Московский авиационный институт
(национальный исследовательский университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Лабораторная работа №2
Задание №2 часть I по курсу «Численные методы»
Вариант №12

Студент: С. М. Власова
Преподаватель: И. Э. Иванов
Группа: М8О-306Б
Дата:
Оценка:
Подпись:

Москва, 2020

Задание №2.2

Реализовать метод Ньютона решения систем нелинейных уравнений в виде программного кода, задавая в качестве входных данных точность вычислений. С использованием разработанного программного обеспечения решить систему нелинейных уравнений (при наличии нескольких решений найти то из них, в котором значения неизвестных являются положительными); начальное приближение определить графически. Проанализировать зависимость погрешности вычислений от количества итераций.

Вариант: 12

$$\begin{cases} x_1 - \cos(x_2) = 3 \\ x_2 - \sin(x_1) = 3 \end{cases}$$

1 Описание метода решения

Систему нелинейных уравнений с n неизвестными можно записать в виде

$$\begin{cases} f_1(x_1, x_2, \ldots, x_n) = 0 \\ f_2(x_1, x_2, \ldots, x_n) = 0 \\ \quad \ldots\ldots\ldots\ldots\ldots \\ f_n(x_1, x_2, \ldots, x_n) = 0 \end{cases}$$

или, более коротко, в векторной форме

$$\boldsymbol{f}(\boldsymbol{x}) = \mathbf{0},$$

где $\boldsymbol{x}$ — вектор неизвестных величин, $\boldsymbol{f}$ — вектор-функция

$$\boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \\ \ldots \\ x_n \end{pmatrix}, \quad \boldsymbol{f} = \begin{pmatrix} f_1(\boldsymbol{x}) \\ f_2(\boldsymbol{x}) \\ \ldots \\ f_n(\boldsymbol{x}) \end{pmatrix}, \quad \mathbf{0} = \begin{pmatrix} 0 \\ 0 \\ \ldots \\ 0 \end{pmatrix}$$

В редких случаях для решения такой системы удается применить метод последовательного исключения неизвестных и свести решение исходной задачи к решению одного нелинейного уравнения с одним неизвестным. Значения других неизвестных величин находятся соответствующей подстановкой в конкретные выражения. Однако в подавляющем большинстве случаев для решения систем нелинейных уравнений используются итерационные методы.

В дальнейшем предполагается, что ищется изолированное решение нелинейной системы.

Как и в случае одного нелинейного уравнения, локализация решения может осуществляться на основе специфической информации по конкретной решаемой задаче (например, по физическим соображениям), и — с помощью методов математического анализа. При решении системы двух уравнений, достаточно часто удобным является графический способ, когда месторасположение корней определяется как точки пересечения кривых $f_1(x_1, x_2) = 0$, $f_2(x_1, x_2) = 0$ на плоскости (x_1, x_2).

Метод Ньютона. Если определено начальное приближение $\boldsymbol{x}^{(0)} = (x_1^{(0)}, x_2^{(0)}, \ldots, x_n^{(0)})^T$, итерационный процесс нахождения решения системы методом Ньютона можно пред-

ставить в виде

$$\begin{cases} x_1^{(k+1)} = x_1^{(k)} + \Delta x_1^{(k)} \\ x_2^{(k+1)} = x_2^{(k)} + \Delta x_2^{(k)} \\ \quad \ldots\ldots\ldots\ldots \\ x_n^{(k+1)} = x_n^{(k)} + \Delta x_n^{(k)} \end{cases}$$

$k = (0, 1, 2, \ldots)$, где значения приращений $\Delta x_1^{(k)}, \Delta x_2^{(k)}, \ldots, \Delta x_n^{(k)}$ определяются из решения системы линейных алгебраических уравнений, все коэффициенты которой выражаются через известное предыдущее приближение $\boldsymbol{x}^{(k)} = (x_1^{(k)}, x_2^{(k)}, \ldots, x_n^{(k)})^T$

