Лабораторна робота №10

*Метод найменших квадратів*

Виконала Гальчинська Софія, студентка ФІТ 2-8. Варіант 4.

[*https://github.com/GalchynskaSofiia/Numerical\_Methods\_Of\_Programming*](https://github.com/GalchynskaSofiia/Numerical_Methods_Of_Programming)

Для функції побудуйте таблицю:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***i*** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| ***xi*** | 0,1 | 0,2 | 0,3 | 0,4 | 0,5 | 0,6 | 0,7 | 0,8 | 0,9 | 1 |
| ***yi*** | f(x0) | f(x1) | f(x2) | f(x3) | f(x4) | f(x5) | f(x6) | f(x7) | f(x8) | f(x9) |

Для визначення значень функції, підставимо ***xi*** у функцію. Ці розрахунки зробимо за допомогою коду:

import numpy as np

def func(x):

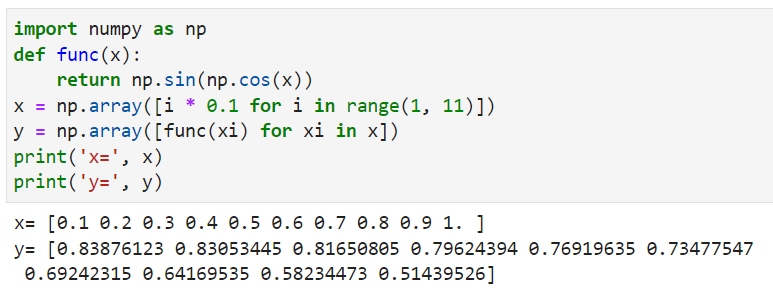
return np.sin(np.cos(x))

x = np.array([i \* 0.1 for i in range(1, 11)])

y = np.array([func(xi) for xi in x])

print('x=', x)

print('y=', y)



Отже, маємо:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ***i*** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** |
| ***xi*** | 0,1 | 0,2 | 0,3 | 0,4 | 0,5 | 0,6 | 0,7 | 0,8 | 0,9 | 1 |
| ***yi*** | 0,83876123 | 0,83053445 | 0,81650805 | 0,79624394 | 0,76919635 | 0,73477547 | 0,69242315 | 0,64169535 | 0,58234473 | 0,51439526 |

**Аналітичні розрахунки:**

**МНК наближення прямою:**

import numpy as np

def func(x):

return np.sin(np.cos(x))

x = np.array([i \* 0.1 for i in range(1, 11)])

y = np.array([func(xi) for xi in x])

x\_ = np.mean(x)

x2\_ = np.mean(x\*\*2)

y\_ = np.mean(y)

xy\_ = np.mean(x\*y)

print('x\_ = ', x\_)

print('x2\_ = ', x2\_)

print('y\_ = ', y\_)

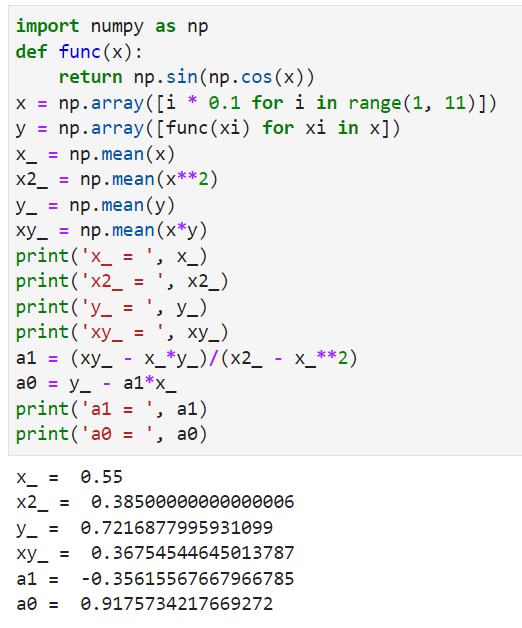
print('xy\_ = ', xy\_)

a1 = (xy\_ - x\_\*y\_)/(x2\_ - x\_\*\*2)

a0 = y\_ - a1\*x\_

print('a1 = ', a1)

print('a0 = ', a0)



