Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ»

КАФЕДРА «ТЕОРЕТИЧЕСКАЯ ИНФОРМАТИКА И КОМПЬЮТЕРНЫЕ ТЕХНОЛОГИИ»

Отчёт по лабораторной работе № 6
«Сравнение приближенных методов решения нелинейных уравнений»

по курсу

«Численные методы»

Студент группы ИУ9-61Б Бакланова А.Д.

Преподаватель Домрачева А.Б.

Москва, 2020 г.

Цель работы

Целью данной работы является реализация и сравнение приближенных методов решения нелинейных уравнений.

Задание

Нарисовать график функции $f(x)$ (с полным исследованием функции) и найти отрезки, где функция имеет простые корни и отличные от нуля первые две производные.

Найти с точностью 0.001 все корни уравнения $f(x) = 0$ методом деления отрезка пополам и методом Ньютона; определить число приближений в каждом случае.

Теоретические сведения:

Метод деления отрезка пополам:

Самый простой метод решения задачи. Поиск корня функции $f(x)$ начинаем с деления отрезка $[a, b]$ пополам точкой $x = \dfrac{(a+b)}{2}$

Из двух получившихся отрезков выберем тот, на котором находится корень уравнения, т.е. где функция меняет знак. Обозначим этот отрезок $[a_1, b_1]$. Если $f(a)f(x) < 0$, то это — отрезок $[a, x]$ $(a_1 = a, b_1 = x)$; в противном случае — отрезок $[x, b]$ $(a_1 = x, b_1 = b)$. Отправляясь от отрезка $[a_1, b_1]$ вдвое меньшей длины, опять находим середину отрезка $x = \dfrac{(a_1 + b_1)}{2}$ и определяем по описанному алгоритму отрезок $[a_2, b_2]$ и т.д. На $k-$ом шаге длина получившегося отрезка $[a_k, b_k]$ будет равна $b_k - a_k = \frac{(b-a)}{2}$. Процесс продолжаем до тех пор, пока не будет выполнено условие $b_k - a_k \le 2\varepsilon$, где ε — требуемая точность нахождения корня. Тогда приближенное значение корня уравнения $x = \frac{(a_k + b_k)}{2}$

Метод Ньютона (касательных)

Метод решения нелинейного уравнения является итерационным. Для получения $(k+1)-$й итерации метода x_{k+1} из точки $(x_k, \quad f(x_k))$, лежащей на графике функции, проводим касательную. Точка пересечения касательной с осью абсцисс и есть следующее, $(k+1)-$е приближение к корню уравнения. Алгебраически метод Ньютона сводится к рекурсивной зависимости

$$x_{k+1} = x_k - f(x_k)/f'(x_k), \quad k = 0,1,2...$$

Достаточным условием сходимости метода является отличие от нуля первых двух производных функции $f(x)$ на отрезке $[a, b]$. В качестве начального приближения x_0 выбираем тот конец отрезка, где знак функции совпадает со знаком второй производной $(f(x_0)f''(x_0) > 0)$. Заданная погрешность ε считается достигнутой, если

$$f(x_k)f(x_k + sgn(x_k - x_{k-1})\varepsilon) < 0, \text{ где } sgn\,x = \begin{cases} 1, & x > 0 \\ 0, & x = 0 \\ -1, & x < 0 \end{cases}$$

Дано:

$$f(x) = Ax^3 + Bx^2 + Cx + D$$

Входные данные:

$$f(x) = x^3 - 3x^2 + 3$$

Программная реализация:

ApproxNonLinear.java

```java
public class ApproxNonLinear {
    public static void main(String[] args) {
        //2 вариант
        // A=1 B=-3 C=0 D=3
        //f(x) = x^3 - 3x^2 +3


        divideOnHalf(-10.0, 0.0);
        divideOnHalf(0.0, 2.0);
        divideOnHalf(2.0, 10.0);

        newton(-1.0, 0.0);
        newton(1.0, 1.5);
        newton(2.0, 3.0);

    }

    static void divideOnHalf(double a, double b){
        double x = 0;
        double a1 = a;
        double b1 = b;
        double e = 0.001;
        double k=0;
        while (b1-a1 > 2*e) {
            x = (a1+ b1)/2;
            if (f(a) * f(x) < 0) {
                b1 = x;
            } else {
                a1 = x;
            }
            k++;
        }
        System.out.println("Divide method: "+(a1+b1)/2);
        System.out.println("Num of steps: "+k);
    }

    public static double sgn(double x) {
        if (x>0) return 1;
        else if (x==0) return 0;
        else return -1;
    }

    static void newton(double a, double b) {
        double x0 = 0;
```

```java
        double x1 = 0;
        double e = 0.001;

        if (f(a)*f_derivative2(a) > 0){
            x0 = a;
        }else{
            x0 = b;
        }
        int k = 0;
        while (true) {
            x1 = x0 -f(x0)/f_derivative(x0);
            k++;
            if ((f(x1)*f( x: x1 + sgn( x: x1-x0)*e))<0){
                break;
            }
            x0 = x1;

        }
        System.out.println("Newton method: "+x1);
        System.out.println("Num of steps: "+k);
    }

    static double f(double x) { return (1)*x*x*x + (-3)*x*x + (0)*x + (3); }

    static double f_derivative(double x) { return 3*x*x + (-6)*x; }

    static double f_derivative2(double x) { return 6*x - 6; }
}
```

Результат:

```
/Library/Java/JavaVirtualMachines/jdk1.8.0_181.jdk/Contents/Home/bin/java ...
Divide method: -0.8795166015625
Num of steps: 13.0
Divide method: 1.3466796875
Num of steps: 10.0
Divide method: 2.5322265625
Num of steps: 12.0
Newton method: -0.8794515669515669
Num of steps: 2
Newton method: 1.3472222222222222
Num of steps: 2
Newton method: 2.5323901618653797
Num of steps: 3

Process finished with exit code 0
```

Вывод:

Итого, в результате лабораторной работы были реализованы приближенные методы решения нелинейных уравнений. Серией тестов было выявлено, что метод Ньютона выполняется за гораздо меньшее количество шагов, в отличие от метода деления отрезка пополам, что очень сильно сказывается на производительности. Также, метод Ньютона дает более точные результаты.