МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский Авиационный Институт»

(Национальный Исследовательский Университет)

Институт№8: «Информационные технологии и прикладная математика»

Кафедра: 806 «Вычислительная математика и программирование»

**КУРСОВОЙ ПРОЕКТ**

По курсу «Вычислительные системы»

I семестр

Тема:

«Процедуры и функции в качестве параметров»

|  |  |
| --- | --- |
| Группа: | М8О-106Б-22 |
| Студент: | Абдисаламов Э. |
| Преподаватель: | Дубинин А. В. |
| Оценка: |  |
| Дата: |  |

Москва, 2022

Содержание

[Введение 3](#_Toc124119847)

[1. Теоретическая часть 4](#_Toc124119848)

[1.1 Метод дихотомии (половинного деления) 4](#_Toc124119849)

[1.2 Метод итераций 5](#_Toc124119850)

[1.3 Метод Ньютона 5](#_Toc124119851)

[2. Практическая часть 7](#_Toc124119852)

[2.1 Задание 7](#_Toc124119853)

[2.1.1 Вариант 7](#_Toc124119854)

[2.2 Использованные переменные и функции 7](#_Toc124119855)

[2.3 Графики функций и их производных 8](#_Toc124119856)

[2.3.1 Функция из варианта 5 8](#_Toc124119857)

[2.3.2 Функция из варианта 6 10](#_Toc124119858)

[2.4 Протокол 12](#_Toc124119859)

[2.5 Выходные данные 16](#_Toc124119860)

[Заключение 18](#_Toc124119861)

[Список использованных источников 19](#_Toc124119862)

# Введение

Нахождение корней трансцендентных уравнений является зачастую достаточно сложной задачей, не решаемой аналитически с помощью конечных формул. Кроме того, иногда на практике уравнение содержит коэффициенты, значения которых заданы приблизительно, так что говорить о точном решении уравнений в таких случаях не стоит, поэтому задачи приближенного определения корней уравнения и соответствующей оценки их точности имеют большое значение.

Цель работы: составить программу на языке Си с процедурами решения трансцендентных алгебраических уравнений различными численными методами (итераций, Ньютона и половинного деления – дихотомии). Применить каждую процедуру к решению двух уравнений.

# 1. Теоретическая часть

## 1.1 Метод дихотомии (половинного деления)

Если на отрезке [a; b] существует корень уравнения, то значения функции на концах отрезка имеют разные знаки: F(a)\*F(b)<0. Метод заключается в делении отрезка пополам и его сужении в два раза на каждом шаге итерационного процесса в зависимости от знака функции в середине отрезка.

Итерационный процесс строится следующим образом: за начальное приближение принимаются границы исходного отрезка . Далее вычисления проводятся по формулам: , если или по формулам если .

Процесс повторяется до тех пор, пока не будет выполнено условие окончания .

Приближенное значение корня к моменту окончания итерационного процесса получается следующим образом: .



Рисунок 1. Метод дихотомии (половинного деления)

## 1.2 Метод итераций

Идея метода заключается в замене исходного уравнения уравнением вида .

Достаточное условие сходимости метода . Это условие необходимо проверить перед началом решения задачи, так как функция может быть выбрана неоднозначно, причем в случае неверного выбора указанной вункции метод расходится.

Начальное приближение корня: (середина исходного отрезка).

Итерационный процесс: .

Условие окончания: .

Приближенное значение корня: .

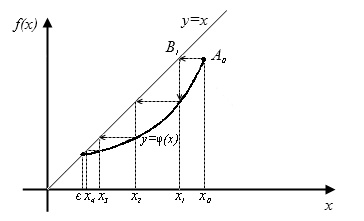


Рисунок 2. Метод итераций

## 1.3 Метод Ньютона

Метод Ньютона является частным случаем метода итераций.

Условие сходимости метода: на отрезке .

Итерационный процесс: . На рисунке 3 представлена геометрическая интерпретация метода Ньютона.