**МНК наближення параболою:**

import numpy as np

import sympy as sp

from sympy.solvers.solveset import linsolve

def func(x):

return np.sin(np.cos(x))

x = np.array([i \* 0.1 for i in range(1, 11)])

y = np.array([func(xi) for xi in x])

x\_ = np.mean(x)

x2\_ = np.mean(x\*\*2)

x3\_ = np.mean(x\*\*3)

x4\_ = np.mean(x\*\*4)

y\_ = np.mean(y)

xy\_ = np.mean(x\*y)

x2y\_ = np.mean(x\*\*2\*y)

print('x\_ = ', x\_)

print('x2\_ = ', x2\_)

print('x3\_ = ', x3\_)

print('x4\_ = ', x4\_)

print('y\_ = ', y\_)

print('xy\_ = ', xy\_)

print('x2y\_ = ', x2y\_)

a0, a1, a2 = sp.symbols("a0 a1 a2")

eq1 = sp.Eq(a0 + a1\*x\_ + a2\*x2\_, y\_)

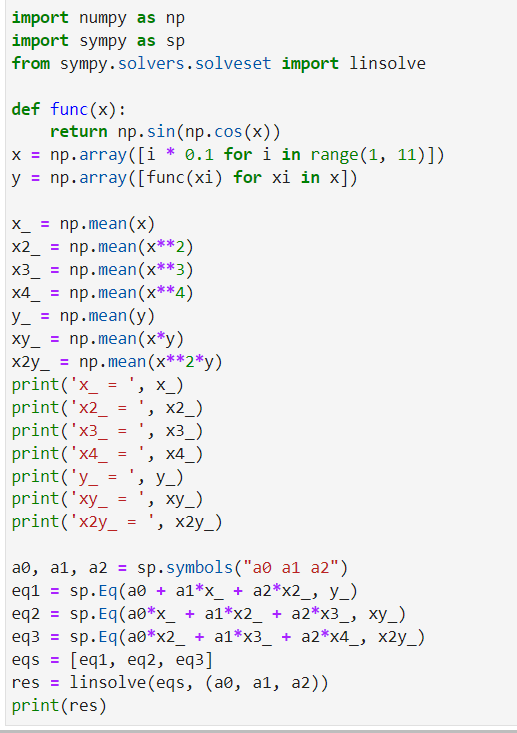
eq2 = sp.Eq(a0\*x\_ + a1\*x2\_ + a2\*x3\_, xy\_)

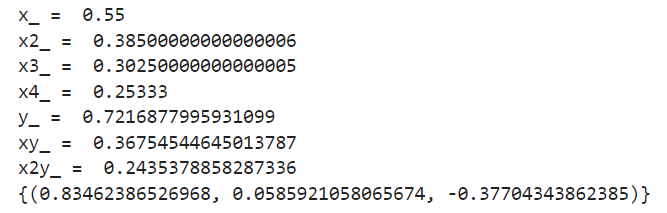
eq3 = sp.Eq(a0\*x2\_ + a1\*x3\_ + a2\*x4\_, x2y\_)

eqs = [eq1, eq2, eq3]

res = linsolve(eqs, (a0, a1, a2))

print(res)





**МНК наближення прямою:**

import numpy as np

import matplotlib.pyplot as plt

from scipy.optimize import least\_squares

def func(x):

return np.sin(np.cos(x))

x = np.array([i \* 0.1 for i in range(1, 11)])

y = np.array([func(xi) for xi in x])

print('x=', x)

print('y=', y)

def fun(a, x, y):

return a[0] + a[1] \* x - y

a0 = np.array([1, 1])

res\_lsq = least\_squares(fun, x0=a0, args=(x, y))

print("a0 = %.2f, a1 = %.2f" % tuple(res\_lsq.x))

f = lambda x: res\_lsq.x[0] + res\_lsq.x[1] \* x

x\_p = np.linspace(min(x), max(x), 20)

y\_p = f(x\_p)

