Міністерство освіти і науки України
Дніпровський національний університет імені Олеся Гончара
Факультет прикладної математики та інформаційних технологій
Кафедра комп’ютерних технологій

ЗВІТ З ЛАБОРАТОРНОЇ РОБОТИ № 2

з курсу «Методи оптимізації»

на тему «Методи розв’язання задач одновимірної оптимізації»

Варіант №14

Виконав:
студент гр. ПА-22-2
Овдієнко Андрій

Дніпро
2025

Зміст

1. Постановка задачі

1.1 Мета

Познайомитись практично з ітераційними методами розв`язання задач одновимірної оптимізації.

1.2 Основна постановка задачі

Знайти мінімум функції f (x) відрізку [a;b] з точністю $\varepsilon = 10^{-4}$ методами ділення навпіл, золотого перерізу, Фібоначчі. Цільова функція f (x) та відрізок [a,b] визначаються номером індивідуального завдання (Рисунок 1).

14	$y = x^3 - 3x + 2 \to \min$	$[0, 2]$

Рисунок 1 – Номер індивідуального завдання, цільова функція f(x) та відрізок [a;b].

2. Теоретична частина

2.1 Метод ділення відрізку навпіл (метод дихотомії)

Нехай дана цільова функція f(x), яка має мінімум на відрізку [a;b]. Метою є знайти наближене значення критичної точки (мінімума) цієї функції.

Першим кроком перепозначимо, що $[a_1; b_1] = [a; b]$.

Другим кроком знайдемо: λ_1, μ_1 – за формулами:

$$\begin{cases} \lambda_1 = \dfrac{a_1 + b_1 - \delta}{2} \\ \mu_1 = \dfrac{a_1 + b_1 + \delta'}{2} \end{cases}$$

де δ – стала, і завдовільняє умові: $0 < \delta < b - a$. Зрозуміло, що λ_1, μ_1 розташовані симетрично відносно центра $[a_1; b_1]$.

Далі обчислюють та порівнюють значення цільової функції $f(\lambda_1)$ та $f(\mu_1)$.

Якщо $f(\lambda_1) \leq f(\mu_1)$ - то маємо наступне: $a_2 = a_1;\ b_2 = \mu_1$.

Якщо $f(\lambda_1) > f(\mu_1)$ - то маємо наступне: $a_2 = \lambda_1;\ b_2 = b_1$.

У результаті ми звузили наш проміжок [a;b], а точка мінімума залишилася лежати в ньому. Довжина відрізка $[a_2; b_2]$ буде наступною:

$$[a_2; b_2] = \frac{b - a - \delta}{2^1} + \delta.$$

Це було для першого кроку. Тепер уявімо, що ми на якомусь k-тому кроці, маємо проміжок $[a_k; b_k]$, функція f(x) має точку мінімума на цьому проміжку.

Довжина цього відрізку буде наступною:

$$b_k - a_k = \frac{b - a - \delta}{2^{k-1}} + \delta, k \geq 2.$$

Точки λ_k, μ_k рахуються аналогічно:

$$\begin{cases} \lambda_k = \dfrac{a_k + b_k - \delta}{2} \\ \mu_k = \dfrac{a_k + b_k + \delta'}{2} \end{cases}$$

де δ – стала, і завдовільняє умові: $0 < \delta < b_k - a_k$.

Далі обчислюють та порівнюють значення цільової функції $f(\lambda_k)$ та $f(\mu_k)$.

Якщо $f(\lambda_k) \leq f(\mu_k)$ - то маємо наступне: $a_{k+1} = a_k;\ b_{k+1} = \mu_k$.

Якщо $f(\lambda_k) > f(\mu_k)$ - то маємо наступне: $a_{k+1} = \lambda_k;\ b_{k+1} = b_k$.

Довжина отриманого відрізку буде наступною:

$$b_{k+1} - a_{k+1} = \frac{b - a - \delta}{2^k} + \delta, k \geq 2.$$

Відрізок $[a_{k+1}; b_{k+1}]$ містить точку мінімуму функції $f(x)$ на $[a,b]$.

Критерієм закінчення ітераційного процесу є умова: $b_{k+1} - a_{k+1} < \varepsilon$, де ε - задана точність, $\varepsilon > \delta$.

Нехай є проміжок $[a_{k+1}; b_{k+1}]$,коли точність вже є допустимою. Слід шукати одне з двох значень для набляження x^*:

Якщо $f(\lambda_k) \leq f(\mu_k)$ - то маємо наступне: $x^* = \lambda_k$;

Якщо $f(\lambda_k) > f(\mu_k)$ - то маємо наступне: $x^* = \mu_k$.

І тоді $x^* = max\{\lambda_k, \mu_k\}$. Помилка буде наступною:

$$|x_* - x^*| \leq \max\{b_{k+1} - x^*; x^* - a_{k+1}\} = \frac{b - a - \delta}{2^k},$$

x_* - точне значення, x^* - наближене значення.

Але помилку можна зменшити, якщо взяти за розв'язок середнє арифметичне a_{k+1}, b_{k+1}. Тобто $x^* = \frac{a_{k+1} + b_{k+1}}{2}$. Помилка буде наступною (меншою):

$$|x_* - x^*| \leq \frac{a_{k+1} + b_{k+1}}{2} = \frac{b - a - \delta}{2^{k+1}} + \frac{\delta}{2},$$

x_* - точне значення, x^* - наближене значення.

Наочно можна зрозуміти, що помилка менша, якщо підставити $\delta = 0$.

$$\frac{b - a - \delta}{2^k} > \frac{b - a - \delta}{2^{k+1}} + \frac{\delta}{2}$$

$$\frac{b - a}{2^k} > \frac{b - a}{2^{k+1}}$$

$$\frac{1}{2^k} > \frac{1}{2^{k+1}}$$

$$2^{-k} > 2^{-(k+1)}$$

$$-k > -(k+1)$$

$$k < k + 1$$

таким чином – середнє арифметичне a_{k+1}, b_{k+1} дає наближенний, більш точний результат, ніж при порівнянні $f(\lambda_k), f(\mu_k)$.

2.2 Метод золотого перетину

Нехай дана цільова функція f(x), яка має мінімум на відрізку [a;b]. Метою є знайти наближене значення критичної точки (мінімума) цієї функції.

Першим кроком перепозначимо, що $[a_1; b_1] = [a; b]$.

Другим кроком знайдемо: λ_1, μ_1 – за формулами:

$$\begin{cases} \lambda_1 = a_1 + \dfrac{3 - \sqrt{5}}{2}(b_1 - a_1) \\ \mu_1 = a_1 + \dfrac{\sqrt{5} - 1}{2}(b_1 - a_1) \end{cases}.$$

Далі обчислюють та порівнюють значення цільової функції $f(\lambda_1)$ та $f(\mu_1)$.

Якщо $f(\lambda_1) \leq f(\mu_1)$ - то маємо наступне: $a_2 = a_1;\ b_2 = \mu_1$.

Якщо $f(\lambda_1) > f(\mu_1)$ - то маємо наступне: $a_2 = \lambda_1;\ b_2 = b_1$.

У результаті ми звузили наш проміжок [a;b], а точка мінімума залишилася лежати в ньому. Довжина відрізка $[a_2; b_2]$ буде наступною:

$$b_2 - a_2 = \frac{\sqrt{5} - 1}{2}(b - a).$$

Це було для першого кроку. Тепер уявімо, що ми на якомусь k-тому кроці, маємо проміжок $[a_k; b_k]$, функція f(x) має точку мінімума на цьому проміжку.

Довжина цього відрізку буде наступною:

$$b_k - a_k = (\frac{\sqrt{5} - 1}{2})^{k-1}(b - a).$$

Точки λ_k, μ_k рахуються аналогічно:

$$\begin{cases} \lambda_k = a_k + \dfrac{3 - \sqrt{5}}{2}(b_k - a_k) \\ \mu_k = a_k + \dfrac{\sqrt{5} - 1}{2}(b_k - a_k) \end{cases}.$$

Довжина отриманого відрізку буде наступною:

$$b_{k+1} - a_{k+1} = (\frac{\sqrt{5}-1}{2})^k (b-a).$$

Критерієм закінчення ітераційного процесу є умова: $b_{k+1} - a_{k+1} < \varepsilon$, де ε - задана точність.

