Федеральное государственное автономное образовательное учреждение высшего образования **«Национальный исследовательский университет ИТМО»**

Факультет Программной Инженерии и Компьютерной Техники

Лабораторная работа №2
по дисциплине «Вычислительная математика»
Вариант 16

Преподаватель:

Машина Е.А.

Выполнила:

Шайхутдинова Н.В.

P3208

Санкт-Петербург

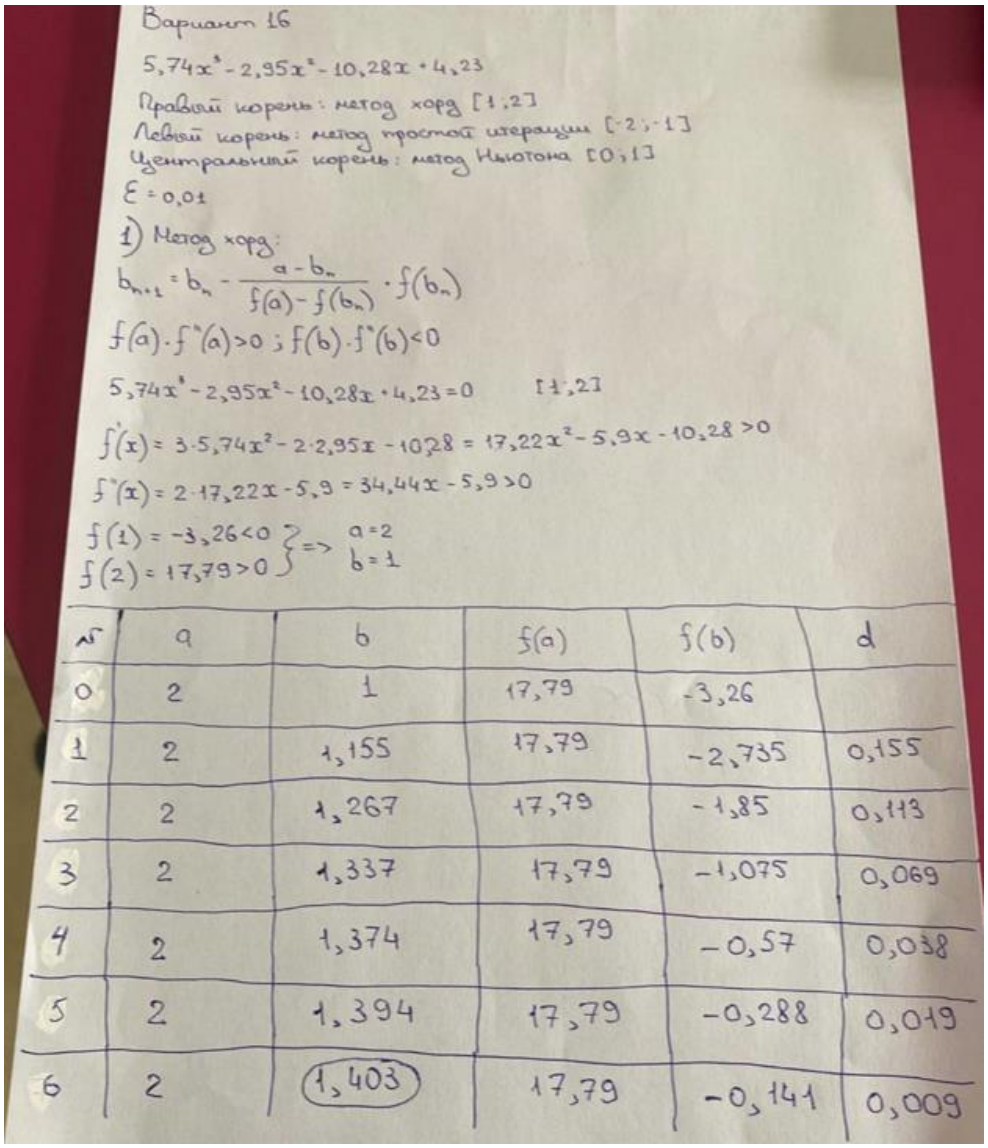
2024 г.

Цель лабораторной работы

Изучить численные методы решения нелинейных уравнений и их систем, найти корни заданного нелинейного уравнения/системы нелинейных уравнений, выполнить программную реализацию методов.

