Лабораторна робота 4
Чисельні методи наближення функцій. Апроксимація та інтерполяція функцій

4.1. Мета роботи

Вивчення методу найменших квадратів (МНК) для практичного розв'язання задач апроксимації функцій. Вивчення методів інтерполяції для практичного розв'язання задачі наближення функцій. Придбання навичок використання вказаних методів для розв'язання задачі наближення функцій із застосуванням комп'ютера.

4.2. Методичні рекомендації щодо організації самостійної роботи

За темою лабораторної роботи студент повинен: *знати* загальне формулювання задачі наближення (апроксимації) функцій й, зокрема, задачі інтерполяції; *вміти* розв'язувати задачу апроксимації функцій методом найменших квадратів [1 – 6]; *вміти* розв'язувати задачу наближення функцій за допомогою кусково-лінійної й кусково-квадратичної інтерполяції й полінома Лагранжа [1 – 6]; *вміти* застосовувати процедури пакета R для розв'язання задачі наближення функцій.

4.2.1. Апроксимація функцій.

Задача апроксимації функцій у загальному вигляді формулюється наступним чином.

Нехай є деяка функція $f(x)$, $f:R^1 \to R^1$, про яку відомо, що в n точках $x_1, x_2, \ldots, x_n$ вона приймає, відповідно, значення $y_1, y_2, \ldots, y_n$, тобто $y_i = f(x_i)$, $i = \overline{1,n}$. Потрібно відновити її значення при інших значеннях $x \in [x_1, x_n]$.

Функція $y = f(x)$ може бути як невідомою, тобто її значення тільки вимірюються, а може бути й просто дуже складною для обчислень. У цих випадках функцію $f(x)$ намагаються замінити більш простою функцією $g(x,a)$, близькою до $f(x)$, тобто

$$f(x) \approx g(x,a), \tag{4.1}$$

де $a \in R^k$, $a = (a_1, a_2, \ldots, a_k)^T$ – деякі параметри функції g, вид якої відомий. Процес наближення однієї функції іншою називають **апроксимацією**, а функцію g при цьому називають **апроксимуючою** для f.

Найбільш відомим й ефективним з методів розв'язання задачі апроксимації функцій є **метод найменших квадратів**, суть якого полягає в наступному. Вводиться функція (від a) виду

$$\Phi(a) \equiv \sum_{i=1}^{n} \left[(y_i - g(x_i, a)\right]^2, \tag{4.2}$$

яку можна розглядати як міру відхилення функції $g(x,a)$ (по аргументу x) від функції $f(x)$ по сукупності точок $x_1, x_2, \ldots, x_n$. Тоді параметри a функції $g(x,a)$ можна визначити з умови найменшого відхилення $g(x,a)$ від $f(x)$, тобто параметри a визначаються як точка, у якій функція $\Phi(a)$ досягає по $a \in R^k$ мінімального значення (точка мінімуму). Нехай a^* – шукані параметри. При цьому величини $r_i \equiv y_i - g(x_i, a^*)$, $i = \overline{1,n}$, називаються **залишками** (**залишковими відхиленнями**) й використовуються для аналізу отриманого розв'язку задачі апроксимації.

У випадках, коли функція $g(x,a)$ є лінійною за параметрами a точку мінімуму функції $\Phi(a)$ можна знайти аналітично з необхідної умови мінімуму 1-го порядку $\Phi'(a) = 0$, тобто:

$$\frac{\partial \Phi}{\partial a_j} = 0, \forall j = \overline{1,k}. \tag{4.3}$$

Розглянемо загальний випадок, коли функція $g(x,a)$ лінійна по параметрам a, тобто має вигляд:

$$g(x,a) = \sum_{m=1}^{k} a_m \psi_m(x), \tag{4.4}$$

де $\psi_m(x)$, $m=1,\dots,k$ – деякі відомі функції. Тоді з (4.2) – (4.4) одержуємо

$$\forall j=\overline{1,k}:\ \frac{\partial \Phi}{\partial a_j}=-2\sum_{i=1}^{n}\left[y_i-g(x_i,a)\right]\psi_j(x_i)=-2\sum_{i=1}^{n}\left[y_i-\sum_{m=1}^{k}a_m\psi_m(x_i)\right]\psi_j(x_i)=0.$$

