Języki i techniki programowania

Wykład 8

Maciej Rybczyński

Legend

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

plt.rcParams['font.size'] = 14
plt.rcParams['legend.fontsize'] = 14

x = np.linspace(0,10,50)
y1 = np.sin(x); y2 = np.cos(x); y3 = np.exp(-x)

plt.plot(x, y1, 'k-', label='sin(x)', lw=2)
plt.plot(x, y2, 'b', dashes=[20,10], label='cos(x)', lw=2)
plt.plot(x[::2], y3[::2], 'rs-', label='exp(-x)', ms=8, lw=2)
```

Co drugi element

```
plt.legend(loc='lower right',
           ncol=2, numpoints=1, labelspacing=0.5,
           handletextpad=0.5, handlelength=2.5,
           borderaxespad=0.5, borderpad=0.05,
           columnspacing=0.9, frameon=False)

plt.axis([-0.5, 10.5, -1.7, 1.5])
plt.show()
```

Zmieniaj wartości parametrów

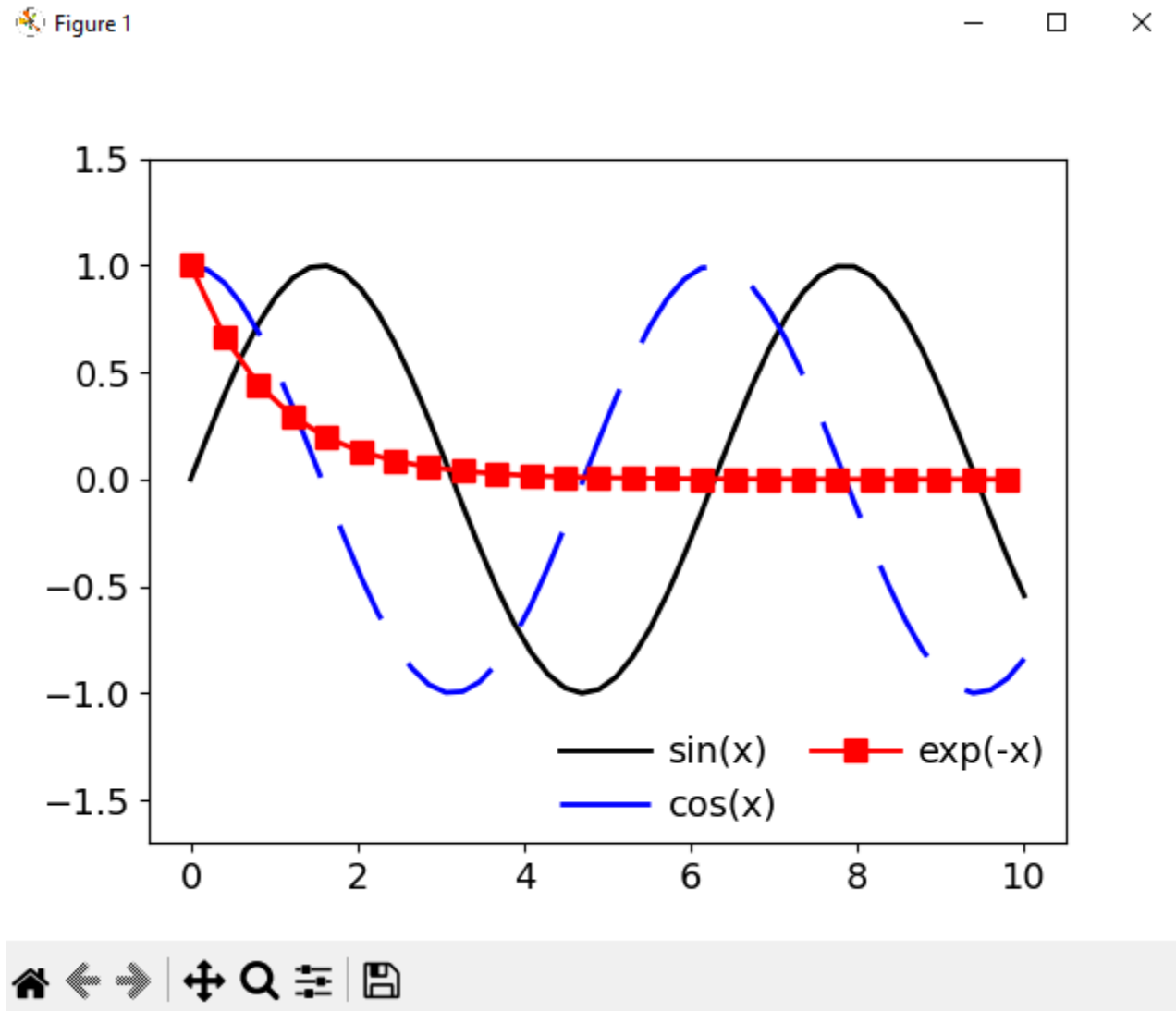


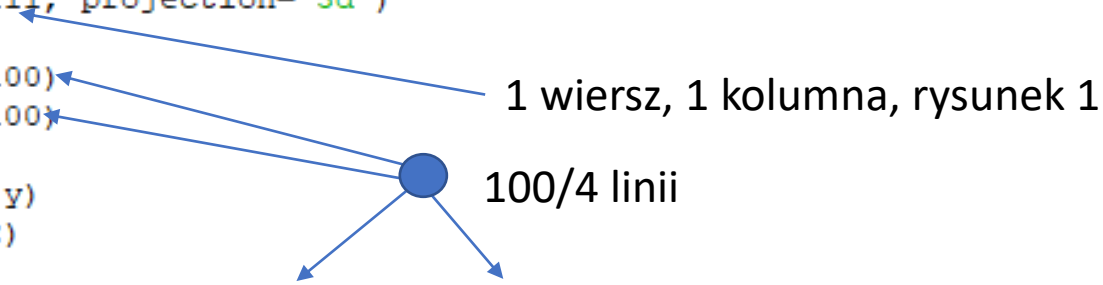
Figure 1
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
0
2
4
6
8
10
sin(x)
cos(x)
exp(-x)

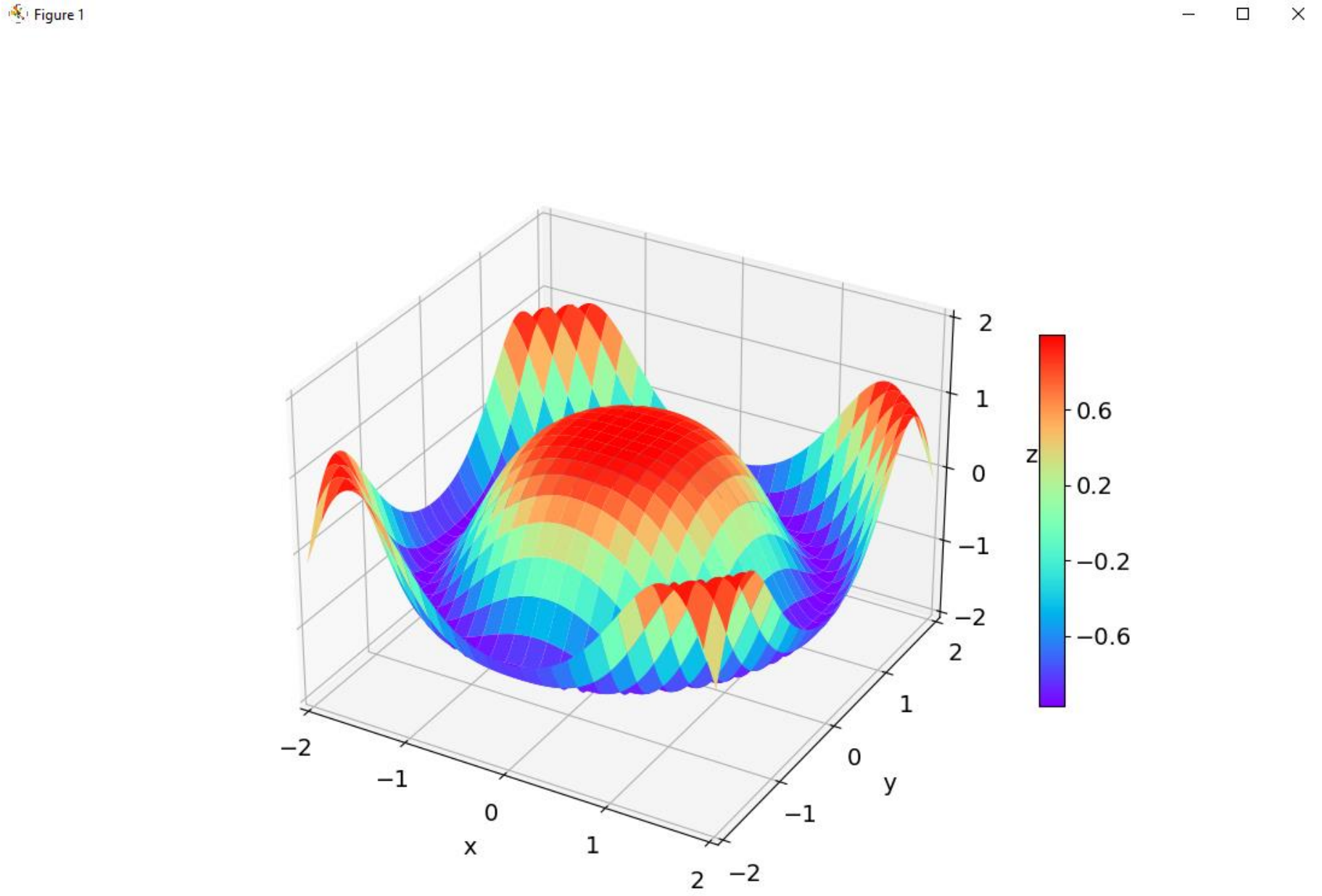
Niektóre parametry wykresu

