











import sympy as sp

import numpy as np

import matplotlib.pyplot as plt

# Задаємо символьну змінну x та функцію f(x)

x = sp.symbols('x')

f = sp.cos(3\*x ) -3\*x + 3

# Знаходимо перші три похідні

f1 = sp.diff(f, x)

f2 = sp.diff(f1, x)

f3 = sp.diff(f2, x)

# Виводимо похідні

print("f'(x) =", f1)

print("f''(x) =", f2)

print("f'''(x) =", f3)

# Знаходимо значення функції та її похідних в точці x=0

x0 = 0

f\_x0 = f.subs(x, x0).evalf()

f1\_x0 = f1.subs(x, x0).evalf()

f2\_x0 = f2.subs(x, x0).evalf()

f3\_x0 = f3.subs(x, x0).evalf()

print("f(0) =", f\_x0)

print("f(1) =", f1\_x0)

print("f(2) =", f2\_x0)

print("f(3) =", f3\_x0)

# Обчислюємо значення многочлена Тейлора в точці x=0

T = f\_x0 + f1\_x0\*(x-0) + (f2\_x0/2)\*(x-0)\*\*2 + (f3\_x0/6)\*(x-0)\*\*3

# Виводимо наближення за багаточленом Тейлора в точці x=0

print("T(x) =", T.evalf())

# Будуємо графіки

x\_vals = np.linspace(-2, 2, 1000)

f\_vals = np.array([f.subs(x, xi).evalf() for xi in x\_vals])

T\_vals = np.array([T.subs(x, xi).evalf() for xi in x\_vals])

fig, ax = plt.subplots()

ax.plot(x\_vals, f\_vals, label="f(x)")

ax.plot(x\_vals, T\_vals, label="T(x)")

ax.legend()

ax.set\_xlabel("x")

ax.set\_ylabel("y")

ax.set\_title("Графік функції та наближення многочленом Тейлора")

plt.grid(True)

plt.show()

#Побудова багаточлена Тейлора за допомогою approximate\_taylor\_polynomial

import numpy as np

import matplotlib.pyplot as plt

from scipy.interpolate import approximate\_taylor\_polynomial

# Задана функція

def f(x):

return np.cos(3\*x)-3\*x + 3

x = np.linspace(-2.0, 2.0, num=400)

plt.figure(figsize=(10, 6))

plt.plot(x, f(x), label="f(x) curve", color='blue')

degree = 3

taylor = approximate\_taylor\_polynomial(f, 0, degree, 1)

print('taylor=', taylor)

plt.plot(x, taylor(x), label=f"degree={degree}", color='red', linestyle='--' )

plt.legend(bbox\_to\_anchor=(1.05, 1), loc='upper left',borderaxespad=0.0, shadow=True)

plt.xlabel("x")

plt.ylabel("y")

plt.title("Графік функції та наближення ,багаточленами Тейлора")

plt.tight\_layout()

plt.grid()

plt.show()