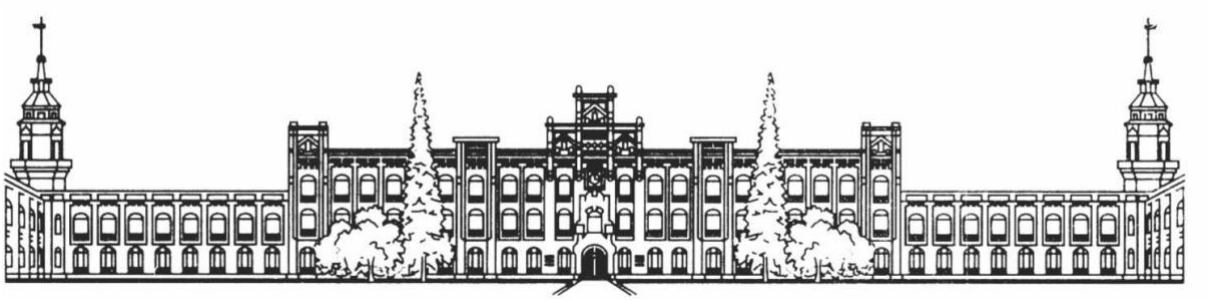


Міністерство освіти і науки України Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»
Інститут прикладного системного аналізу

Методи оптимізації 2
Лабораторна робота №5(7)
«Числові методи одномірної оптимізації»
Бригада (варіант) №6

Виконали:
Студенти 3 курсу
Групи КА-03
Товкач Максим
Муравський Ігор
Страшук Віталій

Перевірили:
Яковлева Алла Петрівна
Спекторський Ігор Якович

Київ-2023

Мета роботи: знайти мінімум заданих функцій за допомогою методів одновимірної оптимізації.

Завдання лабораторної роботи:

У таблиці варіантів завдань знайти цільову функцію f згідно з номером варіанта, вказаного викладачем, та визначити коофіцієнти для квадратичної функції виду $F(x) = x^2 + (p+q)x + pq$, де p - перша, а q - друга цифра в номері варіанту. Якщо номер варіанту менше 10, то $p = 0$.

Скласти програму мінімізації заданих функцій f та F одним із методів

одномірної оптиматизації.

Під час складання програми треба:

- обчислити цільову функцію в окремій підпрогамі;
- частинні похідні цільової функції обчислити чисельно.

Знайти мінімум заданих функції.

7.1. Теоретичні відомості

Кажучи про одномірну оптимізацію, мають на увазі трансформацію загальної задачі оптимізації до часткового випадку

$$f(x) : \mathbb{R}^n \to \mathbb{R} \Rightarrow f(x) : \mathbb{R} \to \mathbb{R} \tag{6.1}$$

Прикладом використання одномірної оптимізації є пошук довжини кроку в ітераційній формулі:

$$x^{k+1} = x^k + \alpha_k h^k \tag{6.2}$$

Оскільки ітераційна формула може використовуватися багаторазово, стільки ж разів необхідно буде розв'язати задачу пошуку довжини кроку (6.2). Тому від ефективності її розв'язку в значній мірі залежить ефективність оптимізаційного процесу в цілому. По ідеології побудови методи одномірного пошуку можна класифікувати таким чином:

1. Методи виключення інтервалів.
2. Методи, що використовують поліноміальну апроксимацію.
3. Методи, які базуються на використанні алгоритмів розв'язку нелінійних рівнянь.

Останні наводитися у даному методичному посібнику не будуть.

Приведемо деякі визначення та факти з теорії унімодальних функцій.

Функція f називається унімодальною на $X = [a, b]$, якщо існує $x* \in X$:

$$\begin{array}{l} f(x_1) > f(x_2), \text{ якщо } x_1 < x_2 < x*, x_1, x_2 \in X; \\ f(x_1) < f(x_2), \text{ якщо } x_2 > x_1 > x*, x_1, x_2 \in X; \end{array} \tag{7.1}$$

Строгоопуклі функції є унімодальними (зворотне твердження неправдиве). Якщо унімодальна функція f досягає $\inf_{x \in X} f(x)$,то неодмінно

у точці x*. Якщо обмежитися лише неперервними функціями, то властивість унімодальної функції означає наявність у неї єдиного локального мінімума.

