Laboratorium 2

Biblioteka Pandas

```
import pandas as pd
dataframe = pd.read_csv('https://marcingabryel.pl/ai/iris.csv')
```

```
dataframe.head(10)
```

	sepal.length	sepal.width	petal.length	petal.width	variety
0	5.1	3.5	1.4	0.2	Setosa
1	4.9	3.0	1.4	0.2	Setosa
2	4.7	3.2	1.3	0.2	Setosa
3	4.6	3.1	1.5	0.2	Setosa
4	5.0	3.6	1.4	0.2	Setosa
5	5.4	3.9	1.7	0.4	Setosa
6	4.6	3.4	1.4	0.3	Setosa
7	5.0	3.4	1.5	0.2	Setosa
8	4.4	2.9	1.4	0.2	Setosa
9	4.9	3.1	1.5	0.1	Setosa

Kolejne kroki: Wygeneruj kod za pomocą zmiennej `dataframe` | Wyświetl polecane wykresy | New interactive sheet

```
dataframe['variety'].unique()
```

```
array(['Setosa', 'Versicolor', 'Virginica'], dtype=object)
```

```
dataframe[dataframe['variety'] == 'Versicolor']
```

	sepal.length	sepal.width	petal.length	petal.width	variety
50	7.0	3.2	4.7	1.4	Versicolor
51	6.4	3.2	4.5	1.5	Versicolor
52	6.9	3.1	4.9	1.5	Versicolor
53	5.5	2.3	4.0	1.3	Versicolor
54	6.5	2.8	4.6	1.5	Versicolor
55	5.7	2.8	4.5	1.3	Versicolor
56	6.3	3.3	4.7	1.6	Versicolor
57	4.9	2.4	3.3	1.0	Versicolor
58	6.6	2.9	4.6	1.3	Versicolor
59	5.2	2.7	3.9	1.4	Versicolor
60	5.0	2.0	3.5	1.0	Versicolor
61	5.9	3.0	4.2	1.5	Versicolor
62	6.0	2.2	4.0	1.0	Versicolor
63	6.1	2.9	4.7	1.4	Versicolor
64	5.6	2.9	3.6	1.3	Versicolor
65	6.7	3.1	4.4	1.4	Versicolor
66	5.6	3.0	4.5	1.5	Versicolor
67	5.8	2.7	4.1	1.0	Versicolor
68	6.2	2.2	4.5	1.5	Versicolor
69	5.6	2.5	3.9	1.1	Versicolor
70	5.9	3.2	4.8	1.8	Versicolor
71	6.1	2.8	4.0	1.3	Versicolor
72	6.3	2.5	4.9	1.5	Versicolor
73	6.1	2.8	4.7	1.2	Versicolor
74	6.4	2.9	4.3	1.3	Versicolor
75	6.6	3.0	4.4	1.4	Versicolor
76	6.8	2.8	4.8	1.4	Versicolor
77	6.7	3.0	5.0	1.7	Versicolor
78	6.0	2.9	4.5	1.5	Versicolor
79	5.7	2.6	3.5	1.0	Versicolor
80	5.5	2.4	3.8	1.1	Versicolor
81	5.5	2.4	3.7	1.0	Versicolor
82	5.8	2.7	3.9	1.2	Versicolor
83	6.0	2.7	5.1	1.6	Versicolor
84	5.4	3.0	4.5	1.5	Versicolor
85	6.0	3.4	4.5	1.6	Versicolor
86	6.7	3.1	4.7	1.5	Versicolor
87	6.3	2.3	4.4	1.3	Versicolor
88	5.6	3.0	4.1	1.3	Versicolor
89	5.5	2.5	4.0	1.3	Versicolor
90	5.5	2.6	4.4	1.2	Versicolor
91	6.1	3.0	4.6	1.4	Versicolor
92	5.8	2.6	4.0	1.2	Versicolor
93	5.0	2.3	3.3	1.0	Versicolor
94	5.6	2.7	4.2	1.3	Versicolor
95	5.7	3.0	4.2	1.2	Versicolor
96	5.7	2.9	4.2	1.3	Versicolor
97	6.2	2.9	4.3	1.3	Versicolor
98	5.1	2.5	3.0	1.1	Versicolor

```
99           5.7          2.8           4.1          1.3  Versicolor
WARNING: Runtime no longer has a reference to this dataframe, please re-run this cell and try again.
```

```
for variety in dataframe['variety'].unique():
  print(f"5 pierwszych dla: {variety}")
  print(dataframe[dataframe['variety'] == variety].head(5))
```

```
5 pierwszych dla: Setosa
   sepal.length  sepal.width  petal.length  petal.width variety
0           5.1          3.5           1.4          0.2  Setosa
1           4.9          3.0           1.4          0.2  Setosa
2           4.7          3.2           1.3          0.2  Setosa
3           4.6          3.1           1.5          0.2  Setosa
4           5.0          3.6           1.4          0.2  Setosa
5 pierwszych dla: Versicolor
