МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Федеральное государственное бюджетное образовательное учреждение высшего образования

Московский Авиационный Институт
(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Курсовой проект
по курсу «Основы информатики»
I семестр

Задание 4
«Процедуры и функции в качестве параметров»

Группа: М80 – 107Б-18

Студент: Цапков Александр Максимович

Преподаватель: Ридли Александра Николаевна

Оценка: ____________________

Дата:______________________

Москва, 2018

Содержание

Введение

В четвертом задание курсового проекта мне требуется создать программу на языка Си с процедурами решения трансцендентных алгебраических уравнений различными численными методами (итерации, Ньютона и половинного деления - Дихотомии). Если метод неприменим к функции, дать математическое обоснование и графическую иллюстрацию, демонстрирующую это.

Краткое описание методов и их реализация в программном коде

Каждую функцию нужно решить с помощью 3 методов: итерации, Ньютона и половинного деления - Дихотомии.

Метод иттерации

Метод итерации — численный метод решения математических задач, приближённый метод решения системы линейных алгебраических уравнений. Суть такого метода заключается в нахождении по приближённому значению величины следующего приближения (являющегося более точным). Каждый следующий итерационный член x^{k+1} рассчитывается как:

$$x_{k+1} = x_k - \lambda(x_k)F(x_k)$$

Где λ(x) - может являться константой со знаком совпадающем со знаком производной.

В программе я реализовал цикл, который каждую новую итерацию считает новый x через предыдущий, и запоминает оба эти значения. Цикл заканчивается когда разница между двумя рядом идущими иксами станет меньше машинного эпсилон. Начальное значение берется как середина отрезка на котором функция является монотонной и имеет корень (концы такого отрезка даны в задании).

```
double iter(double a, double b)
{
    double e = eps(1.0);
    double x;
    double xP1 = (a + b) / 2;
    do {
        x = xP1;
        xP1 = f(x) + x;
    } while (fabs(xP1 - x) > e) ;
    return xP1;
}
```

Метод Ньютона

Метод Ньютона — является частным случаем метода итераций. В данном случае $\lambda(x) = \dfrac{1}{F'(x)}$. Следующий член в данном методе рассчитывается как:

$$x_{k+1} = x_k - \frac{F(x_k)}{F'(x_k)}$$

В программе я также реализовал итерационный цикл, который был немного изменен, в связи с усложнением функции вычисления следующего члена.

```
double newt(double a, double b)
{
    double e = eps(1.0);
    double x;
    double xP1 = (a + b) / 2;
    do {
        x = xP1;
        xP1 = x - f(x) / fd(x);
    } while (fabs(xP1 - x) > (e)) ;
    return xP1;
}
```

Метод Дихотомии

Суть метода дихотомии заключается в том, что произведение значений точек которые лежат по разные стороны от оси абсцисс дает отрицательное значение, тогда когда произведение значений точек по одну сторону от оси ординат дают положительное значение. Можно делить отрезок пополам, и проверять, какой его части принадлежит корень, и делить уже ее. Таким образом с каждым делением наш отрезок все больше будет приближаться к корню уравнения. Когда расстояние между концами отрезка станет меньше чем эпсилон, можно вывести, что искомая точка приблизительно равна:

$$x = \frac{a^k + b^k}{2}$$

В программе это реализовано с помощью цикла. Каждую итерацию он проверяет произведение начала отрезка a и его середины (a + b) / 2. Если это значение отрицательно, значит искомый корень в первой половине отрезка и я переопределяю b как середину нашего отрезка. Если же изначальное произведение не положительно, значит корень находится во второй половине отрезка и я переопределяю a как середину нашего отрезка. Далее цикл повторяется пока b - a не станет меньше эпсилон. Если же произведение значений функции на концах отрезка равно нулю, значит один из этих концов и есть корень. Проверяем каждый и выводим нужный.

```
double dich(double a, double b)
{
    double e = eps(1.0);
    while ((b - a) > e) {
        if (f(a) * f((a + b) / 2) < 0) {
            b = (a + b) / 2;
        } else if (f(a) * f((a + b) / 2) > 0) {
            a = (a + b) / 2;
        } else if (f(a) == 0) {
            b = a;
            break;
        } else if (f(b) == 0) {
            a = b;
            break;
        }
    }
    return (a + b) / 2;
}
```

