

Модульна контрольна робота

Методи обчислень

Виконав:

Студент 3 курсу ФТІ

групи ФІ-92

Поночевний Назар Юрійович

Варіант 12 -> 30

Завдання 1:

Побудувати поліном $P_4(x)$ за оптимально розташованими вузлами на відрізку $[-30; 31]$.

Парні варіанти: формула Лагранжа.

- Спочатку виберемо оптимальні вузли. Можна взяти рівномірно, але вузли Чебишева мають менше похибку, бо знижують вплив феномена Рунге, тому я виберу їх:

```
def chebishef_nodes(n, a, b):
    nodes = []
    for k in range(1, n + 1):
        nodes.append((0.5*(a + b)) + ((0.5*(b - a))*math.cos(((2*k)-1)*math.pi)/(2*n)))
    return nodes

X = sorted(chebishef_nodes(5, -30, 31))
print(X)

[-28.507223747002183, -17.42745019492043, 0.50000000000000019, 18.427450194920432, 29.507223747002183]
```

- По формулі Лагранжа:

$$L(x) = y_0 \frac{(x-x_1)(x-x_2)(x-x_3)(x-x_4)}{(x_0-x_1)(x_0-x_2)(x_0-x_3)(x_0-x_4)} + \dots + y_4 \frac{(x-x_0)(x-x_1)(x-x_2)(x-x_3)}{(x_4-x_0)(x_4-x_1)(x_4-x_2)(x_4-x_3)}$$

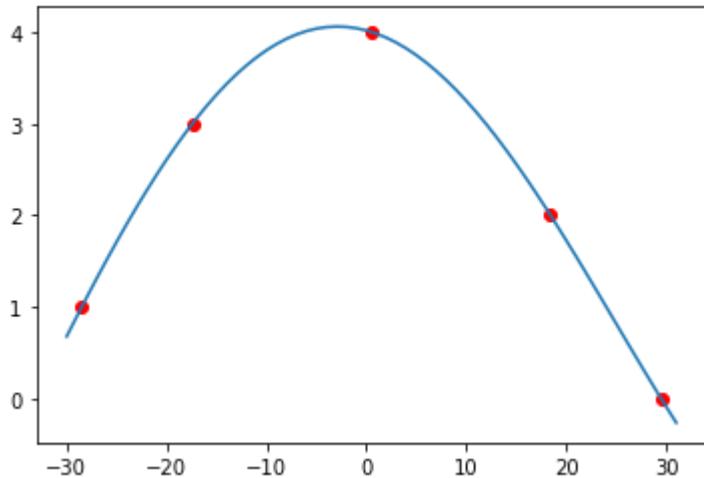
X	Y
-28.507224	1
-17.427450	3
0.500000	4
18.427450	2
29.507224	0

- Отримуємо поліном

```
x = sp.Symbol('x')
L = 0
for i in range(len(y)):
    nominator = y[i]
    for j in range(len(y)):
        if j != i:
            nominator *= (x - X[j])
    denominator = 1
    for j in range(len(y)):
        if j != i:
            denominator *= (X[i] - X[j])
    L += (nominator / denominator)
L = sp.simplify(L)
print(L)

9.75984019461932e-7*x**4 + 1.85337650366828e-5*x**3 - 0.00501011585067425*x**2 - 0.0294784437516135*x + 4.01598937311885
```

- Перевіримо (червоним зображені вузли, синім поліном)



Завдання 2:

Функція задана таблицею:

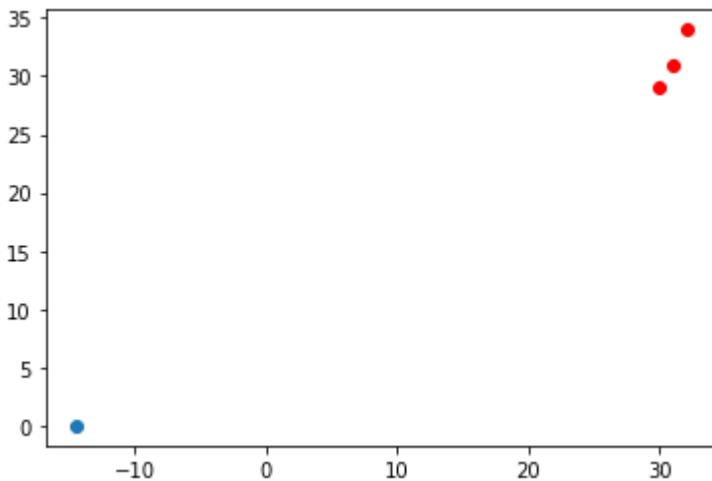
X	Y
30	29
31	31
32	34

Використовуючи зворотну лінійну інтерполяцію, знайти наближене значення кореня.