    sepal.length  sepal.width  petal.length  petal.width     variety
50           7.0          3.2           4.7          1.4  Versicolor
51           6.4          3.2           4.5          1.5  Versicolor
52           6.9          3.1           4.9          1.5  Versicolor
53           5.5          2.3           4.0          1.3  Versicolor
54           6.5          2.8           4.6          1.5  Versicolor
5 pierwszych dla: Virginica
     sepal.length  sepal.width  petal.length  petal.width    variety
100           6.3          3.3           6.0          2.5  Virginica
101           5.8          2.7           5.1          1.9  Virginica
102           7.1          3.0           5.9          2.1  Virginica
103           6.3          2.9           5.6          1.8  Virginica
104           6.5          3.0           5.8          2.2  Virginica
```

```
dataframe['sepal.length'].min()
```

```
4.3
```

```
print("Wartości maksymalne:")
print(dataframe[['sepal.length', 'sepal.width', 'petal.length', 'petal.width']].max())

print("\nWartości minimalne:")
print(dataframe[['sepal.length', 'sepal.width', 'petal.length', 'petal.width']].min())

print("\nWartości średnie:")
print(dataframe[['sepal.length', 'sepal.width', 'petal.length', 'petal.width']].mean())
```

```
Wartości maksymalne:
sepal.length    7.9
sepal.width     4.4
petal.length    6.9
petal.width     2.5
dtype: float64

Wartości minimalne:
sepal.length    4.3
sepal.width     2.0
petal.length    1.0
petal.width     0.1
dtype: float64

Wartości średnie:
sepal.length    5.843333
sepal.width     3.057333
petal.length    3.758000
petal.width     1.199333
dtype: float64
```

```
dataframe['y'] = dataframe['variety'].copy()
dataframe['y'] = dataframe['y'].replace('Setosa', 1.0).replace(['Versicolor', 'Virginica'], 0.0)
print(dataframe)
```

```
     sepal.length  sepal.width  petal.length  petal.width    variety    y
0             5.1          3.5           1.4          0.2     Setosa  1.0
1             4.9          3.0           1.4          0.2     Setosa  1.0
2             4.7          3.2           1.3          0.2     Setosa  1.0
3             4.6          3.1           1.5          0.2     Setosa  1.0
4             5.0          3.6           1.4          0.2     Setosa  1.0
..            ...          ...           ...          ...        ...  ...
145           6.7          3.0           5.2          2.3  Virginica  0.0
146           6.3          2.5           5.0          1.9  Virginica  0.0
147           6.5          3.0           5.2          2.0  Virginica  0.0
148           6.2          3.4           5.4          2.3  Virginica  0.0
149           5.9          3.0           5.1          1.8  Virginica  0.0

[150 rows x 6 columns]
<ipython-input-20-979cb3bccedc>:2: FutureWarning: Downcasting behavior in `replace` is deprecated and will be removed in a future ve
  dataframe['y'] = dataframe['y'].replace('Setosa', 1.0).replace(['Versicolor', 'Virginica'], 0.0)
```

```
X = dataframe[['sepal.length', 'sepal.width', 'petal.length', 'petal.width']]
print(X)
```