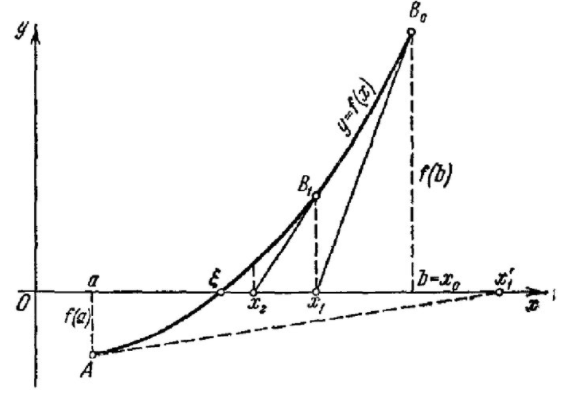


Рисунок 3. Метод Ньютона

2. Практическая часть

2.1 Задание

Составить программу на языке Си с процедурами решения трансцендентных алгебраических уравнений различными численными методами (итераций, Ньютона и половинного деления – дихотомии). Нелинейные уравнения оформить как параметры-функции, разрешив относительно неизвестной величины в случае необходимости. Применить каждую процедуру к решению двух уравнений, заданных двумя строками таблицы, начиная с варианта с заданным номером. Если метод неприменим, дать математическое обоснование и графическую иллюстрацию.

2.1.1 Вариант

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Уравнение | Отрезок, содержащий корень | Базовый метод | Приближенное значение корня |
| 5 |  |  |  |  |
| 6 |  |  |  |  |

2.2 Использованные переменные и функции

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Описание |
| delta | double | Погрешность при нахождении корня |
| res.steps | int | Количество итераций |
| res.root | double | Корень уравнения |
| res.success | bool | Показывает, выполняется ли функция данным методом |
| a5 | double | Начало отрезка для 5 варианта |
| b5 | Конец отрезка для 5 варианта |
| a6 | Начало отрезка для 6 варианта |
| b6 | Конец отрезка для 6 варианта |
| Функции | | |
| F5 | double | Исходная функция 5 |
| F6 | Исходная функция 6 |
| f5 | Выраженный x для функции 5 для метода итераций |
| f6 | Выраженный x для функции 6 для метода итераций |
| df | Первая производная функции в точке |
| ddf | Вторая производная функции в точке |
| diho | result | Вычисление корня методом дихотомии (половинного деления) |
| itera | Вычисление корня методом итераций |
| Newton | Вычисление корня методом Ньютона |
| print\_result | void | Выводит результат работы разных методов для передаваемой функций в указанном отрезке. |

## 2.3 Графики функций и их производных

### 2.3.1 Функция из варианта 5

, отрезок [0, 1]

График функции:

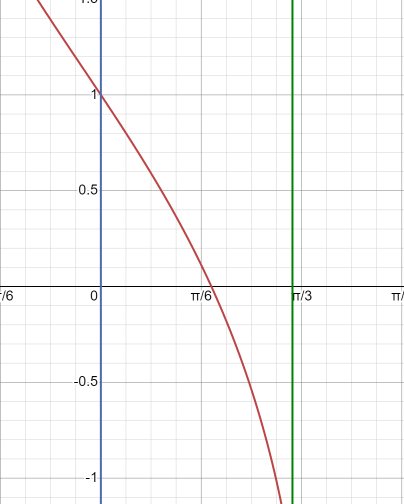
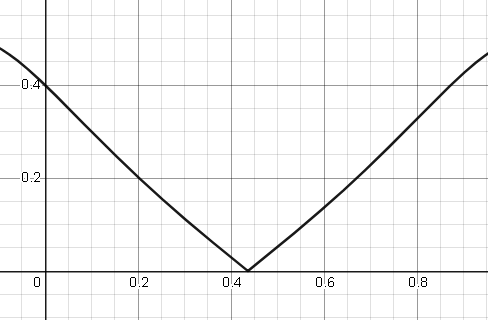
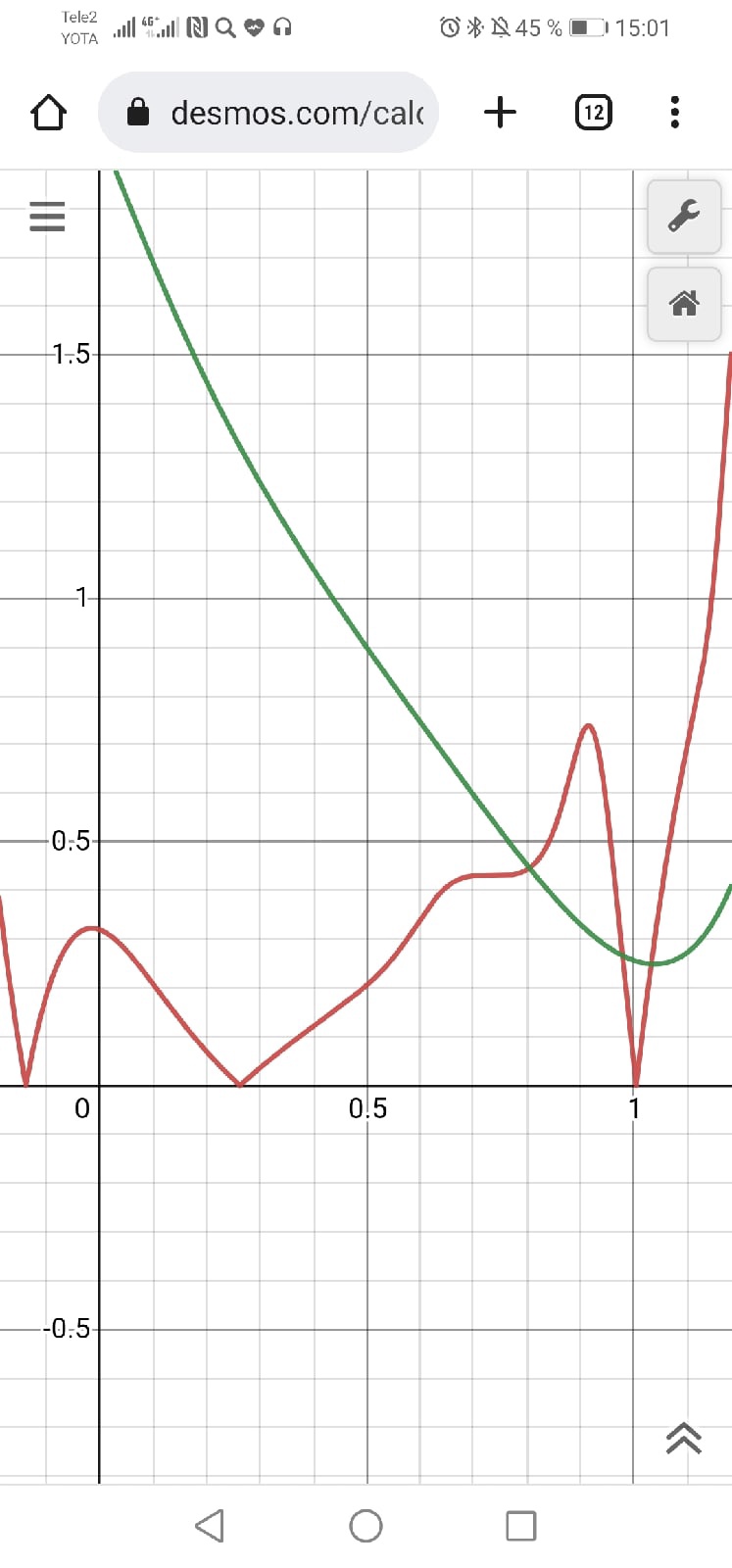


График производной:



Условие выполняется на всём отрезке [0, 1], следовательно, условие сходимости метода итераций выполнено

Графики (красный и зеленый):



Условие не выполняется на всём отрезке [0, 1], однако так как наше начальное приближение корня равно , то метод Ньютона будет выполняться, если корень лежит в области выполнения условия.

### 2.3.2 Функция из варианта 6

, отрезок [0.5;1]

График функции:

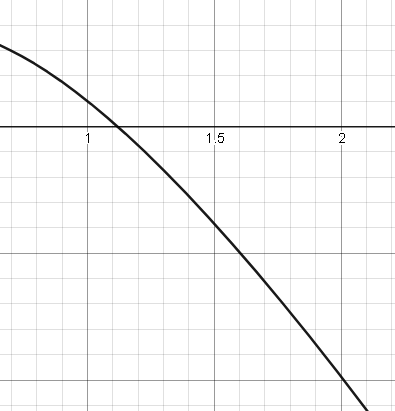
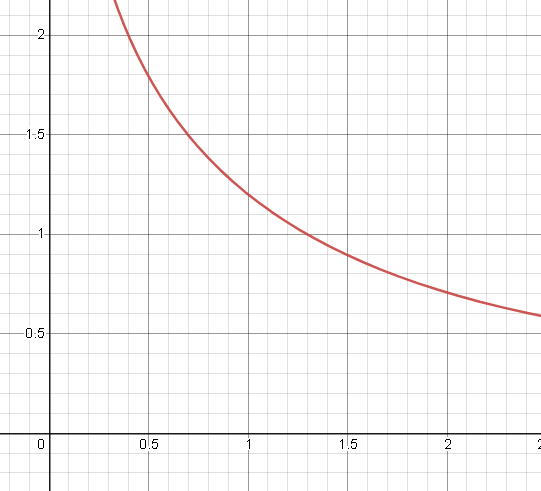
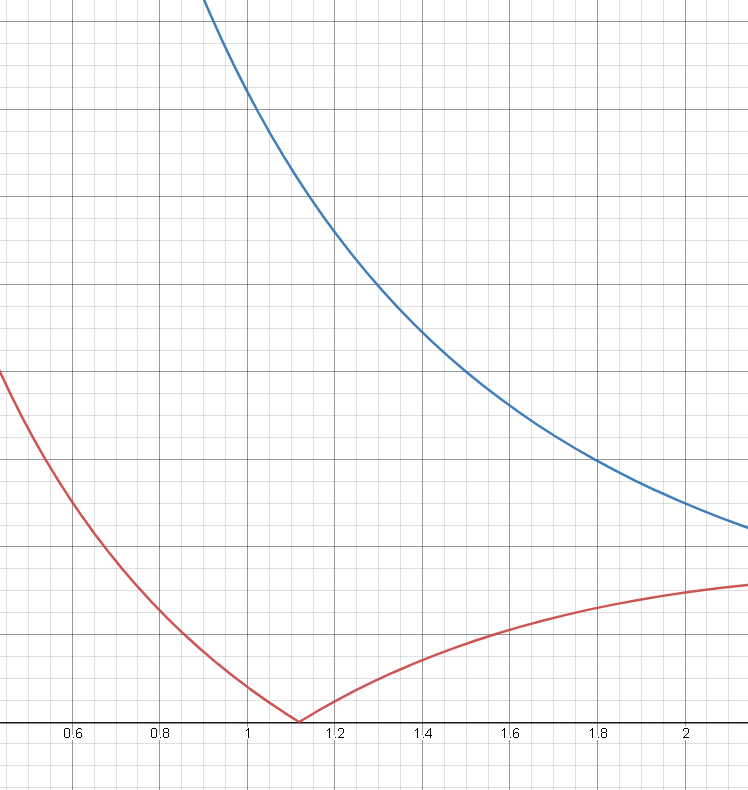


График производной функции:



Условие выполняется на всём отрезке [1, 2], следовательно, условие сходимости метода итераций выполнено

Графики (красный и синий):



Условие < выполняется на всём отрезке , поэтому условие сходимости метода Ньютона выполнено.

## 2.4 Протокол

kp4.c:

#include "func\_kp4.h"

int main(){

    double delta = pow(2, -14);

    double a5 = 0, b5 = 1, a6 = 0.5, b6 = 1;

    print\_result(5, F5, f5, a5, b5, delta);

    print\_result(6, F6, f6, a6, b6, delta);

}

func\_for\_kp4.c:

#include "func\_kp4.h"

double F5(double x){

    return (sqrt(1-x) - tan(x));

}

double f5(double x){

    return 1 - tan(x) \* sqrt(1-x);

}

double F6(double x){

    return (x + cos(pow(x, 0.52) + 2));

}

double f6(double x){

    return -cos(pow(x, 0.52) + 2);

}

double df( double (\*f)(double), double x){

    double dx = pow(2, -21);

    return (f(x + dx) - f(x - dx))/(2 \* dx);

}

double ddf( double (\*f)(double), double x){

    double dx = pow(2, -12);

    return (f(x + dx) - 2 \* f(x) + f(x - dx)) / (dx \* dx);

