

Университет ИТМО  
Мегафакультет компьютерных технологий и управления  
Факультет программной инженерии и компьютерной техники



ЛАБОРАТОРНАЯ РАБОТА №2  
ЧИСЛЕННОЕ РЕШЕНИЕ НЕЛИНЕЙНЫХ УРАВНЕНИЙ И  
СИСТЕМ  
Вариант №10

Группа: Р3211  
Студент: Орчиков Даниил Валерьевич  
Преподаватель: Малышева Татьяна Алексеевна

г. Санкт-Петербург  
2022

## Оглавление

Цель работы .....	2
Вычислительная реализация задачи: .....	2
1 часть. Решение нелинейного уравнения .....	2
Рабочие формулы .....	2
Решение .....	3
2 часть. Решение системы нелинейных уравнений .....	4
Рабочие формулы .....	4
Решение .....	4
Программная реализация задачи .....	5
Рабочие формулы .....	5
Листинг программы .....	5
Примеры и результаты работы программы .....	9
Пример 1 .....	9
Пример 2 .....	10
Пример 3 .....	11
Вывод .....	11

## Цель работы

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

## Вычислительная реализация задачи:

### 1 часть. Решение нелинейного уравнения

$$x^3 - 3.125x^2 - 3.5x + 2.458$$

Рабочие формулы

Метод Ньютона:

$$x_i = x_{i-1} - \frac{f(x_{i-1})}{f'(x_{i-1})}$$

Метод половинного деления:

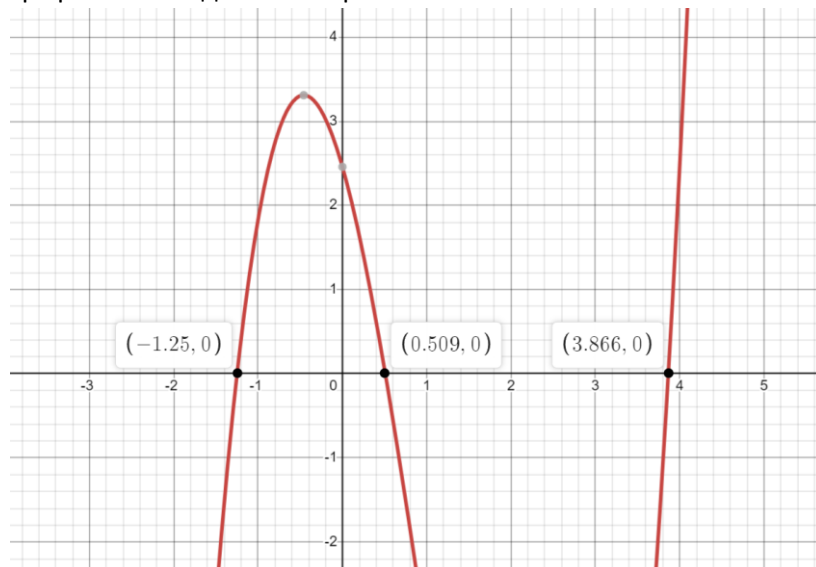
$$x_i = \frac{a_i + b_i}{2}$$

Метод простой итерации:

$$x_{i+1} = \varphi(x_i)$$

## Решение

### 1. Графическое отделение корней



### 2. Интервалы изоляции корней:

$(-2.5; -1)$ ,  $(0; 1)$ ,  $(3.5; 4)$

### 3. Методы, используемые для уточнения корней:

- Метод Ньютона
- Метод половинного деления
- Метод простой итерации

### 4. Первый корень (метод Ньютона):

Интервал изоляции -  $(-2.5; -1)$

№ итерации	$x_k$	$f(x_k)$	$f'(x_k)$	$x_{k+1}$	$ x_{k+1} - x_k $
1	-2.5	-23.948	-30.875	-1.724	0.776
2	-1.724	5.926	-16.197	-1.359	0.365
3	-1.359	1.062	-10.527	-1.258	0.101
4	-1.258	0.072	-9.105	-1.25	0.008