```
import matplotlib.pyplot as plt

# print(plt.rcParams.keys())

# plt.rcParams['font.family'] = 'Arial'
# plt.rcParams['font.family'] = 'Serif'
plt.rcParams['font.family'] = 'Times New Roman'
plt.rcParams['font.size'] = 16
plt.rcParams['legend.fontsize'] = 14
plt.rcParams['lines.linewidth'] = 2.5
plt.rcParams['axes.linewidth'] = 1.0
plt.rcParams['xtick.major.size'] = 8.0
plt.rcParams['xtick.major.width'] = 1.0
plt.rcParams['ytick.major.size'] = 8.0
plt.rcParams['ytick.major.width'] = 1.0
plt.rcParams['xtick.minor.size'] = 4.0
plt.rcParams['xtick.minor.width'] = 1.0
plt.rcParams['ytick.minor.size'] = 4.0
plt.rcParams['ytick.minor.width'] = 1.0
plt.rcParams['xtick.direction'] = 'out'
plt.rcParams['ytick.direction'] = 'out'
```

3D plot

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

plt.rcParams['font.size'] = 14

fig = plt.figure(figsize=(11, 8))
ax = fig.add_subplot(111, projection='3d')

x = np.linspace(-2,2,100)
y = np.linspace(-2,2,100)

(X,Y) = np.meshgrid(x,y)
Z = np.cos(X**2 + Y**2)

fl = ax.plot_surface(X, Y, Z, rstride=4, cstride=4,
                     linewidth=0.1, cmap='rainbow')

fig.colorbar(fl, shrink=0.5, aspect=15,
             ticks=np.linspace(-1,1,6))
ax.set_xlim(-2,2)
ax.set_ylim(-2,2)
ax.set_zlim(-2,2)

ax.set_xlabel('x', labelpad=10)
ax.set_ylabel('y', labelpad=10)
ax.set_zlabel('z', labelpad=10)

ax.set_xticks(np.linspace(-2,2,5))
ax.set_yticks(np.linspace(-2,2,5))
ax.set_zticks(np.linspace(-2,2,5))

plt.show()
```

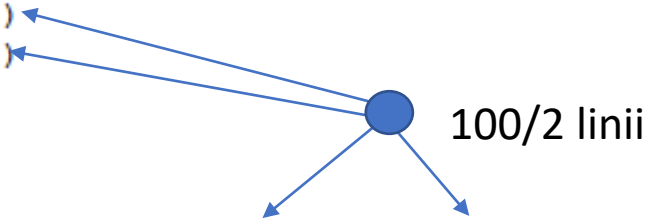




Używając myszki można przybliżać, oddalać, obracać, etc.

3D plot, wireframe

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

plt.rcParams['font.size'] = 14

fig = plt.figure(figsize=(11, 8))
ax = fig.add_subplot(111, projection='3d')

x = np.linspace(-2,2,100)
y = np.linspace(-2,2,100)

(X,Y) = np.meshgrid(x,y)
Z = np.cos(X**2 + Y**2)

fl = ax.plot_wireframe(X, Y, Z, rstride=2, cstride=2,
                       linewidth=1, color='black')

ax.set_xlim(-2,2)
ax.set_ylim(-2,2)
ax.set_zlim(-2,2)

ax.set_xlabel('x', labelpad=10)
ax.set_ylabel('y', labelpad=10)
ax.set_zlabel('z', labelpad=10)

ax.set_xticks(np.linspace(-2,2,5))
ax.set_yticks(np.linspace(-2,2,5))
ax.set_zticks(np.linspace(-2,2,5))

plt.show()
```



100/2 linii

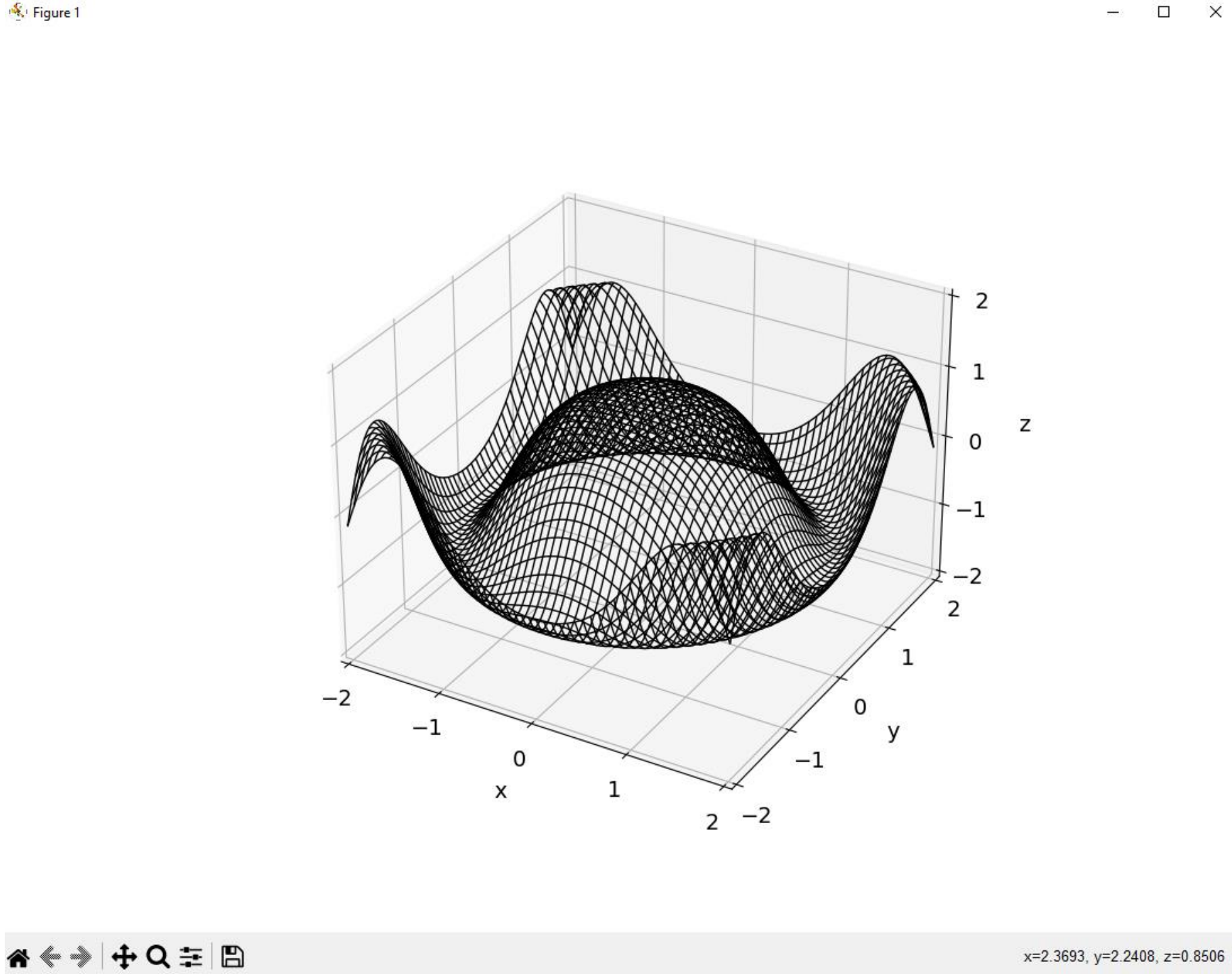


Figure 1
2
1
0
−1
−2
z
−2
−1
0
1
2
x
2
1
0
−1
−2
y
x=2.3693, y=2.2408, z=0.8506

GridSpec

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

2x1 grid (2 wiersze, 1 kolumna)

