Politechnika Wrocławska

Programowanie obiektowe

W11FTE-SI0080G (INP001129WL)

rok akademicki 2023/24

semestr letni

Wykład 7

Karol Tarnowski

karol.tarnowski@pwr.edu.pl

L-1 p. 220

Politechnika Wrocławska

Plan

Biblioteka matplotlib

- plot()
- hierarchia obiektów
- scatter()
- opisy osi
- legenda
- formatowanie obiektów graficznych
- wykresy 3d

Politechnika Wrocławska

Biblioteka matplotlib

- Biblioteka dostarcza zestaw narzędzi do tworzenia wykresów 2D oraz 3D
- Współpraca z biblioteką numpy
- Funkcjonalności wzorowane na funkcjonalnościach pakietu Matlab

https://matplotlib.org/stable/tutorials/introductory/quick_start.html

Politechnika Wrocławska

Biblioteka matplotlib - podstawy

matplotlib_01.py

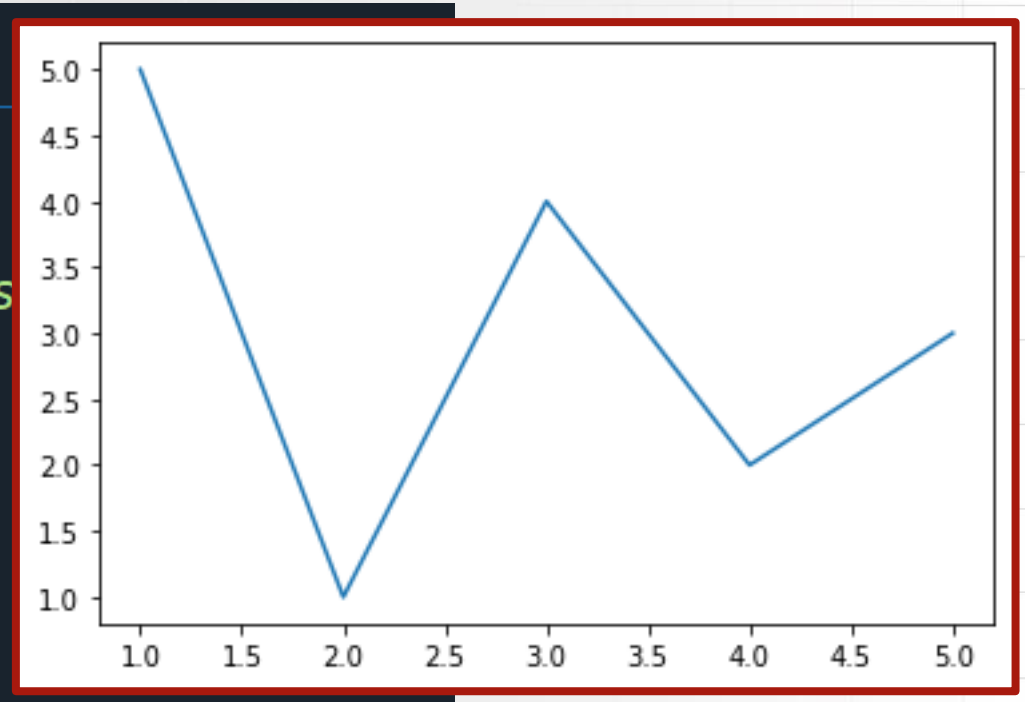
```
1   # -*- coding: utf-8 -*-
2   """
3   Przykład pokazujący tworzenie prostego wykresu.
4   """
5
6   #import matplotlib as mpl
7   import matplotlib.pyplot as plt
8   #import numpy as np
9
10  #utworzenie obiektu wykresu (figure)
11  fig, ax = plt.subplots()
12  x = [1, 2, 3, 4, 5]
13  y = [5, 1, 4, 2, 3]
14  ax.plot(x, y)
```

Politechnika Wrocławska

Biblioteka matplotlib - podstawy

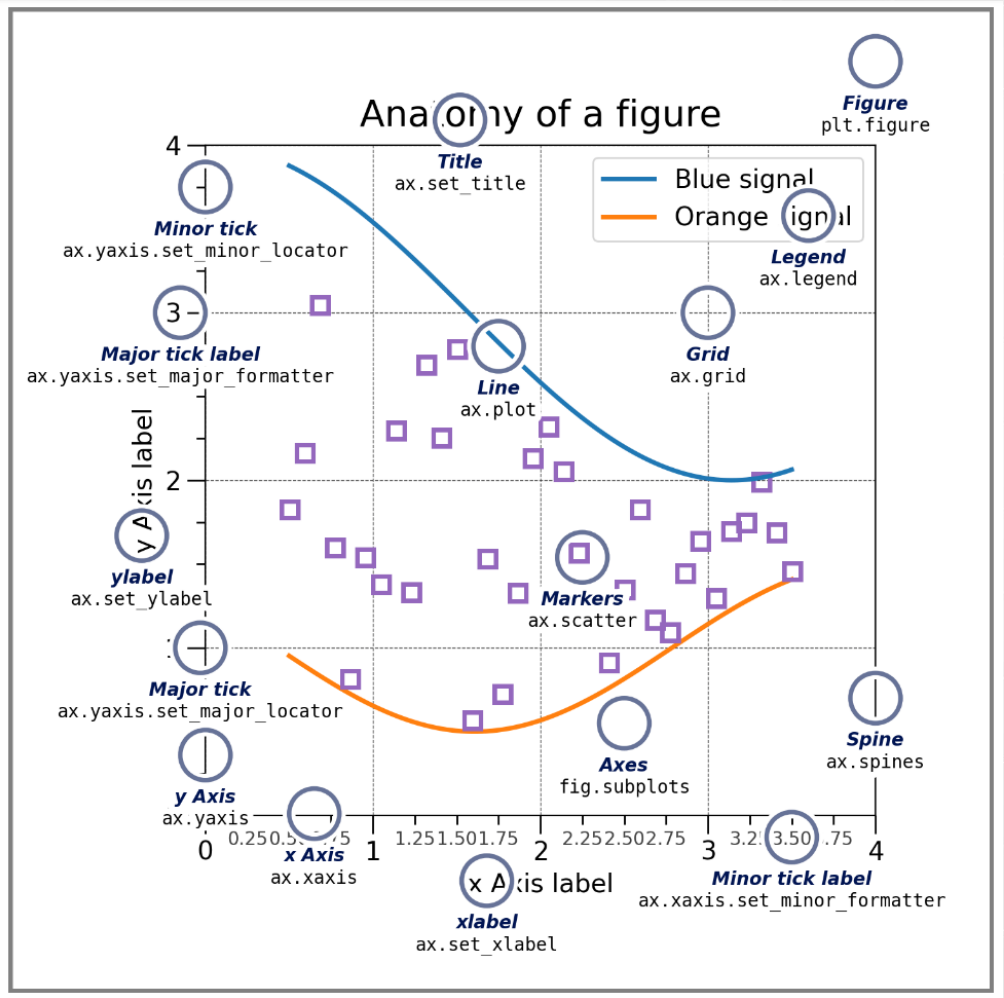
matplotlib_01.py

```
1   # -*- coding: utf-8 -*-
2   """
3   Przykład pokazujący tworzenie pros
4   """
5
6   #import matplotlib as mpl
7   import matplotlib.pyplot as plt
8   #import numpy as np
9
10  #utworzenie obiektu wykresu (figure)
11  fig, ax = plt.subplots()
12  x = [1, 2, 3, 4, 5]
13  y = [5, 1, 4, 2, 3]
14  ax.plot(x, y)
```