Лема 6. 1. Нехай функція f унімодальна на X, $x_1, x_2 \in X$, $x_1 < x_2$. Тоді, якщо $f(x_1) \leq f(x_2)$, то $x^* \leq x_2$; якщо $f(x_1) \geq f(x_2)$, то $x^* \geq x_2$.

Приведемо опис деяких методів мінімізації для унімодальних функцій.

7.1.1. Методи

Метод дихотомії

Початковими даними є інтервал невизначеності $[a, b]$ і похибка ε. Візьмемо за $x_1 = \frac{a+b}{2} - \delta$, $x_2 = \frac{a+b}{2} + \delta$, $\delta > 0$. Порівнюємо значення функції в цих точках. Якщо $f(x_1) < f(x_2)$, то звужуємо інтервал локалізації мінімума до [a,x_2], інакше - до $[x_1, b]$. Довжина відрізку локалізації після першої пари обрахунків дорівнює $\frac{b-a}{2} + \delta$. Друга пара обрахунків проводиться у точках, що знаходяться на відстані δ по обидві сторони від середини відрізка. У результаті отримуємо відрізок локалізації мінімума довжиною $\frac{\frac{b-a}{2}+\delta}{2}+\delta = \frac{b-a}{4}+\frac{3}{2}\delta$, після 3-ої:$\frac{\frac{b-a}{4}+\frac{3}{2}\delta}{2}+\delta = \frac{b-a}{8}+\frac{7}{4}\delta$. Після k-тої пари обрахунків : $\frac{b-a}{2^k} + \frac{2^k-1}{2^{k-1}}\delta$. Після N пар обрахунків довжина відрізку локалізації мінімума дорівнює$\frac{b-a}{2^{[N/2]}} + \frac{2^{[N/2]}-1}{2^{[N/2]-1}}$. Відповідним вибором δ, цю довжину можна зробити як завгодно близькою до$\frac{b-a}{2^{[N/2]}}$.

Метод золотого перетину

Пошук у методі "золотого перетину"базується на розподіленні інтервалу невизначеності на дві частини, яке відоме як "золотий перетин". Використовуються такі два співвідношення із числами Фібоначчі:

$$\lim_{k\to\infty} \frac{F_{k-1}}{F_{k+1}} = 0.382; \quad \lim_{k\to\infty} \frac{F_k}{F_{k+1}} = 0.618 \qquad ((6.8)) \tag{7.2}$$

Нехай маємо відрізок X=[a_0,b_0] та похибку ε. Тоді пошук мінімуму за даним методом буде наступний:

$L_0 = b_0 - a_0$, $x_1^0 = a_0 + 0.382L_0$, $x_2^0 = a_0 + 0.618L_0$, $k = 1$.

Далі: якщо $f(x_1) \leq f(x_2)$, то

a_k=a_{k-1},b_k=x_2, L_k=b_k-a_k, x_2^k=x_1^{k-1}, x_1^k= a_k+0.618L_k.

Якщо f(x_1^{k-1})¿f(x_2^{k-1}), то

a_k=x_1^{k-1}, b_k=b_{k-1}, L_k=b_k-a_k, x_1^k=x_2^{k-1}, x_2^k=a_k+0.618L_k.

Якщо проведено n кроків, то довжина отриманого інтервалу дорівнює $L_n = (0.618)^{n-1}L_0$.

При великих кількостях розрахунків, довжина відрізка локалізації після N кроків за методом золотого перетину приблизно на 17% більше ніж в методі Фібоначчі. Але метод золотого перетину дозволяє зупинити розрахунки в будь-який момент, оскільки точки, над якими ведуться розрахунки, не залежать від числа розрахунків.

Варіант №6:

$f(x) = \frac{x^2}{2} - \sin(x)$

Аналітичне дослідження функції:

$f' = x - \cos(x) = 0$

Це рівняння треба вирішувати чисельно.