Порядок выполнения работы

Вычислительная реализация задачи



Вариант 16

$5{,}74x^3 - 2{,}95x^2 - 10{,}28x + 4{,}23$

Правый корень: метод хорд [1;2]

Левый корень: метод простой итерации [-2;-1]

Центральный корень: метод Ньютона [0;1]

$\varepsilon = 0{,}01$

1) Метод хорд:

$$b_{n+1} = b_n - \frac{a - b_n}{f(a) - f(b_n)} \cdot f(b_n)$$

$f(a) \cdot f''(a) > 0$; $f(b) \cdot f''(b) < 0$

$5{,}74x^3 - 2{,}95x^2 - 10{,}28x + 4{,}23 = 0$ [1;2]

$f'(x) = 3 \cdot 5{,}74x^2 - 2 \cdot 2{,}95x - 10{,}28 = 17{,}22x^2 - 5{,}9x - 10{,}28 > 0$

$f''(x) = 2 \cdot 17{,}22x - 5{,}9 = 34{,}44x - 5{,}9 > 0$

$f(1) = -3{,}26 < 0$
$f(2) = 17{,}79 > 0$ } => $a = 2$, $b = 1$

№	a	b	f(a)	f(b)	d
0	2	1	17,79	-3,26	
1	2	1,155	17,79	-2,735	0,155
2	2	1,267	17,79	-1,85	0,113
3	2	1,337	17,79	-1,075	0,069
4	2	1,374	17,79	-0,57	0,038
5	2	1,394	17,79	-0,288	0,019
6	2	(1,403)	17,79	-0,141	0,009

2) Метод простой итерации:

$5{,}74x^3 - 2{,}95x^2 - 10{,}28x + 4{,}23 = 0$

$[-2;-1]$

$\varepsilon = 0{,}01$

$$x = \left(\frac{2{,}95x^2 + 10{,}28x - 4{,}23}{5{,}74}\right)^{1/3}$$

$$f'(x) = \frac{1}{3}\left(\frac{2{,}95x^2 + 10{,}28x - 4{,}23}{5{,}74}\right)^{-\frac{2}{3}} \cdot \left(\frac{2{,}95 \cdot 2}{5{,}74}x + \frac{10{,}28}{5{,}74}\right) =$$

$$= \left(0{,}514x^2 + 1{,}791x - 0{,}737\right)^{-\frac{2}{3}} \cdot \left(0{,}343x + 0{,}597\right) = \frac{0{,}343x + 0{,}597}{(0{,}514x^2 + 1{,}791x - 0{,}737)^{2/3}}$$

$$\left|\frac{0{,}343x + 0{,}597}{(0{,}514x^2 + 1{,}791x - 0{,}737)^{2/3}}\right| = |f'(x)| < 1$$

Пусть $x_0 = -2$

N	x_k	x_{k+1}	$f(x_{k+1})$	$\lvert x_{k+1} - x_k \rvert$
0	-2	-1,3129	-0,34832	0,687
1	-1,3129	-1,30106	-0,0304	0,0118
2	-1,30106	(-1,30002)	-0,00272	0,001044

394

403

3) Метод Ньютона:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

$5{,}74x^3 - 2{,}95x^2 - 10{,}28x + 4{,}23 = 0$

$[0;1]$

$\varepsilon = 0{,}01$

$f''(x) = 34{,}44x - 5{,}9$ ← не сохраняет знак на $[0;1]$

⇒ не можем использовать метод Ньютона

На своё усмотрение найду корень методом половинного деления:

N	a	b	x	f(a)	f(b)	f(x)	|a-b|
0	0	1	0,5	4,23	-3,26	-0,93	1
1	0	0,5	0,25	4,23	-0,93	1,565	0,5
2	0,25	0,5	0,375	1,565	-0,93	0,263	0,25
3	0,375	0,5	0,4375	0,263	-0,93	-0,351	0,125
4	0,375	0,4375	0,406	0,263	-0,351	-0,048	0,0625
5	0,375	0,406	0,39	0,263	-0,048	0,106	0,03125
6	0,39	0,406	0,398	0,107	-0,048	0,029	0,0156
7	0,398	0,406	(0,4023)	0,029	-0,048	-0,0088	0,007813

Ответ: правый корень = 1,403
левый корень = -1,3
центральный корень = 0,402

Программная реализация задачи

Метод половинного деления, метод секущих, метод простой итерации, метод простой итерации (для систем нелинейных уравнений)