}

result diho(double (\*f)(double), double a, double b, double delta){

    result res;

    res.steps = 0;

    res.success = true;

    if(f(a)\*f(b)>0){

        res.success = false;

        return res;

    }

    while(b - a > delta){

        double c = (a+b)/2;

        if (f(a) \* f(c) < 0) {b = c;}

        else {a = c;}

        res.steps++;

    }

    res.root = (a + b)/2;

    return res;

}

result itera(double (\*f)(double), double a, double b, double delta){

    result res;

    res.steps = 0;

    res.success = true;

    double x0 = (a+b)/2, x1 = x0;

    do {

        if (fabs(df(f, x0)) >= 1){

            res.success = false;

            return res;

        }

        x0 = x1;

        x1 = f(x0);

        res.steps++;

    } while (fabs(x1 - x0) > delta );

    res.root = x1;

    return res;

}

result Newton(double (\*f)(double), double a, double b, double delta){

    result res;

    res.steps = 0;

    res.success = true;

    double x0 = (a+b)/2, x1 = x0;

    do{

        if (fabs(f(x0) \* ddf(f, x0)) >= pow(df(f, x0), 2)){

            res.success = 0;

            return res;

        }

        x0 = x1;

        x1 = x0 - f(x0)/df(f, x0);

        res.steps++;

    } while (fabs(x1 - x0) > delta);

    res.root = x1;

    return res;

}

void print\_result(int variant, double (\*F)(double), double (\*f)(double), double a, double b, double delta){

    printf("For %d variant:\n", variant);

    result dih = diho(F, a, b, delta);

    if (dih.success){

        printf("Dihotomia:  root = %.4lf, steps = %d\n", dih.root, dih.steps);

    } else { printf("The Dihotomia method isn't applicable\n");}

    result iter = itera(f, a, b, delta);

    if (iter.success){

        printf("Iteration:  root = %.4lf, steps = %d\n", iter.root, iter.steps);

    } else { printf("The iteration method isn't applicable\n");}

    result newtn = Newton(F, a, b, delta);

    if (newtn.success){

        printf("Newton:  root = %.4lf, steps = %d\n\n", newtn.root, newtn.steps);

    } else { printf("Newton's method isn't applicable\n\n");}

}

Func\_kp4:

#pragma once

#include <stdio.h>

#include <stdbool.h>

#include <math.h>

typedef struct {

    double root;

    int steps;

    bool success;

} result;

double F5(double x);

double f5(double x);

double F6(double x);

double f6(double x);

double df( double (\*f)(double), double x);

double ddf( double (\*f)(double), double x);

result diho(double (\*f)(double), double a, double b, double delta);

result itera(double (\*f)(double), double a, double b, double delta);

result Newton(double (\*f)(double), double a, double b, double delta);

void print\_result(int variant, double (\*F)(double), double (\*f)(double), double a, double b, double delta);

2.5 Выходные данные

PS C:\Users\elmar\OneDrive - mai.education\Рабочий стол\МАИ\1 курс\Инфа\КП\КП4> gcc -Wall -pedantic -std=c99 func\_for\_kp4.c kp4.c -o kp4.exe

PS C:\Users\elmar\OneDrive - mai.education\Рабочий стол\МАИ\1 курс\Инфа\КП\КП4> ./kp4.exe

For 5 variant:

Dihotomia: root = 0.5768, steps = 14

Iteration: root = 0.5768, steps = 10

Newton: root = 0.5768, steps = 3

For 6 variant:

Dihotomia: root = 0.9892, steps = 13

Iteration: root = 0.9892, steps = 5

Newton: root = 0.9892, steps = 3

# Заключение

В ходе работы над курсовым проектом были рассмотрены три численных метода решения трансцендентных уравнений: метод дихотомии, метод итераций, метод Ньютона. Используя сведения из численных методов, я составил программу на языке С с процедурами решения трансцендентных алгебраический уравнений. Также научился работать со структурами в языке С и еще передавать функции как аргумент к функции.

# Список использованных источников

1. Программирование// Численные методы решения нелинейных уравнений
2. Графический калькулятор Десмос

<https://www.desmos.com/calculator?lang=ru>

1. Сайт Алексея Ларина//Метод итераций

<http://alexlarin.net/Int/zad.htm>