```
gs = gridspec.GridSpec(2, 1, height_ratios=[3, 1])

x = np.linspace(0,10,50)

ax1 = plt.subplot(gs[0,0])
ax1.plot(x, np.sin(x))

ax2 = plt.subplot(gs[1,0])
ax2.plot(x, np.cos(x))

plt.show()
```

wiersz 0, kolumna 0

wiersz 1, kolumna 0

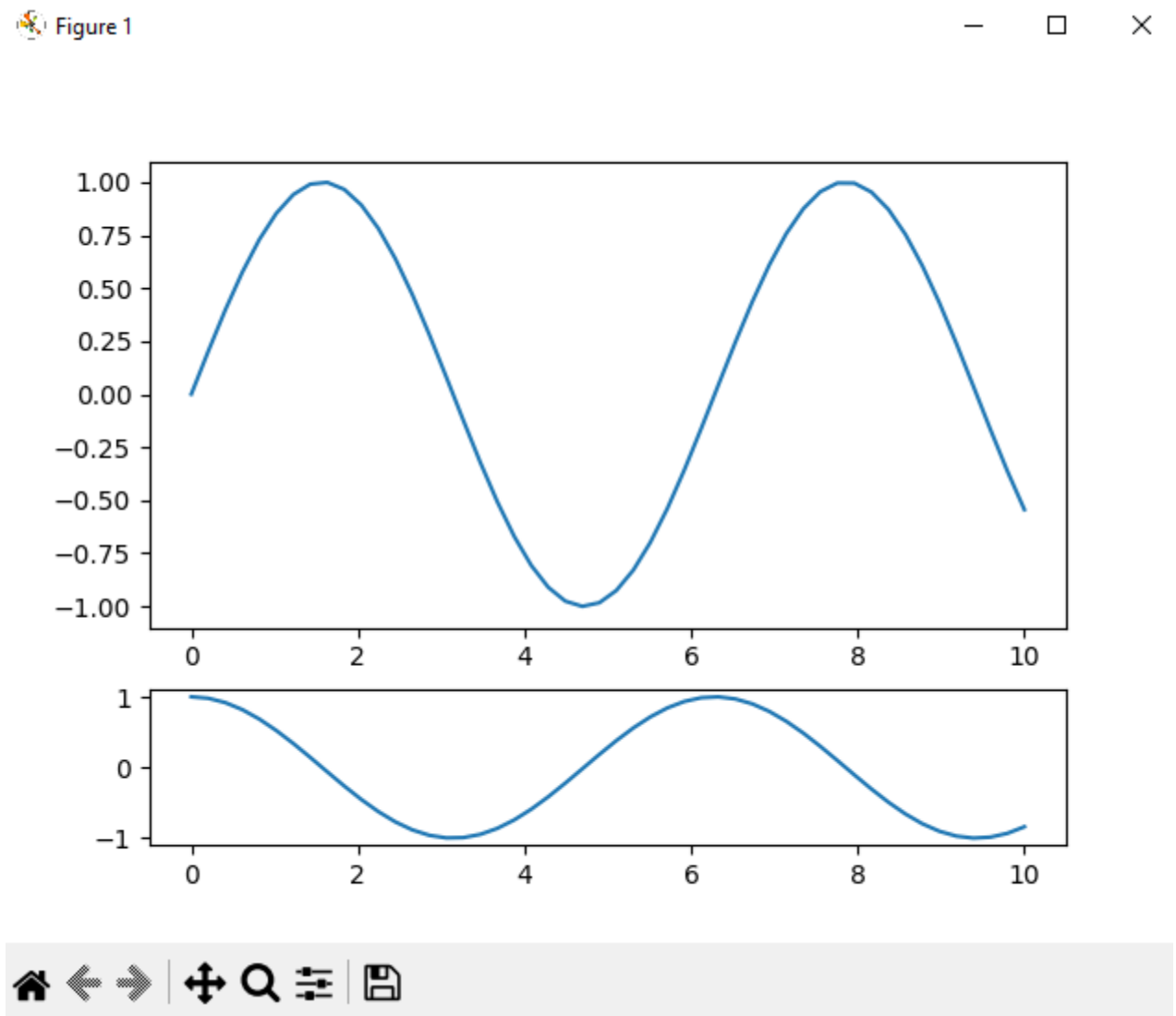


Figure 1
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10
1
0
−1
0
2
4
6
8
10

GridSpec

lec_8f.py - E:/CERNBox/zajecia/Inzynieria_danych/Python/moje/wyklad

File Edit Format Run Options Window Help

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


gs = gridspec.GridSpec(2, 2)

x = np.linspace(0,10,50)

ax1 = plt.subplot(gs[0,:])
ax1.plot(x, np.sin(x))

ax2 = plt.subplot(gs[1,0])
ax2.plot(x, np.cos(x))

ax3 = plt.subplot(gs[1,1])
ax3.plot(x, np.exp(-x/2))

plt.show()
```

2x2 grid (2 wiersze, 2 kolumny)