### 5. Второй корень (метод половинного деления):

Интервал изоляции -  $(0; 1)$

№ итерации	$a$	$b$	$x$	$f(a)$	$f(b)$	$f(x)$	$ a-b $
1	0.000	1.000	0.500	2.458	-3.167	0.052	1.000
2	0.500	1.000	0.750	0.052	-3.167	-1.503	0.500
3	0.500	0.750	0.625	0.052	-1.503	-0.706	0.250
4	0.500	0.625	0.563	0.052	-0.706	-0.322	0.125
5	0.500	0.563	0.531	0.052	-0.322	-0.133	0.063
6	0.500	0.531	0.516	0.052	-0.133	-0.040	0.031
7	0.500	0.516	0.508	0.052	-0.040	0.006	0.016
8	0.508	0.516	0.512	0.006	-0.040	-0.017	0.008

### 6. Третий корень (метод простой итерации):

Интервал изоляции -  $(3.5; 4)$

Применим 3 способ преобразования уравнения:

$$f'(x) = 3x^2 - 6.25x - 3.5$$

$$\max_{[3.5; 4]} |f'(x)| = 19.5$$

$$\lambda = -0.051$$

Преобразованное уравнение:

$$\varphi(x) = -0.051x^3 + 0.16x^2 + 1.179x - 0.125$$

$$\varphi'(x) = -0.153x^2 + 0.32x + 1.179$$

$$\varphi'(3.5) = 0.425$$

$$\varphi'(4) = 0.011$$

Условие сходимости выполняется

№ итерации	$x_k$	$x_{k+1}$	$f(x_{k+1})$	$ x_{k+1} - x_k $
1	3.550	3.795	-1.172	0.245
2	3.795	3.866	0.006	0.071
3	3.886	3.878	0.202	0.011
4	3.878	3.879	0.226	0.001

2 часть. Решение системы нелинейных уравнений

$$\begin{cases} \cos(y - 2) + x = 0 \\ \sin(x + 0.5) - y = 1 \end{cases}$$

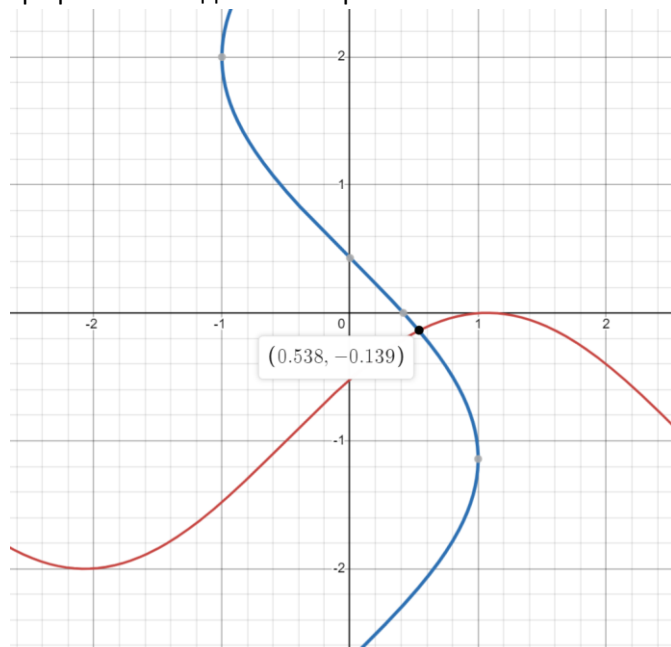
Рабочие формулы

Метод простой итерации

$$\begin{cases} x_1^{i+1} = \varphi_1(x_1^i, x_2^i, \dots, x_n^i) \\ x_2^{i+1} = \varphi_2(x_1^i, x_2^i, \dots, x_n^i) \\ \dots \dots \dots \\ x_n^{i+1} = \varphi_n(x_1^i, x_2^i, \dots, x_n^i) \end{cases}$$

Решение

1. Графическое отделение корней



2. Нахождение корня методом простых итераций:

Решение находится в области  $0 < x < 1$ ,  $-1 < y < 0$

$$\begin{cases} x = -\cos(y - 2) \\ y = \sin(x + 0.5) - 1 \end{cases}$$

Проверим условие сходимости. В области G имеем:

$$\begin{aligned} \frac{\partial \varphi_1}{\partial x} &= 0 & \frac{\partial \varphi_1}{\partial y} &= \sin(y - 2) \\ \frac{\partial \varphi_2}{\partial x} &= \cos(x + 0.5) & \frac{\partial \varphi_2}{\partial y} &= 0 \end{aligned}$$

$$\begin{aligned} \left| \frac{\partial \varphi_1}{\partial x} \right| + \left| \frac{\partial \varphi_1}{\partial y} \right| &= |\sin(y - 2)| \\ 0.141 &\leq |\sin(y - 2)| \leq 0.909 \\ \left| \frac{\partial \varphi_2}{\partial x} \right| + \left| \frac{\partial \varphi_2}{\partial y} \right| &= |\cos(x + 0.5)| \\ 0.071 &\leq |\cos(x + 0.5)| \leq 0.878 \end{aligned}$$

$$\max_{x \in G} |\varphi'(x)| \leq 0.909 < 1 \rightarrow \text{процесс сходящийся}$$

Начальные приближение

$$x^0 = 1, \quad y^0 = 0$$

1 шаг

$$x^1 = -\cos(-2) = 0.416 \quad |x^0 - x^1| = 0.584 > \varepsilon$$

$$y^1 = \sin(1.5) - 1 = -0.003 \quad |y^0 - y^1| = 0.003 < \varepsilon$$

2 шаг

$$x^2 = -\cos(-2.003) = 0.419 \quad |x^1 - x^2| = 0.003 < \varepsilon$$

$$y^2 = \sin(0.916) - 1 = -0.207 \quad |y^1 - y^2| = 0.204 > \varepsilon$$

3 шаг

$$x^3 = -\cos(-2.207) = 0.594 \quad |x^2 - x^3| = 0.175 > \varepsilon$$

$$y^3 = \sin(0.919) - 1 = -0.205 \quad |y^2 - y^3| = 0.002 < \varepsilon$$

4 шаг

$$x^4 = -\cos(-2.205) = 0.593 \quad |x^3 - x^4| = 0.001 < \varepsilon$$

$$y^4 = \sin(1.094) - 1 = -0.112 \quad |y^3 - y^4| = 0.093 > \varepsilon$$

5 шаг

$$x^5 = -\cos(-2.112) = 0.515 \quad |x^4 - x^5| = 0.077 > \varepsilon$$

$$y^5 = \sin(1.093) - 1 = -0.112 \quad |y^4 - y^5| = 0.000 < \varepsilon$$

6 шаг

$$x^6 = -\cos(-2.112) = 0.515 \quad |x^5 - x^6| = 0.000 < \varepsilon$$

$$y^6 = \sin(1.015) - 1 = -0.151 \quad |y^5 - y^6| = 0.039 > \varepsilon$$

7 шаг

$$x^7 = -\cos(-2.151) = 0.548 \quad |x^6 - x^7| = 0.033 > \varepsilon$$

$$y^7 = \sin(1.015) - 1 = -0.151 \quad |y^6 - y^7| = 0.000 < \varepsilon$$

8 шаг

$$x^8 = -\cos(-2.151) = 0.548 \quad |x^7 - x^8| = 0.000 < \varepsilon$$

$$y^8 = \sin(1.048) - 1 = -0.134 \quad |y^7 - y^8| = 0.017 > \varepsilon$$

9 шаг

$$x^9 = -\cos(-2.134) = 0.534 \quad |x^8 - x^9| = 0.014 > \varepsilon$$

$$y^9 = \sin(1.048) - 1 = -0.134 \quad |y^8 - y^9| = 0.000 < \varepsilon$$

10 шаг

$$x^9 = -\cos(-2.134) = 0.534 \quad |x^8 - x^9| = 0.000 < \varepsilon$$

$$y^9 = \sin(1.034) - 1 = -0.141 \quad |y^8 - y^9| = 0.007 < \varepsilon$$

## Программная реализация задачи

Рабочие формулы

$$\begin{pmatrix} \frac{\partial f_1(x,y)}{\partial x} & \frac{\partial f_1(x,y)}{\partial y} \\ \frac{\partial f_2(x,y)}{\partial x} & \frac{\partial f_2(x,y)}{\partial y} \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = - \begin{pmatrix} f_1(x,y) \\ f_2(x,y) \end{pmatrix}$$