Звідки для $\forall j=\overline{1,k}$:

$$\sum_{m=1}^{k}a_m\left[\sum_{i=1}^{n}\psi_m(x_i)\psi_j(x_i)\right]=\sum_{i=1}^{n}y_i\,\psi_j(x_i). \qquad (4.5)$$

Таким чином, параметри a можуть бути знайдені як розв'язок лінійної системи алгебраїчних рівнянь (4.5), яку можна записати в матричному вигляді:

$$Ca=b, \qquad (4.6)$$

де $c_{jm}=\sum_{i=1}^{n}\psi_m(x_i)\psi_j(x_i)$, , $j=\overline{1,k}$, $m=\overline{1,k}$, $b_j=\sum_{i=1}^{n}y_i\,\psi_j(x_i)$, $j=\overline{1,k}$.

Розглянемо ще випадок, коли апроксимуюча функція є поліномом заданої $(k-1)$-ї степені. Тоді вона, у загальному випадку, може бути записана у вигляді $g(x,a)=b_0+b_1x+b_2x^2+\dots+b_{k-1}x^{k-1}$, де $a\in R^k$, $a_1=b_0$, $a_2=b_1,\dots$, $a_k=b_{k-1}$, b_j – коефіцієнти полінома. При цьому функція $g(x,a)$ є лінійною за параметрами a і має вигляд (4.4), де $\psi_1(x)=1$, $\psi_2(x)=x$,…, $\psi_k(x)=x^{k-1}$, тобто $\psi_m(x)=x^{m-1}$.

Тоді параметри a можна знайти, розв'язавши лінійну систему алгебраїчних рівнянь (4.6), з елементами:

$$c_{jm}=\sum_{i=1}^{n}x_i^{\,m-1}x_i^{\,j-1}=\sum_{i=1}^{n}x_i^{\,m+j-2},\ j=\overline{1,k},\ m=\overline{1,k}, \qquad (4.7)$$

$$b_j=\sum_{i=1}^{n}y_i x_i^{\,j-1},\ j=\overline{1,k}. \qquad (4.8)$$

У пакеті SciPy є процедура **minimize**, призначена для знаходження точки мінімуму довільної функції $\varphi(z)$, $z\in R^k$. Для її виклику необхідно визначити функцію $\varphi(z)$ й задати початкове наближення $z^0\in R^k$, тоді розв'язком є z* = minimize (φ, z^0).

У випадку, коли апроксимуюча функція лінійна по параметрам, система (4.5) може бути розв'язана за допомогою процедури *solve*, описаної у рекомендаціях до лабораторної роботи 2.

4.2.2. Інтерполяція функцій.

Загальна задача апроксимації функцій була сформульована в розділі 6.2.1. Якщо при цьому параметри $a_1, a_2, \ldots, a_k$ апроксимуючої функції $g(x,a)$ визначаються з умови збігу вихідної функції $f(x)$ й функції $g(x,a)$ в точках $x_1, x_2, \ldots, x_n$, тобто $g(x_i,a) = f(x_i)$, $\forall\ i = \overline{1,n}$, то такий спосіб апроксимації (наближення) називається **інтерполяцією.**

Кусково-лінійна інтерполяція. Такий вид інтерполяції полягає в тому, що на кожному окремому інтервалі (x_i, x_{i+1}) функція $f(x)$ наближається лінійною функцією $g_i(x)$, що проходить через дві точки (x_i, y_i), (x_{i+1}, y_{i+1}). Неважко перевірити, що така лінійна функція може бути також записана у вигляді:

$$g_i(x) = \frac{(x - x_{i+1})}{(x_i - x_{i+1})} \times y_i + \frac{(x - x_i)}{(x_{i+1} - x_i)} \times y_{i+1}. \tag{4.9}$$

Таким чином, при використанні кусково-лінійної інтерполяції для таблично заданої функції, при обчисленні наближеного значення функції $f(x)$ у деякій точці $\hat{x} \in (x_1, x_n)$ дотримуються наступного алгоритму. Спочатку знаходять інтервал (x_i, x_{i+1}), якому точка $\hat{x}$ належить, а потім обчислюють наближене значення функції $f(x)$ в точці $\hat{x}$ за формулою $f(\hat{x}) \approx g_i(\hat{x})$.