Politechnika Wrocławska

Biblioteka matplotlib - podstawy



https://matplotlib.org/stable/tutorials/introductory/quick_start.html

Politechnika Wrocławska

Klasa Figure

matplotlib_02.py

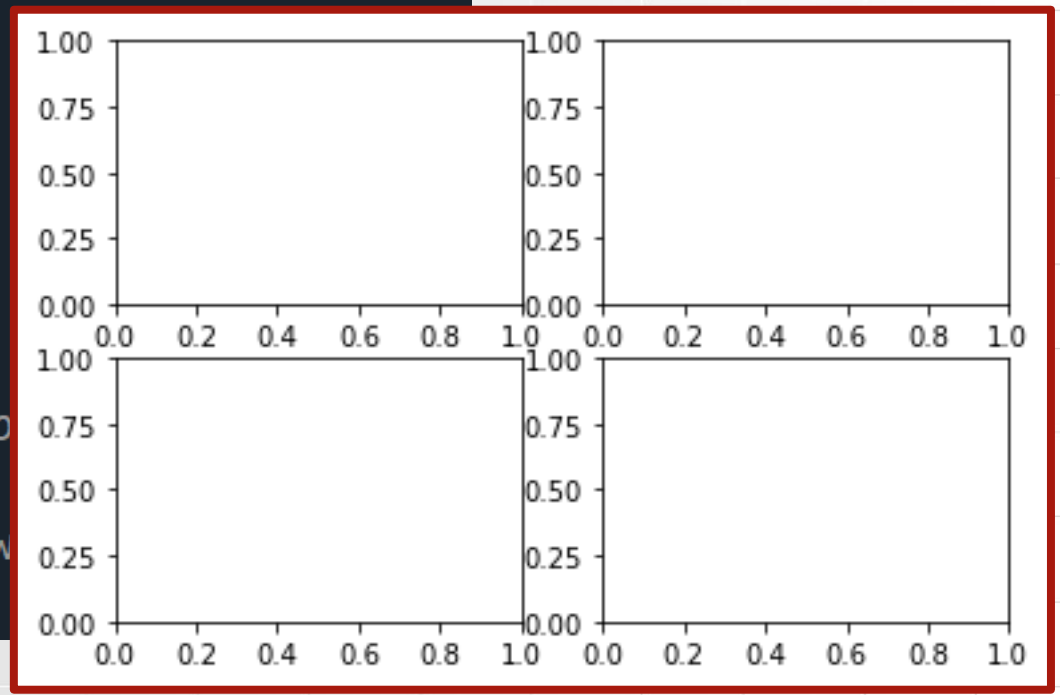
```python
# -*- coding: utf-8 -*-
"""
Przykład pokazujący tworzenie obiektu Figure.
"""

#import matplotlib as mpl
import matplotlib.pyplot as plt
#import numpy as np

# utworzenie obiektu wykresu (Figure)
# tworzenie pustego wykresu bez osi
fig = plt.figure()
# tworzenie wykresu z jednym układem współrzędnych
fig, ax = plt.subplots()
# tworzenie wykresu z siatką 2x2 układów współrzędnych
fig, axs = plt.subplots(2,2)
```

Politechnika Wrocławska

Klasa Figure

matplotlib_02.py

```
1   # -*- coding: utf-8 -*-
2   """
3   Przykład pokazujący tworzenie obiektu Figure.
4   """
5
6   #import matplotlib as mpl
7   import matplotlib.pyplot as plt
8   #import numpy as np
9
10  # utworzenie obiektu wykresu (Figure)
11  # tworzenie pustego wykresu bez osi
12  fig = plt.figure()
13  # tworzenie wykresu z jednym układem wsp
14  fig, ax = plt.subplots()
15  # tworzenie wykresu z siatką 2x2 układów
16  fig, axs = plt.subplots(2,2)
```



Politechnika Wrocławska

Klasa Axes

matplotlib_03.py

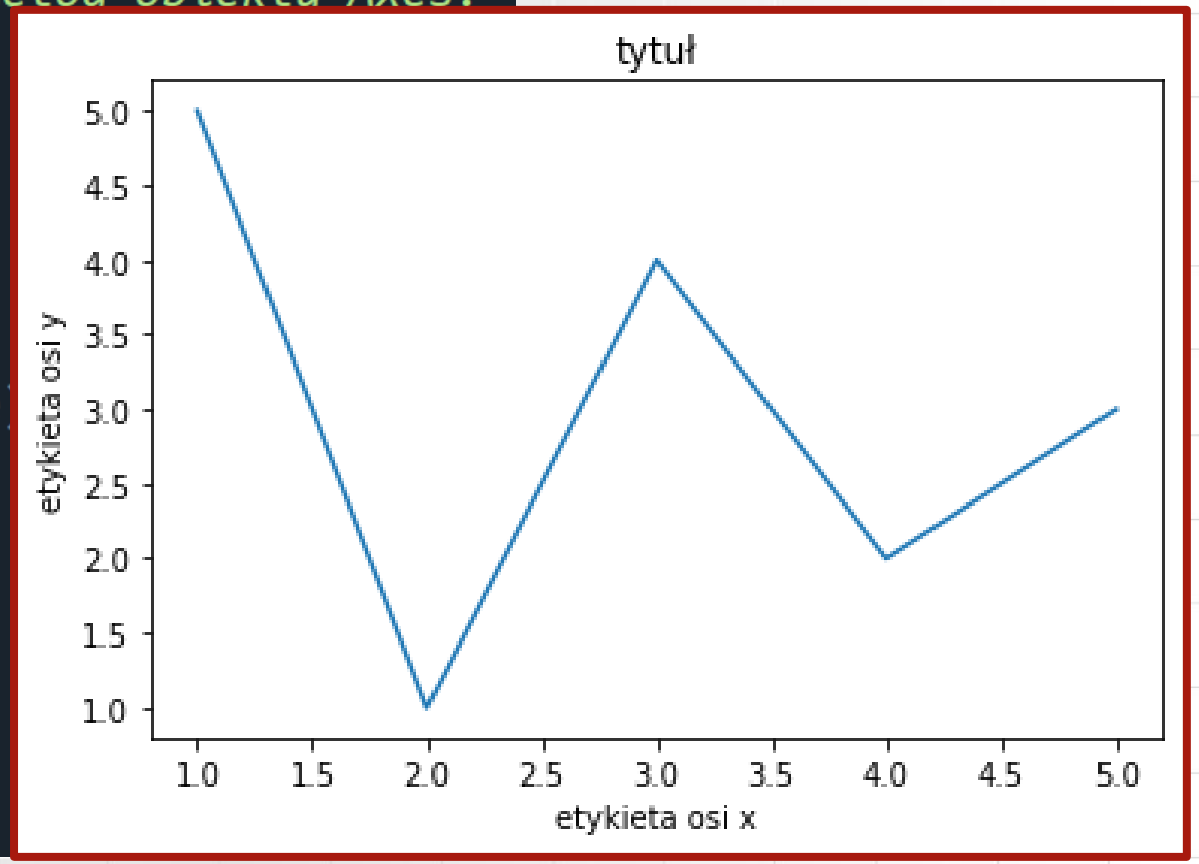
```python
# -*- coding: utf-8 -*-
"""
Przykład pokazujący wykorzystanie metod obiektu Axes.
"""

#import matplotlib as mpl
import matplotlib.pyplot as plt
#import numpy as np

#utworzenie obiektu wykresu (figure)
fig, ax = plt.subplots()
x = [1, 2, 3, 4, 5]
y = [5, 1, 4, 2, 3]
ax.plot(x, y)
ax.set_title('tytuł')
ax.set_xlabel('etykieta osi x')
ax.set_ylabel('etykieta osi y')
```

Politechnika Wrocławska

Klasa Axes

matplotlib_03.py

```python
# -*- coding: utf-8 -*-
"""
Przykład pokazujący wykorzystanie metod obiektu Axes.
"""

#import matplotlib as mpl
import matplotlib.pyplot as plt
#import numpy as np

#utworzenie obiektu wykresu (figure)
fig, ax = plt.subplots()
x = [1, 2, 3, 4, 5]
y = [5, 1, 4, 2, 3]
ax.plot(x, y)
ax.set_title('tytuł')
ax.set_xlabel('etykieta osi x')
ax.set_ylabel('etykieta osi y')
```