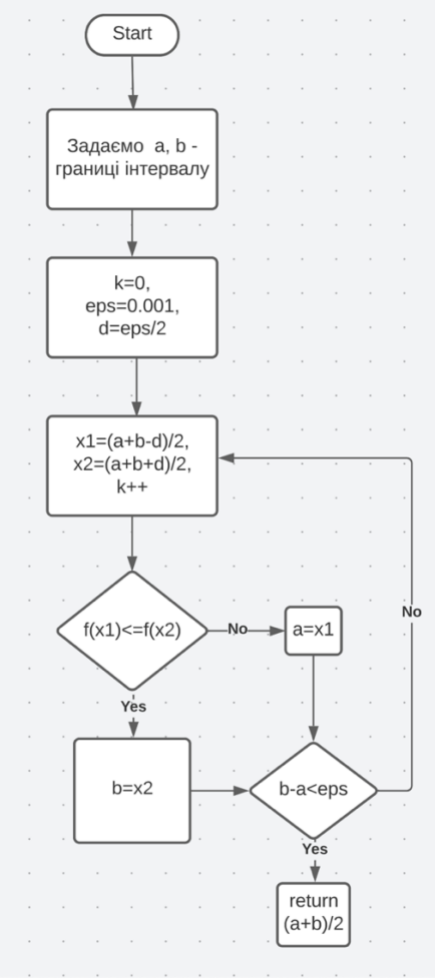
Проте ми можемо побачити, що $\frac{x^2}{2}$ буде завжди ≥ 0, а $\sin(x) \in [-1; 1]$, тобто мінімальне значення функції не може бути меншим за -1

Знайдемо мінімальне значення функції двома методами, аби перевіритися.

1)Метод дихотомії

Обираємо відрізок [a;b] та робимо припущення, що функція на цьому відрізку є унімодальною (має тільки один екстремум на цьому інтервалі). У цьому методі ми рахуємо середину інтервалу [a;b] (точка c) та знаходимо x1=c-d/2 та x2=c+d/2. Ми додаємо та віднімаємо похибку дельта (є(0;eps)) від середини відрізку, аби x1 та x2 залишалися на деякій відстані від середини та не було проблеми, що алгоритм застряг в точці мінімуму функції. Отже ми на кожній інтерації зменшуємо інтервал у 2 рази та відкидаємо той відрізок, де f буде більшим. Робимо такі дії, допоки двожина інтервалу не стане мешою за eps. Повертаємо середину відрізка.

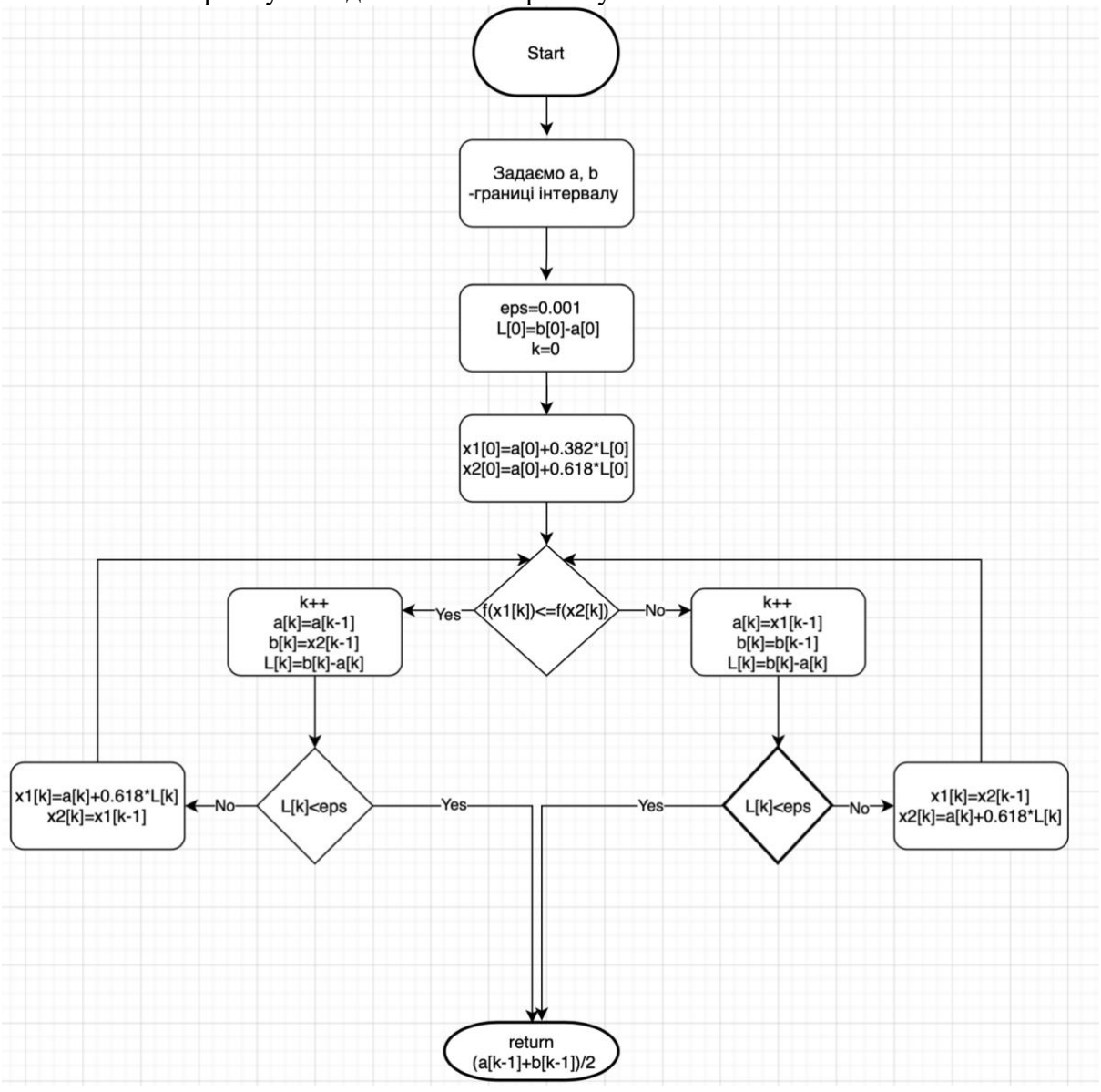
Блок-схема алгоритму метода дихотомії:



