**Белорусский государственный университет**

**Факультет прикладной математики и информатики**

Лабораторная работа №1

Нахождение корня нелинейного уравнения методами простой итерации, Ньютона, Чебышева

Вариант №4

**Выполнил:**

Студент 2 курса 7 группы ФПМИ

Лубенько Алексей Анатольевич

**Преподаватель:**

Будник Анатолий Михайлович

Минск, 2022

**Алгоритм решения**

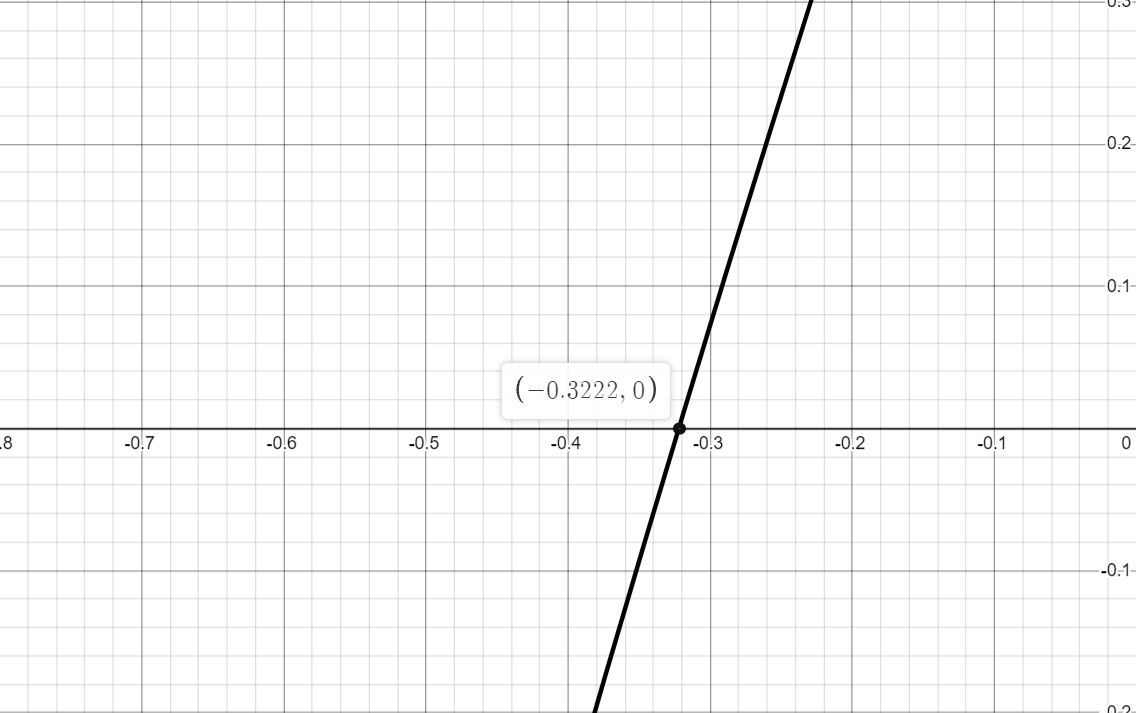
1. **Метод простой итерации**

Рассмотрим заданное уравнение . Приведем его к каноническому виду:

Тогда итерационный процесс будет выглядеть следующим образом:

Критерий останова:

Отделим корни графическим методом (по условию):



Выберем начальное приближение и отрезок

, тогда и проверим выполнение условий теоремы о сходимости метода простой итерации:

***Следовательно, условия теоремы выполнены, и метод простой итерации сходится.***

Найдем априорное число итераций для достижения заданной точности:

Для проверки результата найдем невязку .

**Листинг программы**

#include <iostream>

#include <cmath>

using namespace std;

double f(double x){

return pow(x,3.)+3.\*x+1.;

}

double fi(double x){

return (-1./3.)\*pow(x,3)-(1./3.);

}

int main(){

const double eps=pow(10,-7);

double xn1=-0.45;

double xn=0;

int count=0;

while(abs(xn1-xn)>eps){

xn=xn1;

xn1=fi(xn);

count++;

}

double r=f(xn1);

cout<<"Корень уравнения: "<<xn1<<endl;

cout<<"Невязка: " <<r<<endl;

cout<<"Число итераций:" <<count<<endl;

return 0;

}

**Выходные данные**

Корень уравнения: -0.322185357

Невязка: -7.82568e-09

Число итераций: 16

1. **Метод Чебышева**

Берем промежуток и начальное приближение из первого пункта:

Построим итерационный процесс вида:

Данный процесс *—* метод Чебышева, обладающий кубической скоростью сходимости.

**Листинг программы**

#include <iostream>

#include <cmath>

#include <iomanip>

using namespace std;

double f(double x){

return pow(x,3.)+3.\*x+1.;

}

double f\_1der(double x){

return 3\*pow(x,2)+3;

}

double f\_2der(double x){

return 6\*x;

}

int main(){

setlocale(LC\_ALL, ".1251");

const double eps=pow(10,-7);

double xn1=-0.45;

double xn=0;

int count=0;

while(abs(xn1-xn)>eps){

xn=xn1;

xn1=xn-(f(xn)/ f\_1der(xn))-(f(xn)\*f(xn)\* f\_2der(xn)/2/pow(f\_1der(xn),3));

count++;

}

double r=f(xn1);

cout<<setprecision(9)<<xn1<<endl;

cout<<r<<endl;

cout<<count<<endl;

return 0;

}

**Выходные данные**

Корень уравнения:-0.322185355

Невязка: 0

Число итераций:3

1. **Метод Ньютона**

Берем промежуток и начальное приближение из первого пункта:

Построим итерационный процесс вида:

Он обладает квадратичной скоростью сходимости и погрешность на каждой итерации представима в виде:

Для проверки результата найдем невязку

**Листинг программы**

#include <iostream>

#include <cmath>

#include <iomanip>

using namespace std;

double f(double x){

return pow(x,3.)+3.\*x+1.;

}

double f\_1der(double x){

return 3\*pow(x,2)+3;

}

int main(){

setlocale(LC\_ALL, ".1251");

const double eps=pow(10,-7);

double xn1=-0.45;

double xn=0;

int count=0;

while(abs(xn1-xn)>eps){

xn=xn1;

xn1=xn-(f(xn)/ f\_1der(xn));

count++;

}

double r=f(xn1);

cout<<setprecision(9)<<xn1<<endl;

cout<<r<<endl;

cout<<count<<endl;

return 0;

}

**Выходные данные**

Корень уравнения:-0.322185355

Невязка: 0

Число итераций:6

**Вывод:**

Метод простой итерации сошелся к корню с требуемой точностью, т.к невязка решения меньше, чем требуемая точность. Априорная оценка — 21 итерация, а реальное число итераций — 16. Это связано с тем, что априорная оценка показывает завышенный результат.

Для взятого начального условия метод Ньютона сошелся. Выбранная точность привела к тому, что невязка решения оказалась равной нулю(в пределах точности типа double). Сравнивая метод Ньютона с МПИ, можно заметить, что метод Ньютона сошелся к решению уравнения значительно быстрее (6 итераций против 16), а само приближение точнее, чем в МПИ. Это обусловлено квадратичной скоростью сходимости метода Ньютона в сравнении с линейной скоростью МПИ.

Метод Чебышева, обладающий кубической скоростью сходимости, сошелся к решению быстрее, чем метод Ньютона

(3 итерации против 6) . Невязка решения оказалась равной нулю (в пределах точности типа double).