wiersz 0, wszystkie kolumny

wiersz 1, kolumna 0

wiersz 1, kolumna 1

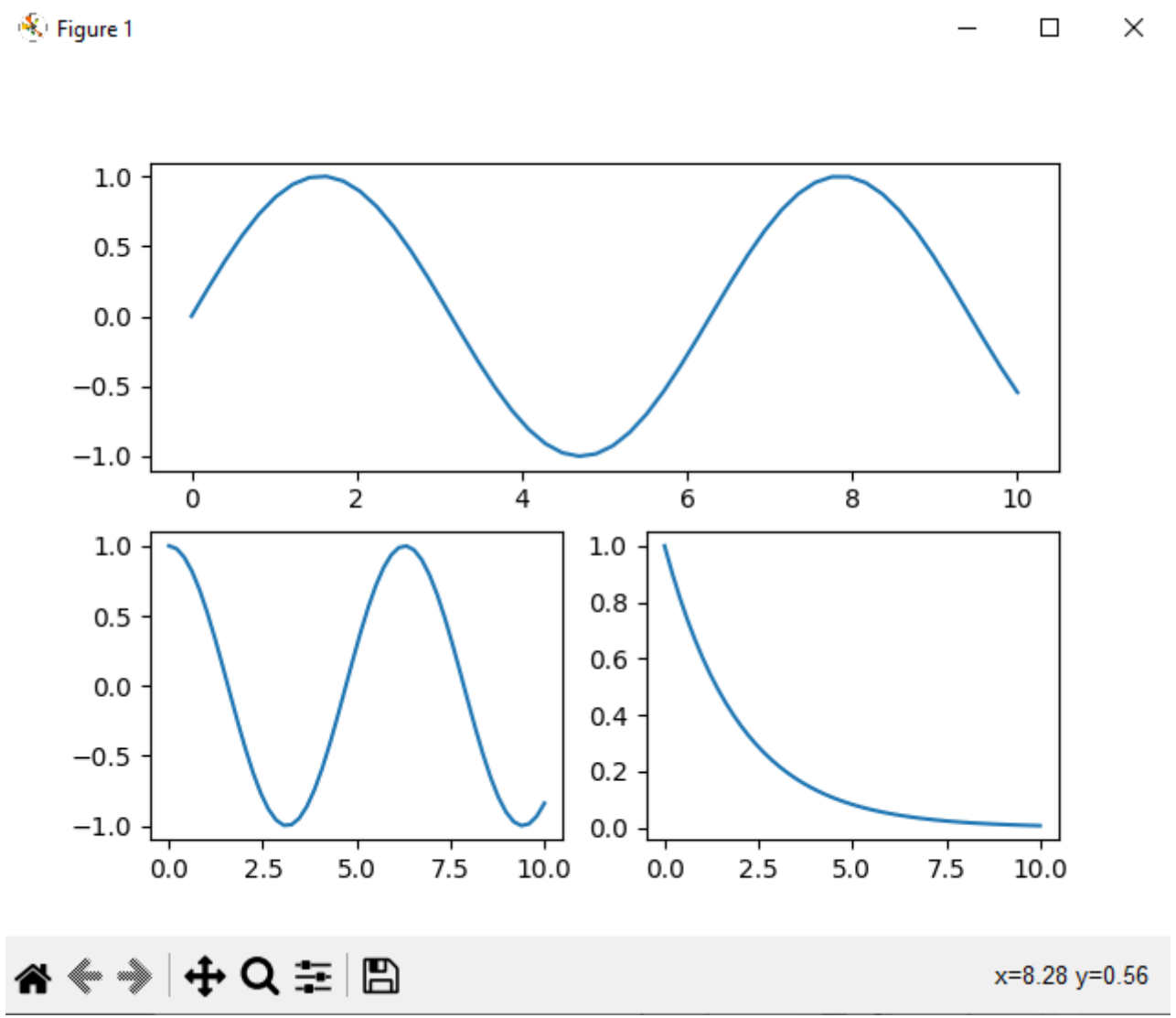
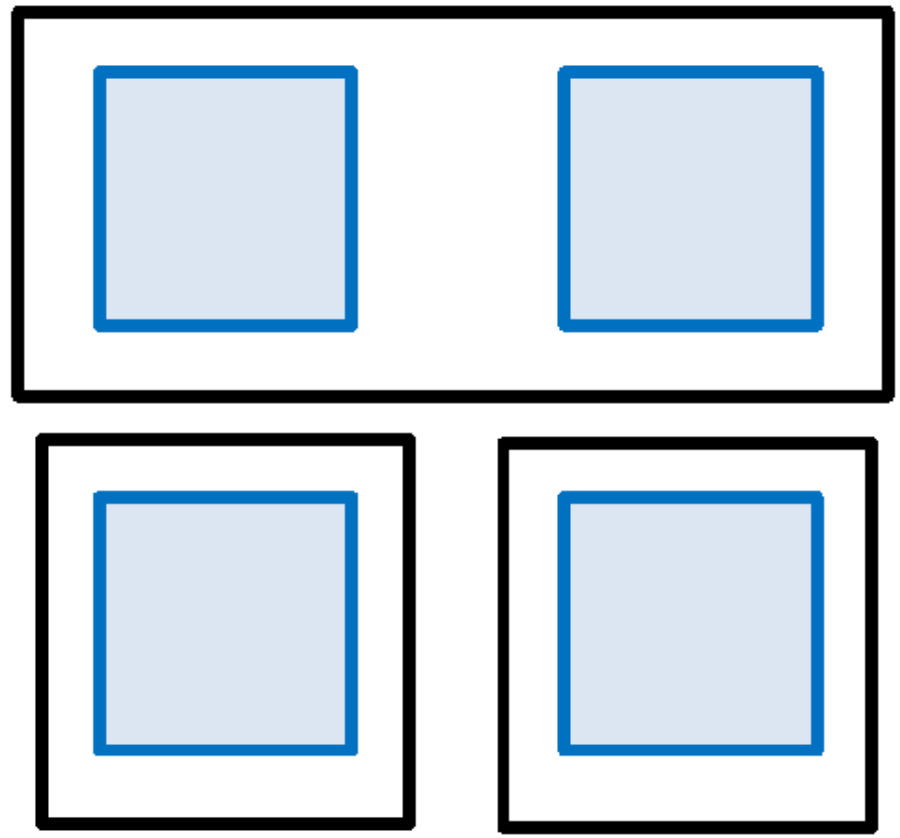


Figure 1
1.0
0.5
0.0
−0.5
−1.0
0
2
4
6
8
10
1.0
0.5
0.0
−0.5
−1.0
0.0
2.5
5.0
7.5
10.0
1.0
0.8
0.6
0.4
0.2
0.0
0.0
2.5
5.0
7.5
10.0
x=8.28 y=0.56

grid 2x2



gs[0, :]

gs[1, 0]

gs[1, 1]

GridSpec

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


gs = gridspec.GridSpec(3, 3, wspace=0.3, hspace=0.5,
                       height_ratios=[1, 1, 2])

x = np.linspace(0,10,10**2)

ax1 = plt.subplot(gs[0,:])
ax1.plot(x, np.sin(x))
ax1.semilogx()

ax2 = plt.subplot(gs[1,:-1])
ax2.plot(x, np.cos(x))

ax3 = plt.subplot(gs[1:,-1])
ax4 = plt.subplot(gs[2,0])
ax3 = plt.subplot(gs[2,1])

plt.show()
```

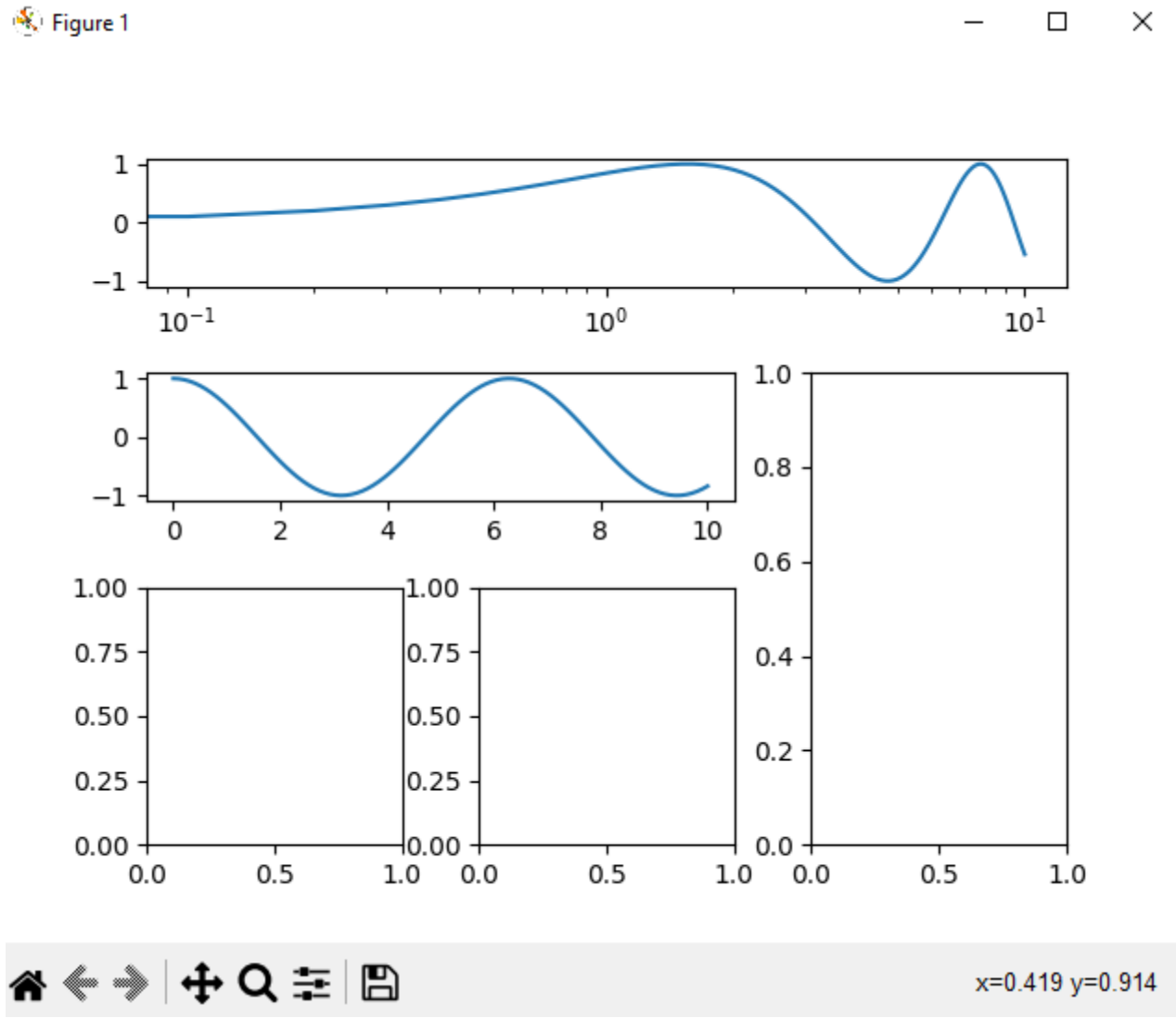


Figure 1
1
0
−1
10^{-1}
10^{0}
10^{1}
1
0
−1
0
2
4
6
8
10
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.5
1.0
1.00
0.75
0.50
0.25
0.00
0.0
0.5
1.0
1.00
0.75
0.50
0.25
0.00
0.0
0.5
1.0
x=0.419 y=0.914

More Stuff
(if you are interested)

Fitowanie funkcji do danych - pakiet SciPy

- Pakiet SciPy jest biblioteką do zastosowań naukowych
- Tutaj wykorzystanie podbiblioteki `optimize`:

  `import scipy.optimize` lub `import scipy.optimize as sp`

  `https://docs.scipy.org/doc/scipy/reference/optimize.html`

- Zastosujemy metodę `curve_fit` do dopasowania funkcji do danych