$$x_{i+1} = x_i + \Delta x_i$$

$$y_{i+1} = y_i + \Delta y_i$$

Листинг программы

Представлен только код, непосредственно выполняющий вычисления

Весь код можно посмотреть [тут \(GitHub\)](#)

```

function chordMethod(a, b, accuracy) {
    function runA() {
        let v = 1
        document.getElementById("res").innerHTML = "<tr><th>№
    }
    let x0 = a
    while (true) {
        let fx0 = f["function"](x0)
        let fb = f["function"](b)
        x = x0 - (b - x0) / (fb - fx0) * fx0
        let fx = f["function"](x)
        document.getElementById("res").innerHTML +=
    `<tr><td>${v}</td><td>${x0}</td><td>${b}</td><td>${x}</td><td>${fx0}</td><td>${fb}</td><td>${fx}</td><td>${Math.
    abs(x0 - x)}</td></tr>`
        if (Math.abs(x0 - x) < accuracy) return
        v++
        x0 = x
        if (v > 100) {
            document.getElementById("res").innerHTML = ""
            document.getElementById("info").innerHTML = "<br>На данном интервале метод хорд не может получить решение"
            return
        }
    }
}

function runB() {
    let v = 1
    document.getElementById("res").innerHTML = "<tr><th>№
    }
    let x0 = b
    while (true) {
        let fx0 = f["function"](x0)
        let fa = f["function"](a)
        x = x0 - (a - x0) / (fa - fx0) * fx0
        let fx = f["function"](x)
        document.getElementById("res").innerHTML +=
    `<tr><td>${v}</td><td>${a}</td><td>${x0}</td><td>${x}</td><td>${fa}</td><td>${fx0}</td><td>${fx}</td><td>${Math.
    abs(x0 - x)}</td></tr>`
        if (Math.abs(x0 - x) < accuracy) return
        v++
        x0 = x
        if (v > 100) {
            document.getElementById("res").innerHTML = ""
            document.getElementById("info").innerHTML = "<br>На данном интервале метод хорд не может получить решение"
            return
        }
    }
}

if (root(a, b, f["derivative"]) || root(a, b, f["derivative2"])) {
    alert("На заданном интервале применение метода может дать некорректный результат")
}
let x
if (f["derivative"]((a + b) / 2) * f["derivative2"]((a + b) / 2) > 0) {
    runA()
}
if (f["derivative"]((a + b) / 2) * f["derivative2"]((a + b) / 2) < 0 || a > x || b < x) {
    runB()
}
if (a > x || b < x) {
    runA()
}
}

function newtonMethod(a, b, accuracy) {
    document.getElementById("res").innerHTML = "<tr><th>№
    }
    let x_k = a
    while (true) {
        let fx_k = f["function"](x_k)
        let fpx_k = f["derivative"](x_k)
        x_{k+1} = x_k - fx_k / fpx_k
        document.getElementById("res").innerHTML +=
    `<tr><td>${k}</td><td>${x_k}</td><td>${fx_k}</td><td>${fpx_k}</td><td>${x_{k+1}}</td><td>${Math.
    abs(x_k - x_{k+1})}</td></tr>`
        if (Math.abs(x_k - x_{k+1}) < accuracy) return
        k++
        x_k = x_{k+1}
        if (k > 100) {
            document.getElementById("res").innerHTML = ""
            document.getElementById("info").innerHTML = "<br>На данном интервале метод Ньютона не может получить решение"
            return
        }
    }
}

function run() {
    if (root(a, b, f["derivative"]) || root(a, b, f["derivative2"])) {
        alert("На заданном интервале применение метода может дать некорректный результат")
    }
    let x
    if (f["derivative"]((a + b) / 2) * f["derivative2"]((a + b) / 2) > 0) {
        runA()
    }
    if (f["derivative"]((a + b) / 2) * f["derivative2"]((a + b) / 2) < 0 || a > x || b < x) {
        runB()
    }
    if (a > x || b < x) {
        runA()
    }
}

