Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут ім. І. Сікорського»

Кафедра автоматизації проектування енергетичних процесів і систем

Лабораторна робота №3
з дисципліни «Чисельні методи в моделюванні енергетичних процесів»
«**Розв'язання нелінійних рівнянь**»
Варіант №1

Виконав:
студент 2-го курсу, ТЕФ
групи ТР-15
Руденко В.І.

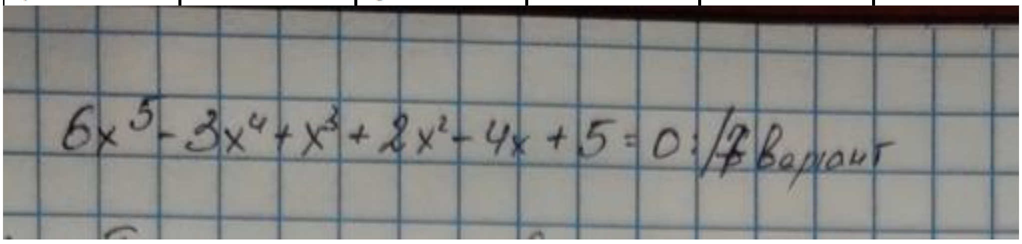
КИЇВ-2022

Мета роботи

Навчитися розв’язувати нелінійні рівняння. Написати універсальну програму для обчислення нелінійного рівняння згідно варіанту. Попередньо виконати допрограмовий етап, результатом якого повинні бути проміжки, щодо яких проводиться уточнення. Перевірити розв’язок уточнення коренів за методами бісекції, хорд, дотичних у Mathcad. Знайти середньоквадратичну похибку обчислень.

Варіант завдання (Вихідна система рівнянь).

№ вар.	Коефіцієнти поліному					
7	2	-3	1	2	-4	1



Теоретична частина

Знаходження коренів рівнянь за допомогою чисельних методів складається з двох етапів:

1)Відокремлення коренів: знаходження сукупності проміжків, кожен з яких містить один з коренів рівняння.

2) Уточнення коренів: знаходження приблизного значення коренів із заданою точністю ε. Розглядається рівняння:

fx=0

Де fx=Pnx=anxn+an-1xn-1+...+a1x+a0=0 таan>0 , для якого потрібно знайти його дійсні корені x*

Перший етап виконується із використанням теорем, які допомагають виділити межі розташування додатніх та від’ємних коренів та знайти проміжки, що містять ці корені.

Теорема про границі усіх (комплексних) коренів рівняння.

Нехай A=maxai, i=0,..., n-1;B=maxai, i=1,..., n

Тоді всі (комплексні) корені рівняння (1) лежать у кільці

|a0|B+|a0| ≤|x*|≤an+A|an|

Теорема про верхню межу додатніх коренів

Нехай A=ai , ai<0;m=maxi, ai<0;

Тоді R=1+n-mAa – верхня межа додатніх коренів:

x* ≤R,fx*=0

Цю ж теорему можна застосувати для визначення нижньої межі додатних коренів заміною x=1yтоді x* ≥1Ry. Заміна знаку x:= - y дозволяє обмежити від’ємні корені.

Теорема Гюа про наявність комплексних коренів

Якщо $\exists$ $k:1<k<n$ $ak2<ak-1\cdot ak+1$, то рівняння має хоча б одну пару комплексно спряжених коренів.

Теорема Штурма про чередування коренів

Нехай $fx=Pn(x)$ поліном без кратних коренів. Утворимо послідовність многочленів:

$$f0=fx;$$
$$f1=f'x;$$
$$fi+1=-fi+1modfi, i=1,\ldots$$

- кожний наступний многочлен є залишком від ділення двох попередніх многочленів, взятим з протилежним знаком. Стверджується, що кількість дійсних коренів полінома $f0(x)$ на довільному відрізку $[a;b]$дорівнює різниці між кількістю змін знаку у цій послідовності при $x = a$ та $x = b$.

Другий етап передбачає застосування одного з нижченаведених методів до кожного з проміжків, отриманих на першому етапі.

Метод бісекції.

Дано: кінці інтервалу a та b, точність . На кожному кроці інтервал ділять навпіл

$$c= (a+b)2$$

та залишають той підінтервал, до якого належить корінь.

Метод хорд.