Нехай є проміжок $[a_{k+1}; b_{k+1}]$,коли точність вже є допустимою. Слід шукати одне з двох значень для набляження $x^* = min\{f(\lambda_k); f(\mu_k)\}$. Помилка буде наступною:

$$|x_* - x^*| \leq \max\{b_{k+1} - x^*; x^* - a_{k+1}\} = (\frac{\sqrt{5}-1}{2})^{k+1}(b-a),$$

x_* - точне значення, x^* - наближене значення.

Можна взяти за розв'язок середнє арифметичне a_{k+1}, b_{k+1}. Тобто $x^* = \frac{a_{k+1}+b_{k+1}}{2}$. Помилка буде наступною:

$$|x_* - x^*| \leq \max\{b_{k+1} - x^*; x^* - a_{k+1}\} = (\frac{\sqrt{5}-1}{2})^{k}(b-a),$$

x_* - точне значення, x^* - наближене значення.

Перевіримо, чи стала похибка менше при середнбому арифметичному.

$$(\frac{\sqrt{5}-1}{2})^{k+1}(b-a) < (\frac{\sqrt{5}-1}{2})^{k}(b-a)$$

$$(\frac{\sqrt{5}-1}{2})^{k+1} < (\frac{\sqrt{5}-1}{2})^{k}$$

$$k+1 > k$$

Тобто середнє арифметичне значення не зробило результат точнішим.

2.3 Метод Фібоначчі

Для початку згадаємо, що таке числа Фібоначі. Числа Фібоначі – це послідовність, яка починається наступним чином: 1, 1, 2, 3, 5, 8, 13, 21, … - і продовжується таким чином, що поточне значення дорівнює сумі двох попередніх.

Зауважимо, що в данному методі є певне число n – кількість ітерацій, відповідно до нього рахується число фібоначі. Наприклад $n = 1{,}2 \Longrightarrow F_1 = F_2 = 1; n = 3 \Longrightarrow F_3 = 2; n = 4 \Longrightarrow F_4 = 3, \ldots$.

Нехай дана цільова функція f(x), яка має мінімум на відрізку [a;b]. Метою є знайти наближене значення критичної точки (мінімума) цієї функції.

Першим кроком обирається число n – кількість ітерацій, за допомогою умови: $F_{n+1} \leq \frac{b-a}{\varepsilon} \leq F_{n+2}$. Якщо працюєте на C++ - то рекомендовано обирати діапазон $n \in [1; 88]$, бо C++ коректно обчисює значення до F_{90} – саме для типу usigned long long int.

Другим кроком перепозначимо, що $[a_1; b_1] = [a; b]$.

Третім кроком знайдемо: λ_1, μ_1 – за формулами:

$$\begin{cases} \lambda_1 = a_1 + \frac{F_n}{F_{n+2}}(b_1 - a_1) \\ \mu_1 = a_1 + \frac{F_{n+1}}{F_{n+2}}(b_1 - a_1) \end{cases}.$$

Далі обчислюють та порівнюють значення цільової функції $f(\lambda_1)$ та $f(\mu_1)$.

Якщо $f(\lambda_1) \leq f(\mu_1)$ - то маємо наступне: $a_2 = a_1;\ b_2 = \mu_1$.

Якщо $f(\lambda_1) > f(\mu_1)$ - то маємо наступне: $a_2 = \lambda_1;\ b_2 = b_1$.

У результаті ми звузили наш проміжок [a;b], а точка мінімума залишилася лежати в ньому. Довжина відрізка $[a_2; b_2]$ буде наступною:

$$b_2 - a_2 = b - \lambda_1 = \mu_1 - a = \frac{F_{n+1}}{F_{n+2}}(b - a).$$

Це було для першого кроку. Тепер уявімо, що ми на якомусь k-тому кроці, маємо проміжок $[a_k; b_k]$, функція f(x) має точку мінімума на цьому проміжку.

Довжина наступного відрізку буде:

$$b_{k+1} - a_{k+1} = \frac{F_{n-k+2}}{F_{n+2}}(b - a).$$

Точки λ_k, μ_k рахуються аналогічно:

$$\begin{cases} \lambda_k = a_k + \frac{F_{n-k+1}}{F_{n-k+3}}(b_k - a_k) \\ \mu_k = a_k + \frac{F_{n-k+2}}{F_{n-k+3}}(b_k - a_k) \end{cases}$$

Нехай $k = n$. Тоді процес закінчуємо і отримуємо $x^* = \lambda_n = \mu_n$, але це дуже дрібні операції, тому помилка буде, саме тому $x^* = \frac{\lambda_n + \mu_n}{2}$.

Довжина отриманого відрізку буде наступною:

$$b_n - a_n = \frac{2(b-a)}{F_{n+2}}.$$

Похибка: $|x_* - x^*| \leq \frac{b-a}{F_{n+2}}$, x_* - точне значення, x^* - наближене значення.

3. Код програми на Python

```
import math
import os
import random
from openpyxl import Workbook, load_workbook


def f(x):
    return x * x * x - 3 * x + 2


def cap(a, b, eps):

    return f"f(x) = x^3 -3x + 2;\n\n[a; b] = [{a}; {b}];\n\nε = {eps}"


def random_number_between(a, b):
    while True:

        value = random.uniform(a, b)

        if value != a and value != b:
            return value


def halving_method(a, b, eps,file_path):
    if not os.path.exists(file_path):
        wb = Workbook()
        wb.save(file_path)
    else:
        wb = load_workbook(file_path)

    sheet_name = "Метод ділення навпіл"
```

```
if sheet_name in wb.sheetnames:
    del wb[sheet_name]
    wb.save(file_path)

wb.create_sheet(sheet_name)
wb.save(file_path)

wb[sheet_name].cell(row=1, column=1, value="K")
wb[sheet_name].cell(row=1, column=2, value="lambda")
wb[sheet_name].cell(row=1, column=3, value="nua")
wb[sheet_name].cell(row=1, column=4, value="a")
wb[sheet_name].cell(row=1, column=5, value="b")
wb[sheet_name].cell(row=1, column=6, value="x")
wb[sheet_name].cell(row=1, column=7, value="F(lambda)")
wb[sheet_name].cell(row=1, column=8, value="F(nua)")
wb[sheet_name].cell(row=1, column=9, value="F(a)")
wb[sheet_name].cell(row=1, column=10, value="F(b)")
wb[sheet_name].cell(row=1, column=11, value="F(x)")

a_k = a
b_k = b

k = 1

while True:

    while True:

        beta_k = random_number_between(0, b_k - a_k)

        if eps > beta_k:
            break

    lamda_k = (a_k + b_k - beta_k) / 2
    nua_k = (a_k + b_k + beta_k) / 2
```

```
        if f(lamda_k) <= f(nua_k) :
            b_k = nua_k
        else:
            a_k = lamda_k

        wb[sheet_name].cell(row=k+1, column=1, value=k)
        wb[sheet_name].cell(row=k+1, column=2, value=lamda_k)
        wb[sheet_name].cell(row=k+1, column=3, value=nua_k)
        wb[sheet_name].cell(row=k+1, column=4, value=a_k)
        wb[sheet_name].cell(row=k+1, column=5, value=b_k)
        wb[sheet_name].cell(row=k+1, column=6, value=(a_k+b_k)/2)
        wb[sheet_name].cell(row=k+1, column=7, value=f(lamda_k))
        wb[sheet_name].cell(row=k+1, column=8, value=f(nua_k))
        wb[sheet_name].cell(row=k+1, column=9, value=f(a_k))
        wb[sheet_name].cell(row=k+1, column=10, value=f(b_k))
        wb[sheet_name].cell(row=k+1, column=11, value=f((a_k+b_k)/2))



        k+=1

        if abs(b_k-a_k)<eps:
            wb.save(file_path)
            return f"f_min({(b_k + a_k) / 2}) = {f((b_k + a_k) / 2)}"



def gold_retin_method(a, b, eps,file_path):
    if not os.path.exists(file_path):
        wb = Workbook()
        wb.save(file_path)
    else:
        wb = load_workbook(file_path)

    sheet_name = "Метод Золотого перетину"
```

```
if sheet_name in wb.sheetnames:
    del wb[sheet_name]
    wb.save(file_path)

wb.create_sheet(sheet_name)
wb.save(file_path)

wb[sheet_name].cell(row=1, column=1, value="K")
wb[sheet_name].cell(row=1, column=2, value="lambda")
wb[sheet_name].cell(row=1, column=3, value="nua")
wb[sheet_name].cell(row=1, column=4, value="a")
wb[sheet_name].cell(row=1, column=5, value="b")
wb[sheet_name].cell(row=1, column=6, value="x")
wb[sheet_name].cell(row=1, column=7, value="F(lambda)")
wb[sheet_name].cell(row=1, column=8, value="F(nua)")
wb[sheet_name].cell(row=1, column=9, value="F(a)")
wb[sheet_name].cell(row=1, column=10, value="F(b)")
wb[sheet_name].cell(row=1, column=11, value="F(x)")

a_k = a
b_k = b

k = 1

while True:

    lamda_k = a_k + (3 - math.sqrt(5)) * (b_k - a_k) / 2.0
    nua_k = a_k + (math.sqrt(5) - 1) * (b_k - a_k) / 2.0

    if f(lamda_k) <= f(nua_k):
        b_k = nua_k
    else:
        a_k = lamda_k

    x_k = lamda_k if f(lamda_k) < f(nua_k) else nua_k
```