```

```

let v = 1
while (true) {
    let fx0 = f["function"](x0)
    let dfx0 = f["derivative"](x0)
    x = x0 - fx0 / dfx0
    document.getElementById("res").innerHTML +=
`<tr><td>${v}</td><td>${x0}</td><td>${fx0}</td><td>${dfx0}</td><td>${x}</td><td>${Math.abs(x0 - x)}</td></tr>`
    if (Math.abs(x0 - x) < accuracy) return
    v++
    x0 = x
    if (v > 100) {
        document.getElementById("res").innerHTML = ""
        document.getElementById("info").innerHTML = "<br>На данном интервале метод Ньютона не может получить решение"
        return
    }
}

if (root(a, b, f["derivative2"])) {
    alert("На заданном интервале применение метода Ньютона может дать некорректный результат")
}
let x0, x
if (f["function"](a) * f["derivative2"](a)) {
    x0 = a
} else x0 = b
run()
if (a > x || b < x) {
    document.getElementById("res").innerHTML = "<tr><th>№
Шара</th><th>x_k</th><th>f(x_k)</th><th>f'(x_k)</th><th>x_{k+1}</th><th>|x_k - x_{k+1}|</th></tr>"
    if (f["function"](a) * f["derivative2"](a)) {
        x0 = b
    } else x0 = a
    run()
}
}

function simpleIterationMethod(a, b, accuracy) {
    document.getElementById("res").innerHTML = "<tr><th>№ Шара</th><th>x_i</th><th>x_{i+1}</th><th>phi(x_{i +
1})</th><th>f(x_{k+1})</th><th>|x_k - x_{k+1}|</th></tr>"

    function run() {
        let v = 1
        let x0 = a
        while (true) {
            x = x0 + lambda * (f["function"](x0))
            document.getElementById("res").innerHTML += `<tr><td>${v}</td><td>${x0}</td><td>${x}</td><td>${x0 + lambda *
(f["function"](x))}</td><td>${f["function"](x)}</td><td>${Math.abs(x0 - x)}</td></tr>`
            if (Math.abs(x0 - x) < accuracy) return
            v++
            x0 = x
            if (v > 100) {
                document.getElementById("info").innerHTML += "<br>Не удалось добиться нужной точности за вменяемое количество
итераций. Метод расходится"
                return
            }
        }
    }
}

if (root(a, b, f["derivative"])) {
    alert("На заданном интервале применение метода простых итераций может дать некорректный результат")
}
let mx = Number.MIN_VALUE
for (let i = a; i <= b; i += Math.abs(a - b) / 100)
    if (mx < Math.abs(f["derivative"](i)))
        mx = Math.abs(f["derivative"](i))
let lambda = 1 / mx
lambda *= f["derivative"](mx) > 0 ? -1 : 1
document.getElementById("info").innerHTML = "Достаточное условие" + (lambda * f["derivative"](mx) + 1 < 1 ? " " : " не ") +
"выполняется"

