**Министерство науки и высшего образования Российской Федерации**

федеральное государственное автономное образовательное учреждение

высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Инженерная школа природных ресурсов

Направление подготовки Химическая технология

Отделение химической инженерии

**ИТЕРАЦИОННЫЕ МЕТОДЫ РЕШЕНИЯ НЕЛИНЕЙНЫХ УРАВНЕНИЙ**

**Лабораторная работа по дисциплине «Углубленный курс информатики»**

Выполнил студент гр. 2Д91 Е.А. Резинкина

(Подпись)

20 апреля 2020 г.

Отчет принят:

Преподаватель

доцент ОХИ ИШПР, к.т.н. В.А. Чузлов

(Подпись)

\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 г.

Томск 2020 г.

**Цель работы:** изучить теорию методов итерации, уделить особое внимание методам половинного деления и Ньютона, закрепить полученные знания на практике, составив программы на языке Pascal.

**Теоретическая часть**

Итерационные методы – это методы последовательных приближений. В них необходимо задать некоторое приближённое решение – начальное приближение. После этого с помощью некоторого алгоритма проводится один цикл вычислений, называемый итерацией. В результате итерации находят новое приближение. Итерации проводятся до получения решения с требуемой точностью.

Наиболее простым в реализации является метод бисекции, или как его еще называют, метод половинного деления. Это итерационный метод, суть которого заключается в том, что на каждой итерации интервал сокращается вдвое до тех пор, пока не будет найдено решение с заданной точностью.

Данный метод достаточно прост и содержит всего два действия. Сначала находится переменная х – середина интервала [a,b]. После чего вычисляется значение функции в середине интервала. Затем определяется, совпадает ли по знаку значение функции в середине интервала, со знаком функции в левой части. В случаи если их знаки равны, то новой левой границей считается середина интервала, в ином же случаи правой граница интервала считается его середина. Таким образом, при каждой итерации интервал сокращается вдовое то справа, то слева.

Еще одним хорошим методом решения уравнений является метод касательных или метод Ньютона. Главное его отличие от представленных ранее методов бисекции и хорд – отсутствие необходимости отделения корня. Вместо этого нужно задать лишь начальное приближение. Однако его главным недостатком остается сложность реализации, связанная, прежде всего с необходимостью определять производные исходного уравнения.

В основе метода Ньютона лежит разложения функции в ряд Тейлора:

http://tpdn.ru/upload/articles/teilor.gif

Обычно значения ряда, содержащие шаг h во второй и более высоких степенях отбрасывают, так как их влияние на результат незначительны.

Суть метода заключается в экстраполяции функции касательными. После того как пользователь задает начальное приближение, программа должна определить точку пересечения касательной к графику функции с осью абсцисс. Для этого используется формула:

http://tpdn.ru/upload/articles/kasatelnih.gif

Затем находится значение функции в точке пересечения касательной с осью абсцисс и если получившиеся значение близко к нулю, то считается, что решение уравнения найдено.

К сожалению, при всех своих достоинствах метод Ньютона не гарантирует сходимости. Отсутствия решения может возникнуть по нескольким причинам. Например, это может произойти из-за того, что касательная будет параллельна оси абсцисс. В этом случае необходимо предусмотреть выход из цикла при достижении большого количества итераций.

Количество итераций напрямую зависит от введенных начальных данных. При удачном стечении обстоятельств решение каждым из методов может быть найдено даже при единственной итерации.