Листинги программ

```
def checkConvergenceForSystem(userChoice):
    x = int(userChoice[9])
    y = int(userChoice[10])
    dfdx = eval(
        userChoice[3].replace("y", "(" + str(y) + ")").replace("x", "(" + str(x) + ")")
    )
    dfdy = eval(
        userChoice[4].replace("y", "(" + str(y) + ")").replace("x", "(" + str(x) + ")")
    )
    dgdx = eval(
        userChoice[5].replace("y", "(" + str(y) + ")").replace("x", "(" + str(x) + ")")
    )
    dgdy = eval(
        userChoice[6].replace("y", "(" + str(y) + ")").replace("x", "(" + str(x) + ")")
    )
    if max(abs(dfdx) + abs(dfdy), abs(dgdx) + abs(dgdy)) >= 1:
        print("Проверка на сходимость не прошла!")
        return False
    else:
        return True



def simpleIterationMethod(userChoice):
    M = 30
    iteration = 0
    if checkConvergenceForSystem(userChoice) == False:
        M = 10
    e = float(userChoice[11])
```

```
x0 = int(userChoice[9])
y0 = int(userChoice[10])
x = 0
y = 0
curEx = 100
curEy = 100
print("№ -> x -> y -> |xn+1 - x| -> |yn+1 - y|")
while iteration < M and (curEx > e or curEy > e):
    iteration += 1
    x = eval(
        userChoice[7]
        .replace("y", "(" + str(y0) + ")")
        .replace("x", "(" + str(x0) + ")")
    )
    y = eval(
        userChoice[8]
        .replace("y", "(" + str(y0) + ")")
        .replace("x", "(" + str(x0) + ")")
    )
    curEx = abs(x - x0)
    curEy = abs(y - y0)
    x0 = x
    y0 = y
    print(iteration, x, "->", y, "->", curEx, "->", curEy)
answer = []
answer.append(x)
answer.append(curEx)
answer.append(y)
answer.append(curEy)
answer.append(iteration)
return answer
```

```
def chooseSystem(userChoice):
    checkerForSystem = True
    while checkerForSystem:
        print(
            'Выберите систему нелинейных уравнений: \n "1" -> 0.1x^2+x+0.2y-0.3=0 \n 0.2x^2+y+0.1xy-0.7=0\n  '
            '      0<x<1, 0<y<1\n "2" -> x^2+y^2+x=0.25\n          0.234x^2+0.23yx+0.2y=0 \n        -1<x<0, -1<y<0'
        )
        selection = input()
        if selection == "1":
            userChoice.append("-0.2*x")
            userChoice.append("-0.4*y")
            userChoice.append("-0.4*x-0.1*y")
            userChoice.append("-0.1*x")
            userChoice.append("0.3-0.1*x**2-0.2*y**2")
            userChoice.append("0.7-0.2*x**2-0.1*x*y")
            userChoice.append("1")
            userChoice.append("1")
            checkerForSystem = False
            chooseE(userChoice)

        elif selection == "2":
            userChoice.append("-2*x")
            userChoice.append("-2*y")
            userChoice.append("-2.34*x-1.15*y")
            userChoice.append("-1.15*x")
            userChoice.append("0.25-x**2-y**2")
            userChoice.append("(-0.234*x**2)/(0.23*x+0.2)")
            userChoice.append("0")
            userChoice.append("0")
            checkerForSystem = False
```

```
            chooseE(userChoice)

        else:
            print("Я Вас не понимаю :(")


def hordaMethod(userChoice):
    iteration = 1
    e = float(userChoice[8])
    a = float(userChoice[6])
    b0 = float(userChoice[7])
    fa = eval(userChoice[3].replace("x", "(" + str(a) + ")"))
    fb0 = eval(userChoice[3].replace("x", "(" + str(b0) + ")"))
    fapp = eval(userChoice[5].replace("x", "(" + str(a) + ")"))
    fb0pp = eval(userChoice[5].replace("x", "(" + str(b0) + ")"))
    x = a
    if fa * fapp < 0 and fb0 * fb0pp > 0:
        a = b0
        b0 = x
    fa = eval(userChoice[3].replace("x", "(" + str(a) + ")"))
    fb0 = eval(userChoice[3].replace("x", "(" + str(b0) + ")"))
    b = b0 - (a - b0) / (fa - fb0) * fb0
    curE = abs(b - b0)
    fb = eval(userChoice[3].replace("x", "(" + str(b) + ")"))
    print("№ -> a -> b -> f(a) -> f(b) -> |bn+1 - b|")
    print(iteration, "->", a, "->", b, "->", fa, "->", fb, "->", curE)
    while curE > e:
        iteration += 1
        b0 = b
        fb0 = eval(userChoice[3].replace("x", "(" + str(b0) + ")"))
        b = b0 - (a - b0) / (fa - fb0) * fb0
        curE = abs(b - b0)
```

```
        print(iteration, "->", a, "->", b, "->", fa, "->", fb, "->", curE)
    answer = []
    answer.append(b)
    fb = eval(userChoice[3].replace("x", "(" + str(b) + ")"))
    answer.append(fb)
    answer.append(iteration)
    return answer



def checkConvergence(userChoice):
    a = float(userChoice[6])
    b = float(userChoice[7])
    fa = eval(userChoice[3].replace("x", "(" + str(a) + ")"))
    fb = eval(userChoice[3].replace("x", "(" + str(b) + ")"))
    fap = eval(userChoice[4].replace("x", "(" + str(a) + ")"))
    fbp = eval(userChoice[4].replace("x", "(" + str(b) + ")"))
    fapp = eval(userChoice[5].replace("x", "(" + str(a) + ")"))
    fbpp = eval(userChoice[5].replace("x", "(" + str(b) + ")"))
    if fa * fb >= 0:
        print(
            "Проверка на сходимость не прошла. Значения функции на концах интервала имеют одинаковый знак!"
        )
        return False
    elif (fap >= 0 and fbp >= 0 or fap <= 0 and fbp <= 0) == False:
        print(
            "Проверка на сходимость не прошла. Производная функции не сохраняет знак на отрезке!"
        )
        return False
    elif (fapp > 0 and fbpp > 0 or fapp < 0 and fbpp < 0) == False:
        print(
```

```
            "Проверка на сходимость не прошла. Вторая производная функции не сохраняет знак на отрезке!"
        )
        return False
    else:
        return True


def methodNewton(userChoice):
    M = 30
    iteration = 0
    e = float(userChoice[8])
    a = float(userChoice[6])
    b = float(userChoice[7])
    fa = eval(userChoice[3].replace("x", "(" + str(a) + ")"))
    fb = eval(userChoice[3].replace("x", "(" + str(a) + ")"))
    fapp = eval(userChoice[5].replace("x", "(" + str(b) + ")"))
    fbpp = eval(userChoice[5].replace("x", "(" + str(b) + ")"))
    x0 = a
    if fa * fapp > 0:
        x0 = a
    elif fb * fbpp > 0:
        x0 = b
    if checkConvergence(userChoice) == False:
        M = 10
    fx0 = eval(userChoice[3].replace("x", "(" + str(x0) + ")"))
    fx0p = eval(userChoice[4].replace("x", "(" + str(x0) + ")"))
    x = x0 - fx0 / fx0p
    curE = abs(x - x0)
    print("№ -> xn -> f(xn) -> f'(xn) -> xn+1 -> |xn+1 - x|")
    print(iteration, "->", x0, "->", fx0, "->", fx0p, "->", x, "->", curE)
    while iteration < M and curE > e:
```