```

```

    console.log(lambda * f["derivative"](mx) + 1)
    let x
    run()
    if (a > x || b < x) {
        document.getElementById("res").innerHTML = "<tr><th>№ Илара</th><th>x_i</th><th>x_{i+1}</th><th>phi(x_{i+1})</th><th>f(x_{k+1})</th><th>|x_k - x_{k+1}|</th></tr>"
        lambda *= -1
        console.log(lambda * f["derivative"](mx) + 1)
        document.getElementById("info").innerHTML = "Достаточное условие" + (lambda * f["derivative"](mx) + 1 < 1 ? " " : " не ") +
"выполняется"
        run()
    }
}

function root(a, b, func) {
    const step = Math.abs(b - a) / 100;
    let previousSign = Math.sign(func(a));
    let currentSign;

    for (let x = a; x <= b; x += step) {
        currentSign = Math.sign(func(x));
        if (currentSign !== previousSign || currentSign === 0) {
            return x;
        }
        previousSign = currentSign;
    }

    return 0;
}

function hasMoreThat1Root(a, b) {
    a = root(a, b, f["function"])
    return root(a, b, f["function"])
}

function systemNewtonMethod(x0, y0, accuracy) {
    document.getElementById("info").innerHTML = ""
    document.getElementById("check").innerHTML = ""
    document.getElementById("res").innerHTML = "<tr><th>№ Илара</th><th>x_i</th><th>y_i</th><th>|x_k - x_{k+1}|</th><th>|y_k - y_{k+1}|</th></tr>"

    let v = 1
    let x, y
    while (true) {
        let a11 = f["derivative1X"](x0, y0)
        let a12 = f["derivative1Y"](x0, y0)
        let a21 = f["derivative2X"](x0, y0)
        let a22 = f["derivative2Y"](x0, y0)
        let b1 = -f["function1"](x0, y0)
        let b2 = -f["function2"](x0, y0)
        let d = a11 * a22 - a12 * a21
        if (Math.abs(d) < Number.EPSILON) {
            document.getElementById("res").innerHTML = ""
            document.getElementById("info").innerHTML = "Определитель матрицы равен нулю"
            return
        }
        if (d === 0) break
        let d1 = b1 * a22 - b2 * a12
        let d2 = a11 * b2 - a21 * b1
        let dx = d1 / d
        let dy = d2 / d
        document.getElementById("res").innerHTML +=
`<tr><td>${v}</td><td>${x0}</td><td>${y0}</td><td>${Math.abs(dx)}</td><td>${Math.abs(dy)}</td></tr>`
        x = x0 + dx
        y = y0 + dy
        if (Math.abs(dx) <= accuracy && Math.abs(dy) <= accuracy) {
            break
        }
        x0 = x
    }
}

```



```
y0 = y
v++
if (v > 300) {
    document.getElementById("info").innerHTML += "<br>Не удалось добиться нужной точности за вменяемое количество
итераций."
    return
}
}
document.getElementById("check").innerHTML = `f1({x}, {y}) = ${f"function1"](x, y)} ≈ ${Math.round(f"function1"](x, y))}`
document.getElementById("check").innerHTML += `<br>f2({x}, {y}) = ${f"function2"](x, y)} ≈ ${Math.round(f"function2"](x, y))}`
}
```

Примеры и результаты работы программы

Пример 1

x^3 - 3.125x^2 - 3.5x + 2.458

▼

Метод Ньютона

▼

Точность: 

