МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

**«Вятский государственный университет»**

Факультет автоматики и вычислительной техники

Кафедра электронных вычислительных машин

ЧИСЛЕННЫЕ МЕТОДЫ РЕШЕНИЯ НЕЛИНЕЙНЫХ УРАВНЕНИЙ

Отчет по лабораторной работе №1 дисциплины

«Вычислительная математика»

Вариант 7

Выполнил студент группы ИВТ-22\_\_\_\_\_\_\_\_\_\_\_\_/Волостных М.И./

Проверил преподаватель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/Архангельский В.В./

Киров 2016

**Задание:**

1. Построить график функции f(x) и отделить один из корней уравнения: f(x).

2. Сузить интервал изоляции корня, если необходимо, проверив условие: M<=2m.

3. Уточнить корень с погрешностью e<=0,00001 двумя численными методами: комбинированным методом и методом итераций.

4. Проверить полученное значение корня, используя систему Mathcad.

Вариант 7

Уравнение: x+ln(x)+0.5

Интервал: [0,1;1,0]

**2 Ход выполнения лабораторной работы**

**2.1 Теоретические сведения**

**2.1.1 Теоретические сведения по уточнению корня комбинированным методом**

Методы хорд и касательных дают приближения корня с разных сторон. Поэтому их часто применяют в сочетании друг с другом, тогда уточнение корня происходит быстрее.

Пусть дано уравнение *f(x) = 0*, корень отделен на отрезке *[a, b]*.

Рассмотрим случай, когда *f ‘(x) f ’’(x)>0*.

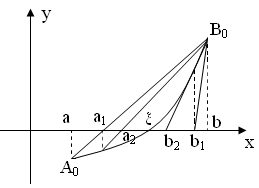


Рисунок 1

В этом случае метод хорд дает приближенное значение корня с недостатком (конец *b* неподвижен), а метод касательных – с избытком (за начальное приближение берем точку *b*).

Тогда вычисления следует проводить по формулам:

http://dit.isuct.ru/ivt/sitanov/Literatura/M501/Pages/Glava2_5.files/image003.gif

http://dit.isuct.ru/ivt/sitanov/Literatura/M501/Pages/Glava2_5.files/image005.gif

Теперь корень *ξ* заключен в интервале *[a1, b1]*. Применяя к этому отрезку комбинированный метод, получим:

http://dit.isuct.ru/ivt/sitanov/Literatura/M501/Pages/Glava2_5.files/image007.gif

http://dit.isuct.ru/ivt/sitanov/Literatura/M501/Pages/Glava2_5.files/image009.gif

и т.д.

|  |  |
| --- | --- |
| http://dit.isuct.ru/ivt/sitanov/Literatura/M501/Pages/Glava2_5.files/image011.gif  http://dit.isuct.ru/ivt/sitanov/Literatura/M501/Pages/Glava2_5.files/image013.gif |  |

Если же *f ‘(x) f ’’(x)<0* (рис. 2.14), то, рассуждая аналогично, получим следующие формулы для уточнения корня уравнения:

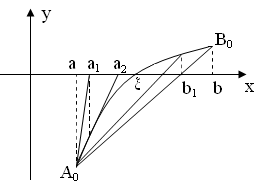


Рисунок 2

http://dit.isuct.ru/ivt/sitanov/Literatura/M501/Pages/Glava2_5.files/image015.gif

http://dit.isuct.ru/ivt/sitanov/Literatura/M501/Pages/Glava2_5.files/image017.gif

Вычислительный процесс прекращается, как только выполнится условие:

http://dit.isuct.ru/ivt/sitanov/Literatura/M501/Pages/Glava2_5.files/image019.gif

**2.1.2 Теоретические сведения по уточнению корня методом простых итераций**

Представим исходное уравнение http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image003.gif в виде http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image290.gif.

Пусть нам известно начальное приближение к корню http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image168.gif (http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image079.gif). Подставив его в правую часть уравнения (2.21) получим новое приближение http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image294.gif, затем аналогичным образом получим http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image296.gif и так далее,

http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image298.gif,    http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image300.gif.

Оказывается, что при определенных свойствах функции http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image302.gif последовательность http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image304.gif, определяемая по формуле (2.22), сходится к корню уравнения http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image003.gif. Необходимо установить при каких условиях итерационный процесс будет сходящимся.

В начале рассмотрим графически процесс получения приближений в методе простых итераций. При решении уравнения необходимо отыскать точку пересечения кривой http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image307.gif и прямой http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image309.gif. На рисунке изображена некоторая кривая http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image307.gif, которая может представлять собой любую функцию, но сейчас для нас важно то обстоятельство, что производная этой функции в окрестности корня положительна и меньше 1. Пусть http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image311.gif  –  корень уравнения, который, естественно, предполагается неизвестным. Выберем начальное приближение в точке http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image168.gif. Следующее приближении будет равно http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image315.gif. Для того, чтобы отобразить http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image081.gif на графике можно провести через точку http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image270.gif прямую, параллельную оси *OX*, до пересечения с прямой http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image309.gif, а затем в точке пересечения этих прямых опустить перпендикуляр на ось *OX*, который и отметит положение точки http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image081.gif. Аналогично получаются все  последующие приближения. Из рисунка видно, что они сходятся к корню. Напомним, что для рассмотрения мы взяли функцию производная которой   
положительна и меньше 1.