$$\begin{cases} f_1(\boldsymbol{x}^{(k)}) + \frac{\partial f_1(\boldsymbol{x}^{(k)})}{\partial x_1}\Delta x_1^{(k)} + \frac{\partial f_1(\boldsymbol{x}^{(k)})}{\partial x_2}\Delta x_2^{(k)} + \ldots + \frac{\partial f_1(\boldsymbol{x}^{(k)})}{\partial x_n}\Delta x_n^{(k)} = 0 \\ f_2(\boldsymbol{x}^{(k)}) + \frac{\partial f_2(\boldsymbol{x}^{(k)})}{\partial x_1}\Delta x_1^{(k)} + \frac{\partial f_2(\boldsymbol{x}^{(k)})}{\partial x_2}\Delta x_2^{(k)} + \ldots + \frac{\partial f_2(\boldsymbol{x}^{(k)})}{\partial x_n}\Delta x_n^{(k)} = 0 \\ \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ f_n(\boldsymbol{x}^{(k)}) + \frac{\partial f_n(\boldsymbol{x}^{(k)})}{\partial x_1}\Delta x_1^{(k)} + \frac{\partial f_n(\boldsymbol{x}^{(k)})}{\partial x_2}\Delta x_2^{(k)} + \ldots + \frac{\partial f_n(\boldsymbol{x}^{(k)})}{\partial x_n}\Delta x_n^{(k)} = 0 \end{cases}$$

В векторно-матричной форме расчетные формулы имеют вид

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \boldsymbol{\Delta x}^{(k+1)}, \qquad k = 0, 1, 2, \ldots$$

где вектор приращений $\boldsymbol{\Delta x}^{(k)} = \begin{pmatrix} \Delta x_1{}^{(k)} \\ \Delta x_2{}^{(k)} \\ \ldots \\ \Delta x_n{}^{(k)} \end{pmatrix}$ находится из решения уравнения

$$\boldsymbol{f(x^{(k)}) + J(x^{(k)})\Delta x^{(k)} = 0}$$

Здесь $\boldsymbol{J(x)} = \begin{pmatrix} \frac{\partial f_1(\boldsymbol{x})}{\partial x_1} & \frac{\partial f_1(\boldsymbol{x})}{\partial x_2} & \ldots & \frac{\partial f_1(\boldsymbol{x})}{\partial x_n} \\ \frac{\partial f_2(\boldsymbol{x})}{\partial x_1} & \frac{\partial f_2(\boldsymbol{x})}{\partial x_2} & \ldots & \frac{\partial f_2(\boldsymbol{x})}{\partial x_n} \\ & \ldots\ldots & & \\ \frac{\partial f_n(\boldsymbol{x})}{\partial x_1} & \frac{\partial f_n(\boldsymbol{x})}{\partial x_2} & \ldots & \frac{\partial f_n(\boldsymbol{x})}{\partial x_n} \end{pmatrix}$ — матрица Якоби первых производных вектор-функции $\boldsymbol{f(x)}$.

Итерационный процесс нахождения решения можно записать в виде

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} - \boldsymbol{J^{-1}(x^{(k)})f(x^{(k)})}, \qquad k = 0, 1, 2, \ldots$$

где $\boldsymbol{J^{-1}(x)}$ — матрица, обратная матрице Якоби.

При реализации алгоритма метода Ньютона в большинстве случаев предпочтительным является не вычисление обратной матрицы $\boldsymbol{J^{-1}(x^{(k)})}$, а нахождение из системы значений приращений $\Delta x_1^{(k)}, \Delta x_2^{(k)}, \ldots, \Delta x_n^{(k)}$ и вычисление нового приближения. Для решения таких линейных систем можно привлекать самые разные методы, как прямые, так и итерационные, с учетом размерности n решаемой задачи и специфики матриц Якоби $\boldsymbol{J(x)}$ (например, симметрии, разреженности и т.п.).

Использование метода Ньютона предполагает дифференцируемость функций $f_1(\boldsymbol{x})$, $f_2(\boldsymbol{x}), \ldots, f_n(\boldsymbol{x})$ и невырожденность матрицы Якоби ($|\boldsymbol{J}(\boldsymbol{x}^{(k)})| \neq 0$). В случае, если начальное приближение выбрано в достаточно малой окрестности искомого корня, итерации сходятся к точному решению, причем сходимость квадратичная.

В практических вычислениях в качестве условия окончания итераций обычно используется критерий

$$||\boldsymbol{x}^{(k+1)} - \boldsymbol{x}^{(k)}|| \leq \varepsilon,$$

где ε — заданная точность.