Якщо $f(x)$ двічі неперервно-диференційовна на відрізку $[x_1, x_n]$, то оцінка похибки при кусково-лінійній інтерполяції має вигляд [1 – 3, 5, 6]:

$$|f(x) - g(x)| \le \frac{M_2 h^2}{2},\ \forall x \in (x_1, x_n),$$

де $M_2 = \max\limits_{\xi \in (x_1, x_n)} |f''(\xi)|$, $h = \max\limits_{i=\overline{1,n}} |x_{i+1} - x_i|$.

Кусково-квадратична інтерполяція. Подібно кусково-лінійній інтерполяції можна використати для наближення $f(x)$ на кожному окремому інтервалі (x_i, x_{i+1}) й поліноми більш високого порядку, наприклад, квадратичну функцію $g_i(x)$, яка проходить вже через три точки (x_i, y_i), (x_{i+1}, y_{i+1}), (x_{i+2}, y_{i+2}). Неважко перевірити, що така квадратична функція може бути записана у вигляді:

$$g_i(x)=\frac{(x-x_{i+1})(x-x_{i+2})}{(x_i-x_{i+1})(x_i-x_{i+2})}\times y_i+\frac{(x-x_i)(x-x_{i+2})}{(x_{i+1}-x_i)(x_{i+1}-x_{i+2})}\times y_{i+1}+$$
$$+\frac{(x-x_i)(x-x_{i+1})}{(x_{i+2}-x_i)(x_{i+2}-x_{i+1})}\times y_{i+2}\,. \quad (4.10)$$

Якщо $f(x)$ тричі неперервно-диференційовна на відрізку $[x_1,x_n]$, то оцінка похибки при кусково-квадратичній інтерполяції має вигляд [1 – 3; 5; 6]:

$$|f(x)-g(x)|\le\frac{M_3h^3}{6},\ \forall x\in(x_1,x_n),$$

де $M_3=\max\limits_{\xi\in(x_1,x_n)}\left|f^{(3)}(\xi)\right|$, $h=\max\limits_{i=\overline{1,n}}|x_{i+1}-x_i|$.

Інтерполяційний многочлен Лагранжа. Якщо відомі значення функції $f(x)$ в n точках, то поліном мінімальної степені, що інтерполює $f(x)$, буде мати степінь $n-1$. Він називається **інтерполяційним многочленом Лагранжа** й має вигляд:

$$L_{n-1}(x)=\sum_{i=1}^{n}\left[\prod_{\substack{j=1\\ j\ne i}}^{n}\left(\frac{x-x_j}{x_i-x_j}\right)\right]y_i\,. \quad (4.11)$$

Неважко перевірити, що $L_{n-1}(x_i)=y_i$ для всіх $i=\overline{1,n}$.

Похибка наближення інтерпольованої функції многочленом Лагранжа сильно залежить від розкиду точок x_i, тобто чим менший інтервал (x_1,x_n), тим точніше $L_{n-1}(x)$ наближає значення функції $f(x)$ в точках $x\in(x_1,x_n)$.

Якщо функція $f(x)$ n разів неперервно-диференційовна на відрізку (x_1,x_n). Тоді для всіх $x\in(x_1,x_n)$:

$$|f(x)-L_{n-1}(x)|\le\frac{M_nh^n}{n!},$$

де $M_n=\max\limits_{\xi\in(x_1,x_n)}\left|f^{(n)}(\xi)\right|$, $h=\max\limits_{i=\overline{1,n}}|x_{i+1}-x_i|$ [1 – 3; 5; 6].

4.3. Контрольні приклади

Приклад 1. Залежність (HQ-характеристика насоса) створюваного насосом напору (H) від витрати води (Q) описується функцією

$H = H_F - S_F Q^2$, де H_F, S_F – параметри залежності. У результаті проведення натурних експериментів з насосом отримані такі дані:

Q, м^3/год.	5	20	30	35
H, м вод. ст.	209	194	165	142

Треба побудувати графік HQ-характеристики насоса та оцінити значення напору, який буде створювати насос при витраті води $Q = 32$ м^3/год.

