**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра МОЭВМ**

отчет

**по лабораторной работе №5**

**по дисциплине «Вычислительная математика»**

Тема: **Решение нелинейных уравнений методом простых итераций**

|  |  |  |
| --- | --- | --- |
| Студент гр. 8381 |  | Кухарев М.А. |
| Преподаватель |  | Щеголева Н.Л. |

Санкт-Петербург

2017

**Цель работы:**

Изучение метода простых итераций для приближенного решения нелинейного уравнения.

**Постановка задачи:**

Необходимо найти корень уравнения

методом простых итераций с заданной точностью *Eps*, исследовать скорость сходимости и обусловленность метода.

**Теоретическое введение:**

Метод простых итераций решения уравнения  состоит в замене исходного уравнения эквивалентным ему уравнением x=(x) и построении последовательности xn+1=(xn), сходящейся при n к точному решению.

Рассмотрим один шаг итерационного процесса. Исходя из найденного на предыдущем шаге значения xn-1, вычисляется y= (xn-1). Если , то полагается xn=y и выполняется очередная итерация. Если же , то вычисления заканчиваются и за приближенное значение корня принимается величина xn=y.

Погрешность результата вычислений зависит от знака производной : при >0 погрешность определения корня составляет qq, а при <0 погрешность не превышает . Здесь q- число, такое, что q на отрезке [a,b]. Существование числа q является условием сходимости метода в соответствии с отмеченной выше теоремой.

Функцию  необходимо подбирать так, чтобы  q  . Это обусловливается тем, что если <0 на отрезке [a,b], то последовательные приближения xn=(xn-1) будут колебаться около корня c, если же >0, то последовательные приближения будут сходиться к корню c монотонно. Следует также помнить, что скорость сходимости последовательности {xn} к корню c функции  тем выше, чем выше число q.

*(Формула 1.Критерий остановки)*

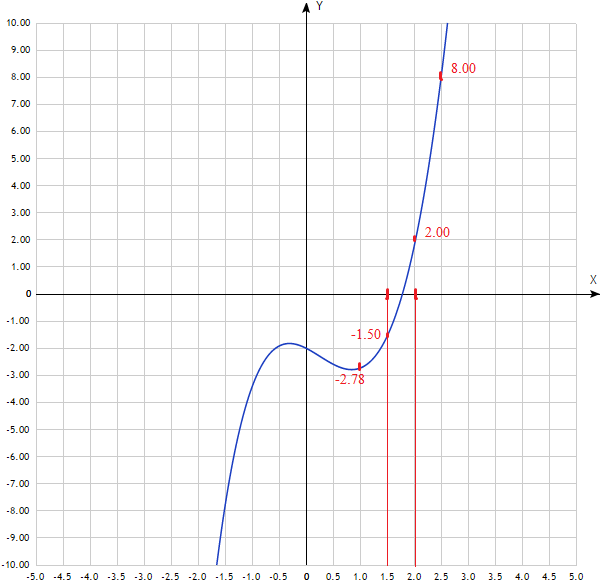
**Ход работы:**

**1)Графически или аналитически отделить корень уравнения** **.**

Для построения графиков функций воспользуемся сайтом y(x).ru.

Сначала определим отрезок, на котором функция принимает и положительное, и отрицательное значение.

*График 1. Определение отрезка.*



По графику видно, что отрезок первого приближения: [1.5; 2.0].

Определим отрезок, где находится корень, аналитически. Для этого составим таблицу:

*Таблица 1.Определение отрезка*

|  |  |
| --- | --- |
| **x** | **f(x)** |
| 1.0 | -2.78 |
| 1.5 | -1.5 |
| 1.78 | 0 |
| 2.0 | 2.0 |
| 2.5 | 8 |

По таблице также видно, что будем искать корень уравнения на промежутке [1.5;2.0].

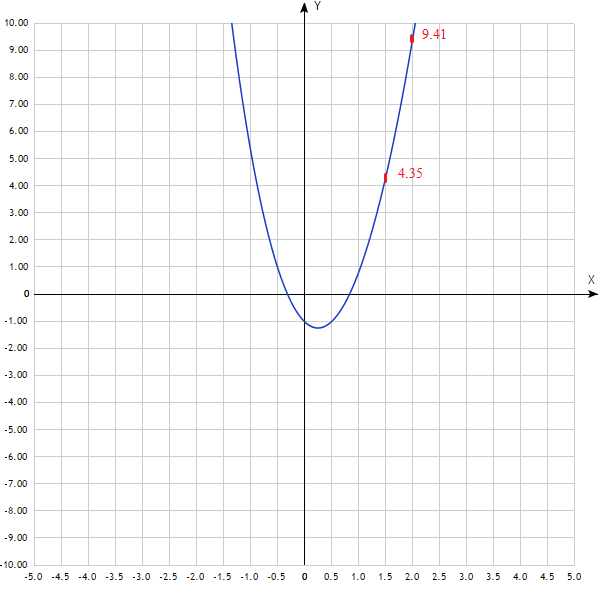
2) **Преобразовать уравнение**  **к виду** **.**

Для того чтобы применить метод, нужно подобрать функцию , которая максимизирует скорость сходимости.

