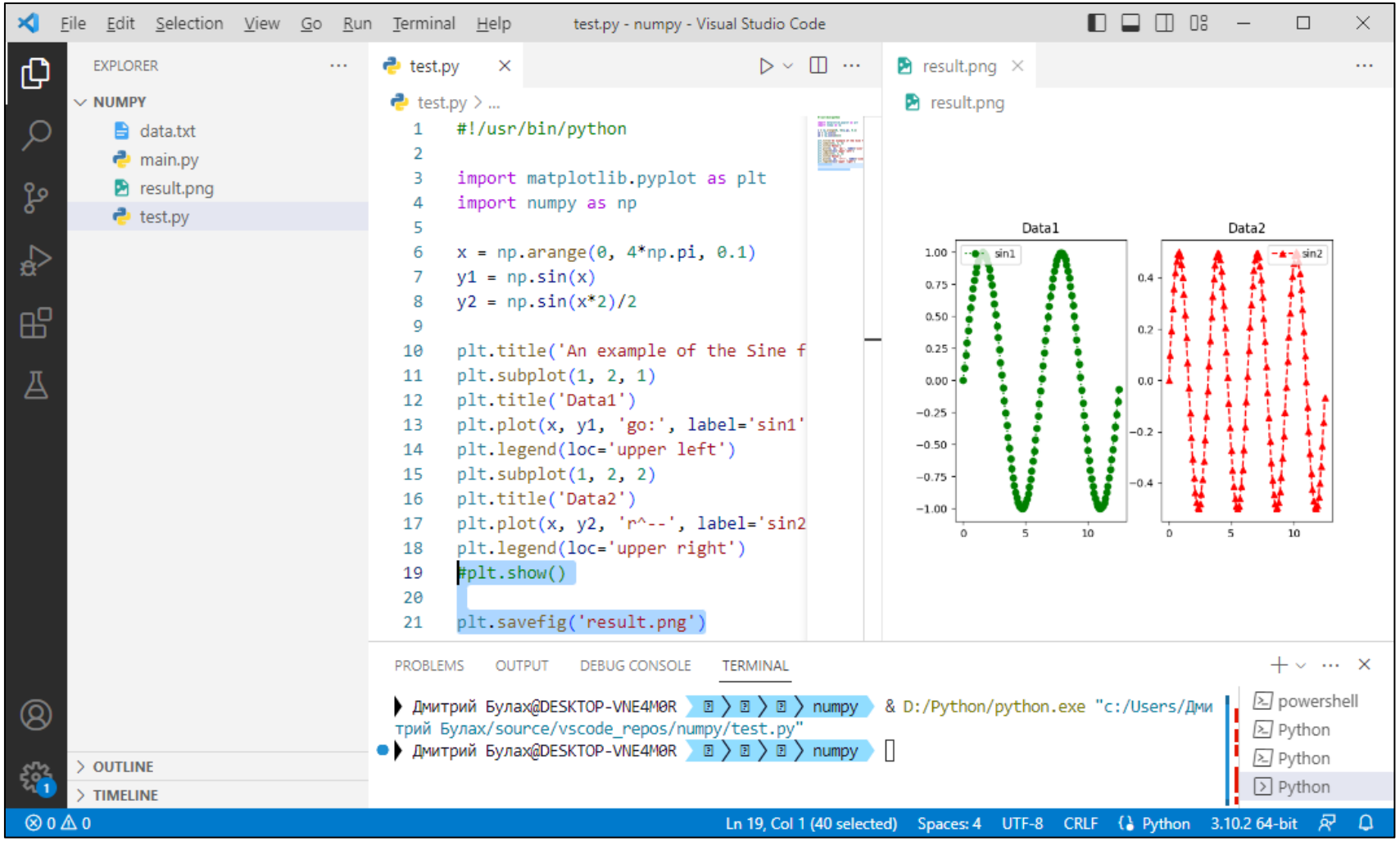


```
plt.plot(x, y2, 'r^--', label='sin2')
```