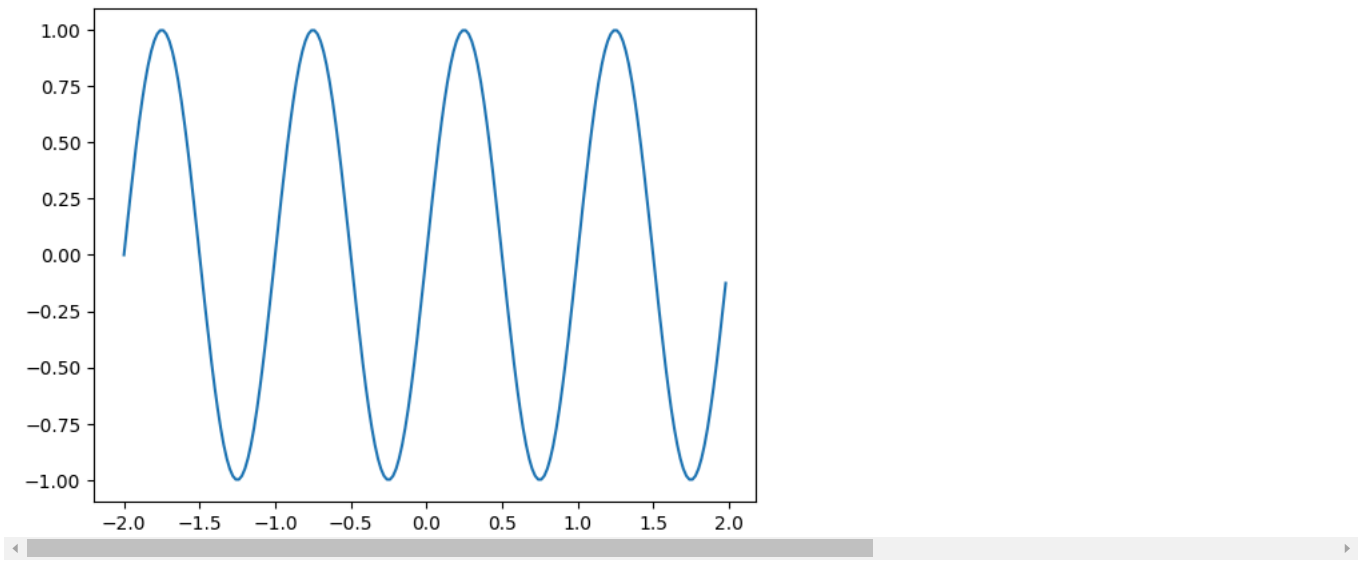
```
     sepal.length  sepal.width  petal.length  petal.width
0             5.1          3.5           1.4          0.2
1             4.9          3.0           1.4          0.2
2             4.7          3.2           1.3          0.2
3             4.6          3.1           1.5          0.2
4             5.0          3.6           1.4          0.2
..            ...          ...           ...          ...
145           6.7          3.0           5.2          2.3
146           6.3          2.5           5.0          1.9
147           6.5          3.0           5.2          2.0
148           6.2          3.4           5.4          2.3
149           5.9          3.0           5.1          1.8

[150 rows x 4 columns]
```

```
Y = dataframe['y']
print(Y)
```

```
0      1.0
1      1.0
2      1.0
3      1.0
4      1.0
      ...
145    0.0
146    0.0
147    0.0
148    0.0
149    0.0
Name: y, Length: 150, dtype: float64
```