=, чтобы при этом сохранялось неравенство

*q=*

*График 2. Значение производной.*



По графику видно: .

Тогда=; .

**3)Выбрать начальное приближение, лежащее на [Left, Right].**

Начальное приближение выбираем таким, чтобы оно принадлежало отрезку, на котором и исходная функция и её производная одного знака.

=1.9

**4)Составить подпрограмму для вычисления значений** **,****, предусмотрев округление вычисленных значений с точностью Delta.**

**5)Составить головную программу, вычисляющую корень уравнения и содержащую обращение к программам** **,**  **и ITER и индикацию результатов.**

**Код программы:**

**Файл .cpp**

#include "stdafx.h"

#include "iostream"

#include <fstream>

#include <math.h>

#include <stdio.h>

#include <stdlib.h>

#include "methods.h"

#include <conio.h>

double delta;

using namespace std;

void main()

{

setlocale(0, "Rus");

ifstream fin("data.txt");

if (!fin)

{

cout << "Файл с исходными данными не существует!\n";

system("pause");

exit(1);

}

ofstream fout("result.txt");

if (!fout)

{

cout << "Файл с исходными данными не существует!\n";

system("pause");

exit(1); }

double x\_real = 1.78546;

int k;

double x0, eps, x;

double F(double);

int i = 0;

while (i < 6)

{

fin >> eps;

if (!fin)

{

cout << "В файле недостаточно данных!\n";

system("pause");

exit(1);

}

fin >> x0;

if (!fin)

{

cout << "В файле недостаточно данных!\n";

system("pause");

exit(1);

}

fin >> delta;

if (!fin)

{

cout << "В файле недостаточно данных!\n";

system("pause");

exit(1);

}

cout << "Delta = " << delta << endl;

fout << "Delta = " << delta << endl;

cout << "x0 = " << x0 << endl;

fout << "x0 = " << x0 << endl;

cout << "Eps = " << eps << endl;

fout << "Eps = " << eps << endl;

x = ITER(x0, eps, k);

printf("x=%f k=%d\n", x, k);

fout << "x = " << x << endl;

fout << "k = " << k << endl;

cout << "abs(x - x\_real) = " << abs(x - x\_real) << endl;

fout << "abs(x - x\_real) = " << abs(x - x\_real) << endl;

fout << "\n";

cout << "\n";

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

fout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

fout << "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*" << endl;

fout << "\n";

cout << "\n";

i++;

fin.get();

}

system("pause");

}

**Файл .h**

#include "Stdafx.h"

#include "iostream"

#include <math.h>

using namespace std;

double Round(double xx, double Delta)

{

if (Delta <= 1E-9)

{

puts("Неверное задание точности округления\n");

exit(1);

}

if (xx > 0.0) return (Delta\*(long((xx / Delta) + 0.5)));

else return (Delta\*(long((xx / Delta) - 0.5)));

}

double F(double x)

{

extern double delta;

double s;

long int fi;

s = x - 0.15\*(x\*x\*x-3\*x-2\*exp(-x));

if (s / delta < 0)

fi = s / delta - 0.5;

else

fi = s / delta + 0.5;

s = fi\*delta;

s = Round(s, delta);

return(s);

}

double ITER(double xx0, double Eps, int &N)

{

if (Eps <= 0.0) { puts("Невер­ое задан­ие точ­ности\n"); exit(1); }

double x1 = F(xx0);

double x2 = F(x1);

N = 2;

while ((x1 - x2)\*(x1 - x2) > fabs((2 \* x1 - xx0 - x2)\*Eps))

{

xx0 = x1;

x1 = x2;

x2 = F(x1);

N++;

}

return(x2);

}

**5) Провести вычисления по программе. Исследовать скорость сходимости и обусловленность метода.**

*Таблица 2. Вычисления. №1*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **X** | **k** | **| x – x\* |** |  | **Eps** | **delta** | **Обуслов.** |
| 1.8 | 2 | 0.01454 | 1.78546 | 0. 1 | 0.1 | плохо |
| 1.8 | 2 | 0.01454 | 1.78546 | 0.01 | хорошо |
| 1.8 | 2 | 0.01454 | 1.78546 | 0.001 | хорошо |
| 1.8 | 2 | 0.01454 | 1.78546 | 0.0001 | хорошо |
| 1.8 | 2 | 0.01454 | 1.78546 | 1e-05 | хорошо |
| 1.8 | 2 | 0.01454 | 1.78546 | 1e-06 | хорошо |