- 1) Треба знайти значення X при Y = 0. Оскільки 0 більше за лівий кінець, то формулу починаємо з нього:

$$\begin{aligned}
 X &= x_0 + (0 - y_0) f(y_0, y_1) + (0 - y_1) f(y_1, y_2) \\
 &= 30 + (-29) \cdot \frac{1}{2} + (-29)(-31) \cdot \frac{f(y_2) - f(y_1)}{y_2 - y_1} - \frac{1}{2} \\
 &= 30 - 14,5 + 8,99 \cdot \frac{\frac{32 - 31}{34 - 31} - \frac{1}{2}}{34 - 29} = \\
 &= 30 - 14,5 + \cancel{8,99} \quad (-29,97) \approx -14,47
 \end{aligned}$$

- 2) Перевіримо (червоним зображені вузли, синім корінь)



Завдання 3:

Парні варіанти: степеневим методом знайти максимальне за модулем власне число (зробити дві ітерації).

```
[[36  1  2]
 [ 1 35  1]
 [ 2  1 32]]
```

- 1) Виконуємо 2 ітерації степеневого метода:

```
X = [np.array([1, 1, 1]).reshape(-1, 1)]
lamb = []
for k in range(2):
    X.append(A.dot(X[k]))
    lamb.append(X[k + 1][0] / X[k][0])
print(f"Власне значення: {lamb[-1]}")
print(f"Власний вектор:\n{X[-1]}")
```

Власне значення: [38.74358974]
 Власний вектор:
 [[1511]
 [1369]
 [1235]]

- 2) Перевіримо

```
print(A.dot(X[-1]))
print(lamb[-1]*X[-1])
```

[[58235]
 [50661]
 [43911]]
[[58541.56410256]
 [53039.97435897]
 [47848.3333333]]

Завдання 4:

Обчислити результат у з точністю до 2x значущих цифр. Вихідні дані округлити до 3x значущих цифр. Оцінити похибку результату - Δ чи $\hat{\Delta}$ (що зручніше):

$$30. y = x_1 + x_2 + 5x_3, x_1 = 0,4321, x_2 = 0,9876, x_3 = 0,1025,$$

- 1) Виконуємо обрахунки:

Округлення вих. даних до 3x знач. цифр:

$$x_1 = 0,432; x_2 = 0,987; x_3 = 0,1025.$$

Рахуємо результат з точ. до 2x знач. цифр:

$$y = 0,432 + 0,987 + 5 \cdot 0,1025 \approx 1,9$$

Оцінюємо похибку: $A = 1,9322$

$$\underline{\Delta = |A - y| = 0,0322}.$$

Завдання 5:

Методом LU-декомпозиції знайти розв'язок системи $Ax=b$.

```
A = np.array([[36, 1, 1], [1, 35, 1], [2, 1, 34]])
b = np.array([33, 4, 30]).reshape(-1, 1)
print(A)
print(b)

[[36  1  1]
 [ 1 35  1]
 [ 2  1 34]]
[[33]
 [ 4]
 [30]]
```

- 1) Виконуємо LU-декомпозицію:

```

def lu(A):
    n = A.shape[0]
    U = A.copy()
    L = np.eye(n, dtype=np.double)
    for i in range(n):
        factor = U[i + 1:, i] / U[i, i]
        L[i + 1:, i] = factor
        U[i + 1:] -= factor[:, np.newaxis] * U[i]
    return L, U
L, U = lu(A)
print(L)
print(U)

[[1.          0.          0.          ]
 [0.02777778 1.          0.          ]
 [0.05555556 0.02700556 1.          ]]
[[36.          1.          1.          ]
 [ 0.          34.97222222 0.97222222]
 [ 0.          0.          33.91818904]]

```

2) Проводимо 2 обратних хода:

```

def backward_pass1(L, b):
    y = np.zeros_like(b, dtype=np.double)
    n = len(y)
    y[0, 0] = b[0, 0] / L[0, 0]
    for i in range(1, n):
        suma = sum([L[i, j] * y[j, 0] for j in range(i)])
        y[i, 0] = (b[i, 0] - suma) / L[i, i]
    return y

def backward_pass2(U, y):
    x = np.zeros_like(y, dtype=np.double)
    n = len(x)
    x[n - 1, 0] = y[n - 1, 0] / U[n - 1, n - 1]
    for i in range(n - 2, -1, -1):
        suma = sum([U[i, j] * x[j, 0] for j in range(n - 1, i, -1)])
        x[i, 0] = (y[i, 0] - suma) / U[i, i]
    return x

y = backward_pass1(U, b)
print(f"\ny:{\n{y}}")

x = backward_pass2(L, y)
print(f"\nx:{\n{x}}")

```

```

y:
[[0.91666667]
 [0.11437649]
 [0.88448118]]

x:
[[0.91666667]
 [0.11437649]
 [0.88448118]]

```

Завдання 6:

Занулити елемент в позиції (4, 1) матриці прямим методом Якобі:

$$\begin{bmatrix} [1 & 2 & 3 & 30] \\ [2 & 3 & 4 & 5] \\ [3 & 4 & 5 & 6] \\ [30 & 5 & 6 & 7] \end{bmatrix}$$

- 1) Виконуємо обчислення:

$$\xrightarrow{\times 10} \begin{pmatrix} 1 & 2 & 3 & 30 \\ 2 & 3 & 4 & 5 \\ 3 & 4 & 5 & 6 \\ 30 & 5 & 6 & 7 \end{pmatrix} = \begin{pmatrix} 1 & 2 & 3 & 30 \\ 2 & 3 & 4 & 5 \\ 3 & 4 & 5 & 6 \\ \boxed{0} & +35 & 44 & 53 \end{pmatrix}$$

Завдання 7:

$30 \bmod 10 + 1 = 1$: Знайти кількість вірних цифр у наближеному числі а, якщо задано $a = 29,00$, $A = 28,99$.