0,001

Введите интервал

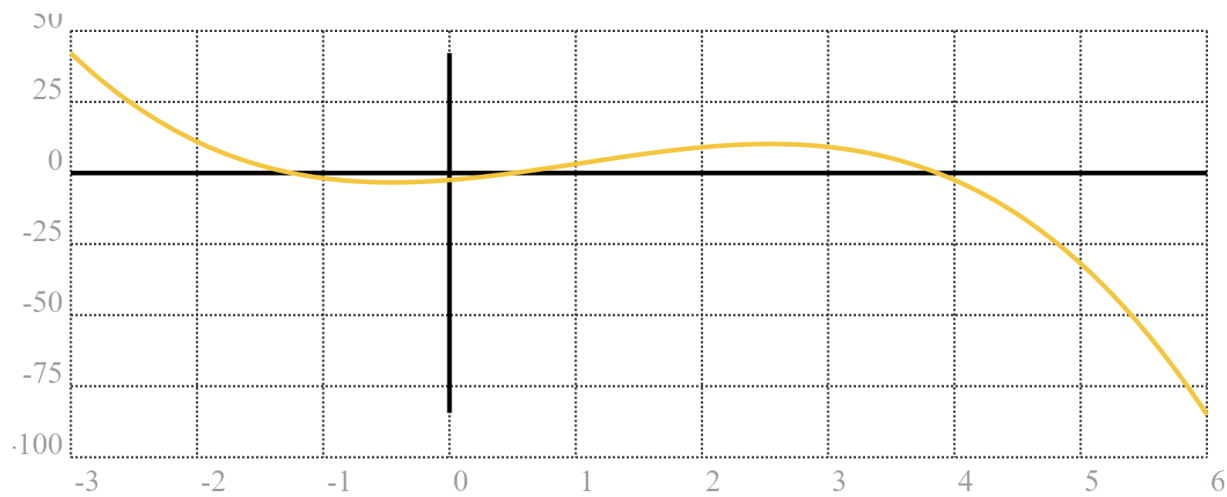
(-2

:

-1

)

Рассчитать



№ Шага	x_k	f(x_k)	f'(x_k)	x_{k+1}	x_k - x_{k+1}
1	-2	11.042	-21	-1.474190476190476	0.525809523809524
2	-1.474190476190476	2.377466822201919	-12.233403156462582	-1.2798482484559175	0.19434222773455856
3	-1.2798482484559175	0.2777233892523787	-9.413086170076525	-1.250344280869257	0.029503967586660407
4	-1.250344280869257	0.0060368427531747315	-9.004734217540236	-1.24967387321398	0.0006704076552770388

Пример 2

cos(x) \* x + 1

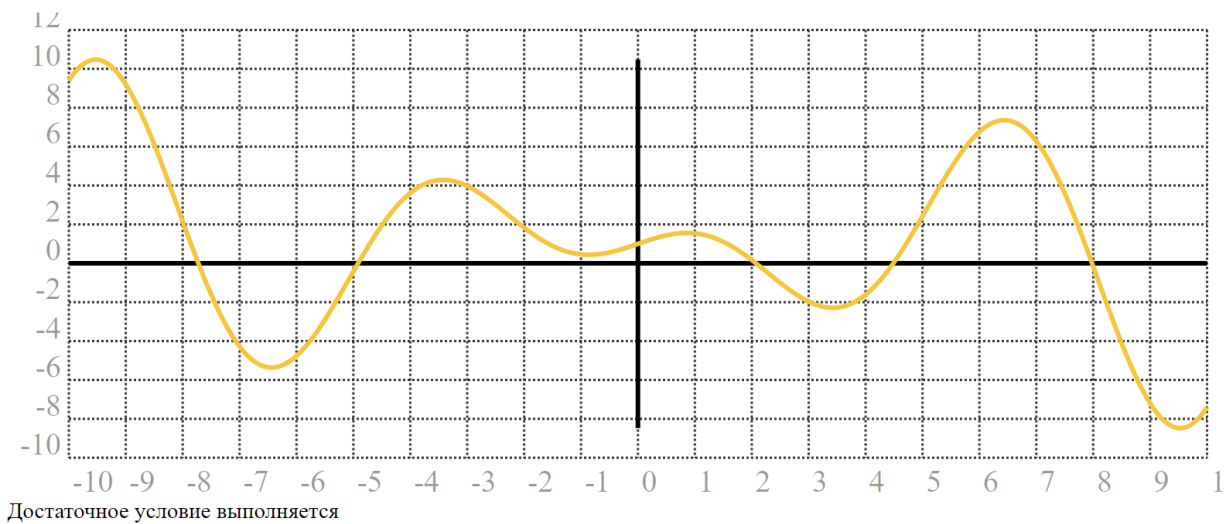
Метод простой итерации

Точность: 0,001

Введите интервал

(-6; 2)