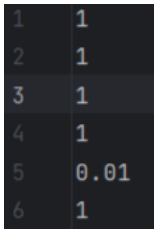
```
        iteration += 1
        x0 = x
        fx0 = eval(userChoice[3].replace("x", "(" + str(x0) + ")"))
        fx0p = eval(userChoice[4].replace("x", "(" + str(x0) + ")"))
        x = x0 - fx0 / fx0p
        curE = abs(x - x0)
        print(iteration, "->", x0, "->", fx0, "->", fx0p, "->", x, "->", curE)
        if iteration == M:
            return False
    answer = []
    answer.append(x)
    fx = eval(userChoice[3].replace("x", "(" + str(x) + ")"))
    answer.append(fx)
    answer.append(iteration)
    return answer


def halfMethod(userChoice):
    a = float(userChoice[6])
    b = float(userChoice[7])
    x = (a + b) / 2
    e = float(userChoice[8])
    fa = eval(userChoice[3].replace("x", "(" + str(a) + ")"))
    fb = eval(userChoice[3].replace("x", "(" + str(b) + ")"))
    fx = eval(userChoice[3].replace("x", "(" + str(x) + ")"))
    curE = abs(a - b)
    iteration = 0
    print("№ -> a -> b -> x -> f(a) -> f(b) -> f(x) -> |a - b|")
    print(
        iteration, "->", a, "->", b, "->", x, "->", fa, "->", fb, "->", fx, "->", curE
    )
    if fx == 0:
```

```
    answer = []
    answer.append(x)
    answer.append(fx)
    answer.append(1)
    return answer
while curE > e:
    iteration += 1
    if fa > 0 and fx > 0 or fa < 0 and fx < 0:
        a = x
    if fb > 0 and fx > 0 or fb < 0 and fx < 0:
        b = x
    x = (a + b) / 2
    fa = eval(userChoice[3].replace("x", "(" + str(a) + ")"))
    fb = eval(userChoice[3].replace("x", "(" + str(b) + ")"))
    fx = eval(userChoice[3].replace("x", "(" + str(x) + ")"))
    curE = abs(a - b)
    print(
        iteration,
        "->",
        a,
        "->",
        b,
        "->",
        x,
        "->",
        fa,
        "->",
        fb,
        "->",
        fx,
        "->",
        curE,
```

```
        )
    answer = []
    answer.append(x)
    answer.append(fx)
    answer.append(iteration)
    return answer
```

