МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Тихоокеанский государственный университет»

Кафедра «Программное обеспечение вычислительной техники и автоматизированных систем»

Решение системы нелинейных уравнений

Лабораторная работа №3

по дисциплине «Вычислительная математика»

Выполнил студент	Чекулаев В. Ю.
Факультет, группа	ФКФН, ПО(аб)-81
Проверил	Резак Е.В.

Хабаровск – 2020г.

Задание: Используя метод Ньютона, решить систему нелинейных уравнений с точностью 0.0001.

Вариант 7. $\begin{matrix} \sin(x-1)/2+y=1.3 \\ x-\sin(y+1)=0.8 \end{matrix}$

Метод Ньютона для решения систем нелинейных уравнений

Формула для нахождения решения системы нелинейных уравнений:

$$x^{k+1}=x^k-W^{-1}(x^k)*F(x^k), k=1,2,\ldots \quad (1)$$

Где $W(x)=\begin{pmatrix} \frac{\partial f_1(x)}{\partial x_1} & \cdots & \frac{\partial f_1(x)}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_n(x)}{\partial x_1} & \cdots & \frac{\partial f_n(x)}{\partial x_n} \end{pmatrix}$ - матрица Якоби.

Так как процесс вычисления обратной матрицы является трудоемким, преобразуем (1) следующим образом:

$$W(x^k)*\varepsilon=-F(x^k), k=1,2,\ldots$$

В результате получена система линейных алгебраических уравнений относительно поправки ε. После ее определения вычисляется следующее приближение $x^{k+1}=x^k+\varepsilon$.

Алгоритм метода Ньютона

1. Задать начальное приближение и точность. Положить *k = 0*.

2. Решить систему линейных алгебраических уравнений относительно поправки ε:

$$W(x^k)*\varepsilon=-F(x^k), k=1,2,\ldots$$

3. Вычислить следующее приближение: $x^{k+1}=x^k+\varepsilon$.

4. Если норма разности векторов x^{k+1} и x^k , равная $\max_i|x_i^{k+1}-x_i^k|$, меньше либо равна заданной точности, процесс закончить и положить ответ равным x^{k+1} . Иначе положить *k = k+1* и перейти к пункту 2.

Теорема о сходимости метода Ньютона

Пусть f($x_1, x_2, \ldots, x_n$) определены, непрерывны и имеют непрерывные первые и вторые производные в области Ω.

Пусть:

1) для начальной точки x^0 выполняется условие $\|x^0 - x\| < \delta$

2) существует $W^{-1}(x^0)$, причем $\|W^{-1}(x^0)\| \leqslant A_0$

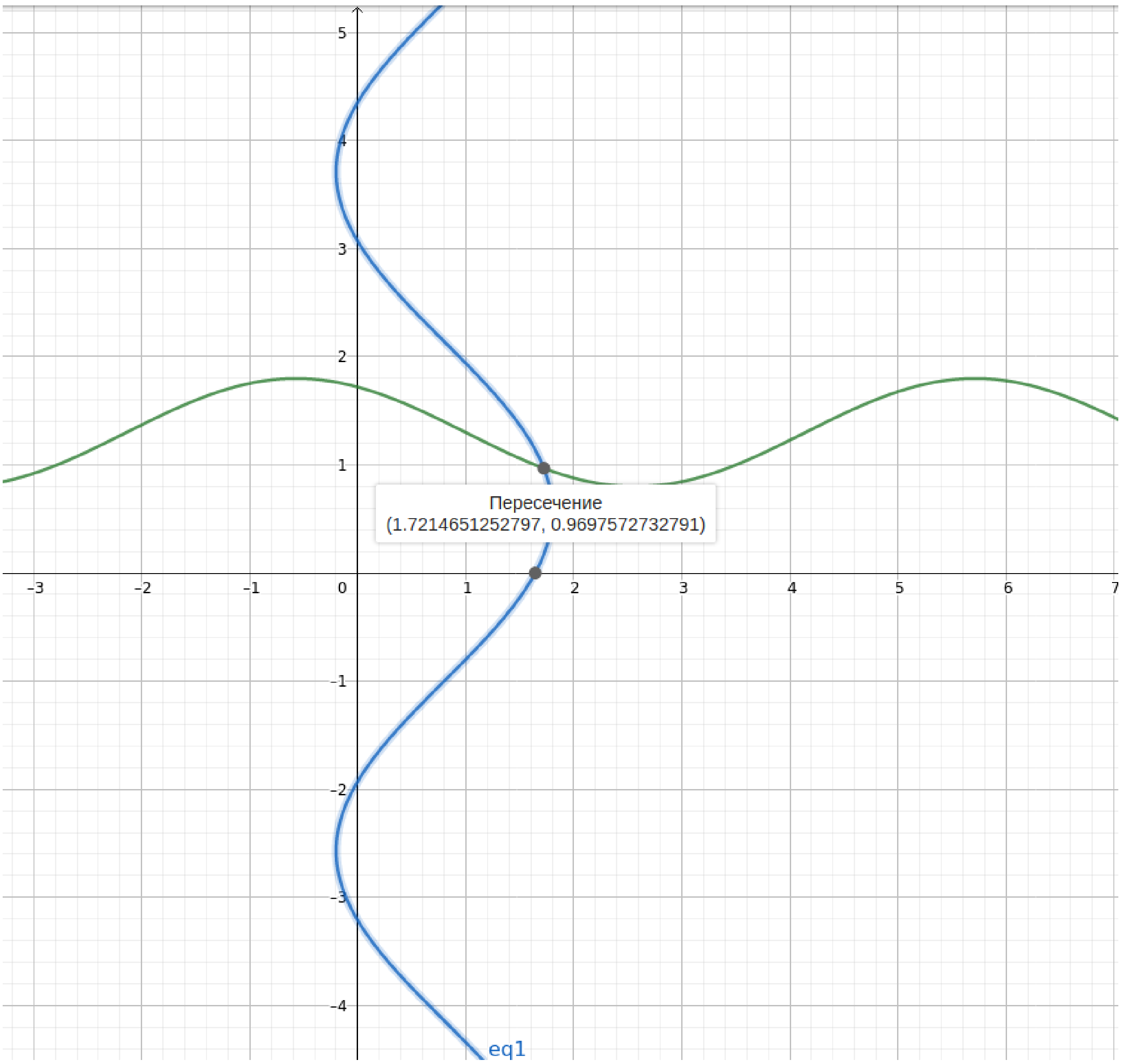
3) $\|W^{-1}(x^0) F(x^0)\| \leqslant B_0 \leqslant \delta/2$

4) $\sum_{k=1}^{n} \left| \frac{\partial^2 f_i(x)}{\partial x_j \partial x_k} \right| \leqslant C; i, j = 0..n$

5) A_0, B_0, C удовлетворяют условию $\mu_0 = 2nA_0 B_0 C \leqslant 1$

Тогда метод Ньютона сходится к точному решению $x = \lim_{k \to \infty} x^k$

Ручной расчет



МЕТОД НЬЮТОНА

k	0	1	2	3	4
x_1^k	1	1,738012	1,7222917	1,721467758	1,721465109
x_2^k	1	0,9309939	0,9694056	0,969755993	0,9697572786
$\max\limits_i \lvert x_i^{k+1} - x_i^k \rvert$		0,738012	0,0384117	0,000823942	2,649E-06
ε (по методу Зейделя)	0,738012	-0,0157203	-0,000823942	-2,649E-06	
	-0,0690061	0,0384117	0,000350393	1,2856E-06	
$-F(x^k)$	-0,3	-0,032596847670313	-4,14739269520847E-05	-2,91905937865877E-07	
	-0,709297426825682	0,002184794537402	0,000690022370191	2,13743787125242E-06	
$W(x^k)$	0,5 1	0,3699037915 1	0,3751463222 1	0,3754185513 1	
	1 0,4161468365	1 0,3524591345	1 0,3881370219	1 0,3884599209	