**Практическая часть**

**Задание**

Составить программу для решения нелинейных уравнений методом половинного деления, простых итераций и методом Ньютона:

Интервал [1; 2], допустимая точность 10-2;

**Программная реализация**

**program** LB09\_1\_1;

**const**

eps = 1e-2;

**function** f(x: real): real;

**begin**

result := exp(4\*ln(x))+3\*x-20;

**end**;

**function** dihotomy(a, b: real; eps: real): real;

**var**

x: real;

**begin**

**repeat**

x := (a + b) / 2;

**if** f(a) \* f(x) > 0 **then**

a := x

**else**

b := x

**until** (abs(a - b) <= eps) **or** (f(x) = 0);

result := x

**end**;

**begin**

writeln('x = ',dihotomy(1, 2, eps))

**end**.

**program** LB09\_1\_2;

**const**

eps = 1e-2;

**function** g(x: real): real;

**begin**

result := x-(exp(4\*ln(x))+3\*x-20)/35;

**end**;

**function** iterations(a, b: real; eps: real): real;

**var**

x: real;

**begin**

result := a;

**repeat**

x := g(result);

result := g(x)

**until** abs(result - x) <= eps;

**end**;

**begin**

writeln('x = ',iterations(1, 2, eps))

**end**.

**program** lb09\_1\_3;

**const**

eps = 1e-2;

**function** f(x: real): real;

**begin**

result := exp(4\*ln(x))+3\*x-20

**end**;

**function** f1(x: real): real;

**begin**

result := 4\*exp(3\*ln(x))+3

**end**;

**function** f2(x: real): real;

**begin**

result := 12\*exp(2\*ln(x))

**end**;

**function** newton(a, b: real; eps: real): real;

**var**

x: real;

**begin**

**if** f(a) \* f2(a) > 0 **then**

result := a

**else**

**if** f(b) \* f2(b) > 0 **then**

result := b

**else**

**begin**

writeln('Метод Ньютона, решений нет!');

**exit**

**end**;

**repeat**

x := result;

result := x - f(x) / f1(x)

**until** abs(result - x) <= eps;

**end**;

**begin**

writeln('x = ',newton(1, 2, eps))

**end**.

**Ответ**

х=1.9453125

Интервал [0; 1], допустимая точность 10-3;

**Программная реализация**

**program** LB9\_2\_1;

**const**

eps = 1e-3;

**function** f(x: real): real;

**begin**

result := exp(x)+x-2;

**end**;

**function** dihotomy(a, b: real; eps: real): real;

**var**

x: real;

**begin**

**repeat**

x := (a + b) / 2;

**if** f(a) \* f(x) > 0 **then**

a := x

**else**

b := x

**until** (abs(a - b) <= eps) **or** (f(x) = 0);

result := x

**end**;

**begin**

writeln('x=',dihotomy(0, 1, eps))

**end**.

**program** LB9\_2\_2;

**const**

eps = 1e-3;

**function** g(x: real): real;

**begin**

result := x-(exp(x)+x-2)/4;

**end**;

**function** iterations(a, b: real; eps: real): real;

**var**

x: real;

**begin**

result := a;

**repeat**

x := g(result);

result := g(x)

**until** abs(result - x) <= eps;

**end**;

**begin**

writeln('x = ',iterations(0, 1, eps))

**end**.

**program** lb9\_2\_3;

**const**

eps = 1e-3;

**function** f(x: real): real;

**begin**

result := exp(x)+x-2

**end**;

**function** f1(x: real): real;

**begin**

result := exp(x)+1

**end**;

**function** f2(x: real): real;

**begin**

result := exp(x)

**end**;

**function** newton(a, b: real; eps: real): real;

**var**

x: real;

**begin**

**if** f(a) \* f2(a) > 0 **then**

result := a

**else**

**if** f(b) \* f2(b) > 0 **then**

result := b

**else**

**begin**

writeln('Метод Ньютона, решений нет!');

**exit**

**end**;

**repeat**

x := result;

result := x - f(x) / f1(x)

**until** abs(result - x) <= eps;

**end**;

**begin**

writeln('x = ',newton(0, 1, eps))

**end**.

**Ответ**

х= 0.4423828125

Интервал [0.5; 1.5], допустимая   
точность 0.2\*10-4;

**Программная реализация**

**program** LB9\_3\_1;

**const**

eps = 2e-5;

**function** f(x: real): real;

**begin**

result := ln(x)+x;

**end**;

**function** dihotomy(a, b: real; eps: real): real;

**var**

x: real;