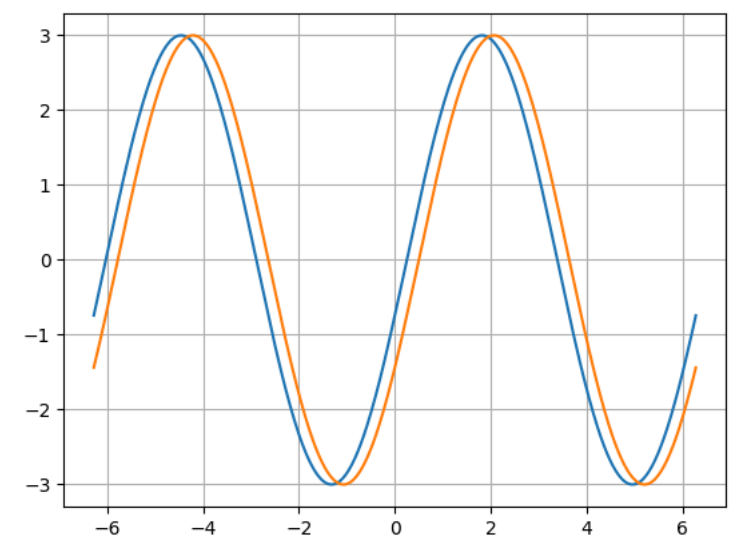
Biblioteka matplotlib

```
import matplotlib.pyplot as plt
import numpy as np
import math as m

def draw_sin():
    x = np.arange(-2.0, 2.0, 0.02)
    y = np.sin(2 * np.pi * x)
    plt.plot(x, y)

draw_sin()
```

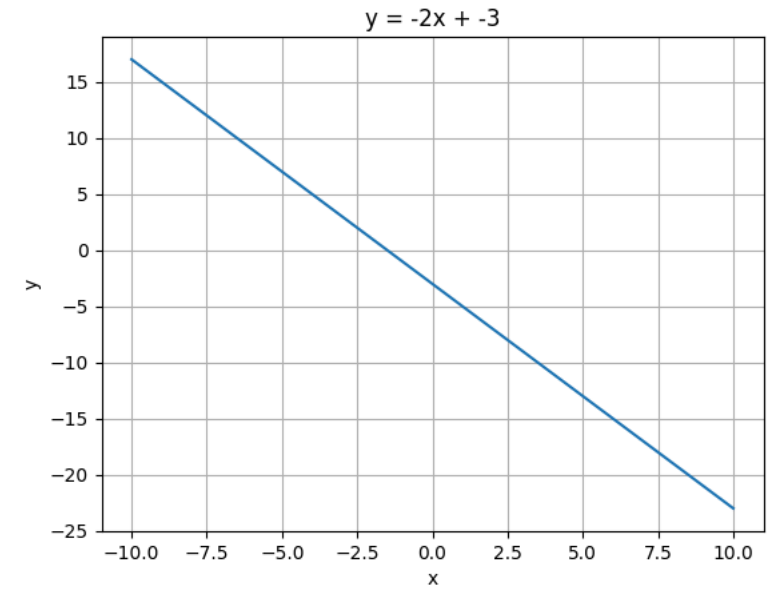



```
def draw_sin_functions(a):
  t = np.linspace(-2 * np.pi, 2 * np.pi, 400)
  y1 = a * np.sin(t - 0.25)
  y2 = a * np.sin(t - 0.5)

  plt.plot(t, y1)
  plt.plot(t, y2)
```

```
  plt.grid(True)
  plt.show()


draw_sin_functions(3)
```