Листинг

```
#include<iostream>
#include<cmath>
using namespace std;

float f1(float x, float y){
    return sin(x-1)/2+y-1.3;
}

float f2(float x, float y){
    return x-sin(y+1)-0.8;
}

float* minusF(float x, float y){
    float* buf = new float[2];
    buf[0] = -1*f2(x, y);
    buf[1] = -1*f1(x, y);
    return buf;
}

float** W(float x, float y){
    float** buf = new float*[2];
    for(int i = 0; i < 2; i++){
        buf[i] = new float[2];
    }

    buf[0][0] = 1;
    buf[0][1] = -cos(y+1);
    buf[1][0] = cos(x-1)/2;
    buf[1][1] = 1;

    return buf;
}

float** preobrC(float* A[]){
    float** C = new float*[2];
    for(int i = 0; i < 2; i++){
        C[i] = new float[2];
    }
    for(int i = 0; i < 2; i++){
        for(int j = 0; j < 2; j++){
            if(i == j){
```

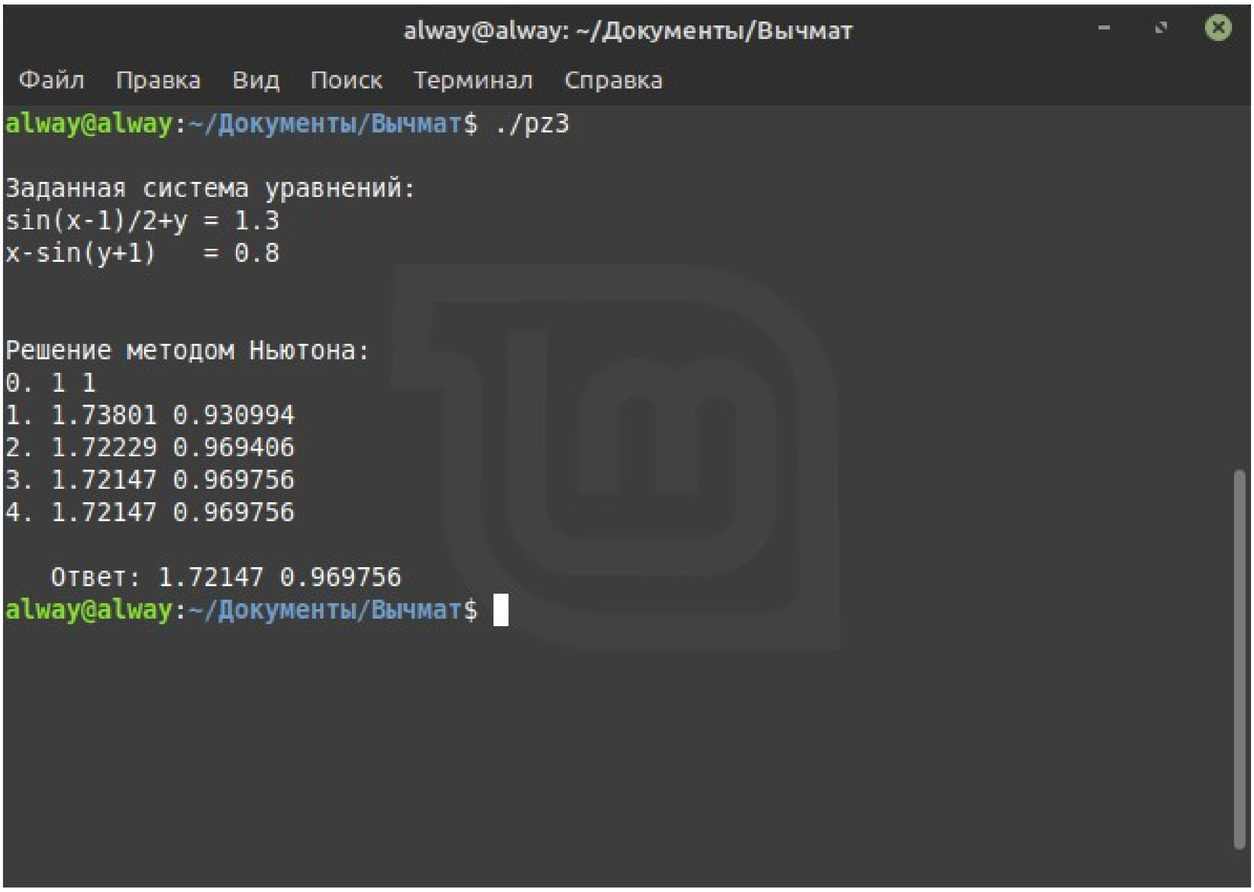
```
                C[i][j] = 0;
            } else{
                C[i][j] = -A[i][j]/A[i][i];
            }
        }
    }
    return C;
}

float* preobrD(float* A[], float* b){
    float* d = new float[2];
    for(int i = 0; i < 2; i++){
        d[i] = b[i]/A[i][i];
    }
    return d;
}

float* multM(float* M[], float* V, int N){ // Умножение матрицы на вектор
    float* buf = new float[N];
    for(int i = 0; i < N; i++){
        buf[i] = 0;
    }

    for(int j = 0; j < N; j++){
        for(int i = 0; i < N; i++){
            buf[i] += M[i][j] * V[j];
        }
    }

    return buf;
}

float* sumM(float* V1, float* V2, int N){
    float* buf = new float[N];
    for(int i = 0 ; i < N; i++){
        buf[i] = V1[i] + V2[i];
    }
    return buf;
}

float* sumM(float* V1, float* V2, float* V3, int N){
    float* buf = new float[N];
```

```
    for(int i = 0 ; i < N; i++){
        buf[i] = V1[i] + V2[i] + V3[i];
    }
    return buf;
}

float norma(float* V1, float* V2, int N){ // Норма двух векторов
    float res = 0;
    for(int i = 0; i < N; i++){
        res += abs(V1[i] - V2[i]);
    }
    return res;
}

float* zeydel(float* C[], float* d, int N){
    float** L = new float*[N];
    float** U = new float*[N];