Розв'язання 1. HQ-характеристика насоса задана таблицею даних, при цьому функція $H(Q) = H_F - S_F Q^2$ є апроксимуючою для неї. Використаємо метод найменших квадратів для оцінки параметрів H_F, S_F апроксимуючої функції.

Використання процедури minimize **пакету** SciPy**:**

Вводимо початкові дані:

```
import numpy as np

x = [5, 20, 30, 35]
y = [209, 194, 165, 142]
n = 4

# Задаємо вид апроксимуючої функції
def FunModel(x, a):
  y = a[0] - a[1]*x**2
  return y
```

Визначаємо цільову функцію методу МНК:

```
# Визначаємо цільову функцію методу МНК
def FunMNK(a):
  S = 0
  for i in range(n):
    S += (y[i] - FunModel(x[i], a))**2
  return S
```

Задаємо початкове значення для параметрів **a** апроксимуючої функції й оцінюємо параметри за допомогою функції **minimize** пакету **SciPy:**

```
from scipy.optimize import minimize

# Задаємо початкове значення для параметрів а апроксимуючої функції й
оцінюємо параметри
a0 = [210, 1]
```

```
res = minimize(FunMNK, a0)
print(res)
```

Для аналізу правильності отриманого розв'язку обчилюємо значення залишкових відхилень:

```
# Обчислюэмо значення апроксимуючої функції в заданих точках x
ym = [0]*n
for i in range(n):
  ym[i] = FunModel(x[i], a1)

# Обчислюэмо значення залишків
r = [0]*n
for i in range(n):
  r[i] = y[i] - ym[i]
print(r)

[-2.796365054851833, 3.2014094743775274, 2.1984421800166274, -
2.6034865613179363]
```

А для наочної інтерпретації результатів побудуємо графіки:

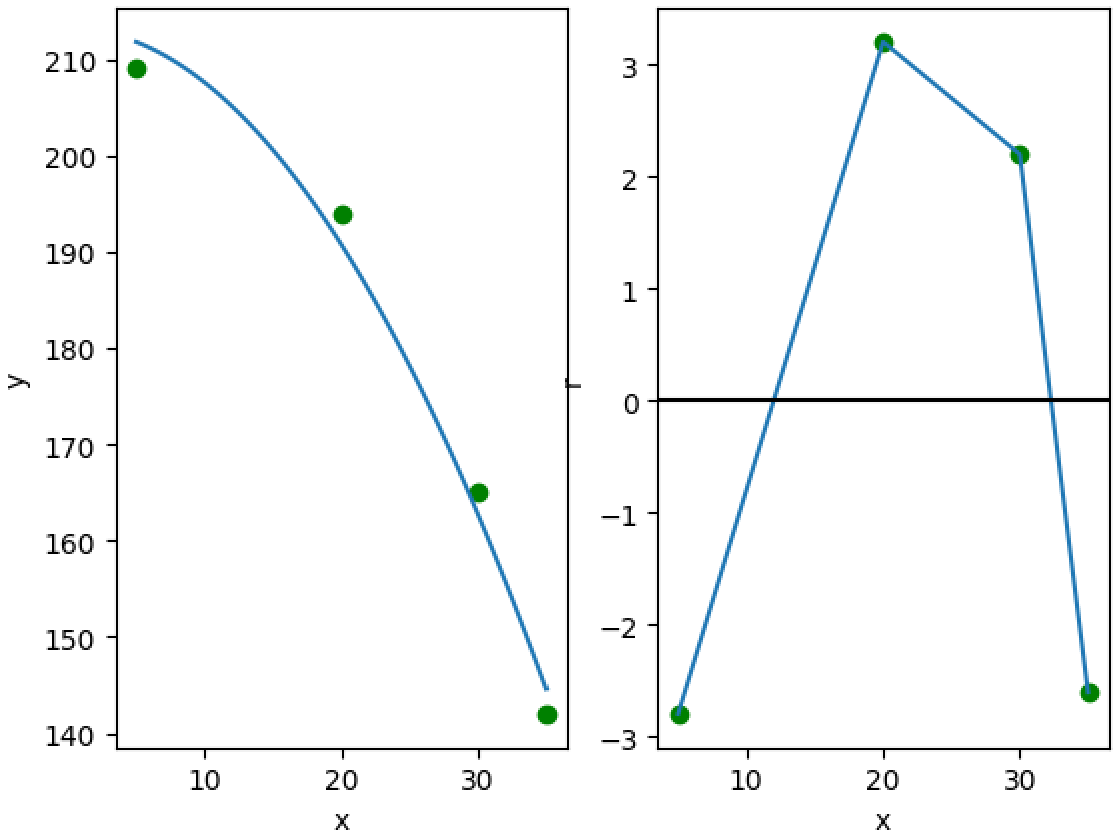
```
import matplotlib.pyplot as plt

a = x[0]
b = x[-1]
n1 = 100
x1 = np.linspace(a, b, n1)
y1 = [0]*n1
for i in range(n1):
  y1[i] = FunModel(x1[i], a1)

fig, (ax1, ax2) = plt.subplots(1, 2)

ax1.scatter(x, y, color = "green")
ax1.plot(x1, y1)
ax1.set(xlabel='x', ylabel='y')

ax2.scatter(x, r, color = "green")
ax2.plot(x, r)
ax2.set(xlabel='x', ylabel='r')
ax2.axhline(0, color='k') #x-axis line
```