```
import scipy.optimize as sp

par,cov = sp.curve_fit(f,x,y,p0=start,sigma=s,
absolute_sigma=True)
```

```
from scipy.optimize import curve_fit

par,cov = curve_fit(f,x,y,p0=start,sigma=s,
absolute_sigma=True)
```

Curve_fit

- Wywołanie funkcji dopasowującej krzywą teoretyczną do danych:

```
par,cov = curve_fit(f,x,y,p0=start,sigma=s,absolute_sigma=True)
```

- Metoda zwraca:
  ⇒ macierz 1D z parametrami (`par`) dopasowania funkcji `f`
  ⇒ 2D macierz kowariancji (`cov`) dopasowania
- Argumenty metody `curve_fit`:
  ⇒ f - dopasowywana funkcja
  ⇒ x - macierz ze współrzędnymi x punktów
  ⇒ y - macierz ze współrzędnymi y punktów
  ⇒ p0 - macierz z początkowymi wartościami parametrów*
  ⇒ sigma - macierz z niepewnościami współrzędnych y*
  ⇒ absolute_sigma - informuje, czy niepewności y są podane w jednostkach absolutnych lub względnych*
  * = parametry opcjonalne

https://docs.scipy.org/doc/scipy/reference/generated/scipy.optimize.curve_fit.html

Dopasowanie funkcji do danych - przykłady

```
import matplotlib
# matplotlib.use('Agg')

import numpy as n
import matplotlib.pyplot as p

data = n.loadtxt('line.txt')
x = data[:, 0]
y = data[:, 1]

p.plot(x, y, 'o')
p.show()

p.savefig('przyklad8.pdf')
```

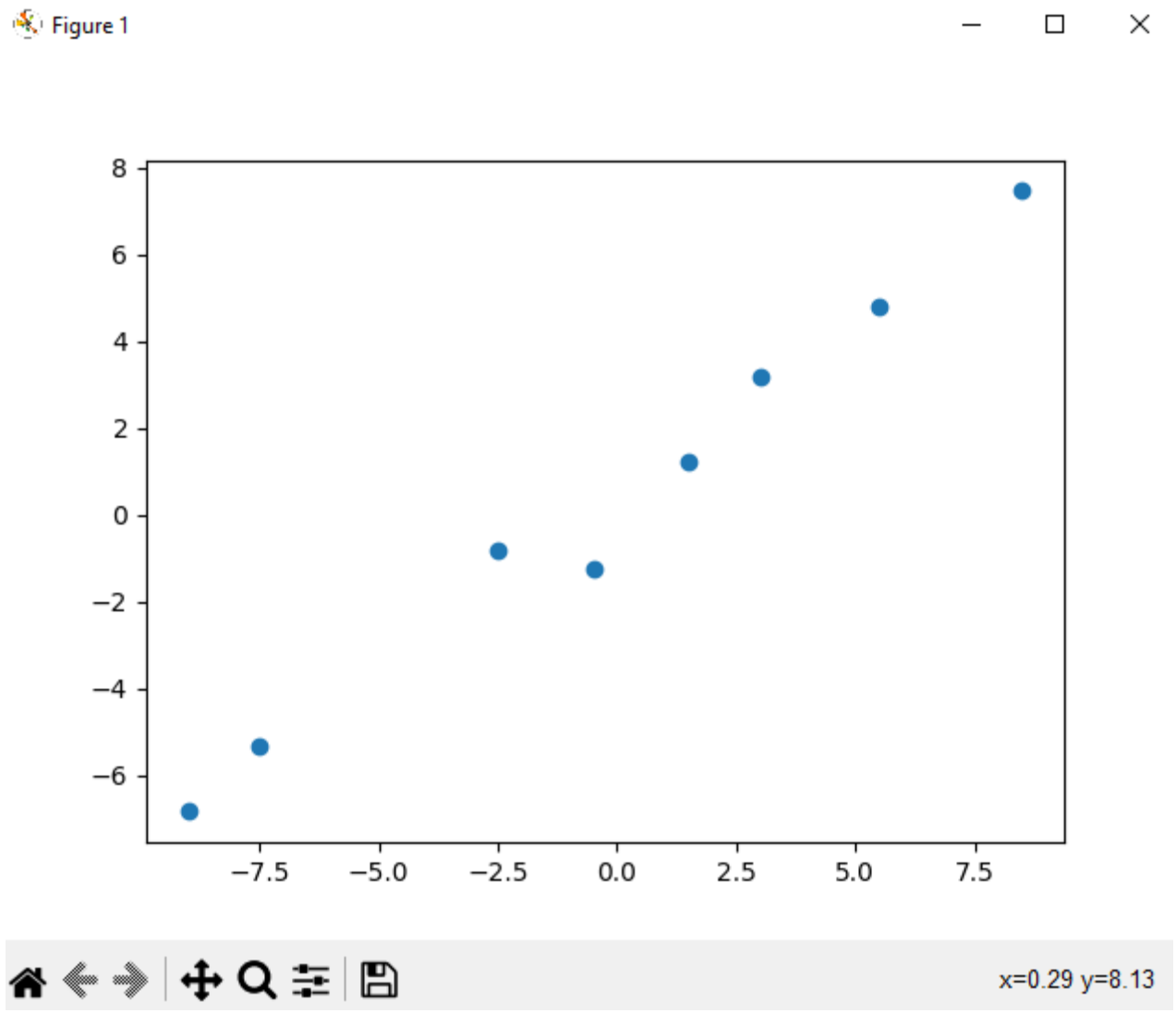


Figure 1
8
6
4
2
0
−2
−4
−6
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
x=0.29 y=8.13

Dopasowanie funkcji do danych - przykłady

```
import matplotlib
# matplotlib.use('Agg')

import numpy as n
import matplotlib.pyplot as p

from scipy.optimize import curve_fit

data = n.loadtxt('line.txt')
x = data[:, 0]
y = data[:, 1]

p.plot(x, y, 'o')
p.show()

def f(x, a, b):
    return a * x + b

par, cov = curve_fit(f, x, y)

# wypisanie parametrów fitu
print(par)
# wypisanie niepewnosci parametrów fitu
print(n.sqrt(n.diag(cov)))

p.savefig('przyklad8.pdf')
```

```
>>>
= RESTART: E:/CERNBox/zajecia/Inzynieria_danych
[0.80040727 0.41041191]
[0.04055835 0.23010386]
>>>
```

Dopasowanie funkcji do danych - przykłady

```
import matplotlib
# matplotlib.use('Agg')

import numpy as n
import matplotlib.pyplot as p

from scipy.optimize import curve_fit

data = n.loadtxt('line.txt')
x = data[:, 0]
y = data[:, 1]

def f(x, a, b):
    return a * x + b

par, cov = curve_fit(f, x, y)