    for(int i = 0; i < N; i++){
        L[i] = new float[N];
        U[i] = new float[N];
    }

    for(int i = 0; i < N; i++){
        for(int j = 0; j < N; j++){
            if(j >= i){
                U[i][j] = C[i][j];
                L[i][j] = 0;
            } else{
                L[i][j] = C[i][j];
                U[i][j] = 0;
            }
        }
    }

    float* x_k = new float[N];
    for(int i = 0; i < N; i++){
        x_k[i] = d[i];
    }
    float* x_k1 = new float[N];
    float E = 0.00001;

    do{
```

```
        x_k = x_k1;
        x_k1 = sumM(multM(C, x_k, N), d, N);
        x_k1 = sumM(multM(L, x_k1, N), multM(U, x_k, N), d, N);
    }while(norma(x_k, x_k1, N) > E);

    return x_k;
}

float norma2(float* V1, float* V2){
    float x1 = abs(V1[0]-V2[0]);
    float x2 = abs(V1[1]-V2[1]);
    if(x1>x2) {return x1;} else {return x2;};
}

void newton(float x0, float y0){
    float* x_k = new float[2];
    float* x_k1 = new float[2];
    float** buf1;
    float* buf2;
    x_k1[0] = x0; x_k1[1] = y0;
    float* deltax;
    float E = 0.0001;
    int k = 0;

    cout << k << ". " << x_k1[0] << " " << x_k1[1] << "\n"; k++;

    do{
        x_k = x_k1;
        buf1 = preobrC(W(x_k[0], x_k[1]));
        buf2 = preobrD(W(x_k[0], x_k[1]), minusF(x_k[0], x_k[1]));
        deltax = zeydel(buf1, buf2, 2);
        x_k1 = sumM(x_k, deltax, 2);
        cout << k << ". " << x_k1[0] << " " << x_k1[1] << "\n"; k++;
    } while(norma2(x_k1, x_k) > E);

    cout << "\n   Ответ: ";
    for(int i = 0; i < 2; i++){
        cout << x_k1[i] << " ";
    } cout << "\n";
}

int main(){
```

```
    system("clear");
    cout << "Заданная система уравнений: \n";
    cout << "sin(x-1)/2+y = 1.3\n";
    cout << "x-sin(y+1)   = 0.8\n\n\n";
    cout << "Решение методом Ньютона:\n";
    newton(1,1);

    return 0;
}
```

Вывод программы



Вывод

В ходе данной лабораторной работы был изучен метод Ньютона для решения систем нелинейных уравнений. На основе теоретических данных была написана программа, результаты и количество итераций которой совпали с ручным расчетом.