Вхідні дані аналогічні тим, що використовуються методом бісекції. Проводиться січна до графіку функції. Точкою перетину її з віссю абсцис ділять інтервал:

$$c= a\cdot fb-b\cdot f(a)fb-f(a)$$

та залишають той підінтервал, до якого належить корінь

Метод Ньютона (дотичних).

Дано: початкове наближення $x0$ та точність . Проводять дотичні до графіку функції, що дає формулу

$$xk-1=xk- f(xk)f'(xk)$$

Перевірка існування кореня на відрізку $[a;b]$здійснюється так: корінь належить відрізку, якщо $fa\cdot fb<0$

якщо $fa\cdot fb>0$, то відрізок не містить коренів.

Проміжні результати виконання:

```
Method Polovinki:
F(-1.0256) : -0.000190984
        Used Iteration: 76
---------------------
---------------------
Method Hordiv:
F(-1.02559) : 5.42582e-06
        Used Iteration: 230
---------------------
---------------------
Method Nbutona:
F(-1.02559) : 5.15624e-07
        Used Iteration: 7
```

Рисунок 1 Фінальні результати трьох методів

```
Method Nbutona:
Step : X = -2-(Function(-2)/Function`(-2)
---------------------
Step : X = -1.6059-(Function(-1.6059)/Function`(-1.6059)
---------------------
Step : X = -1.31549-(Function(-1.31549)/Function`(-1.31549)
---------------------
Step : X = -1.12825-(Function(-1.12825)/Function`(-1.12825)
---------------------
Step : X = -1.04309-(Function(-1.04309)/Function`(-1.04309)
---------------------
Step : X = -1.0262-(Function(-1.0262)/Function`(-1.0262)
---------------------
Step : X = -1.0256-(Function(-1.0256)/Function`(-1.0256)
---------------------
F(-1.02559) : 5.15624e-07
        Used Iteration: 7
```