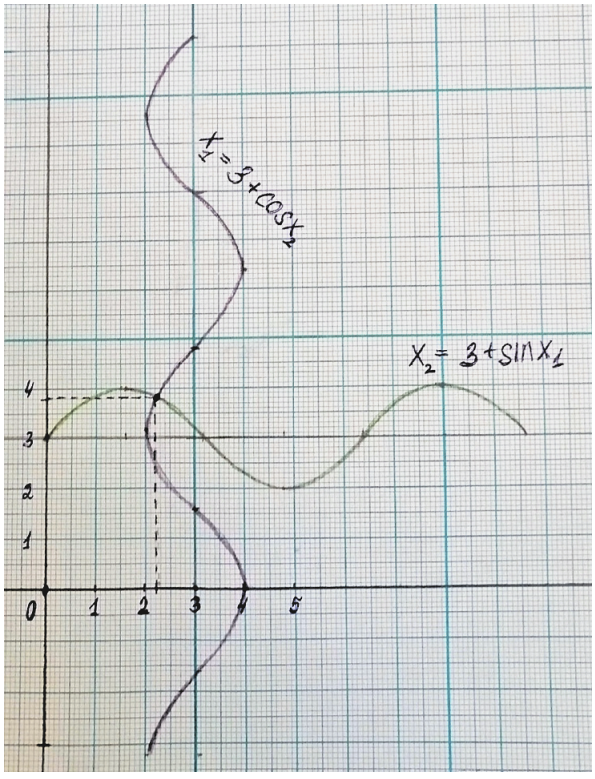
2 Моя задача

Методом Ньютона найти положительное решение системы нелинейных уравнений

$$\begin{cases} x_1 - \cos(x_2) = 3 \\ x_2 - \sin(x_1) = 3 \end{cases}$$

с заданной точностью.

Решение:
Этап I. Локализация: для выбора начального приближения применим графический способ. Построив на плоскости (x_1, x_2) в интересующей нас области кривые $f_1(x_1, x_2) = 0$ и $f_2(x_1, x_2) = 0$, определяем, что положительное решение системы находится в квадрате $2 < x_1 < 3$ и $3 < x_2 < 4$.



В качестве начального приближения возьмем середины выбранных интервалов.

$$x_1^{(0)} = \frac{(2+3)}{2} = 2.5, \quad x_2^{(0)} = \frac{(3+4)}{2} = 3.5$$

Этап II. Уточнение: для уточнения корней будем использовать метод Ньютона.

Для системы двух уравнений расчетные формулы удобно записать в виде, разрешенном относительно $x_1^{(k+1)}, \quad x_2^{(k+1)}$

$$\begin{cases} x_1^{(k+1)} = x_1^{(k)} - \frac{det\boldsymbol{A}_1^{(k)}}{det\boldsymbol{J}^{(k)}} \\ \\ x_2^{(k+1)} = x_2^{(k)} - \frac{det\boldsymbol{A}_2^{(k)}}{det\boldsymbol{J}^{(k)}} \end{cases}$$

$k = 0, 1, 2, \ldots,$ где

$$\boldsymbol{J^{(k)}} = \begin{pmatrix} \frac{\partial f_1(x_1^{(k)}, x_2^{(k)})}{\partial x_1} & \frac{\partial f_1(x_1^{(k)}, x_2^{(k)})}{\partial x_2} \\ \frac{\partial f_2(x_1^{(k)}, x_2^{(k)})}{\partial x_1} & \frac{\partial f_2(x_1^{(k)}, x_2^{(k)})}{\partial x_2} \end{pmatrix},$$

$$\boldsymbol{A_1^{(k)}} = \begin{pmatrix} f_1(x_1^{(k)}, x_2^{(k)}) & \frac{\partial f_1(x_1^{(k)}, x_2^{(k)})}{\partial x_2} \\ f_2(x_1^{(k)}, x_2^{(k)}) & \frac{\partial f_2(x_1^{(k)}, x_2^{(k)})}{\partial x_2} \end{pmatrix}, \quad \boldsymbol{A_2^{(k)}} = \begin{pmatrix} \frac{\partial f_1(x_1^{(k)}, x_2^{(k)})}{\partial x_1} & f_1(x_1^{(k)}, x_2^{(k)}) \\ \frac{\partial f_2(x_1^{(k)}, x_2^{(k)})}{\partial x_1} & f_2(x_1^{(k)}, x_2^{(k)}) \end{pmatrix}$$

В моей задаче:

$$f_1(x_1^{(k)}, x_2^{(k)}) = x_1^{(k)} - \cos(x_2^{(k)}) - 3$$
$$f_2(x_1^{(k)}, x_2^{(k)}) = x_2^{(k)} - \sin(x_1^{(k)}) - 3$$

$$\frac{\partial f_1(x_1^{(k)}, x_2^{(k)})}{\partial x_1} = 1, \qquad \frac{\partial f_1(x_1^{(k)}, x_2^{(k)})}{\partial x_2} = \sin(x_2^{(k)})$$

$$\frac{\partial f_2(x_1^{(k)}, x_2^{(k)})}{\partial x_1} = -\cos(x_1^{(k)}), \qquad \frac{\partial f_2(x_1^{(k)}, x_2^{(k)})}{\partial x_2} = 1.$$