Як видно з першого графіку, апроксимуюча функція «гарно лягає» на данні експерименту. З другого графіку важко судити про поведінку залишків, бо число самих експериментів дуже мале (всього 4).

Розв'язання 2. Використання процедури solve пакета numpy

Апроксимуюча функція $H(Q)=H_F-S_FQ^2$ є поліномом 2-ї степені й може бути записана у вигляді $g(x,a)=a_1+a_2x^2$, де $a\in R^2$ – коефіцієнти полінома $(a_1=H_F, a_2=S_F)$. При цьому функція $g(x,a)$ є лінійною за параметрами a і має вигляд (4.4), де $\psi_1(x)=1$, $\psi_2(x)=-x^2$. Таким чином, у даному прикладі $n=4$, $k=2$. Тоді параметри a можна знайти, розв'язавши лінійну систему рівнянь (4.6).

Для оцінки параметрів апроксимуючої функції виду (4.4) в загальному вигляді запишемо процедуру *MNKsolve*:

```
import numpy as np

def MNKsolve(x, y, n, Fi, k):
  # Формуємо матрицю значень векторної функції Fi в точках x[i] по
рядках
  FiX = np.zeros((n, k))
  for i in range(n):
    FiX[i,] = Fi(x[i])
```

```
    # Формуємо матрицю C
    C = np.zeros((k, k))
    for j in range(k):
      for m in range(k):
        S = 0
        for i in range(n):
          S = S + FiX[i,j]* FiX[i,m]
        C[j,m] = S

    # Формуємо вектор d
    d = np.zeros(k)
    for j in range(k):
      S = 0
      for i in range(n):
        S = S + y[i]* FiX[i,j]
      d[j] = S

    # Розв'язуємо систему C*a = d
    a = np.linalg.solve(C, d)

    return a
```

Застосуємо цю процедуру для розв'язання поставленої задачі:

```
# Вводимо вхідні дані
x = [5, 20, 30, 35]
y = [209, 194, 165, 142]
n = 4
k = 2

# Вводимо векторне функцію Fi
def Fi(x):
  z = [0]*k
  z[0] = 1
  z[1] = -x**2
  return z

# Знаходимо параметри апроксимуючої функції
a2 = MNKsolve(x, y, n, Fi, k)
print(a2)
```

Як бачимо, значення параметрів апроксимуючої функції такі ж самі.

Приклад 2. Використовуючи кусково-лінійну й кусково-квадратичну інтерполяцію, а також поліном Лагранжа, обчислити в точці $\hat{x} = 3$ наближене значення функції $f(x)$, заданої таблицею:

x	0,5	2,5	4,5	6,5	8,5
y	4,0	205,	1240	3840	8750

Розв'язання (вручну).