Politechnika Wrocławska

Biblioteka matplotlib

Przydatne klasy:

- Figure
- Axes
- Axis
- Locator
- Formatter
- Artist

Politechnika Wrocławska

Biblioteka matplotlib

matplotlib_04a.py matplotlib_04b.py

```python
# -*- coding: utf-8 -*-
"""
Przykład pokazujący podejście obiektowe i proceduralne.
"""

#import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy as np

# przygotowanie danych
x = np.linspace(-10,10,100)
y1 = 5*np.sin(x)/x
y2 = x*np.cos(x)

# stworzenie wykresu (podejście obiektowe)
# fig - obiekt wykresu
# ax - obiekt układu współrzędnych
fig, ax = plt.subplots()
# wywołanie metody plot() obiektu ax
ax.plot(x, y1)
ax.plot(x, y2)
```

Politechnika Wrocławska

Biblioteka matplotlib

matplotlib_04a.py matplotlib_04b.py

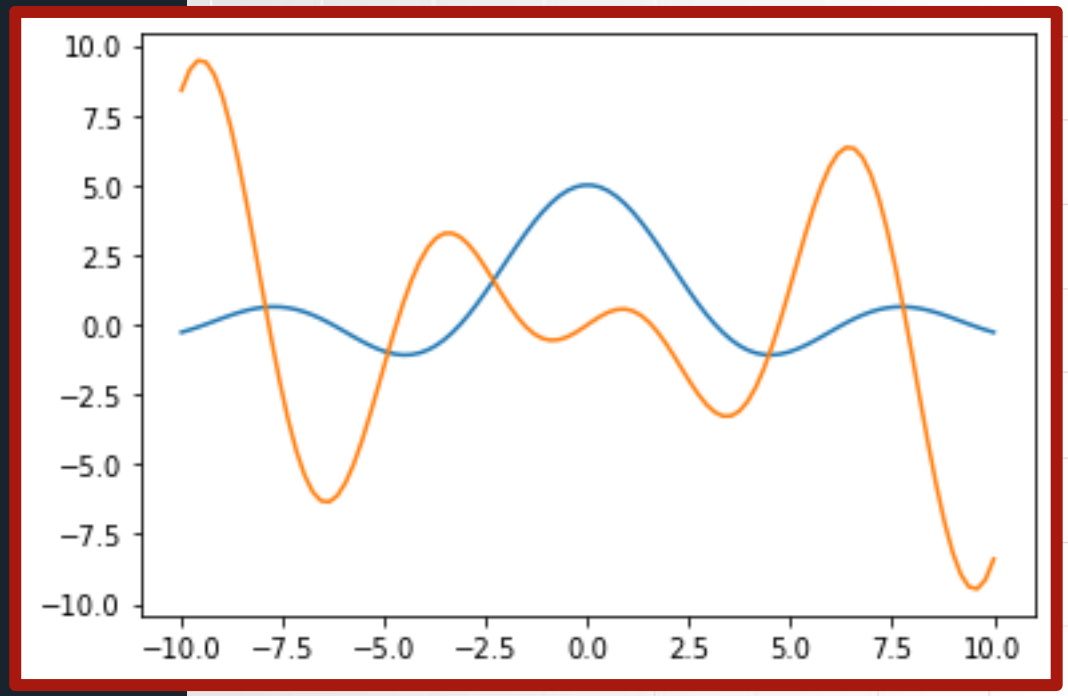
```
1  # -*- coding: utf-8 -*-
2  """
3  Przykład pokazujący podejście obiektowe i proceduralne.
4  """
5
6  #import matplotlib as mpl
7  import matplotlib.pyplot as plt
8  import numpy as np
9
10 # przygotowanie danych
11 x = np.linspace(-10,10,100)
12 y1 = 5*np.sin(x)/x
13 y2 = x*np.cos(x)
14
15 # stworzenie wykresu (podejście proceduralne)
16 # wywołanie funkcji figure() tworzącej wykres
17 plt.figure()
18 # wywołanie funkcji plot() tworzącej obiekt linii
19 # w osiach wykresu
20 plt.plot(x, y1)
21 plt.plot(x, y2)
```

Politechnika Wrocławska

Biblioteka matplotlib

matplotlib_04a.py matplotlib_04b.py

```python
# -*- coding: utf-8 -*-
"""
Przykład pokazujący podejście obiektowe i proceduralne.
"""

#import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy as np

# przygotowanie danych
x = np.linspace(-10,10,100)
y1 = 5*np.sin(x)/x
y2 = x*np.cos(x)

# stworzenie wykresu (podejście proceduralne)
# wywołanie funkcji figure() tworzącej wykres
plt.figure()
# wywołanie funkcji plot() tworzącej obiekt linii
# w osiach wykresu
plt.plot(x, y1)
plt.plot(x, y2)
```



Politechnika Wrocławska

Biblioteka matplotlib

```
15  # stworzenie wykresu (podejście obiektowe)
16  # fig - obiekt wykresu
17  # ax - obiekt układu współrzędnych
18  fig, ax = plt.subplots()
19  # wywołanie metody plot() obiektu ax
20  ax.plot(x, y1)
21  ax.plot(x, y2)
```

```
15  # stworzenie wykresu (podejście proceduralne)
16  # wywołanie funkcji figure() tworzącej wykres
17  plt.figure()
18  # wywołanie funkcji plot() tworzącej obiekt linii
19  # w osiach wykresu
20  plt.plot(x, y1)
21  plt.plot(x, y2)
```

Politechnika Wrocławska

Wykres - scatter

matplotlib_05.py

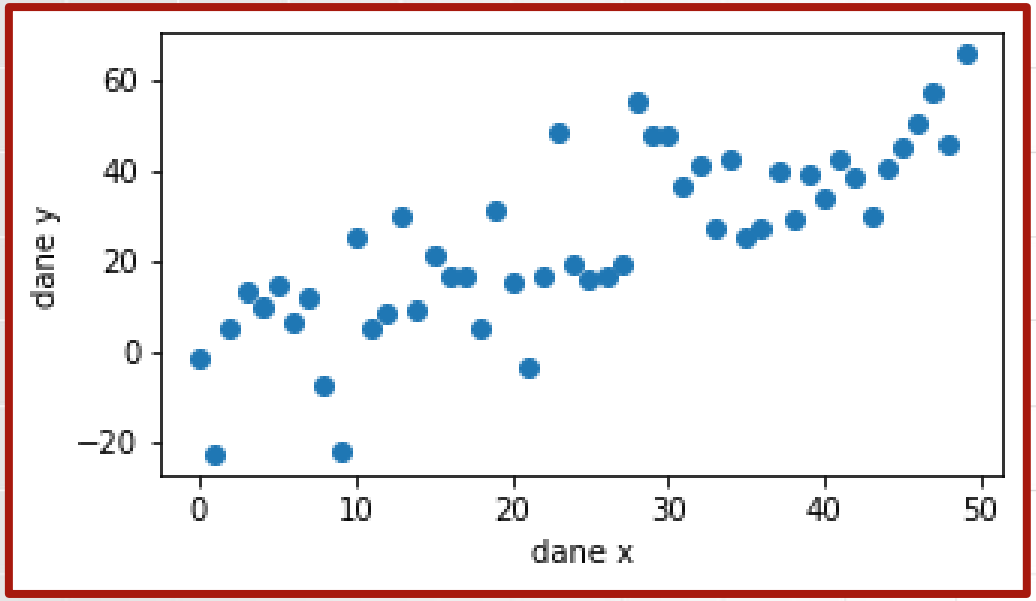
```python
# -*- coding: utf-8 -*-
"""
Przykład pokazujący wykres typu scatter.
Dane do wykresu przekazane jako słownik.
"""

#import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy as np

# przygotowanie danych w postaci słownika
# zawierającego tablice ndarray
data = {'x': np.arange(50) }
data['y'] = data['x'] + 10 * np.random.randn(50)

# utworzenie wykresu
fig, ax = plt.subplots(figsize=(5, 2.7))
# wykres typu scatter z danych w słowniku data
ax.scatter('x', 'y', data=data)
# etykiety osi
ax.set_xlabel('dane x')
ax.set_ylabel('dane y')
```