*№2*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **X** | **k** | **| x – x\* |** |  | **Eps** | **delta** | **Обуслов.** |
| 1.79 | 2 | 0.00454 | 1.78546 | 0. 1 | 0.01 | плохо |
| 1.79 | 2 | 0.00454 | 1.78546 | 0.01 | плохо |
| 1.79 | 3 | 0.00454 | 1.78546 | 0.001 | хорошо |
| 1.79 | 3 | 0.00454 | 1.78546 | 0.0001 | хорошо |
| 1.79 | 3 | 0.00454 | 1.78546 | 1e-05 | хорошо |
| 1.79 | 3 | 0.00454 | 1.78546 | 1e-06 | хорошо |

*№3*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **X** | **k** | **| x – x\* |** |  | **Eps** | **delta** | **Обуслов.** |
| 1.786 | 2 | 0.00054 | 1.78546 | 0. 1 | 0.001 | плохо |
| 1.786 | 2 | 0.00054 | 1.78546 | 0.01 | плохо |
| 1.785 | 3 | 0.00046 | 1.78546 | 0.001 | плохо |
| 1.785 | 3 | 0.00046 | 1.78546 | 0.0001 | хорошо |
| 1.785 | 4 | 0.00046 | 1.78546 | 1e-05 | хорошо |
| 1.785 | 4 | 0.00046 | 1.78546 | 1e-06 | хорошо |

*№4*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **X** | **k** | **| x – x\* |** |  | **Eps** | **delta** | **Обуслов.** |
| 1.7858 | 2 | 0.00034 | 1.78546 | 0. 1 | 0.0001 | плохо |
| 1.7858 | 2 | 0.00034 | 1.78546 | 0.01 | плохо |
| 1.7854 | 3 | 6e-05 | 1.78546 | 0.001 | плохо |
| 1.7854 | 3 | 6e-05 | 1.78546 | 0.0001 | плохо |
| 1.7855 | 5 | 4e-05 | 1.78546 | 1e-05 | хорошо |
| 1.7855 | 5 | 4e-05 | 1.78546 | 1e-06 | хорошо |

*№5*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **X** | **k** | **| x – x\* |** |  | **Eps** | **delta** | **Обуслов.** |
| 1.7858 | 2 | 0.00034 | 1.78546 | 0. 1 | 1e-05 | плохо |
| 1.7858 | 2 | 0.00034 | 1.78546 | 0.01 | плохо |
| 1.78545 | 3 | 1e-05 | 1.78546 | 0.001 | плохо |
| 1.78545 | 3 | 1e-05 | 1.78546 | 0.0001 | плохо |
| 1.78545 | 3 | 1e-05 | 1.78546 | 1e-05 | плохо |
| 1.78546 | 4 | 0 | 1.78546 | 1e-06 | хорошо |

*№6*

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **X** | **k** | **| x – x\* |** |  | **Eps** | **delta** | **Обуслов.** |
| 1.7858 | 2 | 0.000343 | 1.78546 | 0. 1 | 1e-06 | плохо |
| 1.7858 | 2 | 0.000343 | 1.78546 | 0.01 | плохо |
| 1.78545 | 3 | 1.1e-05 | 1.78546 | 0.001 | плохо |
| 1.78545 | 3 | 1.1e-05 | 1.78546 | 0.0001 | плохо |
| 1.78545 | 3 | 1.1e-05 | 1.78546 | 1e-05 | плохо |
| 1.78546 | 4 | 2e-06 | 1.78546 | 1e-06 | плохо |

**Вывод:**

В ходе выполнения данной лабораторной работы была исследована чувствительность метода простых итераций при нахождении корня уравнения *f(x)=0* для нелинейной функции.

Было замечено, что в случае вычисления корня методом простых итераций уравнения ,итераций программе понадобилось меньше, чем при поиске корня методом бисекциии и методом хорд. Поэтому можно сказать, что у метода простых итераций скорость сходимости лучше, чем у метода хорд и бисекции. Но хуже чем у метода Ньютона.

|  |  |  |  |
| --- | --- | --- | --- |
| Название метода | eps | delta | k |
| Метод бисекций |  |  | 17 |
| Метод хорд | 8 |
| Метод Ньютона | 3 |
| Метод простых итераций | 4 |

Обусловленность задачи нахождения корня уравнения *f(x)=0* прямо пропорциональна точности задания исходных данных *delta*. Для сравнения таблицы №1: *delta*=0.1 – обусловленность хорошая, и таблица №6: *delta* =– обусловленность плохая, при соответствующих *eps*.



Метод простых итераций ухудшает обусловленность задачи при *eps > delta* и не ухудшает при *eps <=delta*. Для сравнения таблица №6: *eps=0.1* и *delta= -* обусловленность плохая, и таблица №1: *eps=* и *delta=0.1 -*обусловленность хорошая.



Итерационная последовательность сходится со скоростью геометрической прогрессии со знаменателем *q = 0,37*.

Анализируя таблицы №1-№6 можно сделать вывод, что взяв постоянную *eps*=и уменьшая *delta* на порядок в каждой таблице, обусловленность будет ухудшаться.