Кусково-лінійна інтерполяція

Оскільки точка $\hat{x}=3$ належить інтервалу $(2.5;4.5)$, то наближене значення $f(3)$ можна обчислити за формулою (4.9), тобто

$$f(3)\approx g_2(3)=\frac{(3-4.5)}{(2.5-4.5)}\times 205+\frac{(3-2.5)}{(4.5-2.5)}\times 1240=463.75\,.$$

Кусково-квадратична інтерполяція

Оскільки точка $\hat{x}=3$ належить інтервалу $(2.5;4.5)$, то наближене значення $f(3)$ можна обчислити за формулою (4.10), тобто

$$f(3)\approx g_2(3)=\frac{(3-4.5)(3-6.5)}{(2.5-4.5)(2.5-6.5)}\times 205+\frac{(3-2.5)(3-6.5)}{(4.5-2.5)(4.5-6.5)}\times 1240+$$
$$+\frac{(3-2.5)(3-4.5)}{(6.5-2.5)(6.5-4.5)}\times 3840=497\,.$$

Інтерполяційний многочлен Лагранжа

Проводимо обчислення за формулою (4.11):

$$f(3)\approx L_4(3)=\frac{(3-2.5)(3-4.5)(3-6.5)(3-8.5)}{(0.5-2.5)(0.5-4.5)(0.5-6.5)(0.5-8.5)}\times 4+$$
$$+\frac{(3-0.5)(3-4.5)(3-6.5)(3-8.5)}{(2.5-0.5)(2.5-4.5)(2.5-6.5)(2.5-8.5)}\times 205+$$
$$+\frac{(3-0.5)(3-2.5)(3-6.5)(3-8.5)}{(4.5-0.5)(4.5-2.5)(4.5-6.5)(4.5-8.5)}\times 1240+$$
$$+\frac{(3-0.5)(3-2.5)(3-4.5)(3-8.5)}{(6.5-0.5)(6.5-2.5)(6.5-4.5)(6.5-8.5)}\times 3840+$$
$$+\frac{(3-0.5)(3-2.5)(3-4.5)(3-6.5)}{(8.5-0.5)(8.5-2.5)(8.5-4.5)(8.5-6.5)}\times 8750=357.247\,.$$

Приклад 3. Для функції $f(x)$ з прикладу 2 обчислити в точці $\hat{x}=3$ наближене значення та побудувати графіки інтерполяційних функцій, застосовуючи процедури з пакету *SkyPi* (модуль *interpolate)* для Python.

Розв'язання (Використання процедур пакета SkyPi).

Вводимо початкові дані:

```
x = [0.5, 2.5, 4.5, 6.5, 8.5]
y = [4.0, 205, 1240, 3840, 8750]
```

```
# Кусково-лінійна інтерполяція
```

```
from scipy.interpolate import interp1d

interp_func1 = interp1d(x, y)

xnew = 3.0
ynew = interp_func1(xnew)
print(ynew)
```

```
463.75
```

```
# Spline Interpolation
from scipy.interpolate import UnivariateSpline

interp_func2 = UnivariateSpline(x, y)

ynew = interp_func2(xnew)
print(ynew)
```

```
357.3156249999997
```

Як бачимо, наближене значення функції $f(x)$ в точці $\hat{x} = 3$, що обчислено за допомогою сплайн-інтерполяції, дуже близьке до значення, що обчислено за допомогою інтерполяційного поліному Лагранжа в прикладі 2.