Politechnika Wrocławska

Wykres - scatter

matplotlib_05.py

```python
# -*- coding: utf-8 -*-
"""
Przykład pokazujący wykres typu scatter.
Dane do wykresu przekazane jako słownik.
"""

#import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy as np

# przygotowanie danych w postaci słownika
# zawierającego tablice ndarray
data = {'x': np.arange(50) }
data['y'] = data['x'] + 10 * np.random.randn(50)

# utworzenie wykresu
fig, ax = plt.subplots(figsize=(5, 2.7))
# wykres typu scatter z danych w słowniku data
ax.scatter('x', 'y', data=data)
# etykiety osi
ax.set_xlabel('dane x')
ax.set_ylabel('dane y')
```



Politechnika Wrocławska

Wykres - scatter

matplotlib_06.py

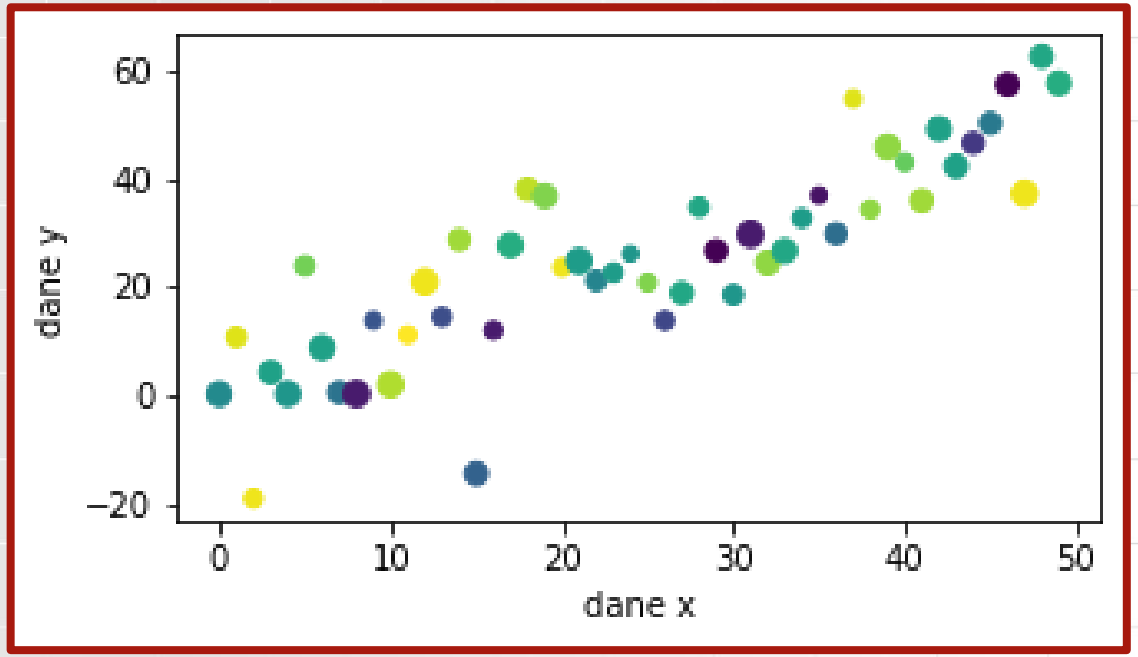
```python
# -*- coding: utf-8 -*-
"""
Przykład pokazujący wykres typu scatter.
Dane do wykresu przekazane jako słownik.
Punkty mają różne rozmiary oraz kolory.
"""

#import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy as np

# przygotowanie danych w postaci słownika
# zawierającego tablice ndarray
data = {'x': np.arange(50) }
data['y'] = data['x'] + 10 * np.random.randn(50)
data['s'] = np.random.randint(20, 60, 50)
data['c'] = np.random.randint(0, 50, 50)

# utworzenie wykresu
fig, ax = plt.subplots(figsize=(5, 2.7))
# argument c - pozwala przekazać kolor znacznika
# argument s - pozwala przekazać rozmiar znacznika
ax.scatter('x', 'y', c='c', s='s', data=data)
ax.set_xlabel('dane x')
ax.set_ylabel('dane y')
```

Politechnika Wrocławska

Wykres - scatter

matplotlib_06.py

```python
# -*- coding: utf-8 -*-
"""
Przykład pokazujący wykres typu scatter.
Dane do wykresu przekazane jako słownik.
Punkty mają różne rozmiary oraz kolory.
"""

#import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy as np

# przygotowanie danych w postaci słownika
# zawierającego tablice ndarray
data = {'x': np.arange(50) }
data['y'] = data['x'] + 10 * np.random.randn(50)
data['s'] = np.random.randint(20, 60, 50)
data['c'] = np.random.randint(0, 50, 50)

# utworzenie wykresu
fig, ax = plt.subplots(figsize=(5, 2.7))
# argument c - pozwala przekazać kolor znacznika
# argument s - pozwala przekazać rozmiar znacznika
ax.scatter('x', 'y', c='c', s='s', data=data)
ax.set_xlabel('dane x')
ax.set_ylabel('dane y')
```