**begin**

**repeat**

x := (a + b) / 2;

**if** f(a) \* f(x) > 0 **then**

a := x

**else**

b := x

**until** (abs(a - b) <= eps) **or** (f(x) = 0);

result := x

**end**;

**begin**

writeln('x = ',dihotomy(0.5, 1.5, eps))

**end**.

**program** LB9\_3\_2;

**const**

eps = 2e-5;

**function** g(x: real): real;

**begin**

result := x-(ln(x)+x)/3;

**end**;

**function** iterations(a, b: real; eps: real): real;

**var**

x: real;

**begin**

result := a;

**repeat**

x := g(result);

result := g(x)

**until** abs(result - x) <= eps;

**end**;

**begin**

writeln('x = ',iterations(0.5, 1.5, eps))

**end**.

**program** lb9\_3\_3;

**const**

eps = 2e-5;

**function** f(x: real): real;

**begin**

result := ln(x)+x

**end**;

**function** f1(x: real): real;

**begin**

result := 1/x+1

**end**;

**function** f2(x: real): real;

**begin**

result := -1/sqr(x)

**end**;

**function** newton(a, b: real; eps: real): real;

**var**

x: real;

**begin**

**if** f(a) \* f2(a) > 0 **then**

result := a

**else**

**if** f(b) \* f2(b) > 0 **then**

result := b

**else**

**begin**

writeln('Метод Ньютона, решений нет!');

**exit**

**end**;

**repeat**

x := result;

result := x - f(x) / f1(x)

**until** abs(result - x) <= eps;

**end**;

**begin**

writeln('x = ',newton(0.5, 1.5, eps))

**end**.

**Ответ**

х=0.567153930664063

Интервал [0.2; 1.5], допустимая   
точность 0.5\*10-4;

**Программная реализация**

**program** LB9\_4\_1;

**const**

eps = 5e-5;

**function** f(x: real): real;

**begin**

result := 2\*x-exp(-0.1\*x);

**end**;

**function** dihotomy(a, b: real; eps: real): real;

**var**

x: real;

**begin**

**repeat**

x := (a + b) / 2;

**if** f(a) \* f(x) > 0 **then**

a := x

**else**

b := x

**until** (abs(a - b) <= eps) **or** (f(x) = 0);

result := x

**end**;

**begin**

writeln('x = ',dihotomy(0.2, 1.5, eps))

**end**.

**program** LB9\_4\_2;

**const**

eps = 5e-5;

**function** g(x: real): real;

**begin**

result := x-(2\*x-exp(-0.1\*x))/2;

**end**;

**function** iterations(a, b: real; eps: real): real;

**var**

x: real;

**begin**

result := a;

**repeat**

x := g(result);

result := g(x)

**until** abs(result - x) <= eps;

**end**;

**begin**

writeln('x = ',iterations(0.2, 1.5, eps))

**end**.

**program** lb9\_4\_3;

**const**

eps = 5e-5;

**function** f(x: real): real;

**begin**

result := 2\*x-exp(-0.1\*x)

**end**;

**function** f1(x: real): real;

**begin**

result := 2+exp(-0.1\*x)/10

**end**;

**function** f2(x: real): real;

**begin**

result := -exp(-0.1\*x)/100

**end**;

**function** newton(a, b: real; eps: real): real;

**var**

x: real;

**begin**

**if** f(a) \* f2(a) > 0 **then**

result := a

**else**

**if** f(b) \* f2(b) > 0 **then**

result := b

**else**

**begin**

writeln('Метод Ньютона, решений нет!');

**exit**

**end**;

**repeat**

x := result;

result := x - f(x) / f1(x)

**until** abs(result - x) <= eps;

**end**;

**begin**

writeln('x = ',newton(0.2, 1.5, eps))

**end**.

**Ответ**

х=0.476718139648438

**Выводы**

В ходе работы была изучена теория по заданной теме, а также полученные знания были применены на практике в ходе написания программ.