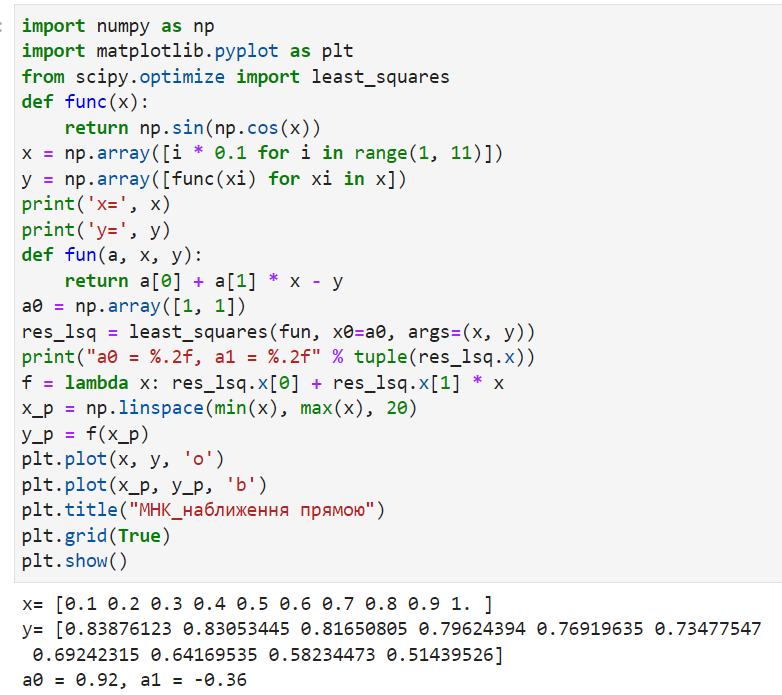
plt.plot(x, y, 'o')

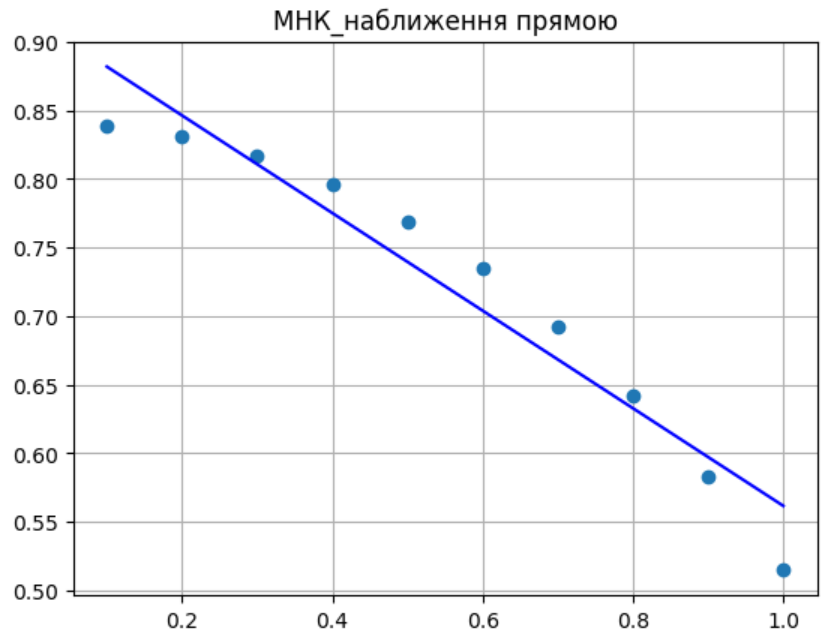
plt.plot(x\_p, y\_p, 'b')

plt.title("МНК\_наближення прямою")

plt.grid(True)

plt.show()





**МНК наближення параболою:**

import numpy as np

import matplotlib.pyplot as plt

from scipy.optimize import least\_squares

def func(x):

return np.sin(np.cos(x))

x = np.array([i \* 0.1 for i in range(1, 11)])

y = np.array([func(xi) for xi in x])

print('x=', x)

print('y=', y)

def fun(a, x, y):

return a[0] + a[1] \* x + a[2] \* x\*\*2 - y

a0 = np.array([1, 1, 1])

res\_lsq = least\_squares(fun, x0=a0, args=(x, y))

print("a0 = %.2f, a1 = %.2f, a2 = %.2f" % tuple(res\_lsq.x))

f = lambda x: sum([u \* v for u, v in zip(res\_lsq.x, [1, x, x\*\*2])])

x\_p = np.linspace(min(x), max(x), 20)

y\_p = f(x\_p)