Итерации продолжаются до выполнения условия

$$||x^{(k+1)}x^{(k)}|| = \max_i |x_i^{(k+1)}x_i^{(k)}|.$$

3 Протокол

Входные данные через командную строку я задаю значение точности ε.
Выходные данные я вывожу на экран.
Скриншот консоли:
$\varepsilon = 0.1$.

```
(base) vlasochka@vlasochka-VPCSB11FX:~/Документы/ЧМ/Lab2$ g++ Newton2.2.cpp -o newton2.2
(base) vlasochka@vlasochka-VPCSB11FX:~/Документы/ЧМ/Lab2$ ./newton2.2
Введите точность вычислений: 0.1
Метод Ньютона:
Положительный корень системы нелинейных уравнений:(2.21, 3.8028)
Число итераций: 3
```

$\varepsilon = 0.01$.

```
(base) vlasochka@vlasochka-VPCSB11FX:~/Документы/ЧМ/Lab2$ ./newton2.2
Введите точность вычислений: 0.01
Метод Ньютона:
Положительный корень системы нелинейных уравнений:(2.21044, 3.80231)
Число итераций: 4
```

$\varepsilon = 0.0001$.

```
(base) vlasochka@vlasochka-VPCSB11FX:~/Документы/ЧМ/Lab2$ ./newton2.2
Введите точность вычислений: 0.0001
Метод Ньютона:
Положительный корень системы нелинейных уравнений:(2.21045, 3.80231)
Число итераций: 5
```

$\varepsilon = 0.00001$.

```
(base) vlasochka@vlasochka-VPCSB11FX:~/Документы/ЧМ/Lab2$ ./newton2.2
Введите точность вычислений: 0.00001
Метод Ньютона:
Положительный корень системы нелинейных уравнений:(2.21045, 3.80231)
Число итераций: 5
(base) vlasochka@vlasochka-VPCSB11FX:~/Документы/ЧМ/Lab2$
```

Можно заметить, что сходимость метода Ньютона очень хорошая даже при малом значении оценки погрешности. При $\varepsilon = 0.0001$ и $\varepsilon = 0.00001$ найденный корень одинаковый. Решение исходного уравнения следующее

$$x^{(\star)} \approx \begin{pmatrix} 2.21045 \\ 3.80231 \end{pmatrix}$$

Алгоритм находит решение за 5 итераций с точностью $\varepsilon = 0.0001$.

4 Исходный код

Листинг 1: Метод Ньютона

```
#include<iostream>
#include<map>
#include<vector>
#include<math.h>

double F1(double x1, double x2)
{
    double val = x1 - cos(x2) - 3;
    return val;
}

double F2(double x1, double x2)
{
    double val = x2 - sin(x1) - 3;
    return val;
}

int main()
{
    double eps;
    int it_count = 0;
    std::vector<std::vector<double>> X(2, std::vector<double> (2, 0));

    X[0][1] = (2 + 3)/2;
    X[1][1] = (3 + 4)/2;

    std::cin >> eps;

    do
    {
        it_count++;
        X[0][0] = X[0][1];
        X[1][0] = X[1][1];

        X[0][1] = X[0][0] - (F1(X[0][0], X[1][0]) - F2(X[0][0], X[1][0])*sin(X[1][0]))/(1 + sin(X
            [1][0])*cos(X[0][0]));
        X[1][1] = X[1][0] - (F2(X[0][0], X[1][0]) + F1(X[0][0], X[1][0])*cos(X[0][0]))/(1 + sin(X
            [1][0])*cos(X[0][0]));

```

```
38      }
39      while(std::max(fabs(X[0][1] - X[0][0]), fabs(X[1][1] - X[1][0])) >= eps);
40
41      std::cout << "Newton Method:" << std::endl;
42      std::cout << "The positive solution of a system of nonlinaral equations:" << "(" << X[0][1]
            << ", " << X[1][1] << ")" << std::endl;
43      std::cout << "The number of iterations: " << it_count << std::endl;
44      return 0;
45  }
```

5 Выводы

Выполнив первую часть второго задания второй лабораторной работы, я узнала, что методом Ньютона применим и для решения систем нелинейных уравнений. Этот метод является итерационным и базируется на последовательном приближении, согласно итерационной формуле. Чтобы метод Ньютона сходился, необходимо, чтобы выполнялся ряд условий. Одно из этих условий — дифференцируемость функций, входящих системы, и невырожденность соответствующей матрицы Якоби. В случае, если начальное приближение выбрано в достаточно малой окрестности искомого корня, итерации сходятся к точному решению, причем сходимость квадратичная.