# wypisanie parametrów fitu
print(par)
# wypisanie niepewnosci parametrów fitu
print(n.sqrt(n.diag(cov)))

# przygotowanie wspolrzednych x z danego zakresu
xpar = n.arange(-10, 10, 0.1)

# rysowanie dopasowanej zaleznosci
p.plot(xpar, f(xpar,par[0],par[1]))
#p.plot(xpar, f(xpar, *par))
p.plot(x, y, 'o')
p.show()

p.savefig('przyklad8.pdf')
```

```
>>>
= RESTART: E:/CERNBox/zajecia/Inzynieria_danych
[0.80040727 0.41041191]
[0.04055835 0.23010386]
>>>
```

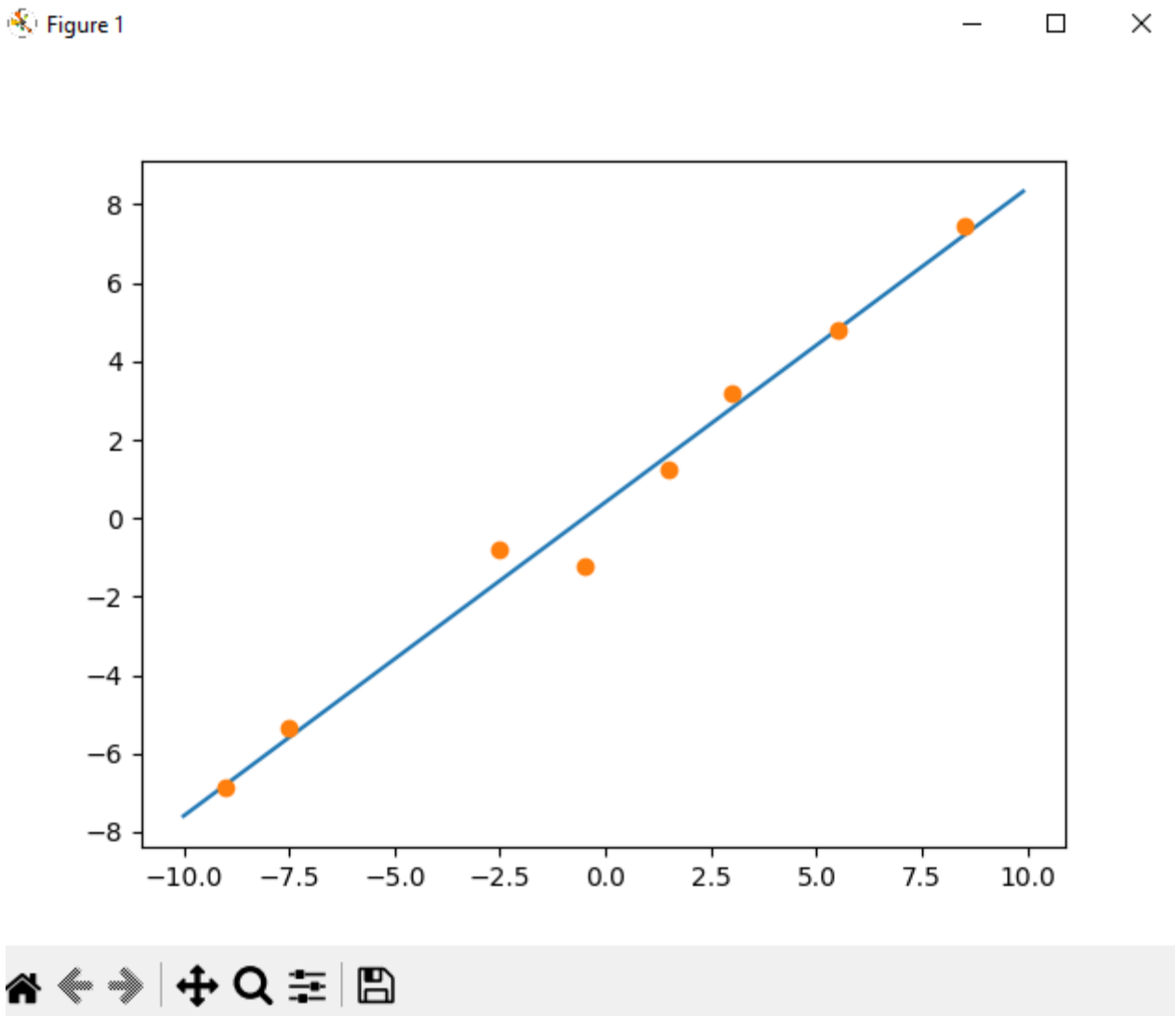


Figure 1
8
6
4
2
0
−2
−4
−6
−8
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

Dopasowanie funkcji do danych - przykłady

```
import matplotlib
# matplotlib.use('Agg')

import numpy as n
import matplotlib.pyplot as p

from scipy.optimize import curve_fit

data = n.loadtxt('line.txt')
x = data[:, 0]
y = data[:, 1]

def f(x, a, b):
    return a * x + b

# par, cov = curve_fit(f, x, y)
par, cov = curve_fit(f, x, y, p0=n.array([0.7, 0.5]))


# wypisanie parametrów fitu
print(par)
# wypisanie niepewnosci parametrów fitu
print(n.sqrt(n.diag(cov)))

# przygotowanie wspolrzednych x z danego zakresu
xpar = n.arange(-10, 10, 0.1)

# rysowanie dopasowanej zaleznosci
p.plot(xpar, f(xpar,par[0],par[1]))
#p.plot(xpar, f(xpar, *par))
p.plot(x, y, 'o')
p.show()

p.savefig('przyklad8.pdf')
```

zadawanie wartości początkowych parametrów fitu

```
>>>
= RESTART: E:/CERNBox/zajecia/Inzynieria_danych
[0.80040727 0.41041191]
[0.04055835 0.23010386]
>>>
```

Dopasowanie funkcji do danych - przykłady

```
import matplotlib
# matplotlib.use('Agg')

import numpy as n
import matplotlib.pyplot as p

from scipy.optimize import curve_fit

data = n.loadtxt('line.txt')
x = data[:, 0]
y = data[:, 1]
s = data[:, 2]
p.errorbar(x,y,yerr=s,fmt='o')

def f(x, a, b):
    return a * x + b

# par, cov = curve_fit(f, x, y)
# par, cov = curve_fit(f, x, y, p0=n.array([0.7, 0.5]))
par,cov=curve_fit(f, x, y,
    p0 = n.array([0.7,0.5]),
    sigma = s,
    absolute_sigma = True)

# wypisanie parametrów fitu
print(par)
# wypisanie niepewnosci parametrów fitu
print(n.sqrt(n.diag(cov)))

# przygotowanie wspolrzednych x z danego zakresu
xpar = n.arange(-10, 10, 0.1)

# rysowanie dopasowanej zaleznosci
p.plot(xpar, f(xpar,par[0],par[1]))
#p.plot(xpar, f(xpar, *par))
p.plot(x, y, 'o')
p.show()

p.savefig('przyklad8.pdf')
```

uwzględnianie błędów na osi rzędnych

inne parametry fitu