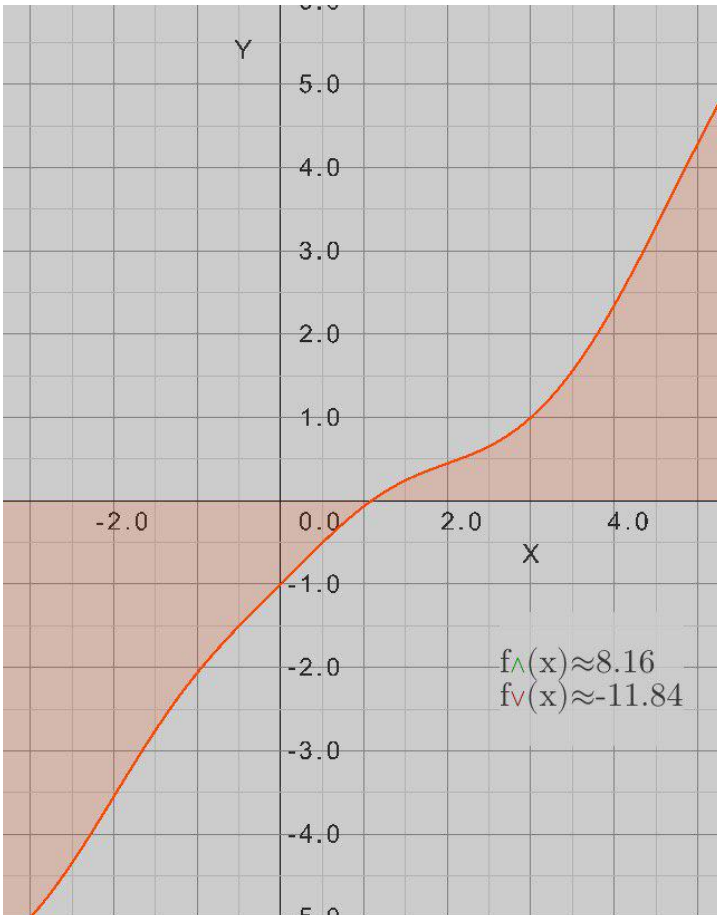
Представление функций и их графиков

В задании мне дано 2 функции.

Первая:

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

Ее график в районе заданной области выглядит так:



Формальная запись в коде программы этой функции и ее производной:

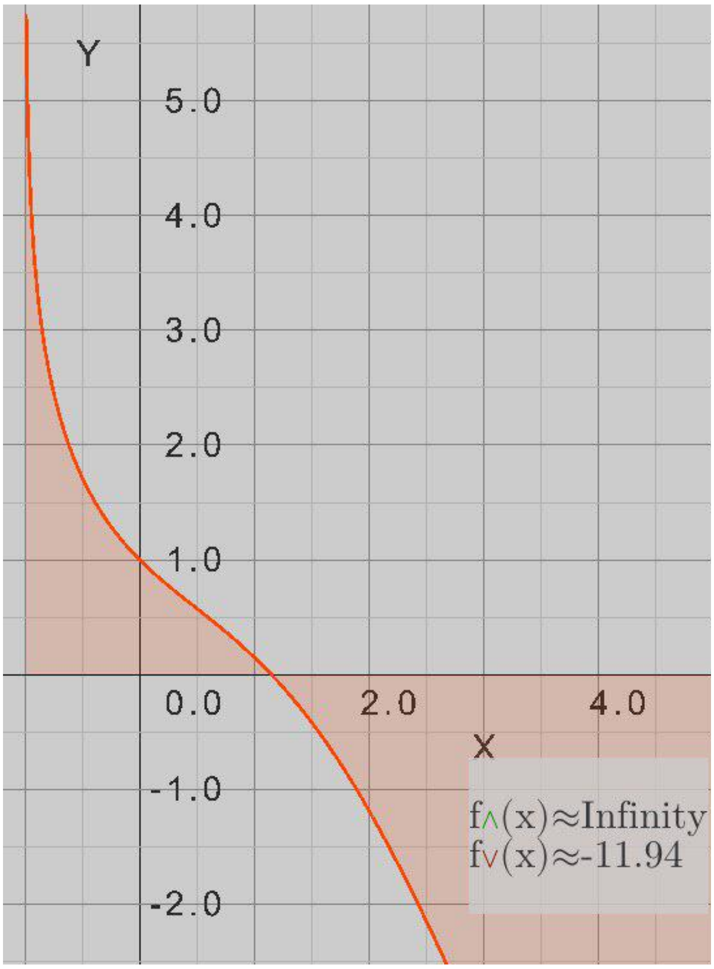
```
double f(double x)
{
    return cos(x) − exp(− (x ∗ x) / 2) + x − 1;
}

double fd(double x)
{
    return −sin(x) + exp(− (x ∗ x) / 2) ∗ x + 1;
}
```

Вторая функция:

$$1 - x + \sin x - \ln(1 + x) = 0$$

Ее график в районе заданной области выглядит так:



Формальная запись в коде программы этой функции и ее производной:

```
double f(double x)
{
    return 1 − x + sin(x) −log(1 + x);
}

double fd(double x)
{
    return −1 + cos(x) −1 / (1 + x);
}
```

Анализ получившихся результатов

Результаты работы все трех методов для 1-1 функции таковы:

```
Alexs-MacBook-Air:4 alex$ gcc -std=c99 -Wall -pedantic -g  main.c
Alexs-MacBook-Air:4 alex$ ./a.out
Дихотомия       1.0894428008491106041333296
Итерационный  1.0894428008491112702671444
Ньютьон         1.0894428008491110482225395
```

Для 2-й функции это:

```
Alexs-MacBook-Air:4 alex$ gcc -std=c99 -Wall -pedantic -g  main.c
Alexs-MacBook-Air:4 alex$ ./a.out
Дихотомия       1.1474388506295438716620083
Итерационный 1.1474388506295443157512182
Ньютьон         1.1474388506295443157512182
```

Все результаты сходятся и очень близки друг к другу.

Заключение

В нашем случае все уравнения сошлись и решились, но такое может быть не всегда. У метода итераций (а значит и у метода Ньютона) есть свои условия и ограничения. Да он более эффективный, чем метод Дихотомии, зато метод дихотомии является “бронебойным”. У него нет никаких других условий, кроме непрерывности на отрезки и одного корня. Метод Дихотомии - является одним из мощнейших инструментов в решении уравнений на компьютере.

Источники

1. https://ru.wikipedia.org/wiki/Метод_Ньютона
2. http://school-collection.edu.ru/dlrstore-wrapper/21808a0c-95ed-448f-9098-68061e6e8fe0/Metod_iteraciy.html