4. Приклад роботи програми

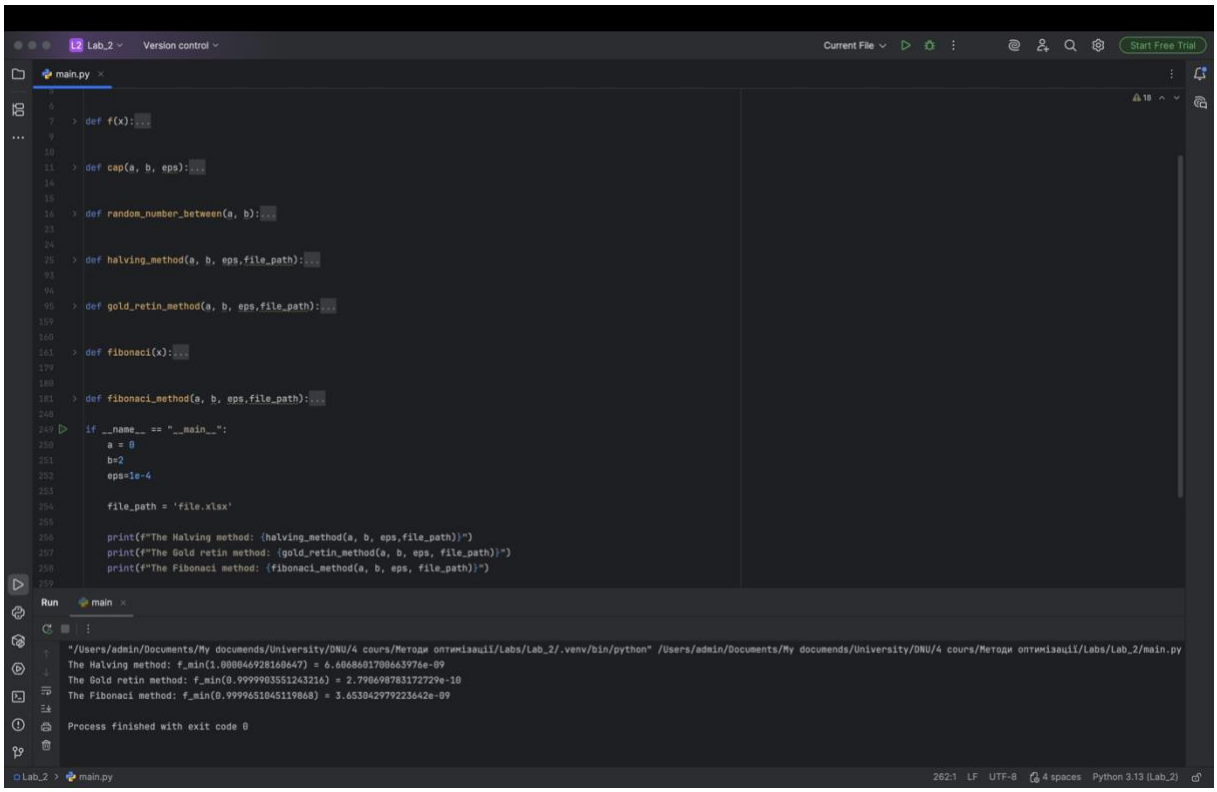
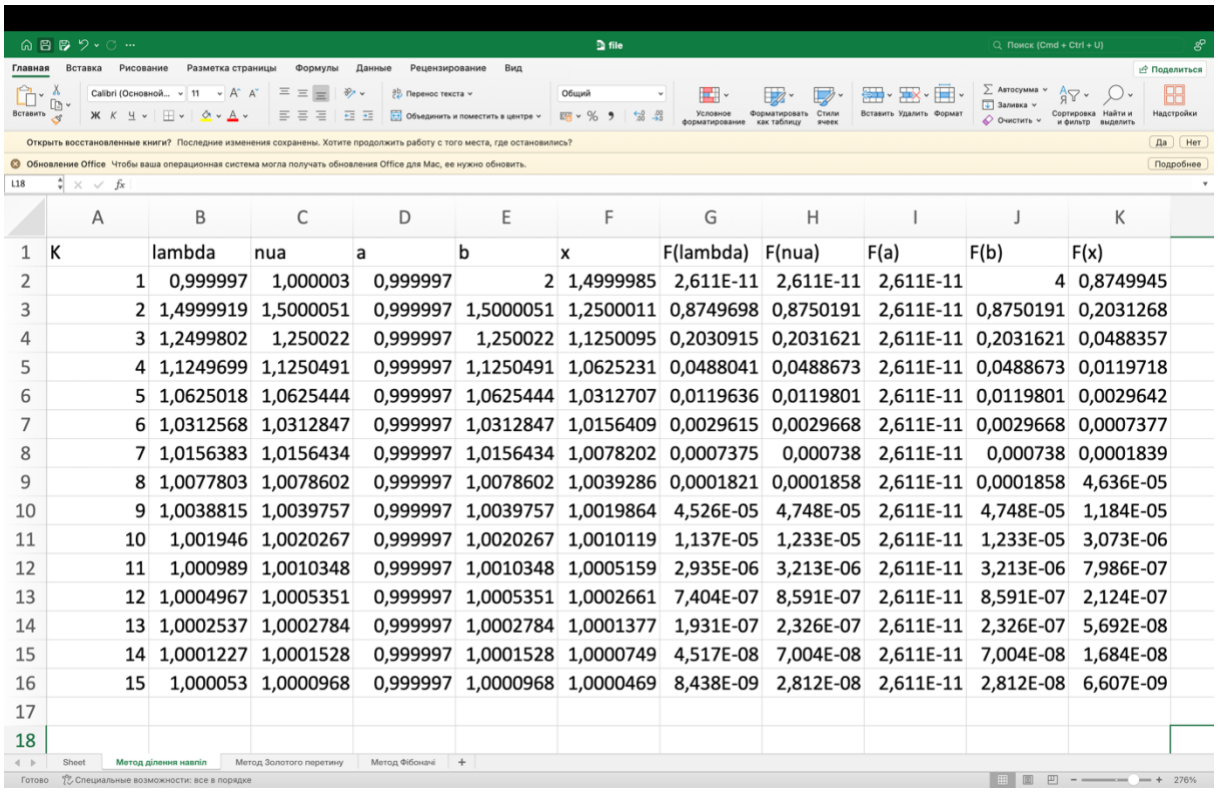
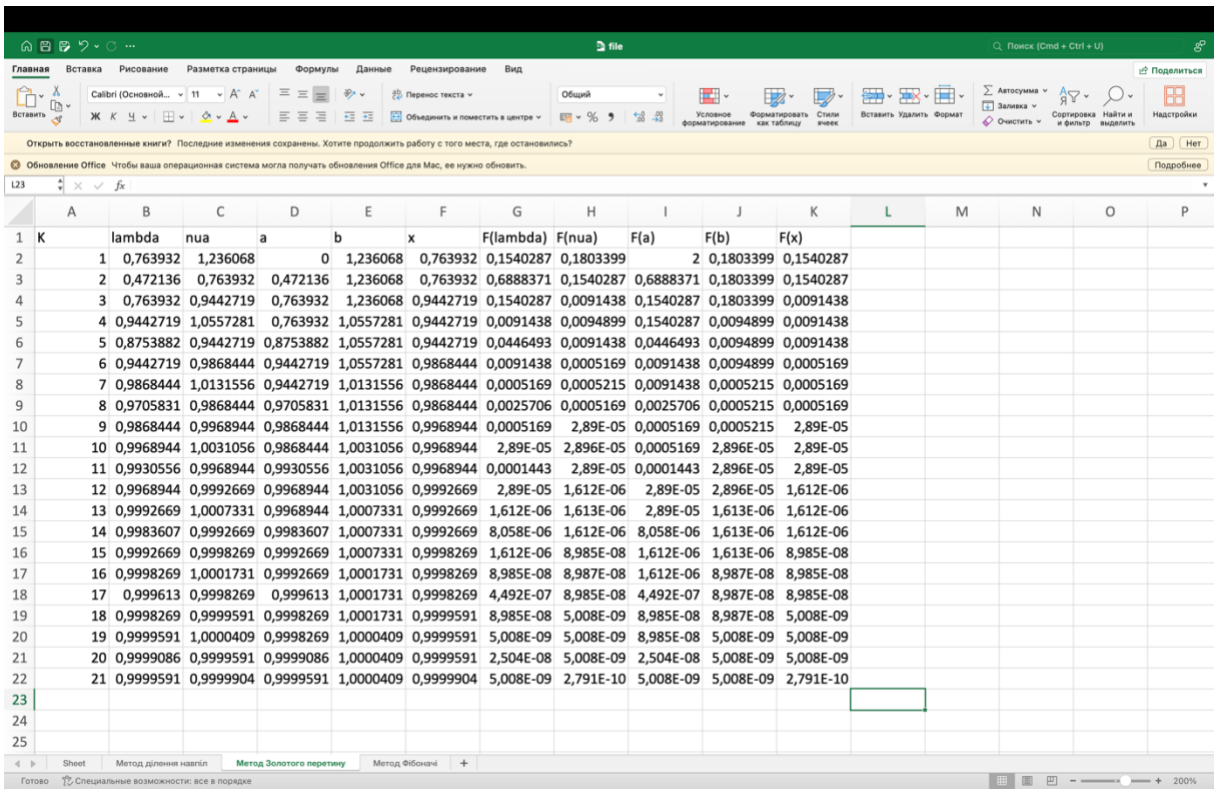


