МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский Авиационный Институт»
(Национальный Исследовательский Университет)

Институт: №8 «Компьютерные науки и прикладная математика»
Кафедра: 806 «Вычислительная математика и программирование»

Курсовая работа по
курсу
«Фундаментальная информатика»
I семестр
Задание 4 «Процедуры и функции в качестве
параметров»

Группа	М8О-109Б-22
Студент	Недосекин А.А.
Преподаватель	Сысоев М.А.
Оценка	
Дата	

Москва, 2022

Постановка задачи

Составить программу на Си с процедурами решения трансцендентных алгебраических уравнений резличными 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, например, с использованием gnuplot.

Вариант 23:

Функция:

$$3x - 4\ln x - 5 = 0$$

Отрезок содержащий корень: [2, 4]

Метод Ньютона **Вариант**

24:

Функция:

$$\cos\frac{2}{x} - 2\sin\frac{1}{x} + \frac{1}{x} = 0$$

Отрезок содержащий корень: [1, 2]

Метод Дихотомии

Теоретическая часть

Метод Ньютона

Метод Ньютона является частным случаем метода итераций.

Условие сходимости метода: $|F(x)\cdot F''(x)| < (F'(x))^2$ на отрезке $[a,b]$.

Итерационный процесс: $x^{(k+1)} = x^{(k)} - F(x^{(k)}) / F'(x^{(k)})$.

Метод дихотомии (половинного деления)

Очевидно, что если на отрезке $[a,b]$ существует корень уравнения, то значения функции на концах отрезка имеют разные знаки: $F(a)\cdot F(b)<0$. 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 $a^{(0)}=a$, $b^{(0)}=b$. Далее вычисления проводятся по формулам: $a^{(k+1)}=(a^{(k)}+b^{(k)})/2$, $b^{(k+1)}=b^{(k)}$, если $F(a^{(k)})\cdot F((a^{(k)}+b^{(k)})/2)>0$; или по формулам: $a^{(k+1)}=a^{(k)}$, $b^{(k+1)}=(a^{(k)}+b^{(k)})/2$, если $F(b^{(k)})\cdot F((a^{(k)}+b^{(k)})/2)>0$.

Процесс повторяется до тех пор, пока не будет выполнено условие окончания $\left|a^{(k)}-b^{(k)}\right|<\varepsilon$.

Приближенное значение корня к моменту окончания итерационного процесса получается следующим образом $x^*\approx(a^{(конечное)}+b^{(конечное)})/2$.

Описание алгоритма

Составляю программу для нахождения корня с помощью метода Ньютона и проверяю найденный корень, либо вывожу, что метод не применим.

Аналогично поступаю и с методом дихотомии.

Использованные в программе переменные

Название переменной	Тип переменной	Смысл переменной
LDBL_EPSILON	long double	Машинный эпсилон 1.0842e-19
step	long double	Шаг для проверки
a	long double	Левая граница отрезка
b	long double	Правая граница отрезка
x_0	long double	значение x
x	long double	Следующее значение x

Исходный код программы:

Вариант 23

```
#include <stdio.h>
#include <float.h>
#include <math.h>

long double f23(long double x) {
    return (3*x - 4*logl(x) - 5);      //3x - 4ln(x) - 5 = 0
}
long double f23_der1(long double x) {
    return 3 - 4/x;
}

long double f23_der2(long double x) {
    return 4/pow(x, 2);
}

int IsNewtonConvergent(
        long double (*f)(long double),
        long double (*der1)(long double),
        long double (*der2)(long double),
        long double a,
        long double b) {
    int res = 1;
    long double step = (b - a) / 100;
    for (long double x = a; x <= b; x += step) {
        if (fabsl(f(x) * der2(x)) >= powl(der1(x), 2)) {
            res = 0;
        }
    }
    return res;
}

long double NewtonMethod(
        long double (*f)(long double),
        long double (*der)(long double),
        long double a,
        long double b) {
    long double x0 = (a + b) / 2;
    long double x1 = x0 - f(x0) / der(x0);
    while (fabsl(x1 - x0) >= LDBL_EPSILON) {
        x0 = x1;
        x1 = x0 - f(x0) / der(x0);
```

```
    }
    return x1;
}



int main() {

    int a = 2;
    int b = 4;
    printf("Function 23 (Newton Method):\n\t\t3x-4lnx-5 = 0\n");
    if (IsNewtonConvergent(f23, f23_der1, f23_der2, a, b)) {
        printf("Convergence check:\tOK!\n");

        long double root = NewtonMethod(f23, f23_der1, a, b);
        printf("Approximated root:\t%.10Lf\n", root);

        long double result = f23(root);
        printf("Result of inserting root:\t%.10Lf\n", result);
    } else {
        printf("Convergence check:\t FAIL!\n\n");
    }
}
```

Вариант 24:

```
#include <stdio.h>
#include <math.h>
#include <float.h>

//Метод дихотомии
long double func24(long double x){
    return ( cos(2/x) - 2*sin(1/x) + 1/x );
}

long double dixit(long double (*f)(long double), long double a, long double b){
    long double c;
    long double ans;
    while (fabsl(a - b) > LDBL_EPSILON) {
        c = (a + b) / 2.0;
        if (f(c) * f(a) < 0) {
            b = c;
```

```
        } else {
            a = c;
        }
    }
    ans = c;
    return ans;
}


int main() {
    //24 var function
    long double a = 1.0,
                b = 2.0;
    printf("Func24 : cos(2/x) - 2*sin(1/x) + 1/x = 0 Method: dixit.\n%.19Lf", dixit(func24, a, b));

    printf("\n\n");


    return 0;
}
```

Входные данные

Нет

Выходные данные

24

```
Func24 : cos(2/x) - 2*sin(1/x) + 1/x = 0 Method: dixit.
1.8756173924904572353



...Program finished with exit code 0
Press ENTER to exit console.
```

23

```
Function 23 (Newton Method):
              3x-4lnx-5 = 0
Convergence check:      FAIL!
```

Вывод

В работе описаны и использованы различные численные методы для решения трансцендентных алгебраических уравнений. Даны обоснования сходимости и расходимости тех или иных методов. Имплементирована функция вычисления производной от заданной функции в точке. На основе алгоритма составлена программа на языке Си, сделана проверка полученных значений путем подстановки. Работа представляется довольно полезной для понимания принципов работы численных методов и способов их имплементации.

Список литературы

1. Численное дифференецирование – URL:

   Численное дифференцирование — Википедия (wikipedia.org)

2. Конечная разность – URL:

   Численное дифференцирование — Википедия (wikipedia.org)