Politechnika Wrocławska

Wykres - plot

matplotlib_07.py

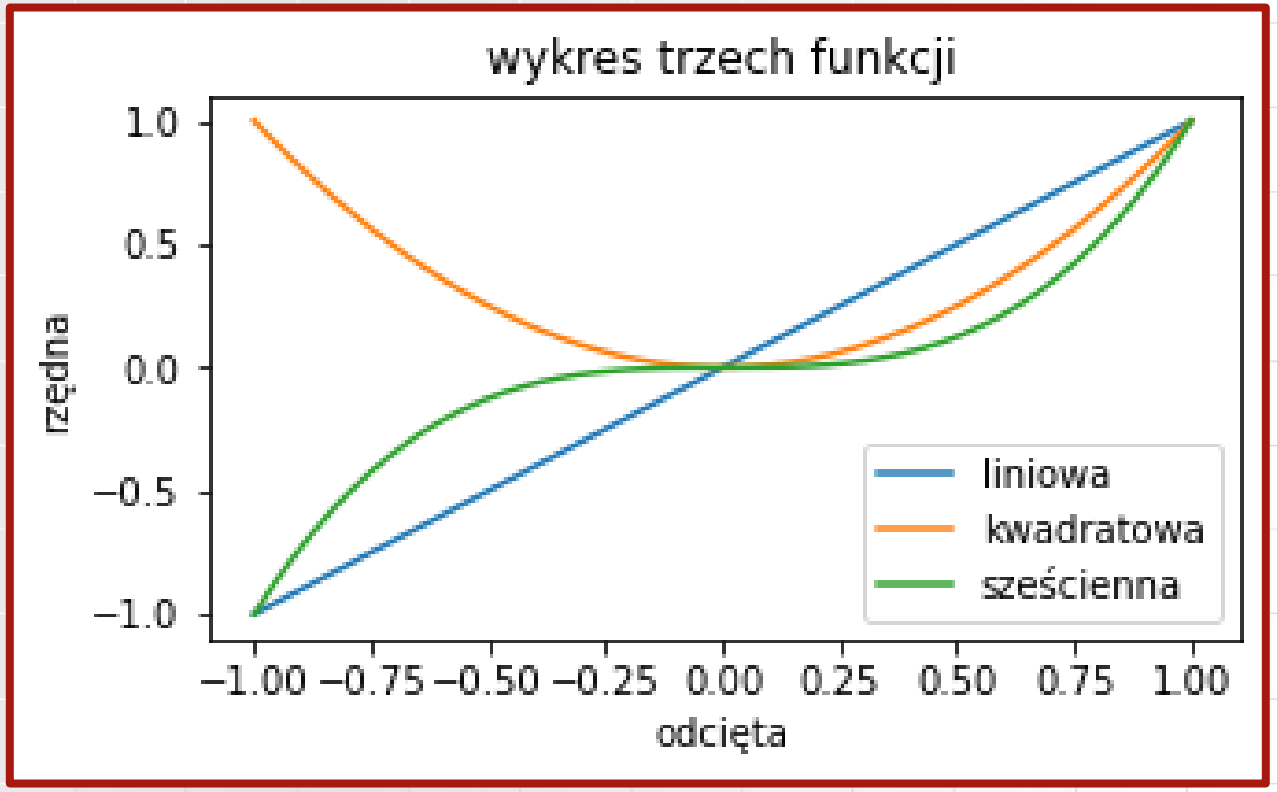
```python
# -*- coding: utf-8 -*-
"""
Przykład pokazujący wykres zawierający trzy
serie danych oraz legendę.
"""

#import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy as np

# przygotowanie wektora argumentów (odciętych)
x = np.linspace(-1,1)

# utworzenie wykresu
fig, ax = plt.subplots(figsize=(5, 2.7))
# dodanie funkcji liniowej do wykresu
ax.plot(x, x, label="liniowa")
# dodanie funkcji kwadratowej
ax.plot(x, x**2, label="kwadratowa")
# dodanie funkcji sześciennej
ax.plot(x, x**3, label="sześcienna")

ax.set_title('wykres trzech funkcji')
ax.set_xlabel('odcięta')
ax.set_ylabel('rzędna')
# umieszczenie legendy
ax.legend()
```

Politechnika Wrocławska

Wykres - plot

matplotlib_07.py

```
# -*- coding: utf-8 -*-
"""
Przykład pokazujący wykres zawierający trzy
serie danych oraz legendę.
"""

#import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy as np

# przygotowanie wektora argumentów (odciętych)
x = np.linspace(-1,1)

# utworzenie wykresu
fig, ax = plt.subplots(figsize=(5, 2.7))
# dodanie funkcji liniowej do wykresu
ax.plot(x, x, label="liniowa")
# dodanie funkcji kwadratowej
ax.plot(x, x**2, label="kwadratowa")
# dodanie funkcji sześciennej
ax.plot(x, x**3, label="sześcienna")

ax.set_title('wykres trzech funkcji')
ax.set_xlabel('odcięta')
ax.set_ylabel('rzędna')
# umieszczenie legendy
ax.legend()
```



Politechnika Wrocławska

Tworzenie wykresu w funkcji

matplotlib_08.py

```python
# -*- coding: utf-8 -*-
"""
Przykład pokazujący wykorzystanie funkcji pomocniczej
do tworzenia wykresu.
"""

#import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy as np

def plotter(ax, data_x, data_y, params):
    """
    Funkcja pomocnicza do tworzenia wykresów.
    """
    out = ax.plot(data_x, data_y, **params)
    return out
```

Politechnika Wrocławska

Tworzenie wykresu w funkcji

matplotlib_08.py

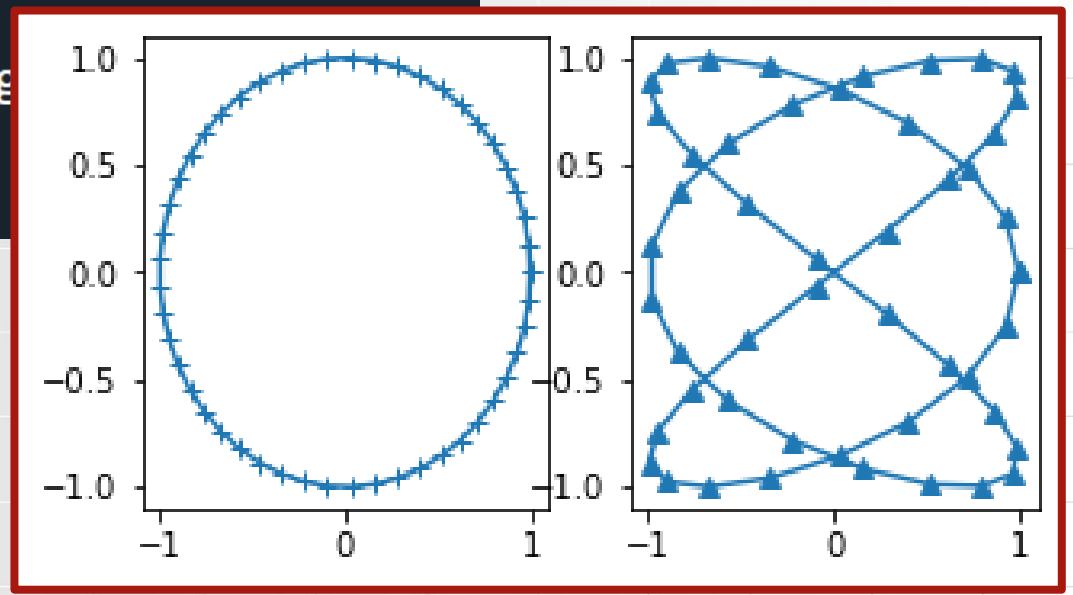
```
18  # przygotowanie danych wejściowych
19  phi = np.linspace(0,2*np.pi)
20  x1 = np.cos(phi)
21  y1 = np.sin(phi)
22  x2 = np.cos(3*phi)
23  y2 = np.sin(2*phi)
24
25  # utworzenie wykresu
26  fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(5, 2.7))
27  plotter(ax1, x1, y1, {"marker": "+"})
28  plotter(ax2, x2, y2, {"marker": "^"})
```

Politechnika Wrocławska

Tworzenie wykresu w funkcji

matplotlib_08.py

```
18  # przygotowanie danych wejściowych
19  phi = np.linspace(0,2*np.pi)
20  x1 = np.cos(phi)
21  y1 = np.sin(phi)
22  x2 = np.cos(3*phi)
23  y2 = np.sin(2*phi)
24
25  # utworzenie wykresu
26  fig, (ax1, ax2) = plt.subplots(1, 2, fig
27  plotter(ax1, x1, y1, {"marker": "+"})
28  plotter(ax2, x2, y2, {"marker": "^"})
```