Рисунок 2 Метод Ньютона

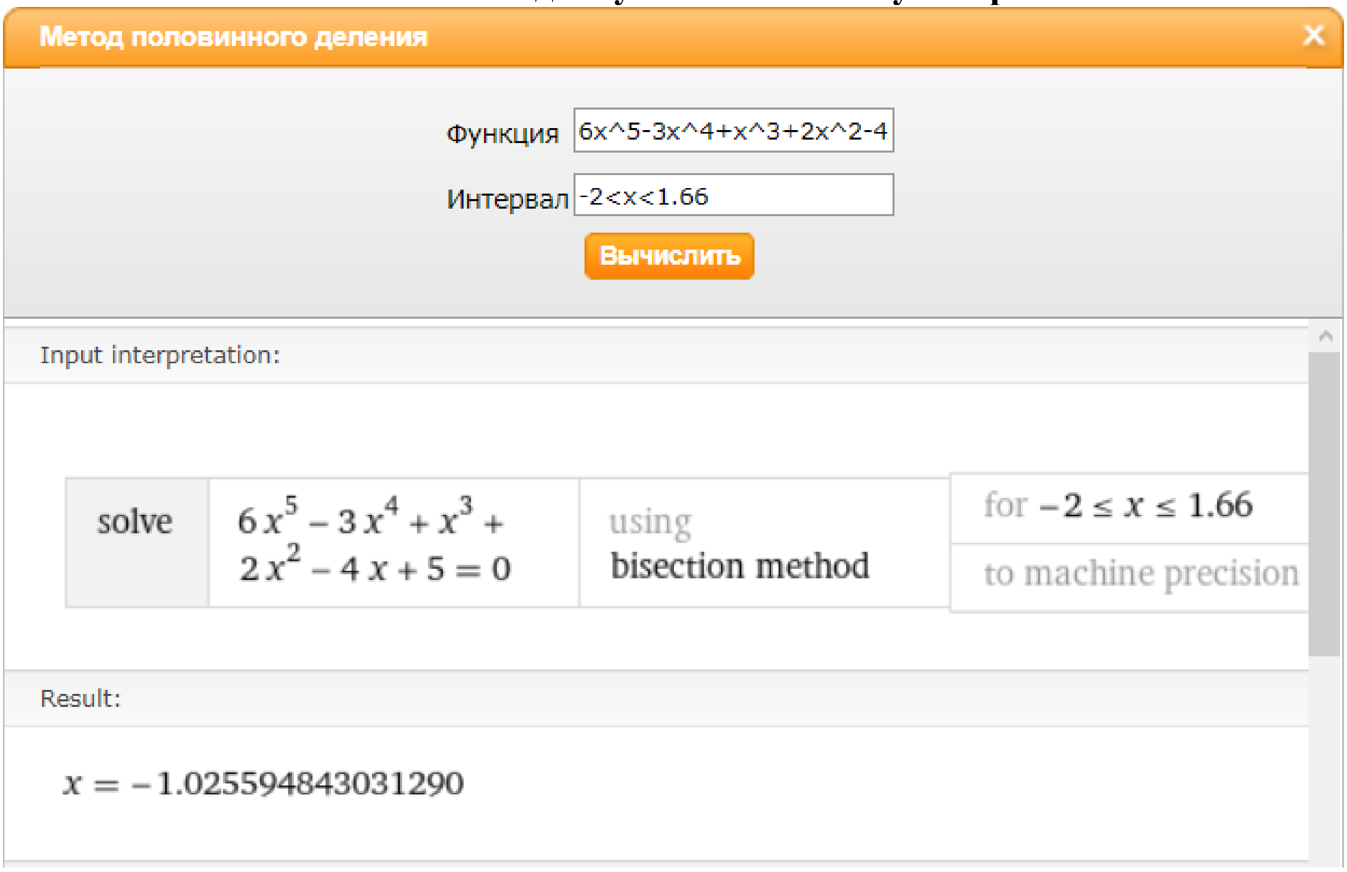
```
Method Hordiv:
Step : -(Function(-2) * (1.66 - -2) / (Function(1.66) - Function(-2))) + -2
Function(-2)*Function(0.881839)<0 : B=X
---------------------
Step : -(Function(-2) * (0.881839 - -2) / (Function(0.881839) - Function(-2))) +
 -2
Function(-2)*Function(0.818526)<0 : B=X
---------------------
Step : -(Function(-2) * (0.818526 - -2) / (Function(0.818526) - Function(-2))) +
 -2
Function(-2)*Function(0.764071)<0 : B=X
---------------------
Step : -(Function(-2) * (0.764071 - -2) / (Function(0.764071) - Function(-2))) +
 -2
Step : -(Function(-2) * (0.544648 - -2) / (Function(0.544648) - Function(
---------------------
Step : -(Function(-2) * (0.715063 - -2) / (Function(0.715063) - Function(-2))) +
 -2
Function(-2)*Function(0.669605)<0 : B=X
---------------------
Step : -(Function(-2) * (0.669605 - -2) / (Function(0.669605) - Function(-2))) +
 -2
Function(-2)*Function(0.626526)<0 : B=X
---------------------
```

Рисунок 3 Метод Хорд

```
Method Polovinki:
Step : C=(A+B)/2        C:-0.17
Step : F(-2)*F(1.66) < 0
B=C
---------------------
Step : C=(A+B)/2        C:-1.085
Step : F(-2)*F(-0.17) > 0
A=C
---------------------
Step : C=(A+B)/2        C:-0.6275
Step : F(-1.085)*F(-0.17) < 0
B=C
---------------------
Step : C=(A+B)/2        C:-0.85625
Step : F(-1.085)*F(-0.6275) < 0
B=C
---------------------
Step : C=(A+B)/2        C:-0.970625
Step : F(-1.085)*F(-0.85625) < 0
B=C
---------------------
Step : C=(A+B)/2        C:-1.02781
Step : F(-1.085)*F(-0.970625) > 0
A=C
---------------------
Step : C=(A+B)/2        C:-0.999219
Step : F(-1.02781)*F(-0.970625) < 0
B=C
---------------------
```

Рисунок 4 Метод Половинок

Розв’язок задачі у Онлайн Калькуляторі



Власний варіант розв’язку виконання допрограмового етапу:

$6x^5-3x^4+x^3+2x^2-4x+5=0$; 7 варіант

1 За Теоремою про верхню та нижню межу

$R=1+\sqrt[k]{A}$ $A=\max|a_i|,\ a_i<0$

$6x^5-3x^4+x^3+2x^2-4x+5=0 \quad /:6$

$x^5-0{,}5x^4+0{,}16x^3+0{,}33x^2-0{,}66x+0{,}83=0$

$A=0{,}66;\ k=1$

$R=1+\sqrt[1]{0{,}66}=1{,}66$ - верхня межа

$f(-x)=-x^5+0{,}5x^4-0{,}16x^3-0{,}33x^2+0{,}66x-0{,}83=0$

$A=0{,}83,\ k=0$

$-R=-2$ - нижня межа

2 За Теоремою про границі усіх (комплексних) коренів рівняння

$A=|\max a_i|\ i=0,\dots,n-1$ $B=|\max a_i|,\ i=1,\dots,n$

$A=5,\ B=6$

$\frac{5}{5+6}\le|x|\le\frac{6+5}{6} \Rightarrow x\in[0{,}45;\ 1{,}83]$

3 За Теоремою Гюа про наявність комплексних ~~чисел~~ коренів

$\exists k\ 1<k<n$ $[6;-3;1;2;-4;5=0]$

$9>6$
$1>-6$
$4>-4$
$16>10$

комплексних чисел немає

$6x^5-3x^4+x^3+2x^2-4x+5=0$

$f'() = 30x^4-12x^3+3x^2+4x-4$

$f_2()$

$$\begin{array}{l|l} 6x^5-3x^4+x^3+2x^2-4x+5 & 30x^4-12x^3+3x^2+4x-4 \\ -\ 6x^5+2{,}4x^4+0{,}6x^3-0{,}4x^2+0{,}4x & -\frac{1}{5}x+\frac{1}{50} \\ \quad -0{,}6x^4+1{,}6x^3+1{,}6x^2-3{,}6+5 & \\ \quad 0{,}6x^4-0{,}24x^3+0{,}06+0{,}08x-0{,}08 & \\ \qquad 1{,}36x^3+1{,}66x^2-3{,}52x+4{,}92 \Rightarrow x^3+1{,}22x^2-2{,}58x+3{,}61 & \end{array}$$

$f_3()$

$$\begin{array}{l|l} 30x^4-12x^3+3x^2+4x-4 & x^3+1{,}22x^2-2{,}58x+3{,}61 \\ -\ 30x^4+36{,}6x^3-77{,}4x^2+108{,}3x & -30x-24{,}6 \\ \quad 24{,}6x^3-74{,}4x^2+112{,}3-4 & \\ \quad 24{,}6x^3-30x^2+63{,}46x-88 & \\ \qquad -104{,}4x^2+175{,}76x-92{,}8 \Rightarrow x^2-1{,}68x-0{,}8 & \end{array}$$

$f_4()$

$$\begin{array}{l|l} x^3+1{,}22x^2-2{,}58x+3{,}61 & x^2-1{,}68x-0{,}8 \\ -\ x^3+1{,}68x^2+0{,}84 & -x-2{,}9 \\ \quad 2{,}9x^2-2{,}5x+3{,}61 & \\ \quad 2{,}9x^2+4{,}87x+2{,}32 & \\ \qquad 2{,}37x+5{,}93 \Rightarrow x+2{,}5 & \end{array}$$

$f_5()$

$$\begin{array}{l|l} x^2-1{,}68x-0{,}8 & x+2{,}5 \\ -\ x^2-2{,}5x & x+4{,}18 \\ \quad -4{,}18x-0{,}8 & \\ \quad 4{,}18x+10{,}45 & \\ \qquad 9{,}65 & \end{array}$$

	-2	1,66
f_0	−	+
f_1	+	+
f_2	−	−
f_3	−	+
f_4	−	−
f_5	−	−
	2	3

1 корінь рівняння

Порівняння результатів

	Програма	Калькулятор
Точка X	**-1.02559**	**-1.02559**

Висновок:

В ході виконання лабораторної роботи №3 було вивчено та опрацьовано методи розв'язку нелінійних рівнянь. Написано універсальну програму що здатна обчислювати нелінійні рівняння трьома заданими методами. Виконано допрограмовий етап, на якому знайдені проміжки на якому наявні шукані X в кількості 1 одиниці. В результаті досліджень виявлено що метод Ньютона лейбніца є найшвидшим, оскільки займає найменшу кількусть ітерацій, середній варіантом є метод бісекції, що потребує найменшу кількість практичних знань для написання коду, та метод з найгіршими результатами виявися метод хорд, який також потребує більшої кількості навичок для написання – що робить його неефективним в будь якому направлені. Всі три методи мають правильний розв'язок, оскільки він сходиться з результатами виконання різних калькуляторів.