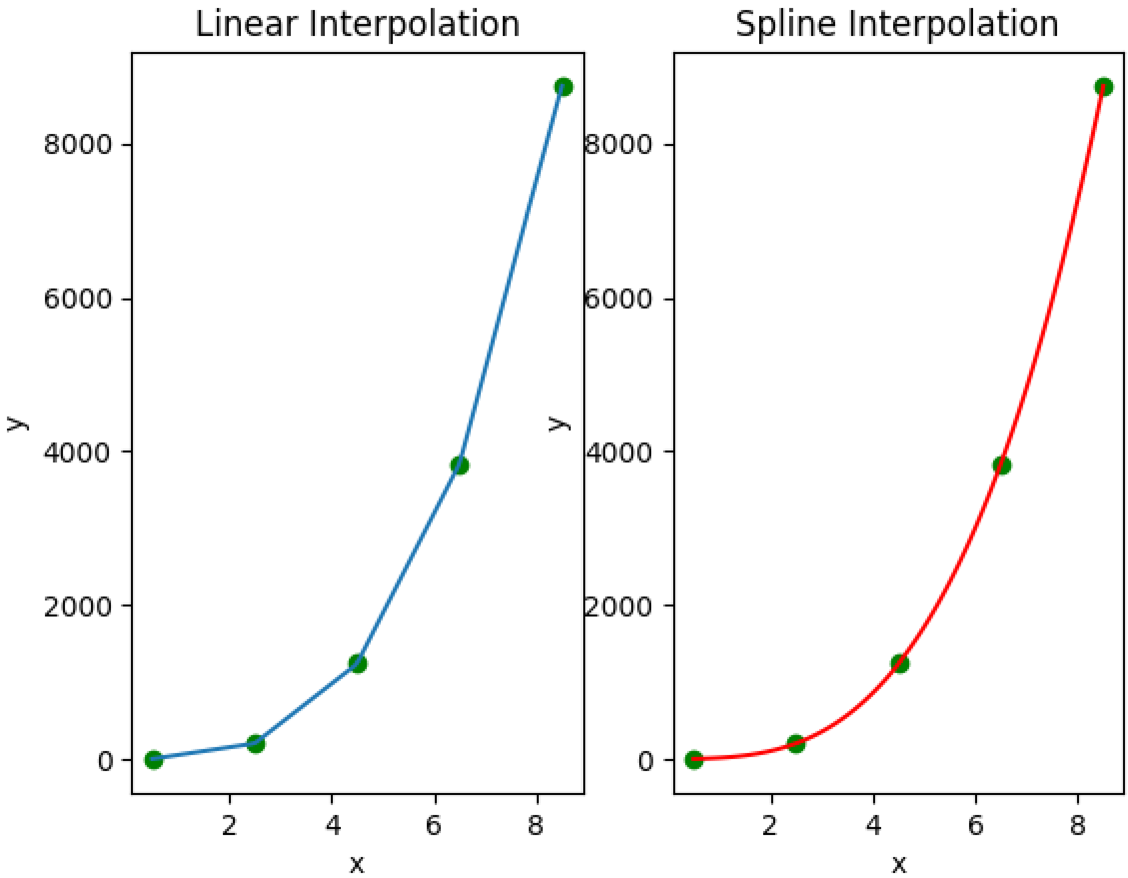
Побудуємо графіки інтерполяційних функцій

```
import matplotlib.pyplot as plt

n1 = 100
x1 = np.linspace(x[0], x[-1], n1)
y1 = [0]*n1
y2 = [0]*n1
for i in range(n1):
  y1[i] = interp_func1(x1[i])
  y2[i] = interp_func2(x1[i])

fig, (ax1, ax2) = plt.subplots(1, 2)
ax1.scatter(x, y, color = "green")
ax1.plot(x1, y1)
ax1.set(xlabel='x', ylabel='y')
ax1.set(title='Linear Interpolation')

ax2.scatter(x, y, color = "green")
ax2.plot(x1, y2, color = "red")
ax2.set(xlabel='x', ylabel='y')
ax2.set(title='Spline Interpolation')
```



4.4. Порядок виконання роботи і варіанти завдань

4.4.1. Зміст звіту

У теоретичній частині роботи необхідно коротко описати:

- постановку задачі наближення функцій;
- метод найменших квадратів для апроксимації функцій;
- методи інтерполяції функцій.