Politechnika Wrocławska

Właściwości wykresów

matplotlib_09.py

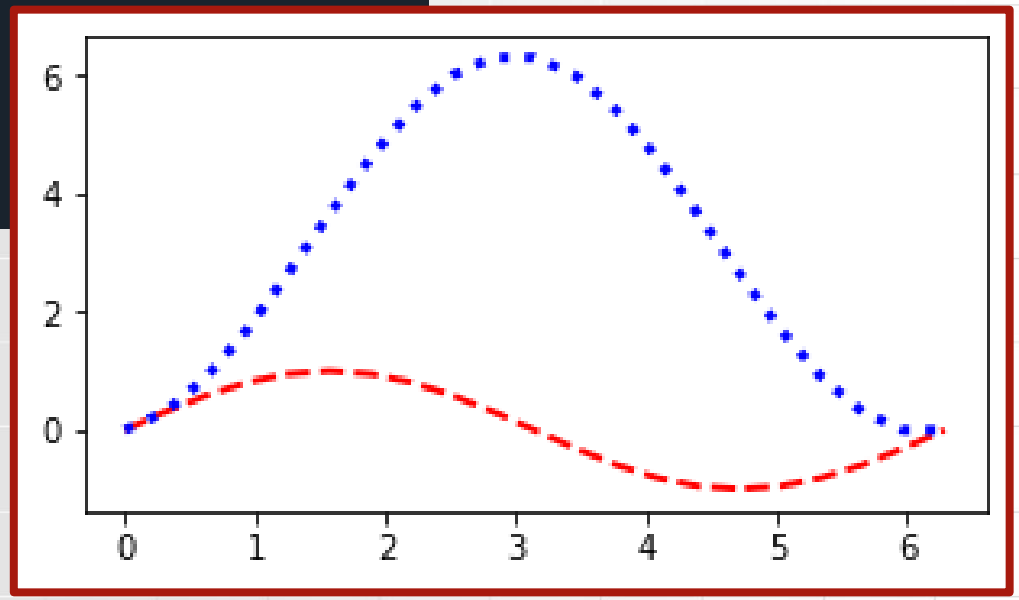
```
1  # -*- coding: utf-8 -*-
2  """
3  Przykład pokazujący różne sposoby
4  modyfikowania właściwości serii danych.
5  """
6
7  #import matplotlib as mpl
8  import matplotlib.pyplot as plt
9  import numpy as np
10
11 # przygotowanie danych wejciowych
12 phi = np.linspace(0,2*np.pi,21)
13 x = np.sin(phi)
14 y = np.cumsum(x)
```

Politechnika Wrocławska

Właściwości wykresów

matplotlib_09.py

```
16  # utworzenie wykresu
17  fig, ax = plt.subplots(figsize=(5, 2.7))
18  # wykres pierwszej serii danych
19  # ustalone kolor, styl i szerokość linii
20  ax.plot(phi, x, color="red", linestyle="--", linewidth=2)
21  # wykres drugiej serii danych
22  l2, = ax.plot(phi, y)
23  # ustalone kolor, styl i szerokość linii
24  l2.set_color("blue")
25  l2.set_linestyle(":")
26  l2.set_linewidth(3)
```



Politechnika Wrocławska

Kolory w bibliotece matplotlib

matplotlib_10.py

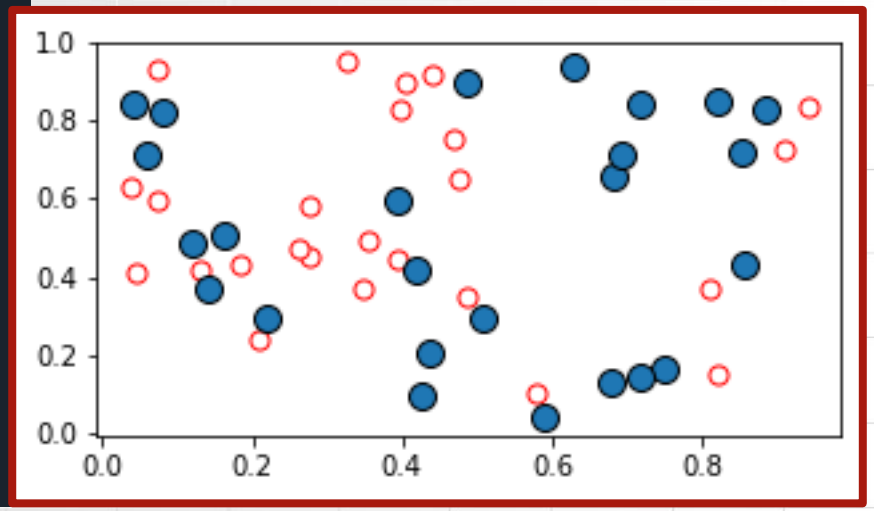
```python
# -*- coding: utf-8 -*-
"""
Przykład pokazujący różne sposoby
definiowania kolorów.
"""

#import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy as np

# przygotowanie danych wejciowych
x1, y1, x2, y2 = np.random.rand(4, 25)

# utworzenie wykresu
fig, ax = plt.subplots(figsize=(5, 2.7))
# kolor wypełnienia znaczników - biały
# kolor krawędzi znaczników - czerwony
ax.scatter(x1, y1, s=50, facecolor="white", edgecolor="r")
# kolor wypełnienia znaczników - niebieski
# kolor krawędzi znaczników - czarny
ax.scatter(x2, y2, s=100, facecolor="C0", edgecolor=(0., 0., 0.))
# więcej o nazywaniu kolorów
# https://matplotlib.org/stable/tutorials/colors/colors.html
```

Politechnika Wrocławska

Kolory w bibliotece matplotlib

matplotlib_10.py

```python
# -*- coding: utf-8 -*-
"""
Przykład pokazujący różne sposoby
definiowania kolorów.
"""

#import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy as np

# przygotowanie danych wejciowych
x1, y1, x2, y2 = np.random.rand(4, 25)

# utworzenie wykresu
fig, ax = plt.subplots(figsize=(5, 2.7))
# kolor wypełnienia znaczników - biały
# kolor krawędzi znaczników - czerwony
ax.scatter(x1, y1, s=50, facecolor="white", edgecolor="r")
# kolor wypełnienia znaczników - niebieski
# kolor krawędzi znaczników - czarny
ax.scatter(x2, y2, s=100, facecolor="C0", edgecolor=(0., 0., 0.))
# więcej o nazywaniu kolorów
# https://matplotlib.org/stable/tutorials/colors/colors.html
```



Politechnika Wrocławska

Opisy na wykresach

matplotlib_11.py

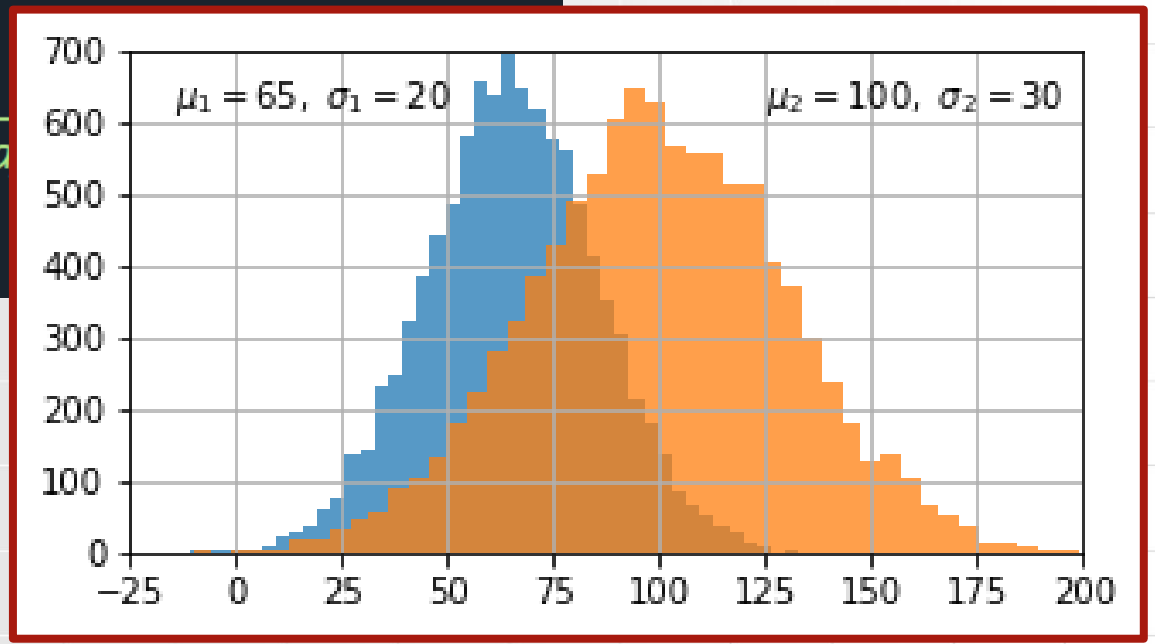
```
# -*- coding: utf-8 -*-
"""
Przykład pokazujący możlwość wprowadzania
oznaczeń tekstowych na wykresie.

"""

#import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy as np

# przygotowanie dwóch zestawów danych losowych
mu1, sigma1 = 65, 20
x1 = mu1 + sigma1 * np.random.randn(10000)
mu2, sigma2 = 100, 30
x2 = mu2 + sigma2 * np.random.randn(10000)
```