Результат программы

test.txt:



Вывод:

Введите путь к файлу:

test

№ -> a -> b -> x -> f(a) -> f(b) -> f(x) -> |a - b|

0 -> -2.0 -> -1.0 -> -1.5 -> -32.92999999999999 -> 5.8199999999999985 -> -6.360000000000003 -> 1.0

1 -> -1.5 -> -1.0 -> -1.25 -> -6.360000000000003 -> 5.8199999999999985 -> 1.2596875 -> 0.5

2 -> -1.5 -> -1.25 -> -1.375 -> -6.360000000000003 -> 1.2596875 -> -2.1341015625000015 -> 0.25

3 -> -1.375 -> -1.25 -> -1.3125 -> -2.1341015625000015 -> 1.2596875 -> -0.337397460937499 -> 0.125

4 -> -1.3125 -> -1.25 -> -1.28125 -> -0.337397460937499 -> 1.2596875 -> 0.48557189941405987 -> 0.0625

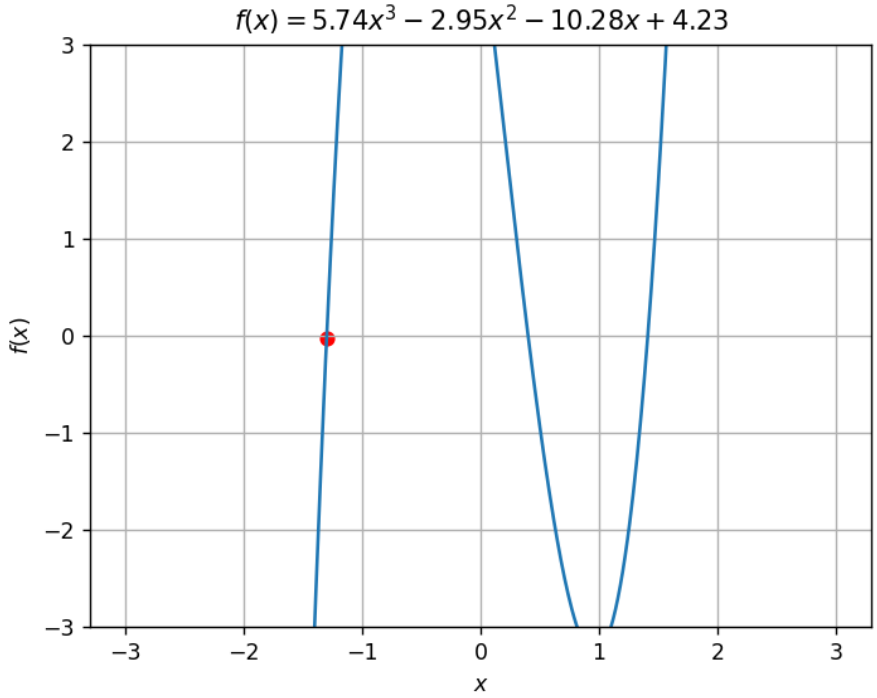
5 -> -1.3125 -> -1.28125 -> -1.296875 -> -0.337397460937499 -> 0.48557189941405987 -> 0.08025962829589695 -> 0.03125

6 -> -1.3125 -> -1.296875 -> -1.3046875 -> -0.337397460937499 -> 0.08025962829589695 -> -0.12701760292053343 -> 0.015625

7 -> -1.3046875 -> -1.296875 -> -1.30078125 -> -0.12701760292053343 -> 0.08025962829589695 -> -0.02299218535423364 -> 0.0078125

Ответ:

-1.30078125 -> -0.02299218535423364 -> 7



Выводы

В ходе выполнения лабораторной работы я изучила численные методы для решения нелинейных уравнений и систем нелинейных уравнений, нашла их корни. На основе полученных знаний были программно реализованы следующие методы: для решения нелинейных уравнений метод половинного деления, метод секущих, метод простой итерации, для решения систем нелинейных уравнений метод простой итерации.