```
= RESTART: E:/CERNBox/zajecia/Inzynieria_da
[0.7760127  0.40379309]
[0.02583416 0.16769905]
```

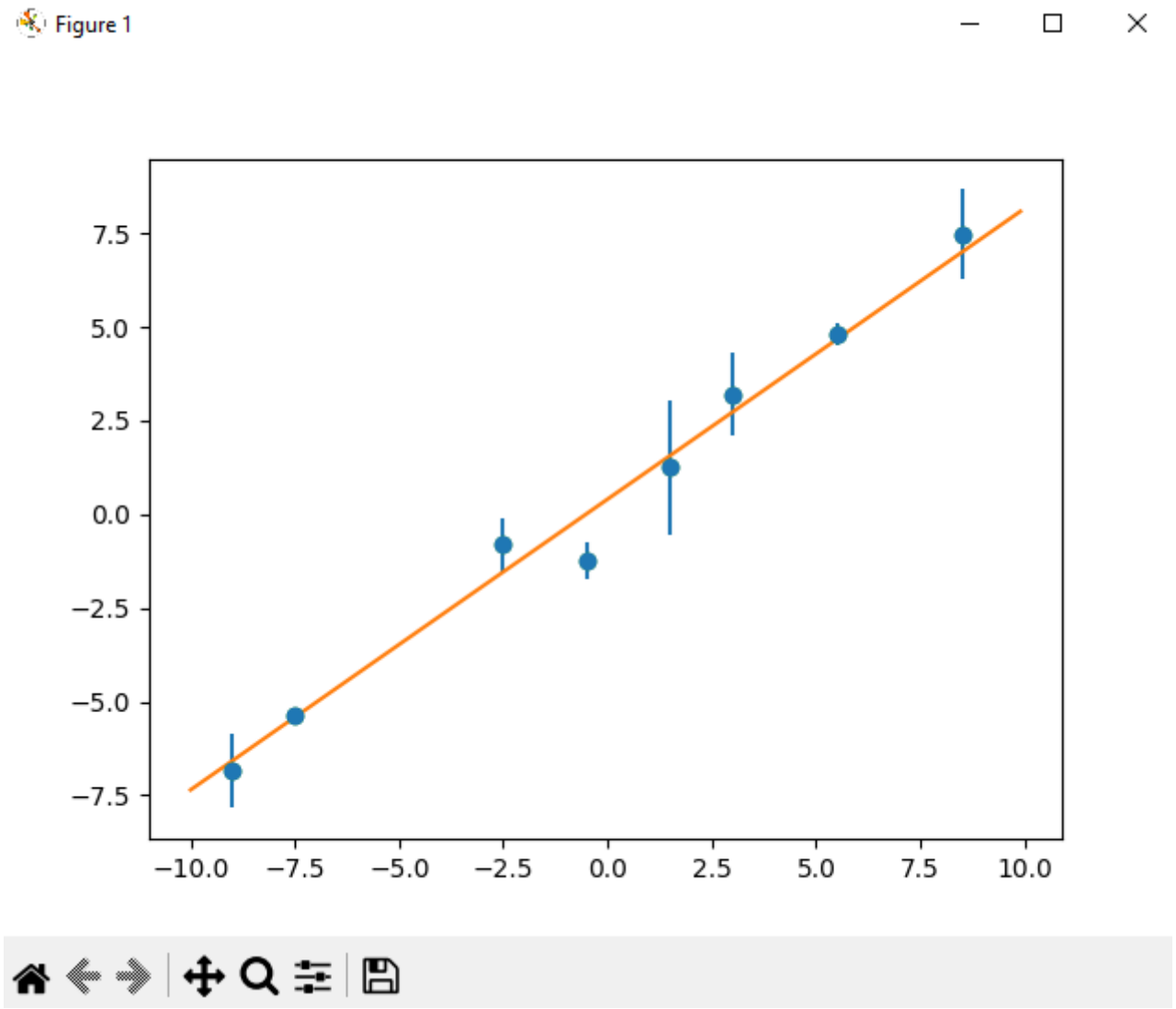
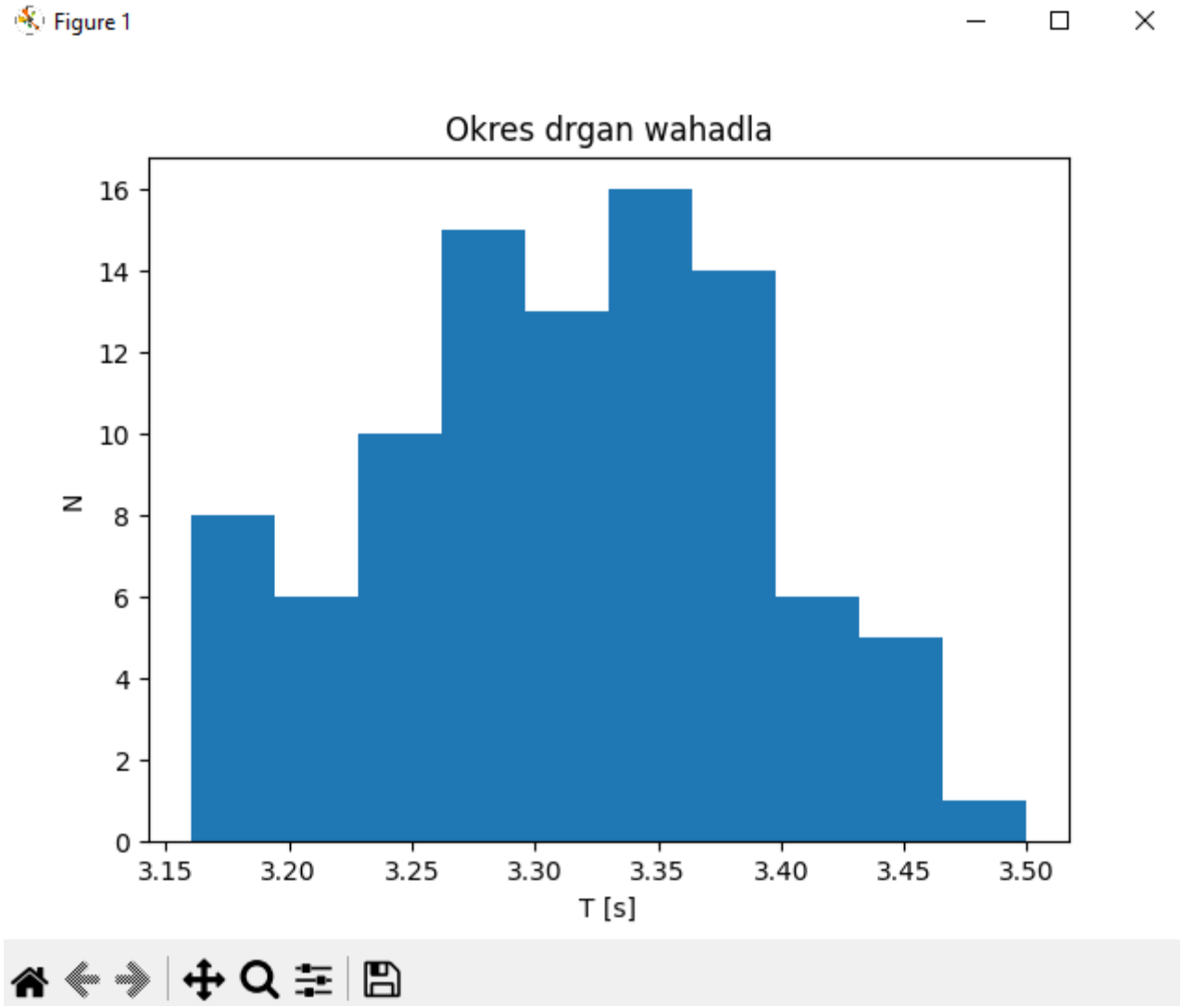


Figure 1
7.5
5.0
2.5
0.0
−2.5
−5.0
−7.5
−10.0
−7.5
−5.0
−2.5
0.0
2.5
5.0
7.5
10.0

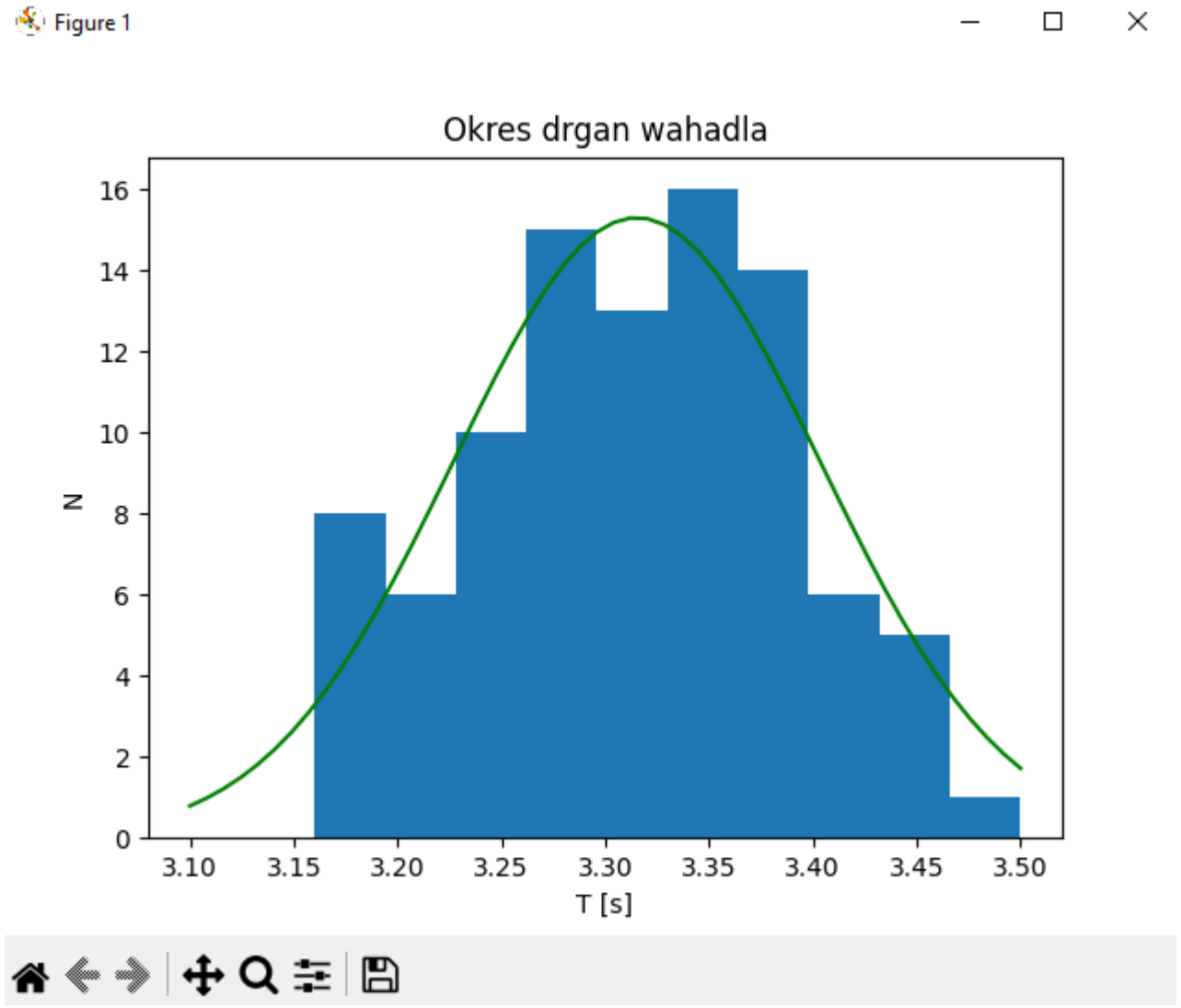
Dopasowanie funkcji do danych - przykłady