Сохранение результата в файл

```
#plt.show()

plt.savefig('result.png')
```

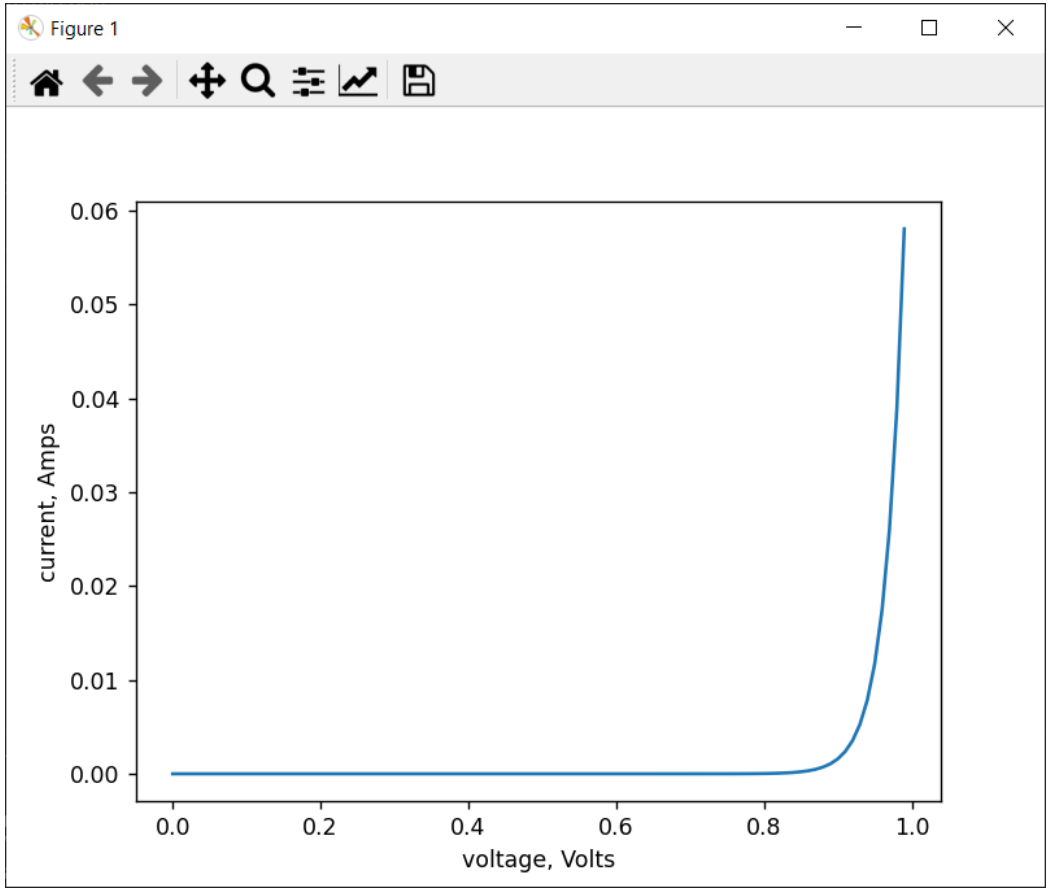


Задание на лабораторную работу (1)

Разработать программу на языке Python с использованием библиотек numpy и matplotlib, которая нарисует график согласно заданию.

Уровень «минимум»

1. Построить график – ВАХ диода в прямом смещении.

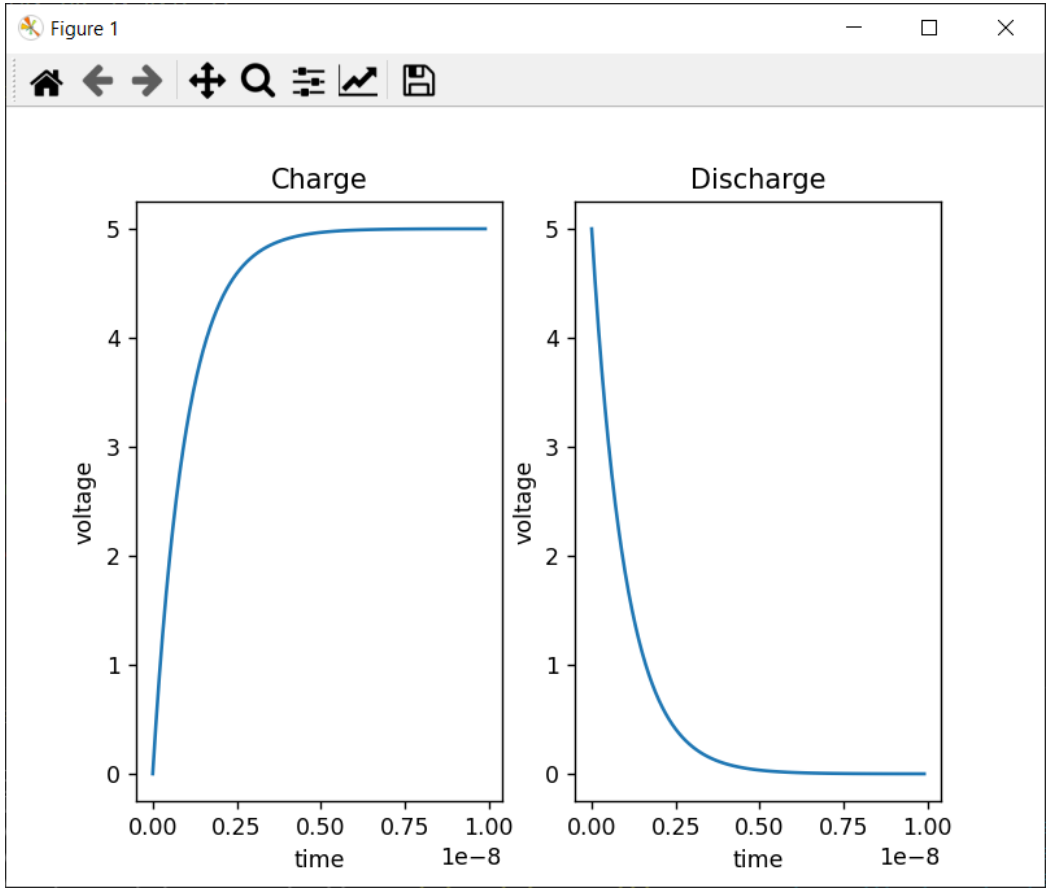


Задание на лабораторную работу (2)

Разработать программу на языке Python с использованием библиотек numpy и matplotlib, которая нарисует график согласно заданию.

Уровень «норм»

1. Построить график – напряжение на RC-цепочке в момент заряда.
2. Построить график – напряжение на RC-цепочке в момент разряда.
3. Совместить два графика на одном окне (см. пример).



Задание на лабораторную работу (3)

Разработать программу на языке Python с использованием библиотек numpy и matplotlib, которая нарисует график согласно заданию.

Уровень «крутыш»

1. Выполнить задание уровня «норм».
2. Построить оба графика как две части единого графика (см. рисунок).
3. Легенды и цвета двух частей графика задаются в коде.