Рассчитать



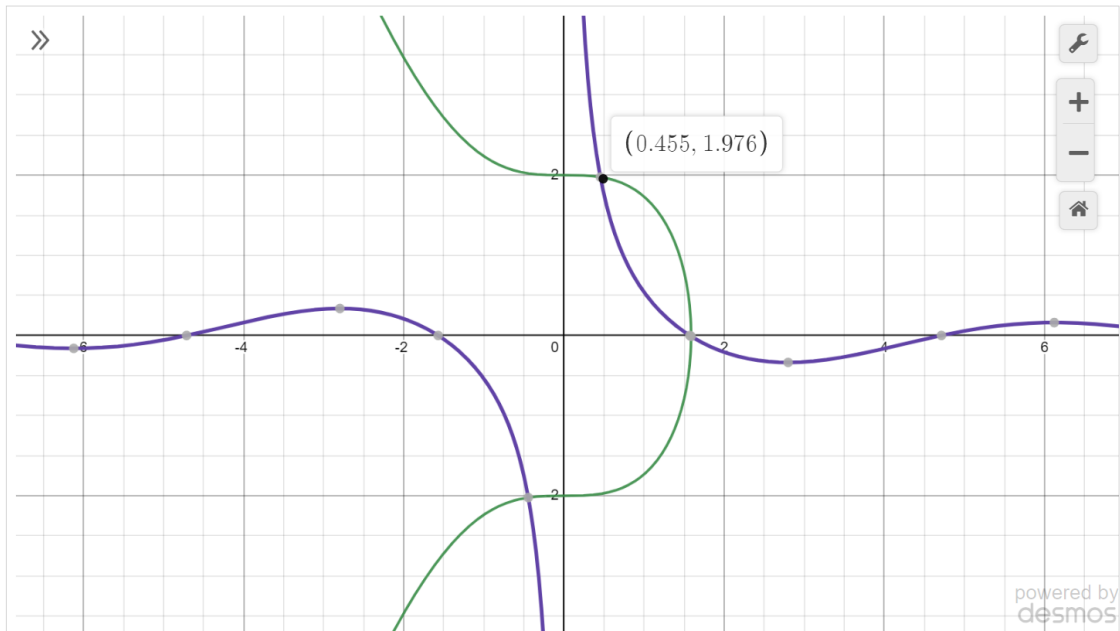
№ Шага	x_i	x_{i+1}	phi(x_{i+1})	f(x_{k+1})	x_k - x_{k+1}
1	-6	-5.065905513615052	-5.852105029037262	-0.7538115033126536	0.9340944863849483
2	-5.065905513615052	-4.918010542652314	-5.065093636504922	-0.004138087325813	0.14789497096273774
3	-4.918010542652314	-4.917198665542184	-4.917998000203651	-0.00006392808369382053	0.0008118771101299771

Пример 3

- ☐ {x ^ 2 + cos(y) = 1; y = x ^ 4}
- ☐ {y = cos(x) - x; (x + 1) ^ 2 + y ^ 10 = 3)}
- ☒ {x ^ 3 + y ^ 2 = 4; xy = cos(x)}
- ☐ {cos(x \* cos(1) + y \* sin(1)) = y \* cos(1); sinh(3x) = cosh(y \* sinh(x)) - sin(8x)}

Точность:

Введите начальные приближения: {  ;  }



№ Шага	x <sub>i</sub>	y <sub>i</sub>	x <sub>k</sub> - x <sub>{k+1}</sub>	y <sub>k</sub> - y <sub>{k+1}</sub>
1	0.5	2	0.044764024437333474	0.022856745417999973
2	0.45523597556266654	1.977143254582	0.0006453857993140507	0.0007680803649039132
3	0.4545905897633525	1.976375174217096	2.3442900696558874e-7	2.5632558997838516e-7

f1(0.4545903553343455, 1.976374917891506) = 1.4033219031261979e-13 ≈ 0  
f2(0.4545903553343455, 1.976374917891506) = 8.471001677889944e-14 ≈ 0

Вывод

Во время выполнения данной лабораторной работы я познакомился с различными итерационными методами решения нелинейных уравнений и систем нелинейных уравнений и запрограммировал некоторые из них.