Рисунок 2 – Приклад роботи програми, яка знаходить точку, де є мінімальне значення функції $f(x) = x^3 - 3x + 2$ на проміжку [0;2] і видає його на єкран консолі.



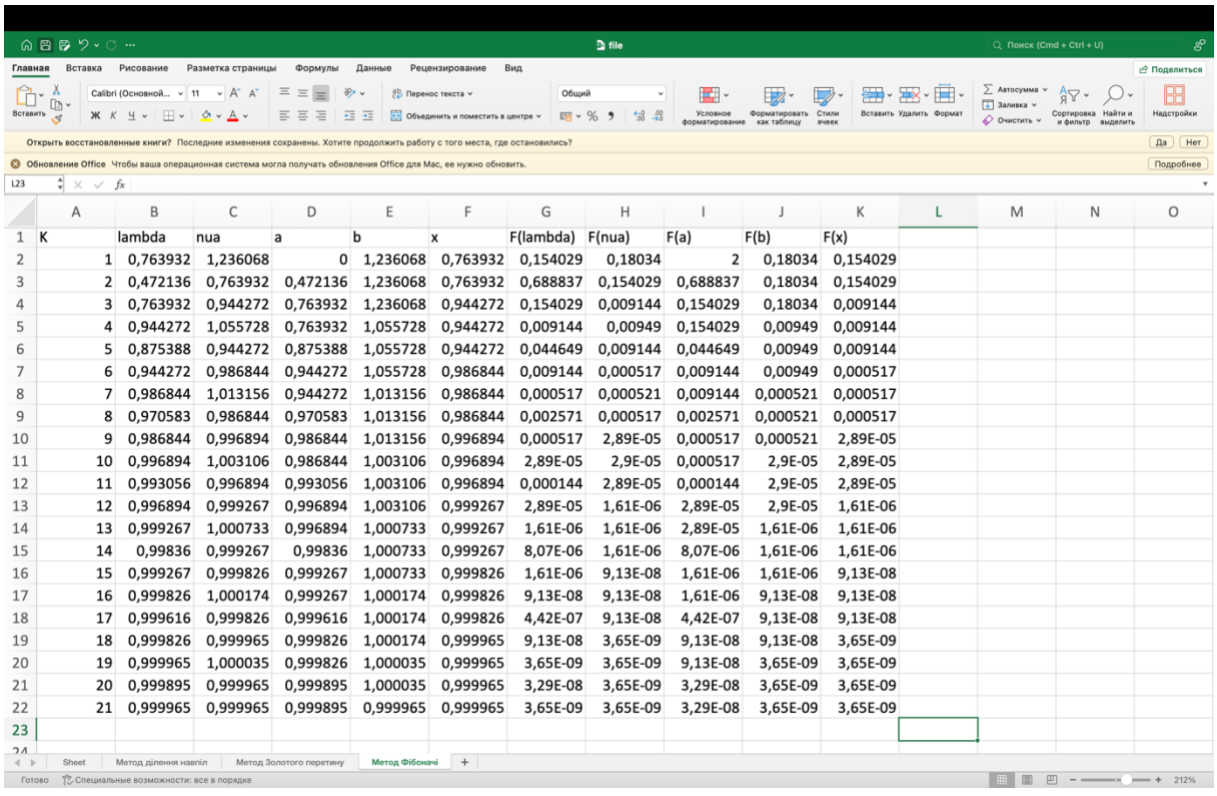
	A	B	C	D	E	F	G	H	I	J	K
1	K	lambda	nua	a	b	x	F(lambda)	F(nua)	F(a)	F(b)	F(x)
2	1	0,999997	1,000003	0,999997	2	1,4999985	2,611E-11	2,611E-11	2,611E-11	4	0,8749945
3	2	1,4999919	1,5000051	0,999997	1,5000051	1,2500011	0,8749698	0,8750191	2,611E-11	0,8750191	0,2031268
4	3	1,2499802	1,250022	0,999997	1,250022	1,1250095	0,2030915	0,2031621	2,611E-11	0,2031621	0,0488357
5	4	1,1249699	1,1250491	0,999997	1,1250491	1,0625231	0,0488041	0,0488673	2,611E-11	0,0488673	0,0119718
6	5	1,0625018	1,0625444	0,999997	1,0625444	1,0312707	0,0119636	0,0119801	2,611E-11	0,0119801	0,0029642
7	6	1,0312568	1,0312847	0,999997	1,0312847	1,0156409	0,0029615	0,0029668	2,611E-11	0,0029668	0,0007377
8	7	1,0156383	1,0156434	0,999997	1,0156434	1,0078202	0,0007375	0,000738	2,611E-11	0,000738	0,0001839
9	8	1,0077803	1,0078602	0,999997	1,0078602	1,0039286	0,0001821	0,0001858	2,611E-11	0,0001858	4,636E-05
10	9	1,0038815	1,0039757	0,999997	1,0039757	1,0019864	4,526E-05	4,748E-05	2,611E-11	4,748E-05	1,184E-05
11	10	1,001946	1,0020267	0,999997	1,0020267	1,0010119	1,137E-05	1,233E-05	2,611E-11	1,233E-05	3,073E-06
12	11	1,000989	1,0010348	0,999997	1,0010348	1,0005159	2,935E-06	3,213E-06	2,611E-11	3,213E-06	7,986E-07
13	12	1,0004967	1,0005351	0,999997	1,0005351	1,0002661	7,404E-07	8,591E-07	2,611E-11	8,591E-07	2,124E-07
14	13	1,0002537	1,0002784	0,999997	1,0002784	1,0001377	1,931E-07	2,326E-07	2,611E-11	2,326E-07	5,692E-08
15	14	1,0001227	1,0001528	0,999997	1,0001528	1,0000749	4,517E-08	7,004E-08	2,611E-11	7,004E-08	1,684E-08
16	15	1,000053	1,0000968	0,999997	1,0000968	1,0000469	8,438E-09	2,812E-08	2,611E-11	2,812E-08	6,607E-09
17											
18											

Рисунок 3 – Покрокові ітерації для методу Діленння навпіл.