- 1) Виконуємо обчислення:

$$\Delta = |A - a| = 0,99;$$

цифра 0 більше $0,99$ - суміжна
 цифра 0 більше $0,99$ - суміжна
 цифра 9 більше $0,99$ - вірна \Rightarrow цифра 2 - вірна.
Маємо 2 вірні цифри.

Завдання 8:

Задати таблицю значень $y = \cos x$ від 45 до 65 гр. з кроком 5 . Обчислити значення $\cos 44$ та $\cos 66$. Точність - 0.01 . Для заданої точності достатньо полінома 1 степеня.

- 1) Задамо таблицю:

X = [i for i in range(45, 70, 5)]
Y = [math.cos(x) for x in X]
df = pd.DataFrame({'X': X, 'Y': Y})
df
x y
0 45 0.525322
1 50 0.964966
2 55 0.022127
3 60 -0.952413
4 65 -0.562454

- 2) Будуємо за допомогою 1 інт. формулі Ньютона (бо 44 більше зліва до 45):

$$x_0 = 45^\circ, x = 44^\circ, q = (44 - 45)/5 = -0,2$$

$$\cos 44^\circ = 0,53 + (-0,2) \cdot 0,43 \approx 0,44$$

- 3) Будуємо за допомогою 2 інт. формулі Ньютона (бо 66 більше справа до 65):

$$x_n = 65^\circ, x = 66^\circ, q = (66 - 65)/5 = 0,2$$

$$\cos 66^\circ = -0,56 + 0,2 \cdot 0,39 \approx -0,48$$

- 4) Помилка: тільки зараз помітив, що побудував таблицю в радіанах, а не градусах. Проте спосіб вирішення такий самий.

Завдання 9:

Знайти формулі для 1,2 похідної, побудованих по інт. поліномах. Парні варіанти: записати 1 інтерполяційну формулу Ньютона, поліном 2 степеня. Знайти 1 похідну безпосереднім диференціюванням формулі з завдання 8.

- 1) Порахуємо похідну інтерполяційної формули Ньютона, поліному 2 степеня:

$$P_n(x) = f(x_0) + (x - x_0) \frac{f(x_1) - f(x_0)}{x_1 - x_0} +$$

$$+ (x - x_0)(x - x_1) \frac{\frac{f(x_2) - f(x_1)}{x_2 - x_1} - \frac{f(x_1) - f(x_0)}{x_1 - x_0}}{x_2 - x_0}$$

$$\frac{\partial P_n}{\partial x} = \frac{f(x_1)(-2x_0 + 2x_2 + x_0^2 - x_2^2)}{(x_0 - x_1)(x_0 - x_2)(x_1 - x_2)} +$$

$$+ \frac{f(x_0)(2x - x_1 - x_2)}{(x_0 - x_1)(x_0 - x_2)} + \frac{f(x_2)(2x - x_0 - x_1)}{(x_0 - x_2)(x_1 - x_2)}$$

- 2) Порахуємо поліном 4 степеня з 8 задачі та теж візьмемо похідну:

```
print(P)
```

```
3.0e-6*x**4 + 0.001165*x**3 - 0.247922*x**2 + 14.444744*x - 266.040329
```

```
print(sp.diff(P, x))
```

```
1.2e-5*x**3 + 0.003495*x**2 - 0.495844*x + 14.444744
```

Завдання 10:

Обчислити інтеграл використовуючи розкладення в ряд та обчислення відповідних інтегралів, що виникли після цього, по методу трапецій. A = 30. Точність = 0,01.

- 1) Розкладемо $\sin x$ в степеневий ряд ($n = 2$ виходячи з точності):

$$\int_0^1 \frac{\sin x}{1+30x} dx = \int_0^1 \frac{x}{1+30x} dx + \int_0^1 \frac{x^3}{3!(1+30x)} dx$$

- 2) Методом трапецій порахуємо інтеграли:

```
def f1(x):
    return x / (1 + 30 * x)

def f2(x):
    return x**3 / (2 * 3 * (1 + 30 * x))

def trap(f, a, b, n):
    h = (b - a) / n
    x = a
    s = f(x) - f(b)
    for k in range(1, n + 1):
        x = x + h
        s = s + 2 * f(x)
    result = (h / 2) * s
    return result

integ1 = trap(f1, 0, 1, 5)
integ2 = trap(f2, 0, 1, 5)
print(f"Result = {integ1} - {integ2} = {round(integ1 - integ2, 2)}")

Result = 0.027809727793428985 - 0.0018014462458876724 = 0.03
```