2)Метод золотого перетину

Спочатку обираємо інтервал невизначеності- відрізок [a;b] та робимо припущення, що функція на цьому відрізку є унімодальною (має тільки один екстремум на цьому інтервалі). Принцип методу полягає в ділені інтервалі на дві такі частини, що відношення меншої частини до більшої буде рівне золотому перетину (0.618) (0.382=1-0.618). Отже маємо відрізок [a;x2], для якого x1 – золотий перетин, а такод відрізок [x1;b], де x2-золотий перетин. Потім обираємо той інтервал, де функція приймає менше значення. Робимо цей процес поки довжина інтервалу не стане меншою за eps.

Блок-схема алгоритму метода золотого перетину:



Код програми:

```
#include <iostream>
#include <math.h>
using namespace std;

float func(float x){
    float f;
    f=((pow(x,2))/2)-sin(x);
    return f;
}

int kkk=0,k=0;

float find_x(){//дихотомии

    float a=0,b=1,d,e=0.001, Aa, AA,f1,f2;
    d=e/2;

    do{kkk++;

        Aa=(a+b-d)/2;
        AA=(a+b+d)/2;

        f1=func(Aa);
        f2=func(AA);


        if(f1<=f2){
            b=AA;
        }
```

```
        else{
            a=Aa;
        }

        if(b-a<e)break;

    }while(1);

    return (a+b)/2;
}


float findX (){//метод золотого перетину
    float a[1000],b[1000],x1[1000],x2[1000],L[1000],eps=0.001;
    a[0]=0;
    b[0]=1;

    L[0]=b[0]-a[0];

    x1[0]=a[0]+0.382*L[0];
    x2[0]=a[0]+0.618*L[0];

    do{
//        cout<<a[k]<<"  "<<b[k]<<" "<<func((a[k]+b[k])/2)<<endl;

        if(func(x1[k])<=func(x2[k])){
            k++;
            a[k]=a[k-1];
            b[k]=x2[k-1];
            L[k]=b[k]-a[k];
```

```
            if(L[k]<eps)break;

            x1[k]=a[k]+0.618*L[k];
            x2[k]=x1[k-1];
        }

        else{
            k++;
            a[k]=x1[k-1];
            b[k]=b[k-1];
            L[k]=b[k]-a[k];

            if(L[k]<eps)break;

            x1[k]=x2[k-1];
            x2[k]=a[k]+0.618*L[k];
        }



    }while(1);


    return (a[k-1]+b[k-1])/2;
}


int main(){
    float x=find_x();
```

```
    cout<<"Методом дихотомії: "<<endl<<"x= "<<x<<endl<<"f(x)=
"<<func(x)<<endl<<"Ітерацій: "<<kkk<<endl<<endl;



    float t=findX();
    cout<<"Методом золотого перетину: "<<endl<<"x= "<<t<<endl<<"f(x)=
"<<func(t)<<endl<<"Ітерацій: "<<k<<endl<<endl;
}
```

Результат роботи:

```
Методом дихотомії:
x= 0.738894
f(x)= -0.400489
Ітерацій: 11

Методом золотого перетину:
x= 0.736053
f(x)= -0.400481
Ітерацій: 6
```

Бачимо, що метод золотого перетину виявився набагато швидшим.
Розв’язокки співпали з заданою точністю eps, а значення функції майже повністю співпали.