K	lambda	nua	a	b	x	F(lambda)	F(nua)	F(a)	F(b)	F(x)
1	0,763932	1,236068	0	1,236068	0,763932	0,1540287	0,1803399	2	0,1803399	0,1540287
2	0,472136	0,763932	0,472136	1,236068	0,763932	0,6888371	0,1540287	0,6888371	0,1803399	0,1540287
3	0,763932	0,9442719	0,763932	1,236068	0,9442719	0,1540287	0,0091438	0,1540287	0,1803399	0,0091438
4	0,9442719	1,0557281	0,763932	1,0557281	0,9442719	0,0091438	0,0094899	0,1540287	0,0094899	0,0091438
5	0,8753882	0,9442719	0,8753882	1,0557281	0,9442719	0,0446493	0,0091438	0,0446493	0,0094899	0,0091438
6	0,9442719	0,9868444	0,9442719	1,0557281	0,9868444	0,0091438	0,0005169	0,0091438	0,0094899	0,0005169
7	0,9868444	1,0131556	0,9442719	1,0131556	0,9868444	0,0005169	0,0005215	0,0091438	0,0005215	0,0005169
8	0,9705831	0,9868444	0,9705831	1,0131556	0,9868444	0,0025706	0,0005169	0,0025706	0,0005215	0,0005169
9	0,9868444	0,9968944	0,9868444	1,0131556	0,9968944	0,0005169	2,89E-05	0,0005169	0,0005215	2,89E-05
10	0,9968944	1,0031056	0,9868444	1,0031056	0,9968944	2,89E-05	2,896E-05	0,0005169	2,896E-05	2,89E-05
11	0,9930556	0,9968944	0,9930556	1,0031056	0,9968944	0,0001443	2,89E-05	0,0001443	2,896E-05	2,89E-05
12	0,9968944	0,9992669	0,9968944	1,0031056	0,9992669	2,89E-05	1,612E-06	2,89E-05	2,896E-05	1,612E-06
13	0,9992669	1,0007331	0,9968944	1,0007331	0,9992669	1,612E-06	1,613E-06	2,89E-05	1,613E-06	1,612E-06
14	0,9983607	0,9992669	0,9983607	1,0007331	0,9992669	8,058E-06	1,612E-06	8,058E-06	1,613E-06	1,612E-06
15	0,9992669	0,9998269	0,9992669	1,0007331	0,9998269	1,612E-06	8,985E-08	1,612E-06	1,613E-06	8,985E-08
16	0,9998269	1,0001731	0,9992669	1,0001731	0,9998269	8,985E-08	8,987E-08	1,612E-06	8,987E-08	8,985E-08
17	0,999613	0,9998269	0,999613	1,0001731	0,9998269	4,492E-07	8,985E-08	4,492E-07	8,987E-08	8,985E-08
18	0,9998269	0,9999591	0,9998269	1,0001731	0,9999591	8,985E-08	5,008E-09	8,985E-08	8,987E-08	5,008E-09
19	0,9999591	1,0000409	0,9998269	1,0000409	0,9999591	5,008E-09	5,008E-09	8,985E-08	5,008E-09	5,008E-09
20	0,9999086	0,9999591	0,9999086	1,0000409	0,9999591	2,504E-08	5,008E-09	2,504E-08	5,008E-09	5,008E-09
21	0,9999591	0,9999904	0,9999591	1,0000409	0,9999904	5,008E-09	2,791E-10	5,008E-09	5,008E-09	2,791E-10

Рисунок 4 – Покрокові ітерації для методу Золотого перетина.



K	lambda	nua	a	b	x	F(lambda)	F(nua)	F(a)	F(b)	F(x)
1	0,763932	1,236068	0	1,236068	0,763932	0,154029	0,18034	2	0,18034	0,154029
2	0,472136	0,763932	0,472136	1,236068	0,763932	0,688837	0,154029	0,688837	0,18034	0,154029
3	0,763932	0,944272	0,763932	1,236068	0,944272	0,154029	0,009144	0,154029	0,18034	0,009144
4	0,944272	1,055728	0,763932	1,055728	0,944272	0,009144	0,00949	0,154029	0,00949	0,009144
5	0,875388	0,944272	0,875388	1,055728	0,944272	0,044649	0,009144	0,044649	0,00949	0,009144
6	0,944272	0,986844	0,944272	1,055728	0,986844	0,009144	0,000517	0,009144	0,00949	0,000517
7	0,986844	1,013156	0,944272	1,013156	0,986844	0,000517	0,000521	0,009144	0,000521	0,000517
8	0,970583	0,986844	0,970583	1,013156	0,986844	0,002571	0,000517	0,002571	0,000521	0,000517
9	0,986844	0,996894	0,986844	1,013156	0,996894	0,000517	2,89E-05	0,000517	0,000521	2,89E-05
10	0,996894	1,003106	0,986844	1,003106	0,996894	2,89E-05	2,9E-05	0,000517	2,9E-05	2,89E-05
11	0,993056	0,996894	0,993056	1,003106	0,996894	0,000144	2,89E-05	0,000144	2,9E-05	2,89E-05
12	0,996894	0,999267	0,996894	1,003106	0,999267	2,89E-05	1,61E-06	2,89E-05	2,9E-05	1,61E-06
13	0,999267	1,000733	0,996894	1,000733	0,999267	1,61E-06	1,61E-06	2,89E-05	1,61E-06	1,61E-06
14	0,99836	0,999267	0,99836	1,000733	0,999267	8,07E-06	1,61E-06	8,07E-06	1,61E-06	1,61E-06
15	0,999267	0,999826	0,999267	1,000733	0,999826	1,61E-06	9,13E-08	1,61E-06	1,61E-06	9,13E-08
16	0,999826	1,000174	0,999267	1,000174	0,999826	9,13E-08	9,13E-08	1,61E-06	9,13E-08	9,13E-08
17	0,999616	0,999826	0,999616	1,000174	0,999826	4,42E-07	9,13E-08	4,42E-07	9,13E-08	9,13E-08
18	0,999826	0,999965	0,999826	1,000174	0,999965	9,13E-08	3,65E-09	9,13E-08	9,13E-08	3,65E-09
19	0,999965	1,000035	0,999826	1,000035	0,999965	3,65E-09	3,65E-09	9,13E-08	3,65E-09	3,65E-09
20	0,999895	0,999965	0,999895	1,000035	0,999965	3,29E-08	3,65E-09	3,29E-08	3,65E-09	3,65E-09
21	0,999965	0,999965	0,999895	0,999965	0,999965	3,65E-09	3,65E-09	3,29E-08	3,65E-09	3,65E-09

Рисунок 5 – Покрокові ітерації для методу Фібоначі.

5. Таблички покрокових ітерацій та графічне зображення

Щоб намалюати функцію був використан додаток WOLFRAM CLOUD:

“

x=.

```
Plot[x^3-3x+2, {x, 0, 2},
  AxesLabel -> {"x", "y"},
  PlotLabel -> "y(x) = x^3-3x+2",
 PlotRange ->{-5,5},
 GridLines -> Automatic,
 Frame ->True]
```

”.