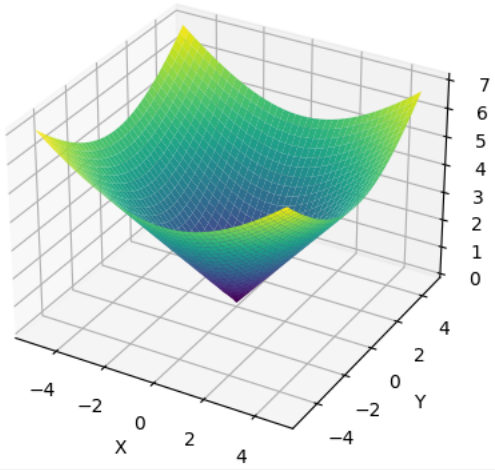



```
def draw_ab(a, b):
  x = np.linspace(-10, 10, 100)
  y = a * x + b
  plt.xlabel('x')
  plt.ylabel('y')
  plt.title(f'y = {a}x + {b}')

  plt.grid(True)
  plt.plot(x, y)


draw_ab(-2, -3)
```




```
from mpl_toolkits.mplot3d import Axes3D
import matplotlib.pyplot as plt
from matplotlib import cm
import numpy as np

def plot_surface_function():
  fig = plt.figure()
  ax = fig.add_subplot(111, projection='3d')

  x = np.arange(-5, 5, 0.25)
  y = np.arange(-5, 5, 0.25)
  X, Y = np.meshgrid(x, y)

  Z = np.sqrt(X**2 + Y**2)
```

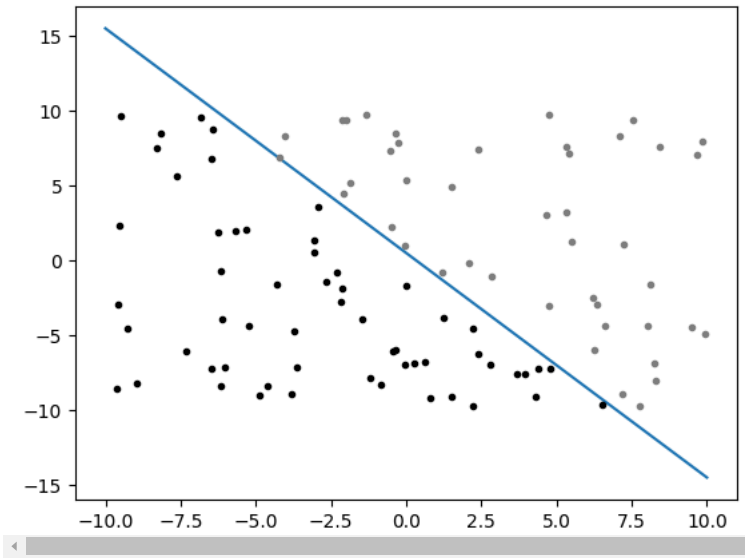
```
  surf = ax.plot_surface(X, Y, Z, cmap=cm.viridis)

  ax.set_xlabel('X')
  ax.set_ylabel('Y')
  ax.set_zlabel('Z')

  plt.show()

plot_surface_function()
```