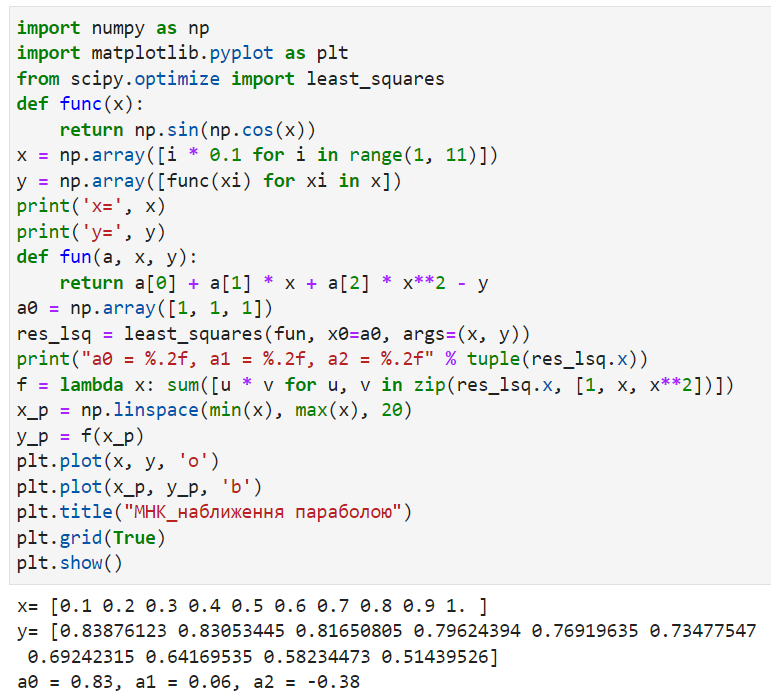
plt.plot(x, y, 'o')

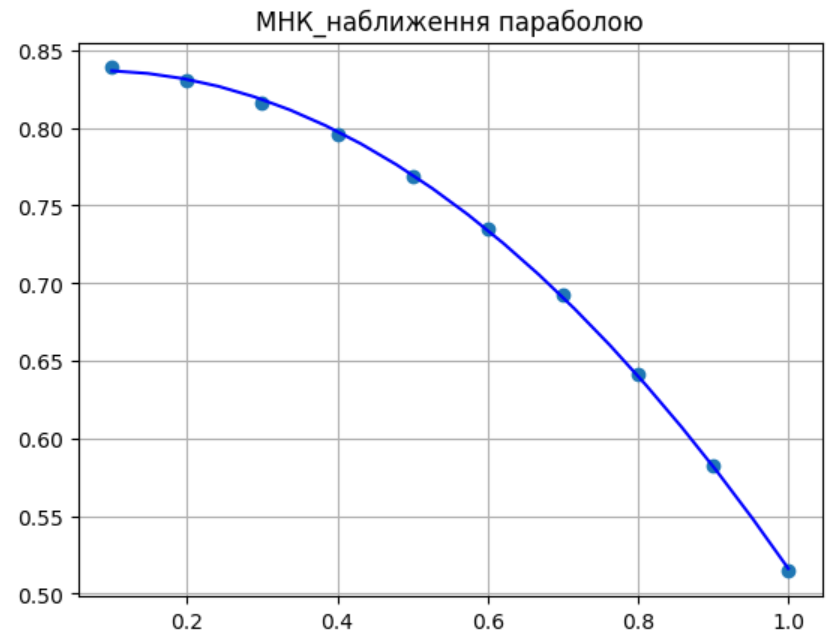
plt.plot(x\_p, y\_p, 'b')

plt.title("МНК\_наближення параболою")

plt.grid(True)

plt.show()





**Висновки:** Отже, під час виконання практичної роботи ознайомились з методом найменших квадратів, зокрема наближення прямою та параболою. Також написали код для обрахування значень ***yi*** при даних ***xi***. Потім було проведено аналітичні розрахунки, де обчислено коефіцієнти, потрібні для побудови рівняння прямої/параболи. Аналітичні розрахунки було проведено також за допомогою Python. Після цього реалізували МНК наближення прямою/параболою у коді та побудували графіки функції з відображенням прямої/параболи та точок, отриманих на початку практичної роботи. Відмітимо, що коефіцієнти, обраховані «вручну» під час аналітичних розрахунків та під час реалізації МНК цілком зійшлись, що свідчить про їх правильність.