Politechnika Wrocławska

Opisy na wykresach

matplotlib_11.py

```
18  # utworzenie wykresu
19  fig, ax = plt.subplots(figsize=(5, 2.7))
20  # zawierającego dwa histogramy
21  ax.hist(x1, 50, alpha=0.75) # kanał alpha - przezroczystość
22  ax.hist(x2, 50, alpha=0.75)
23  # dobranie zakresów osi
24  ax.axis([-25, 200, 0, 700])
25  # umieszczenie opisów histogramów
26  ax.text(-15, 620, r"$\mu_1=65,\ \sigma_
27  ax.text(125, 620, r"$\mu_2=100,\ \sigma
28  # dodanie siatki na wykresie
29  ax.grid(True)
```



Politechnika Wrocławska

Skale wykresów

matplotlib_12.py

```
1  # -*- coding: utf-8 -*-
2  """
3  Przykład pokazujący wykorzystanie skali logarytmicznej.
4
5  """
6
7  #import matplotlib as mpl
8  import matplotlib.pyplot as plt
9  import numpy as np
10
11 # przygotowanie danych losowych
12 mu1, sigma1 = 1, 1
13 y = mu1 + sigma1 * np.random.randn(50)
14 z = 10**y
15 x = np.arange(len(y))
```

Politechnika Wrocławska

Skale wykresów

matplotlib_12.py

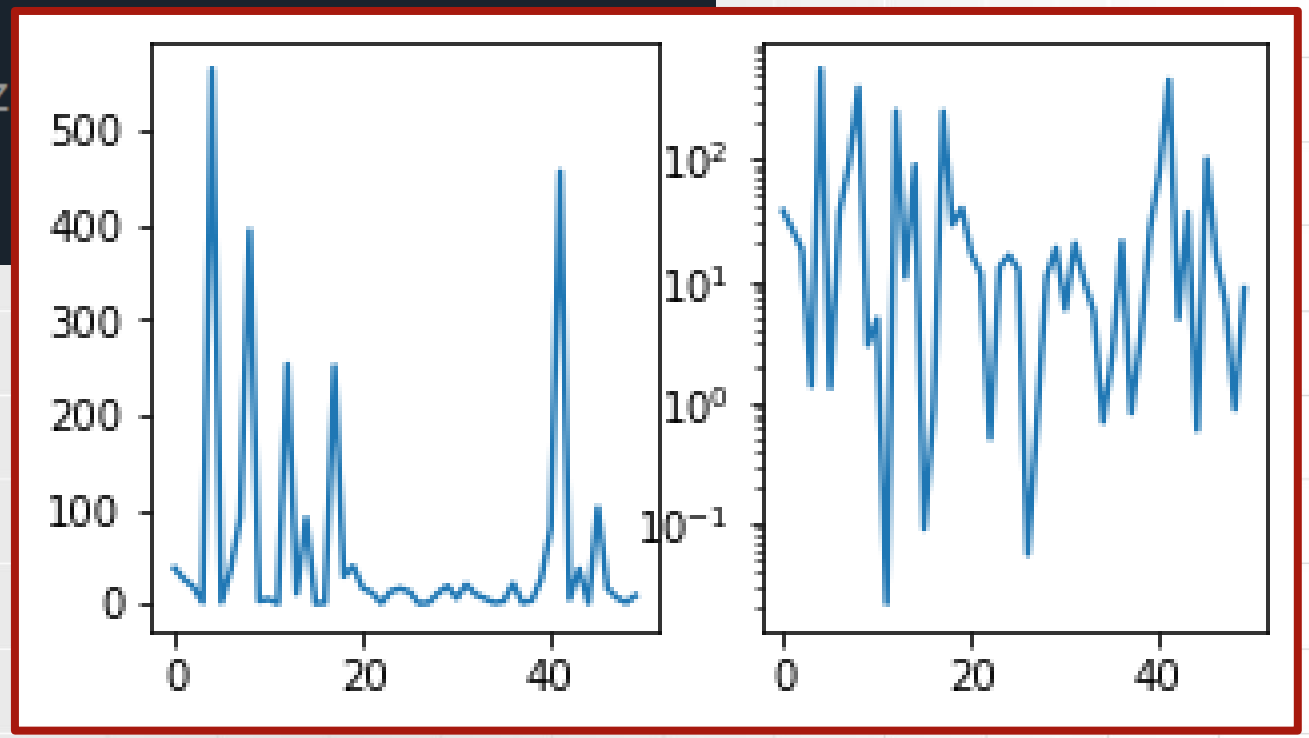
```
17  # utworzenie wykresu o dwóch układach współrzędnych
18  fig, axs = plt.subplots(1, 2, figsize=(5, 2.7))
19  # wykres w skali liniowej
20  axs[0].plot(x,z)
21
22  # wykres w skali logarytmicznej
23  axs[1].plot(x,z)
24  axs[1].set_yscale('log')
```

Politechnika Wrocławska

Skale wykresów

matplotlib_12.py

```
17  # utworzenie wykresu o dwóch układach współrzędnych
18  fig, axs = plt.subplots(1, 2, figsize=(5, 2.7))
19  # wykres w skali liniowej
20  axs[0].plot(x,z)
21
22  # wykres w skali logarytmicz
23  axs[1].plot(x,z)
24  axs[1].set_yscale('log')
```



Politechnika Wrocławska

Podwójne osie

matplotlib_13.py

```
1  # -*- coding: utf-8 -*-
2  """
3  Przykład pokazujący umieszczanie dodatkowych osi
4  w układzie współrzędnych.
5
6  """
7
8  #import matplotlib as mpl
9  import matplotlib.pyplot as plt
10 import numpy as np
11
12 t = np.linspace(0, 6*np.pi, 101)
13 s = np.sin(t)
```