2) $F(x) = x^2 + (p+q) * x + p * q,$
де $p = 0,\ q = 6$

Отже $\boldsymbol{F(x) = x^2 + 6x}$

Спочатку знайдемо розв'язок аналітично:
Знаходимо першу похідну та прирівнюємо її до нуля:
$F'(x) = 2x + 6 = 0$
$x = -3$ – стаціонарна точка
$F(-5) = -4,\quad F(0) = 6,$ отже похідна змінює знак с $-$ на $+$ $\Longrightarrow$ це точка min
$\boldsymbol{F(-3) = -9}$

Тепер знайдемо мінімум функції чисельно – за допомогою тих же методів (дихотомії та золотого перетину), але за інтервал [a;b] ми тепер візьмемо [-10;0]

Код програми:

```
#include <iostream>
#include <math.h>
using namespace std;

float func(float x){
    float f;
    f=pow(x,2)+6*x;
    return f;
}

int kkk=0,k=0;

float find_x(){//дихотомии

    float a=-10,b=0,d,e=0.001, Aa, AA,f1,f2;
    d=e/2;
```

```
    do{kkk++;

        Aa=(a+b-d)/2;
        AA=(a+b+d)/2;

        f1=func(Aa);
        f2=func(AA);


        if(f1<=f2){
            b=AA;
        }

        else{
            a=Aa;
        }

        if(b-a<e)break;

    }while(1);

    return (a+b)/2;
}


float findX (){//метод золотого перетину
    float a[1000],b[1000],x1[1000],x2[1000],L[1000],eps=0.001;
    a[0]=-10;
    b[0]=0;

    L[0]=b[0]-a[0]; //должно быть >0
```

```
    x1[0]=a[0]+0.382*L[0];
    x2[0]=a[0]+0.618*L[0];

    do{
//        cout<<a[k]<<"   "<<b[k]<<" "<<func((a[k]+b[k])/2)<<endl;

        if(func(x1[k])<=func(x2[k])){
            k++;
            a[k]=a[k-1];
            b[k]=x2[k-1];
            L[k]=b[k]-a[k];

            if(L[k]<eps)break;

            x1[k]=a[k]+0.618*L[k];
            x2[k]=x1[k-1];
        }

        else{
            k++;
            a[k]=x1[k-1];
            b[k]=b[k-1];
            L[k]=b[k]-a[k];

            if(L[k]<eps)break;

            x1[k]=x2[k-1];
            x2[k]=a[k]+0.618*L[k];
        }
```

```
    }while(1);


    return (a[k-1]+b[k-1])/2;
}


int main(){
    float x=find_x();
    cout<<"Методом дихотомії: "<<endl<<"x= "<<x<<endl<<"f(x)=
"<<func(x)<<endl<<"Ітерацій: "<<kkk<<endl<<endl;


    float t=findX();
    cout<<"Методом золотого перетину: "<<endl<<"x= "<<t<<endl<<"f(x)=
"<<func(t)<<endl<<"Ітерацій: "<<k<<endl<<endl;
}
```

Результат роботи програми:

Методом дихотомії:
x= -3.00105
f(x)= -9
Ітерацій: 15

Методом золотого перетину:
x= -3.09044
f(x)= -8.99182
Ітерацій: 8

Бачимо, що метод золотого перетину знову виявився швидшим. Проте, метод дихотомії дав більш точний результат (але було здійснено аж 15 ітерацій). Вцілому, обидва методи знайшли мінімальне значення функції з похибкою eps=0.001

Висновок:
Було знайдено мінімальне значення двох функцій двома методами одномірної оптиматизації – методом дихотомії та методом золотого перетину. Метод золотого перетину виявився більш ефективним, аніж метод дихотомії, проте метод дихотомії може дати більш точний результат, адже він використовує більше ітерацій. Знайдені значення мінімуму функцій співпали в обох методах (з заданою точністю eps).