```
import matplotlib
# matplotlib.use('Agg')

import matplotlib.pyplot as p
import numpy as n

data = n.loadtxt('wahadlo.txt')

p.hist(data)

p.xlabel('T [s]')
p.ylabel('N')
p.title('Okres drgan wahadla')

p.show()
p.savefig('przyklad9.pdf')
```



Dopasowanie funkcji do danych - przykłady

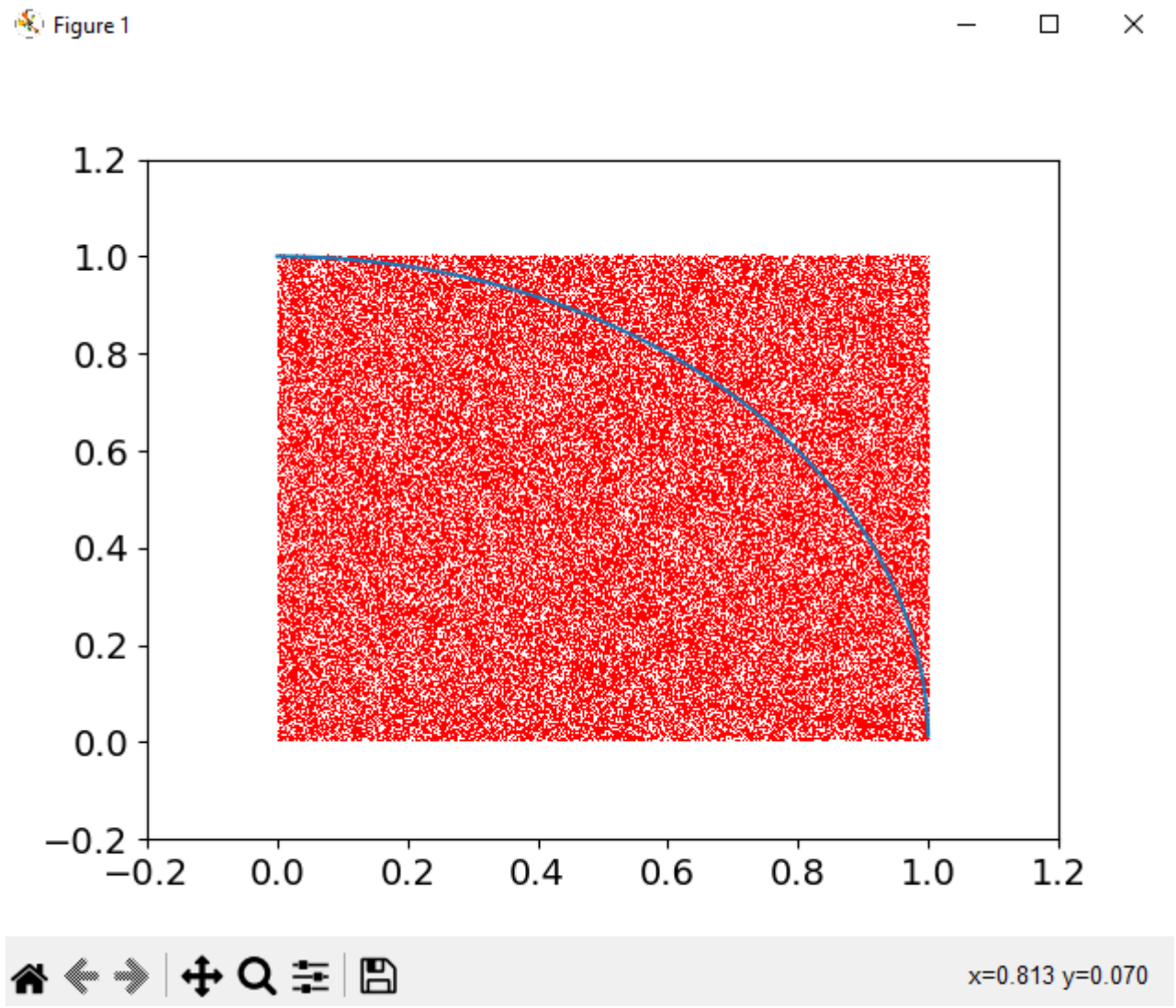
```
import matplotlib
# matplotlib.use('Agg')

import matplotlib.pyplot as p
import numpy as n

from scipy.optimize import curve_fit

data = n.loadtxt('wahadlo.txt')
#p.hist(data)
num, bins, patches = p.hist(data)

bincenters=[]
for i in range(0, len(bins)-1):
    bincenters.append(0.5*(bins[i]+bins[i+1]))
#bincenters=0.5*(bins[1:]+bins[:-1])

def gauss(x, mu, sig, p):
    return p*n.exp(-((x-mu)**2)/(2*sig*sig))

par, cov = curve_fit(gauss, bincenters, num)
# wypisanie parametrów fitu
print(par)
# wypisanie niepewnosci parametrów fitu
print(n.sqrt(n.diag(cov)))

x = n.linspace(3.1, 3.5, 50)

p.xlabel('T [s]')
p.ylabel('N')
p.title('Okres drgan wahadla')

p.plot(x,gauss(x, *par), 'g-')

p.show()
p.savefig('przyklad9.pdf')
```




```
= RESTART: E:/CERNBox/zajecia/Inzynieria_danych/P
[ 3.31518088  0.08835386 15.30066929]
[0.00913993 0.01013877 1.36292115]
```

Prawdopodobieństwo geometryczne – wartość liczby pi

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

plt.rcParams['font.size'] = 14

x = np.random.uniform(0,1,100000)
y = np.random.uniform(0,1,100000)

x1 = np.arange(0,1,0.0001)
y1 = np.sqrt(1.0 - x1**2)

plt.plot(x, y, 'r,')
plt.plot(x1, y1)

plt.axis([-0.2,1.2,-0.2,1.2])
plt.show()
```



Prawdopodobieństwo geometryczne – wartość liczby pi

N – liczba punktów w kwadracie o boku R – proporcjonalna do pola powierzchni kwadratu, P_{Kw}
n_1 – liczba punktów pod krzywą y=sqrt(R**2 – x**2) – proporcjonalna do 1/4 pola powierzchni koła, P_K

$$\frac{P_K}{P_{Kw}} = \frac{n_1}{N} = \frac{{}^1\!/_4\,\pi R^2}{R^2} \longrightarrow \pi \approx \frac{4n_1}{N}$$

```
import numpy as np

N=10**6
n1 = 0
for i in range(N):
    x = np.random.uniform()
    y = np.random.uniform()
    if (y<np.sqrt(1.0-x**2)):
        n1 = n1 + 1


print('pi=', 4.0 * n1 /N)
```

```
= RESTART: E:/CERNBox/zajecia/In
pi= 3.142268
```