Лістинг програми:

```
#include <iostream>
#include <vector>

#define ZalikValue 227
#define variant 7
#define k 5
#define a 2


using namespace std;

//////////////////////////////
float   a0=1,
        a1=-4,
        a2=2,
        a3=1,
        a4=-3,
        a5=2;
float XRange[2] = {-2,1.66};
float point = 0.00001;
bool Complex = false;
//////////////////////////////

vector<float> TempArray;
vector<float> Array = { a5*(1+a), a4, a3, a2, a1, k*a0 };

void Polovinki();
void Hord();
void Nbuton();

double Function(float Temp);
double PFunction(float Temp);

int main()
```

```
{
	cout << "Method Polovinki:" << endl;
	Polovinki();
	cout << "----------------------" << endl;
	cout << "----------------------" << endl;
	cout << "Method Hordiv:" << endl;
	Hord();
	cout << "----------------------" << endl;
	cout << "----------------------" << endl;
	cout << "Method Nbutona:" << endl;
	Nbuton();

}

double Function(float Temp)
{
	double result=0;
	for(int i=Array.size()-1;i>=0;i--)
	{
		if(i==0)
			result += Array[Array.size()-1-i];
		else
			result += Array[Array.size()-1-i]*pow(Temp, i);
	}
	return result;
}
double PFunction(float Temp)
{
	double result=0;
	for(int i=Array.size()-1;i>=0;i--)
	{
		if(i==0) continue;
		result += Array[Array.size()-1-i]*pow(Temp, i-1)*i;
	}
	return result;
}

double PPFunction(float Temp)
{
	double result=0;
	for(int i=Array.size()-1;i>=0;i--)
	{
		if(i==0) continue;
		result += Array[Array.size()-1-i]*pow(Temp, i-2)*i*(i-1);
	}
	return result;
}

void Polovinki()
{
	int INTER = 0;
	float A=XRange[0], B=XRange[1], C=XRange[0];
	do
	{
		C=(A+B)/2; INTER++;
#ifndef Test
		cout << "Step : C=(A+B)/2\tC:" << C << endl;
		cout << "Step : F("<< A <<")*F("<<B<<") "<<
(Function(A)*Function(C)<0? "< 0\nB=C" : "> 0\nA=C") << endl;
		cout << "---------------------" << endl;
#endif
```

```
				INTER+=2;
				if(Function(A)*Function(C)<0)
						B=C;
				else
						A=C;
				INTER++;
		}
		while (Function(C)!=0 && (B-A) > point);
		cout << "F("<< C <<") : " << Function(C) << endl << "\tUsed Iteration: " <<
INTER << endl;
}


void Hord()
{
		int INTER = 0;
		double A=XRange[0], B=XRange[1], X;
		do
		{
				INTER++;
				X = -(Function(A) * (B - A) / (Function(B) - Function(A))) + A;
#ifndef Test
				cout << "Step : -(Function("<<A<<") * ("<<B<<" - "<<A<<") /
(Function("<<B<<") - Function("<<A<<"))) + "<<A<<"" << endl;
				cout << "Function("<<A<<")*Function("<<X<<")<0 : B=X" << endl;
				cout << "----------------------" << endl;
#endif
				INTER++;
				if(Function(A)*Function(X)<0)
						B=X;

				if(B-A <= point)
						break;


		}
		while (abs(Function(X))>point);
		cout << "F("<< X <<") : " << Function(X) << endl << "\tUsed Iteration: " <<
INTER << endl;
}

void Nbuton()
{
		int INTER = 0,i=0;
		double A=XRange[0], B=XRange[1], X=0;

		if(Function(A)*PPFunction(A)>0)
				X=A;
		else if(Function(B)*PPFunction(B)>0)
				X=B;

		do
		{
				INTER++;
#ifndef Test
						cout << "Step : X = "<<X<<"-
(Function("<<X<<")/Function`("<<X<<")" << endl;
						cout << "----------------------" << endl;
#endif

				X = X-(Function(X)/PFunction(X));
```

```
        }
        while (abs(Function(X))>point);
        cout << "F("<< X <<") : " << Function(X) << endl << "\tUsed Iteration: " <<
INTER << endl;
}
```