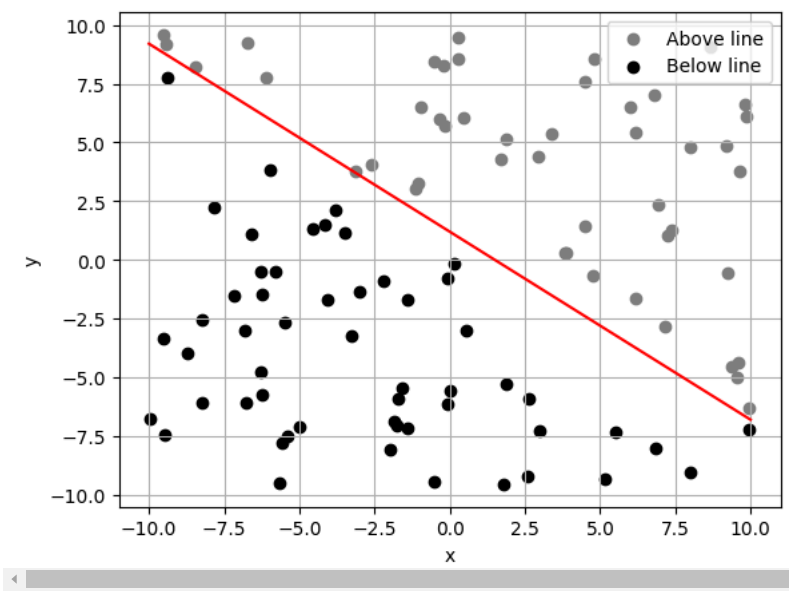
Generowanie i prezentacja danych

Dla funkcji liniowej

```
import matplotlib.pyplot as plt
import numpy as np
import random
N=100
p = np.random.random([N,2]) * 20 - 10

x = np.linspace(-10, 10, 100)
y = -1.5 * x + 0.5
plt.plot(x, y)

for points in p:
  a = points[0]
  b = points[1]
  if b > -1.5 * a + 0.5:
    plt.plot(a, b, '.', color='gray')
  else:
    plt.plot(a, b, '.', color='black')


#plt.plot(p[:,0], p[:,1],'.',color='black')
```



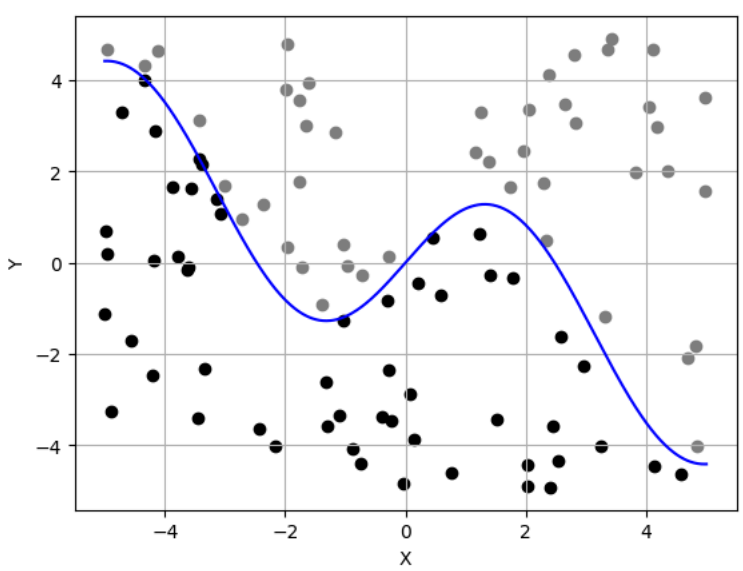

```
import matplotlib.pyplot as plt
import numpy as np

N = 100
a = -0.8
b = 1.2

p = np.random.random([N,2]) * 20 - 10

#plt.plot(p[:, 0], p[:, 1], '.', color='black')

x_vals = np.linspace(-10, 10, 100)
y_vals = a * x_vals + b
plt.plot(x_vals, y_vals, color='red')

above_line = p[:, 1] > (a * p[:, 0] + b)
below_line = p[:, 1] <= (a * p[:, 0] + b)

plt.scatter(p[above_line, 0], p[above_line, 1], color='gray', label='Above line')
plt.scatter(p[below_line, 0], p[below_line, 1], color='black', label='Below line')

plt.grid(True)
plt.xlabel('x')
plt.ylabel('y')
plt.legend()
plt.show()
```