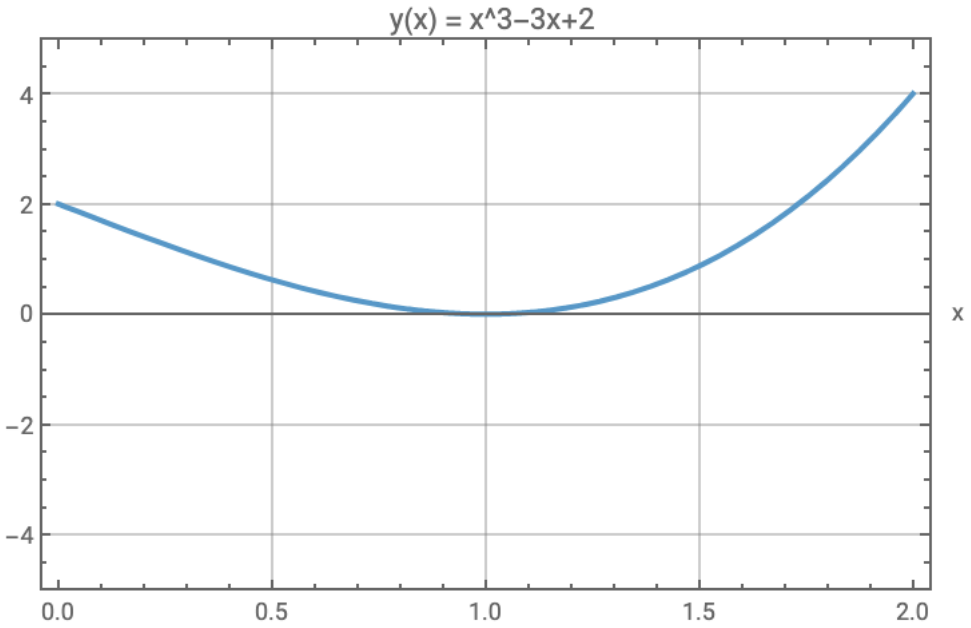


Рисунок 6 – Графік цільової функції на проміжку [0;2].

Побудуємо таблички для наближень.

Таблиця 1 - Таблиця порівнянь для методу Ділення навпіл.

K	lambda	nua	a	b	x	F(lambda)	F(nua)	F(a)	F(b)	F(x)
1	0,999997	1,000003	0,999997	2	1,499999	2,61E-11	2,61E-11	2,61E-11	4	0,874994
2	1,499992	1,500005	0,999997	1,500005	1,250001	0,87497	0,875019	2,61E-11	0,875019	0,203127
3	1,24998	1,250022	0,999997	1,250022	1,12501	0,203092	0,203162	2,61E-11	0,203162	0,048836
4	1,12497	1,125049	0,999997	1,125049	1,062523	0,048804	0,048867	2,61E-11	0,048867	0,011972
5	1,062502	1,062544	0,999997	1,062544	1,031271	0,011964	0,01198	2,61E-11	0,01198	0,002964
6	1,031257	1,031285	0,999997	1,031285	1,015641	0,002961	0,002967	2,61E-11	0,002967	0,000738
7	1,015638	1,015643	0,999997	1,015643	1,00782	0,000737	0,000738	2,61E-11	0,000738	0,000184
8	1,00778	1,00786	0,999997	1,00786	1,003929	0,000182	0,000186	2,61E-11	0,000186	4,64E-05
9	1,003882	1,003976	0,999997	1,003976	1,001986	4,53E-05	4,75E-05	2,61E-11	4,75E-05	1,18E-05
10	1,001946	1,002027	0,999997	1,002027	1,001012	1,14E-05	1,23E-05	2,61E-11	1,23E-05	3,07E-06
11	1,000989	1,001035	0,999997	1,001035	1,000516	2,94E-06	3,21E-06	2,61E-11	3,21E-06	7,99E-07
12	1,000497	1,000535	0,999997	1,000535	1,000266	7,4E-07	8,59E-07	2,61E-11	8,59E-07	2,12E-07
13	1,000254	1,000278	0,999997	1,000278	1,000138	1,93E-07	2,33E-07	2,61E-11	2,33E-07	5,69E-08
14	1,000123	1,000153	0,999997	1,000153	1,000075	4,52E-08	7E-08	2,61E-11	7E-08	1,68E-08
15	1,000053	1,000097	0,999997	1,000097	1,000047	8,44E-09	2,81E-08	2,61E-11	2,81E-08	6,61E-09

Зробимо наочно перші 2 ітерації.

$$\varepsilon = 0.0001; a = 0; b = 2; \delta\epsilon[0; 2], \varepsilon > \delta, \delta = 0.000006;$$

$$\begin{cases} \lambda = \dfrac{2 - 0.000006}{2} = 0.999997 \\ \mu = \dfrac{2 + 0.000006}{2} = 1.000003 \end{cases}$$

$$\begin{cases} f(\lambda) = f(0.999997) = 0.999997^3 - 3 \cdot 0.999997 + 2 \approx 2.7 \cdot 10^{-11} \\ f(\mu) = f(1.000003) = 1.000003^3 - 3 \cdot 1.000003 + 2 \approx 2.7 \cdot 10^{-11} \end{cases}$$

$$f(\lambda) = f(\mu) \Longrightarrow b = 1.000003;$$

Значення вже дуже малі, тому похибка очевидна. Але в данному випадку ми змінюємо b, а не a.

$$\varepsilon = 0.0001; a = 0; b = 1.000003; \delta\epsilon[0; 1.000003], \varepsilon > \delta, \delta = 0.000013;$$

$$b - a = 1.000003 > \varepsilon = 0.0001$$

$$\begin{cases} \lambda = \dfrac{1.000003 - 0.000013}{2} = 0.499995 \\ \mu = \dfrac{1.000003 + 0.000013}{2} = 0.500008 \end{cases}$$

$$\begin{cases} f(\lambda) = f(0.499995) = 0.499995^3 - 3 \cdot 0.499995 + 2 \approx 0.625011 \\ f(\mu) = f(0.500008) = 0.500008^3 - 3 \cdot 0.500008 + 2 \approx 0.624982 \end{cases}$$

$$f(\lambda) > f(\mu) \Longrightarrow a = 0.499995$$

$$\varepsilon = 0.0001; a = 0.499995; b = 1.000003; \delta\epsilon[0; 0.500008], \varepsilon > \delta, \delta = 0.00001;$$

$$b - a = 0.500008 > \varepsilon = 0.0001..$$

Таблиця 2 - Таблиця порівнянь для методу Золотого перетину.

K	lambda	nua	a	b	x	F(lambda)	F(nua)	F(a)	F(b)	F(x)
1	0,763932	1,236068	0	1,236068	0,763932	0,154029	0,18034	2	0,18034	0,154029
2	0,472136	0,763932	0,472136	1,236068	0,763932	0,688837	0,154029	0,688837	0,18034	0,154029
3	0,763932	0,944272	0,763932	1,236068	0,944272	0,154029	0,009144	0,154029	0,18034	0,009144
4	0,944272	1,055728	0,763932	1,055728	0,944272	0,009144	0,00949	0,154029	0,00949	0,009144
5	0,875388	0,944272	0,875388	1,055728	0,944272	0,044649	0,009144	0,044649	0,00949	0,009144
6	0,944272	0,986844	0,944272	1,055728	0,986844	0,009144	0,000517	0,009144	0,00949	0,000517
7	0,986844	1,013156	0,944272	1,013156	0,986844	0,000517	0,000521	0,009144	0,000521	0,000517
8	0,970583	0,986844	0,970583	1,013156	0,986844	0,002571	0,000517	0,002571	0,000521	0,000517
9	0,986844	0,996894	0,986844	1,013156	0,996894	0,000517	2,89E-05	0,000517	0,000521	2,89E-05
10	0,996894	1,003106	0,986844	1,003106	0,996894	2,89E-05	2,9E-05	0,000517	2,9E-05	2,89E-05
11	0,993056	0,996894	0,993056	1,003106	0,996894	0,000144	2,89E-05	0,000144	2,9E-05	2,89E-05
12	0,996894	0,999267	0,996894	1,003106	0,999267	2,89E-05	1,61E-06	2,89E-05	2,9E-05	1,61E-06
13	0,999267	1,000733	0,996894	1,000733	0,999267	1,61E-06	1,61E-06	2,89E-05	1,61E-06	1,61E-06
14	0,998361	0,999267	0,998361	1,000733	0,999267	8,06E-06	1,61E-06	8,06E-06	1,61E-06	1,61E-06
15	0,999267	0,999827	0,999267	1,000733	0,999827	1,61E-06	8,99E-08	1,61E-06	1,61E-06	8,99E-08
16	0,999827	1,000173	0,999267	1,000173	0,999827	8,99E-08	8,99E-08	1,61E-06	8,99E-08	8,99E-08
17	0,999613	0,999827	0,999613	1,000173	0,999827	4,49E-07	8,99E-08	4,49E-07	8,99E-08	8,99E-08
18	0,999827	0,999959	0,999827	1,000173	0,999959	8,99E-08	5,01E-09	8,99E-08	8,99E-08	5,01E-09
19	0,999959	1,000041	0,999827	1,000041	0,999959	5,01E-09	5,01E-09	8,99E-08	5,01E-09	5,01E-09
20	0,999909	0,999959	0,999909	1,000041	0,999959	2,5E-08	5,01E-09	2,5E-08	5,01E-09	5,01E-09
21	0,999959	0,99999	0,999959	1,000041	0,99999	5,01E-09	2,79E-10	5,01E-09	5,01E-09	2,79E-10