Politechnika Wrocławska

Podwójne osie

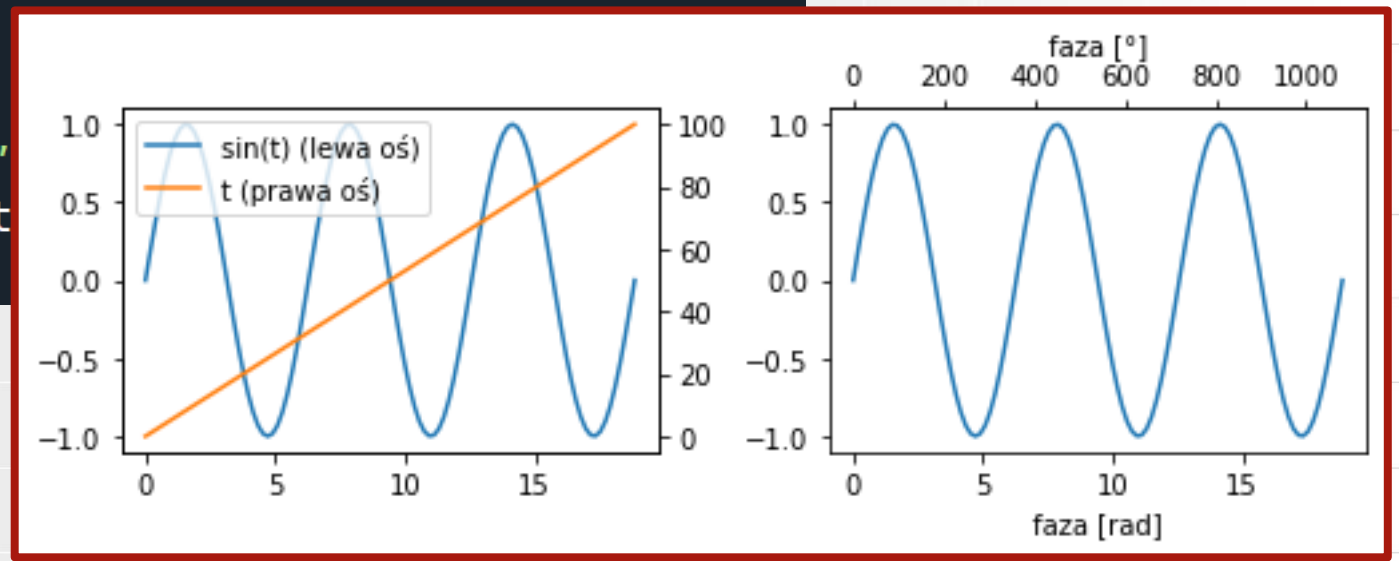
matplotlib_13.py

```
15  fig, (ax1, ax3) = plt.subplots(1,
16                                 2,
17                                 figsize=(7, 2.7),
18                                 constrained_layout=True)
19  l1, = ax1.plot(t, s)
20  ax2 = ax1.twinx()
21  l2, = ax2.plot(t, range(len(t)), 'C1')
22  ax2.legend([l1, l2], ['sin(t) (lewa oś)', 't (prawa oś)'])
23
24  ax3.plot(t, s)
25  ax3.set_xlabel('faza [rad]')
26  ax4 = ax3.secondary_xaxis('top',
27                            functions=(np.rad2deg, np.deg2rad))
28  ax4.set_xlabel('faza [°]')
```

Podwójne osie

matplotlib_13.py

```
15  fig, (ax1, ax3) = plt.subplots(1,
16                                 2,
17                                 figsize=(7, 2.7),
18                                 constrained_layout=True)
19  l1, = ax1.plot(t, s)
20  ax2 = ax1.twinx()
21  l2, = ax2.plot(t, range(len(t)), 'C1')
22  ax2.legend([l1, l2], ['sin(t) (lewa oś)', 't (prawa oś)'])
23
24  ax3.plot(t, s)
25  ax3.set_xlabel('faza [rad]')
26  ax4 = ax3.secondary_xaxis('top'
27                            funct
28  ax4.set_xlabel('faza [°]')
```



Politechnika Wrocławska

Wykresy trójwymiarowe

matplotlib_14.py

```python
# -*- coding: utf-8 -*-
"""
Przykład pokazujący tworzenie wykresów trójwymiarowych.

"""

#import matplotlib as mpl
import matplotlib.pyplot as plt
import numpy as np

# dane do wykkresów
x = np.linspace(-10, 10, 100) # współrzędne x
y = np.linspace(-10, 10, 100) # współrzędne y
X, Y = np.meshgrid(x, y) # siatka współrzędnych X, Y
R = (X**2 + Y**2)**0.5   # obliczone R
Z = np.sin(R)/R          # sinc(r)
```

Politechnika Wrocławska

Wykresy trójwymiarowe

matplotlib_14.py

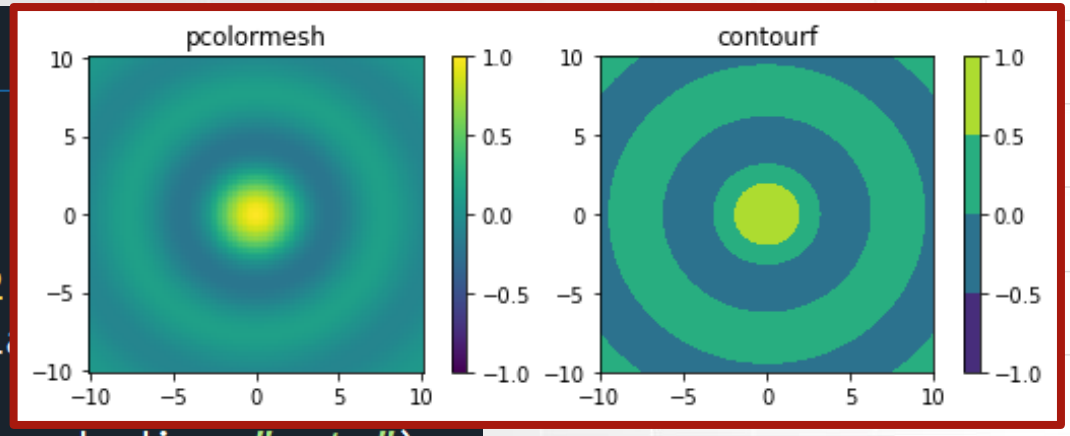
```
18  # przykładowe wykresy 2d
19  fig, (ax1, ax2) = plt.subplots(1,
20                                 2,
21                                 figsize=(7, 2.7),
22                                 constrained_layout=True)
23
24  pc = ax1.pcolormesh(X, Y, Z, vmin=-1, vmax=1, shading="auto")
25  fig.colorbar(pc, ax=ax1)
26  ax1.set_title('pcolormesh')
27
28  co = ax2.contourf(X, Y, Z, levels=np.linspace(-1,1,5))
29  fig.colorbar(co, ax=ax2)
30  ax2.set_title('contourf')
31
32  # przykładowy wykres 3d
33  fig, ax3 = plt.subplots(1, subplot_kw = {'projection': '3d'})
34  ax3.plot_surface(X,Y,Z)
35  ax3.set_title('plot_surface')
```

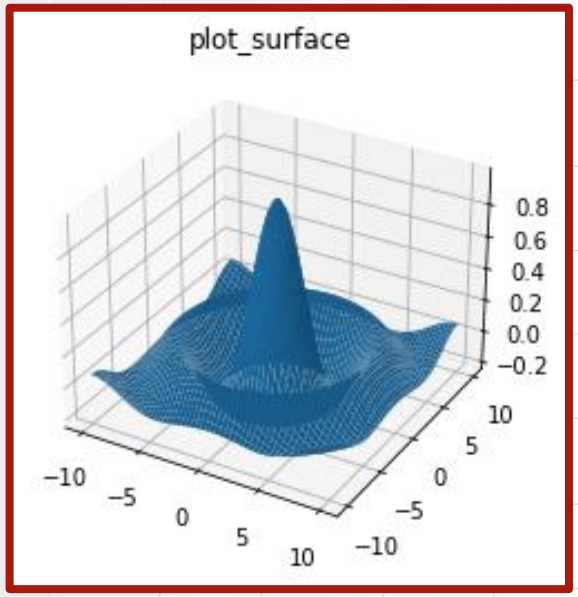
Politechnika Wrocławska

Wykresy trójwymiarowe

matplotlib_14.py

```
18  # przykładowe wykresy 2d
19  fig, (ax1, ax2) = plt.subplots(1,
20                                 2,
21                                 figsize=(7, 2
22                                 constrained_l
23
24  pc = ax1.pcolormesh(X, Y, Z, vmin=-1, vmax=1, shading="auto")
25  fig.colorbar(pc, ax=ax1)
26  ax1.set_title('pcolormesh')
27
28  co = ax2.contourf(X, Y, Z, levels=np.linspace(-1,1,5))
29  fig.colorbar(co, ax=ax2)
30  ax2.set_title('contourf')
31
32  # przykładowy wykres 3d
33  fig, ax3 = plt.subplots(1, subplot_kw = {'projection': '3d'})
34  ax3.plot_surface(X,Y,Z)
35  ax3.set_title('plot_surface')
```





Politechnika Wrocławska

Podsumowanie

Wprowadzenie w możliwości bilioteki matplotlib