Dla funkcji $y = 2sin(x) - x/2$

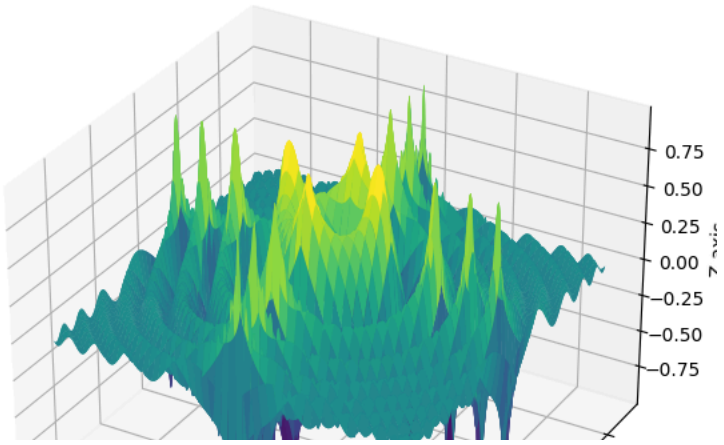
```
N = 100
x_range = (-5, 5)

p = np.random.random([N, 2]) * (x_range[1] - x_range[0]) + x_range[0]
```

```
curve_y = 2 * np.sin(p[:, 0]) - p[:, 0] / 2

d = np.zeros(N)
d[p[:,1] > curve_y] = 1

d

x_vals = np.linspace(x_range[0], x_range[1], 500)
y_vals = 2 * np.sin(x_vals) - x_vals / 2

plt.plot(x_vals, y_vals, color='blue')

plt.scatter(p[d == 1, 0], p[d == 1, 1], color='gray',)
plt.scatter(p[d == 0, 0], p[d == 0, 1], color='black')


plt.xlabel('X')
plt.ylabel('Y')
plt.grid(True)
plt.show()
```

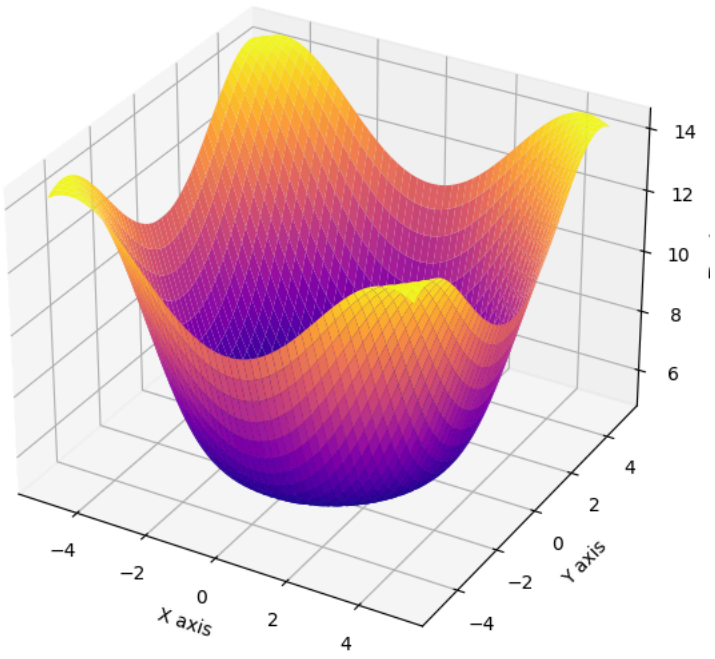


Wykresy 3D

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

x = np.linspace(-5, 5, 500)
y = np.linspace(-5, 5, 500)
x, y = np.meshgrid(x, y)

z = np.sin(x**2 + y**2) / (np.abs(x * y) + 1)

fig = plt.figure(figsize=(10, 7))
ax = fig.add_subplot(111, projection='3d')

ax.plot_surface(x, y, z, cmap='viridis')
ax.plot_surface(x, y, z, cmap='viridis')

ax.set_xlabel('X axis')
ax.set_ylabel('Y axis')
ax.set_zlabel('Z axis')

plt.show()
```




```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

x = np.linspace(-5, 5, 500)
y = np.linspace(-5, 5, 500)
x, y = np.meshgrid(x, y)

z = np.sqrt(x**2 + y**2) + 3 * np.cos(np.sqrt(x**2 + y**2)) + 5

fig = plt.figure(figsize=(10, 7))
ax = fig.add_subplot(111, projection='3d')

ax.plot_surface(x, y, z, cmap='plasma')

ax.set_xlabel('X axis')
ax.set_ylabel('Y axis')
ax.set_zlabel('Z axis')

plt.show()
```



Histogram