Зробимо наочно перші 2 ітерації.

$$\varepsilon = 0.0001; a = 0; b = 2;$$

$$\begin{cases} \lambda = 0 + \dfrac{3 - \sqrt{5}}{2} \cdot (2 - 0) \approx 0.763932 \\ \mu = 0 + \dfrac{\sqrt{5} - 1}{2} \cdot (2 - 0) \approx 1.23607 \end{cases}$$

$$\begin{cases} f(\lambda) = f(0.763932) = 0.763932^3 - 3 \cdot 0.763932 + 2 \approx 0.154029 \\ f(\mu) = f(1.23607) = 1.23607^3 - 3 \cdot 01.23607 + 2 \approx 0.180343 \end{cases}$$

$$f(x) = min\{f(\lambda); f(\mu)\} = min\{0.154029; 180343\} = 0.154029$$

$$f(\lambda) < f(\mu) \Longrightarrow a = 0; b = 1.23607$$

$$b - a = 1.23607 > \varepsilon = 0.0001$$

$$\begin{cases} \lambda = 0 + \dfrac{3 - \sqrt{5}}{2} \cdot (1.23607 - 0) \approx 0.472137 \\ \mu = 0 + \dfrac{\sqrt{5} - 1}{2} \cdot (1.23607 - 0) \approx 0.763933 \end{cases}$$

$$\begin{cases} f(\lambda) = f(0.472137) = 0.472137^3 - 3 \cdot 0.472137 + 2 \approx 0.688835 \\ f(\mu) = f(0.763933) = 0.763933^3 - 3 \cdot 0.763933 + 2 \approx 0.154027 \end{cases}$$

$$f(x) = min\{f(\lambda); f(\mu)\} = min\{0.688835; 0.154027\} = 0.154027$$

$$f(\lambda) > f(\mu) \Longrightarrow a = 0.472137; b = 1.23607$$

$$b - a = 0.763933 > \varepsilon = 0.0001..$$

Таблиця 3 - Таблиця порівнянь для методу Фібоначі.

K	lambda	nua	a	b	x	F(lambda)	F(nua)	F(a)	F(b)	F(x)
1	0,763932	1,236068	0	1,236068	0,763932	0,154029	0,18034	2	0,18034	0,154029
2	0,472136	0,763932	0,472136	1,236068	0,763932	0,688837	0,154029	0,688837	0,18034	0,154029
3	0,763932	0,944272	0,763932	1,236068	0,944272	0,154029	0,009144	0,154029	0,18034	0,009144
4	0,944272	1,055728	0,763932	1,055728	0,944272	0,009144	0,00949	0,154029	0,00949	0,009144
5	0,875388	0,944272	0,875388	1,055728	0,944272	0,044649	0,009144	0,044649	0,00949	0,009144
6	0,944272	0,986844	0,944272	1,055728	0,986844	0,009144	0,000517	0,009144	0,00949	0,000517
7	0,986844	1,013156	0,944272	1,013156	0,986844	0,000517	0,000521	0,009144	0,000521	0,000517
8	0,970583	0,986844	0,970583	1,013156	0,986844	0,002571	0,000517	0,002571	0,000521	0,000517
9	0,986844	0,996894	0,986844	1,013156	0,996894	0,000517	2,89E-05	0,000517	0,000521	2,89E-05
10	0,996894	1,003106	0,986844	1,003106	0,996894	2,89E-05	2,9E-05	0,000517	2,9E-05	2,89E-05
11	0,993056	0,996894	0,993056	1,003106	0,996894	0,000144	2,89E-05	0,000144	2,9E-05	2,89E-05
12	0,996894	0,999267	0,996894	1,003106	0,999267	2,89E-05	1,61E-06	2,89E-05	2,9E-05	1,61E-06
13	0,999267	1,000733	0,996894	1,000733	0,999267	1,61E-06	1,61E-06	2,89E-05	1,61E-06	1,61E-06
14	0,99836	0,999267	0,99836	1,000733	0,999267	8,07E-06	1,61E-06	8,07E-06	1,61E-06	1,61E-06
15	0,999267	0,999826	0,999267	1,000733	0,999826	1,61E-06	9,13E-08	1,61E-06	1,61E-06	9,13E-08
16	0,999826	1,000174	0,999267	1,000174	0,999826	9,13E-08	9,13E-08	1,61E-06	9,13E-08	9,13E-08
17	0,999616	0,999826	0,999616	1,000174	0,999826	4,42E-07	9,13E-08	4,42E-07	9,13E-08	9,13E-08
18	0,999826	0,999965	0,999826	1,000174	0,999965	9,13E-08	3,65E-09	9,13E-08	9,13E-08	3,65E-09
19	0,999965	1,000035	0,999826	1,000035	0,999965	3,65E-09	3,65E-09	9,13E-08	3,65E-09	3,65E-09
20	0,999895	0,999965	0,999895	1,000035	0,999965	3,29E-08	3,65E-09	3,29E-08	3,65E-09	3,65E-09
21	0,999965	0,999965	0,999895	0,999965	0,999965	3,65E-09	3,65E-09	3,29E-08	3,65E-09	3,65E-09

Зробимо наочно перші 2 ітерації.

Таблиця 4 – Таблиця Фібоначі.

F(1)	F(2)	F(3)	F(4)	F(5)	F(6)	F(7)	F(8)	F(9)	F(10)
1	1	2	3	5	8	13	21	34	55

Таблиця 5 – Таблиця Фібоначі.

F(11)	F(12)	F(13)	F(14)	F(15)	F(16)	F(17)	F(18)	F(19)	F(20)
89	144	233	377	610	987	1597	2584	4181	6765

Таблиця 6 – Таблиця Фібоначі.

F(21)	F(22)	F(23)	F(24)	F(25)	F(26)	F(27)	F(28)	F(29)	F(30)
10946	17711	28657	46368	75025	121393	196418	317811	514229	832040

Знайдемо число n.

$$\mathrm{F_{n+1}} \le \frac{2}{10^{-4}} \le \mathrm{F_{n+2}}$$

$$\mathrm{F_{n+1}} \le 20000 \le \mathrm{F_{n+2}}$$

$$17711 \le 20000 \le 28657$$

$$\mathrm{F_{22}} \le 20000 \le \mathrm{F_{23}}$$

n=21.