У практичній частині роботи необхідно:

- запрограмувати у вигляді окремого модуля метод найменших квадратів для оцінки параметрів будь-якої апроксимуючої функції;
- запрограмувати у вигляді окремого модуля метод найменших квадратів для оцінки параметрів апроксимуючого полінома деякої степені;
- привести тексти складених програм;
- підібрати з застосуванням складених програм найкращу апроксимацію для функції, що задана таблично, серед функцій виду:

1) $y = b_0 + b_1 x$;

2) $y = b_0 + b_1 x + b_2 x^2$;

3) $y = b_0 + b_1 x + b_2 x^2 + b_3 x^3$;

4) $y = b_0 + b_1 \sin(b_2 x) + b_3 \cos(b_2 x)$;

5) $y = b_0 \exp(b_1 x)$.

Побудувати графіки апроксимуючих функцій і відповідних залишків.

- запрограмувати у вигляді окремих модулів методи кусково-лінійної та кусково-квадратичної інтерполяції та інтерполяційний поліном Лагранжа;
- привести тексти складених програм;
- обчислити в точці $\hat{x} = \dfrac{(x_1 + x_2)}{2}$ наближене значення функції $f(x)$, заданої таблично в індивідуальному завданні;
- побудувати графіки всіх інтерполяційних функцій;
- побудувати графік інтерполяції сплайном з застосуванням функцій пакету *SkyPi (*модуль *interpolate)*.

4.4.2. Варіанти індивідуальних завдань

Варіант 1

x	–5	–4	–3	–2	–1	0	1
y	54,9	37,6	23,7	14,4	8,3	6,6	8,9

Варіант 2

x	0,5	1,0	1,5	2,0	2,5	3,0	3,5	4,0
y	0,4	0,3	1,0	1,7	2,1	3,4	4,1	5,8

Варіант 3

x	4,5	5,0	5,5	6,0	6,5	7,0	7,5	8,0
y	7,7	9,4	11,4	13,6	15,6	18,6	21,2	24,1

Варіант 4

x	0,4	0,8	1,2	1,6	2,0	2,4	2,8	3,2	3,6
y	0,43	0,94	1,91	3,01	4,0	4,56	6,45	8,59	11,15

Варіант 5

x	4,0	4,4	4,8	5,2	5,6	6,0	6,4	6,8
y	13,88	16,93	20,47	24,15	28,29	32,61	37,41	42,39

Варіант 6

x	1	2	3	4	5	6
y	2,11	2,45	2,61	2,73	2,75	2,81

Варіант 7

x	0,3	0,6	0,9	1,2	1,5	1,8
y	4,39	4,75	4,98	5,11	5,12	5,18

Варіант 8

x	1	2	3	4	5	6
y	0,1	0,21	0,43	0,51	0,62	0,81

Варіант 9

x	1,0	1,5	2,0	2,5	3,0	3,5	4,0
y	4,11	4,16	4,23	4,29	4,36	4,42	4,53

Варіант 10

x	0,5	1,0	1,5	2,0	2,5	3,0	3,5	4,0
y	2,47	2,86	3,01	2,91	2,55	2,11	2,61	1,25

Варіант 11

x	–1	0	1	2	3	4	5
y	2,4	4,3	4,3	0,1	–6,1	–15,5	–26,5

Варіант 12

x	0,16	0,17	0,18	0,19	0,20	0,21
y	6,61	6,5	6,1	5,80	4,3	3,81

Варіант 13

x	–5	–4	–3	–2	–1	0	1
y	45,0	32,5	20,7	13,9	10,4	6,5	11,1

Варіант 14

x	–5	–4	–3	–2	–1	0	1
y	37,0	29,6	18,5	16,2	9,2	9,5	9,1

Варіант 15

x	–1	0	1	2	3	4	5
y	–0,6	3,4	3,4	–2,2	–12,6	–28,8	–48,9

4.5. Контрольні запитання

1. Сформулюйте постановку задачі наближення функцій.
2. У чому полягає метод найменших квадратів для апроксимації функцій?
3. Яким чином проводиться аналіз побудованої апроксимуючої функції?
4. У чому полягає кусково-лінійна інтерполяція функцій? Коли її слід застосовувати?
5. У чому полягає кусково-квадратична інтерполяція функцій? Коли її слід застосовувати?
6. Що називається інтерполяційним поліномом Лагранжа? Коли його слід застосовувати?
7. Що називають сплайном? Чим він відрізняється від інших методів інтерполяції?