Зробимо наочно перші 2 ітерації.

$$n = 21; k = 1; a = 0; b = 2;$$

$$\begin{cases} \lambda = 0 + \dfrac{F_{21-1+1}}{F_{21-1+3}}(2-0) = 2\dfrac{F_{21}}{F_{23}} = 2\dfrac{10946}{28657} \approx 0.763932 \\ \mu = 0 + \dfrac{F_{21-1+2}}{F_{21-1+3}}(2-0) = 2\dfrac{F_{22}}{F_{23}} = 2\dfrac{17711}{28657} \approx 1.23607 \end{cases}$$

$$\begin{cases} f(\lambda) = f(0.763932) = 0.763932^3 - 3 \cdot 0.763932 + 2 \approx 0.154029 \\ f(\mu) = f(1.23607) = 1.23607^3 - 3 \cdot 1.23607 + 2 \approx 0.180343 \end{cases}$$

$$f(\lambda) < f(\mu) \Longrightarrow b = 1.23607$$

$$\lambda \neq \mu; n \neq k$$

$$n = 21; k = 2; a = 0; b = 1.23607;$$

$$\begin{cases} \lambda = 0 + \dfrac{F_{21-2+1}}{F_{21-2+3}}(1.23607-0) = 1.23607\dfrac{F_{20}}{F_{22}} = 1.23607\dfrac{6765}{17711} \approx 0.472137 \\ \mu = 0 + \dfrac{F_{21-2+2}}{F_{21-2+3}}(1.23607-0) = 1.23607\dfrac{F_{21}}{F_{22}} = 1.23607\dfrac{10946}{17711} \approx 0.763933 \end{cases}$$

$$\begin{cases} f(\lambda) = f(0.472137) = 0.472137^3 - 3 \cdot 0.472137 + 2 \approx 0.688835 \\ f(\mu) = f(0.763933) = 0.763933^3 - 3 \cdot 0.763933 + 2 \approx 0.154027 \end{cases}$$

$$f(\lambda) > f(\mu) \Longrightarrow a = 0.472137$$

$$\lambda \neq \mu; n \neq k;$$

$$n = 21; k = 3; a = 0.472137; b = 1.23607; \ldots$$

6. Аналіз оцінок похибки

Нехай довжина вихідного відрізка дорівнює$(b_1 - a_1)$. При заданій величині останнього відрізка $(b_n - a_n)$, яка задовільняє потрібній точності $\varepsilon > 0$, необхідне число обчислень цільової функції n може бути визначено як найменше додатнє ціле, яке задовільняє таким співвідношенням:

Метод ділення навпіл $(\frac{1}{2})^{\frac{n}{2}} \leq \frac{(b_n - a_n)}{(b_1 - a_1)}$;

Метод золотого перетину $(\frac{\sqrt{5}-1}{2})^{n-1} \leq \frac{(b_n - a_n)}{(b_1 - a_1)}$;

Метод Фібоначчі $F_n \geq \frac{(b_1 - a_1)}{(b_n - a_n)}$ або $\frac{1}{F_n} \geq \frac{(b_1 - a_1)}{(b_n - a_n)}$.

Пропишемо перші значення для кожного методу:

$$(\frac{1}{2})^{\frac{n}{2}} = \{0.707107; 0.5; 0.353553; 0.25; 0.176777; 0.125; 0.0883883; \dots\}$$

$$(\frac{\sqrt{5}-1}{2})^{n-1} = \{1; 0.618034; 0.381966; 0.236068; 0.145898; 0.0901699; \dots\}$$

$$\frac{1}{F_n} = \{1; 1; 0.5; 0.333; 0.2; 0.125; 0.0769231; 0.047619; 0.0294118; \dots\}$$

Якщо уважно подивитися на послідовності, то можна побачити, що метод Фібоначчі найшвидше прямує до 0 при великих n, метод золотого перетину значно повільніше, але краще за метод ділення навпіл.

Мені це здалося замалим висновком, тому пропоную використати знання сайта znannya.org, щоб впевнетися. Також, те ж саме при запиті в google: “есть 2 последовательности какая стремиться быстрее к 0” видаст Огляд від II, хоча його ніхто не питав, але він підтвердив інформацію з попереднього сайта. Української інформації просто немає, що ще раз доказує, що ми відстаємо від розвитку, навіть від Росії.

Знайдемо наступні границі:

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \begin{cases} 0, f(n) \text{ прямує до 0 швидше за } g(n) \\ \infty, g(n) \text{ прямує до 0 швидше за } f(n) \end{cases}$$

Використаємо код у Wolfram Cloud:

“

```
halw[n_]:=(1/2)^(n/2)
gold[n_]:=((Sqrt[5]-1)/2)^(n-1)
fibo[n_]:=1/(Fibonacci[n])


Print["\n\nLimit[(halw[n])/(gold[n]),n->\[Infinity] = ",Limit[(halw[n])/(gold[n]),n->\[Infinity]],"\n"]
Print["Limit[(gold[n])/(halw[n]),n->\[Infinity] = ",Limit[(gold[n])/(halw[n]),n->\[Infinity]],"\n\n"]
Print["\n\nLimit[(halw[n])/(fibo[n]),n->\[Infinity] = ",Limit[(halw[n])/(fibo[n]),n->\[Infinity]],"\n"]
Print["Limit[(fibo[n])/(halw[n]),n->\[Infinity] = ",Limit[(fibo[n])/(halw[n]),n->\[Infinity]],"\n\n"]
Print["\n\nLimit[(gold[n])/(fibo[n]),n->\[Infinity] = ",N[Limit[(gold[n])/(fibo[n]),n->\[Infinity]]],"\n"]
Print["Limit[(fibo[n])/(gold[n]),n->\[Infinity] = ",N[Limit[(fibo[n])/(gold[n]),n->\[Infinity]]],"\n\n"]
".
```

```
Limit[(halw[n])/(gold[n]),n→∞ = ∞

Limit[(gold[n])/(halw[n]),n→∞ = 0
```

```
Limit[(halw[n])/(fibo[n]),n→∞ = ∞

Limit[(fibo[n])/(halw[n]),n→∞ = 0
```

```
Limit[(gold[n])/(fibo[n]),n→∞ = 0.723607

Limit[(fibo[n])/(gold[n]),n→∞ = 1.38197
```

Рисунок 7 – Обчислення границь похибок у Wolfram Cloud.

Отримали, що метод ділення навпіл повільніше прямує до нуля ніж метод золотого перетину, метод ділення навпіл також повільніше прямує до нуля ніж метод Фібоначчі. А у порівнянні методу золотого перетину та Фібоначчі отримали дійсні числа (const) – а це значить, що вони однаково швидко прямують до 0. Тобто метод ділення навпіл є найгіршим, а метод золотого перетину та Фібоначі однакові (кращі), але якщо подивитися на послідовності і перші числа – можна зробити висновок, що метод Фібоначі кращій за метод золотого перетину.

Назва метода	$x^{\wedge}$	F($x^{\wedge}$)	Кількість ітерацій	Абсолютна похибка: $\lvert x^{\wedge} - 1\rvert$
Ділення навпіл	1.00004692816065	6.6068601700664E-09	15	0.00004692816065

Золотого перетину	0.999990355124322	2.79069878317273E-10	21	0.0000009644875678
Фібоначчі	0.999965104511987	3.65304297922364E-09	21	0.000034895488013

Судячи з таблиці можна зробити висновок, що найточнішим методом виявився метод золотого перетину, далі метод Фібоначчі, і найгірший метод ділення навпіл.

7. Висновки

Познайомилися практично з ітераційними методами розв`язання задач одновимірної оптимізації.

Знайшли мінімум функції $f(x) = x^3 - 3x + 2$ відрізку $[0; 2]$ з точністю $\varepsilon = 10^{-4}$ методами:

- Ділення навпіл:
  - $\mathrm{f(x)_{min}} = \mathrm{f}(1.00004692816065) = 6{,}6068601700664\mathrm{E} - 09$,
  - (15 ітерацій);
- Золотого перерізу:
  - $\mathrm{f(x)_{min}} = \mathrm{f}(0{,}999990355124322) = 2{,}79069878317273\mathrm{E} - 10$,
  - (21 ітерація);
- Фібоначчі:
  - $\mathrm{f(x)_{min}} = \mathrm{f}(0{,}999965104511987) = 3{,}65304297922364\mathrm{E} - 09$,
  - (21 ітерація);

Можна зробити висновок, що найточнішим методом виявився метод золотого перетину, далі метод Фібоначчі, і найгірший метод ділення навпіл.