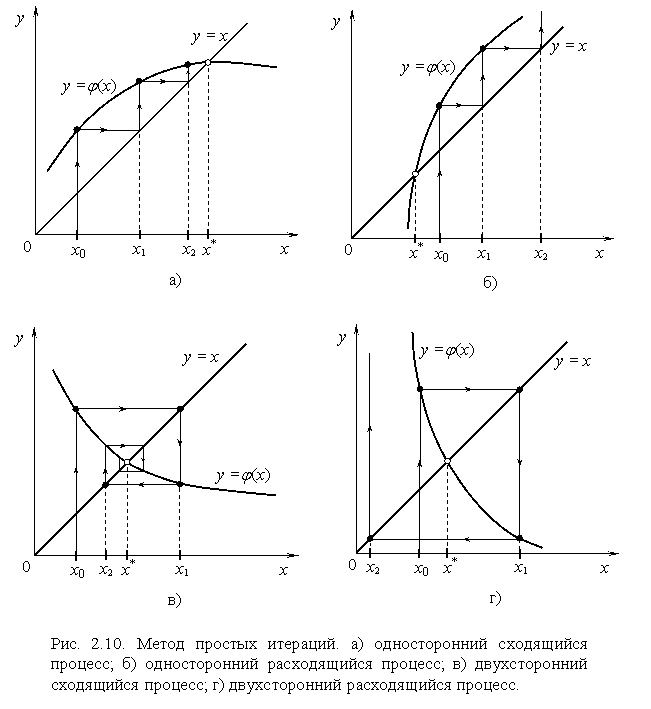


Рисунок 3 – а) Односторонний сходящийся процесс; б) односторонний расходящийся процесс; в) двухсторонний сходящийся процесс; г) двухсторонний расходящийся процесс

Рассмотрим теперь другую функцию http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image307.gif, производная которой отрицательна, но меньше 1 по абсолютному значению. Последовательные приближения также сходятся к корню, но на этот раз каждое последующее приближение находится с противоположной стороны от корня. В то время как в первом случае все последовательные приближения находились с одной стороны от корня.

Наконец, рассмотрим случай, когда производная функции http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image307.gif больше 1 и меньше -1. В обоих случаях каждое последующее приближение отстоит дальше от корня, т.е.  итерационный процесс расходится. Из сказанного выше можно предположить, что итерационный процесс, определяемый формулой сходится при условии, что производная http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image319.gif меньше 1 по абсолютной величине.

Математически условие сходимости можно установить следующим образом. Представим *k*-е и (*k*+1)-е приближения в форме

http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image321.gif,  http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image323.gif,

где http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image325.gif и http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image327.gif – отклонения приближений от корня.

Функцию http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image302.gif вблизи точки http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image075.gif приближенно заменим первыми двумя членами ряда Тейлора. Тогда итерационная формула примет вид

http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image331.gif,

но поскольку http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image075.gif является корнем уравнения, то первые слагаемые в правой и левой частях этого выражения тождественно равны и, следовательно

http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image334.gif.

Для сходимости итерационного процесса необходимо, чтобы погрешность на каждом шаге убывала

http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image336.gif,

откуда следует, что в окрестности корня должно выполняться условие

http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image338.gif.

Таким образом, для того чтобы итерационный процесс был сходящимся, необходимо, чтобы абсолютная величина производной http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image340.gif в окрестности корня была меньше единицы. Если это условие выполняется на отрезке http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image206.gif на котором локализован корень, то в качестве начального приближения можно взять любую точку из этого отрезка http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image079.gif. Скорость сходимости зависит от абсолютной величины производной http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image344.gif:  чем меньше http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image344.gif вблизи корня, тем быстрее сходится процесс.

**2.1.3 Преобразование уравнения к итерационному виду.** Переход от уравнения к уравнению в итерационной форме можно осуществить различными способами в зависимости от вида функции http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image005.gif. При таком переходе необходимо построить функцию http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image302.gif так, чтобы выполнялось условие сходимости (2.23).

Теперь рассмотрим один из общих алгоритмов перехода от уравнения http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image003.gif к уравнению http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image372.gif. Умножим левую и правую части уравнения http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image003.gif на произвольную константу http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image375.gif и добавим к обеим частям неизвестное *x*. При этом корни исходного уравнения не изменятся

http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image377.gif

или

http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image379.gif.

Уравнение эквивалентно уравнению с функцией http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image381.gif. Произвольный выбор константы **  позволяет обеспечить выполнение условия сходимости. Поскольку в данном случае http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image383.gif, значение **  следует выбирать, так чтобы в окрестности корня выполнялось условие

http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image385.gif.

Желательно выбрать величину **  такой, чтобы http://physics.herzen.spb.ru/library/01/01/nm_labs/nonlineareq.files/image387.gif, тогда сходимость будет двухсторонней (рис. 2.11, в).

**2.2 Необходимые расчёты**

Исходное уравнение: x+ln(x)-0.5

Исходная функция: f(x)=x+ln(x)-0.5

Первая производная исходной функции:

Вторая производная исходной функции:

Преобразование исходного уравнения к каноничному виду φ(x) = x:

Первая производная от φ(x):

**2.3 Скриншоты выполнения программы**

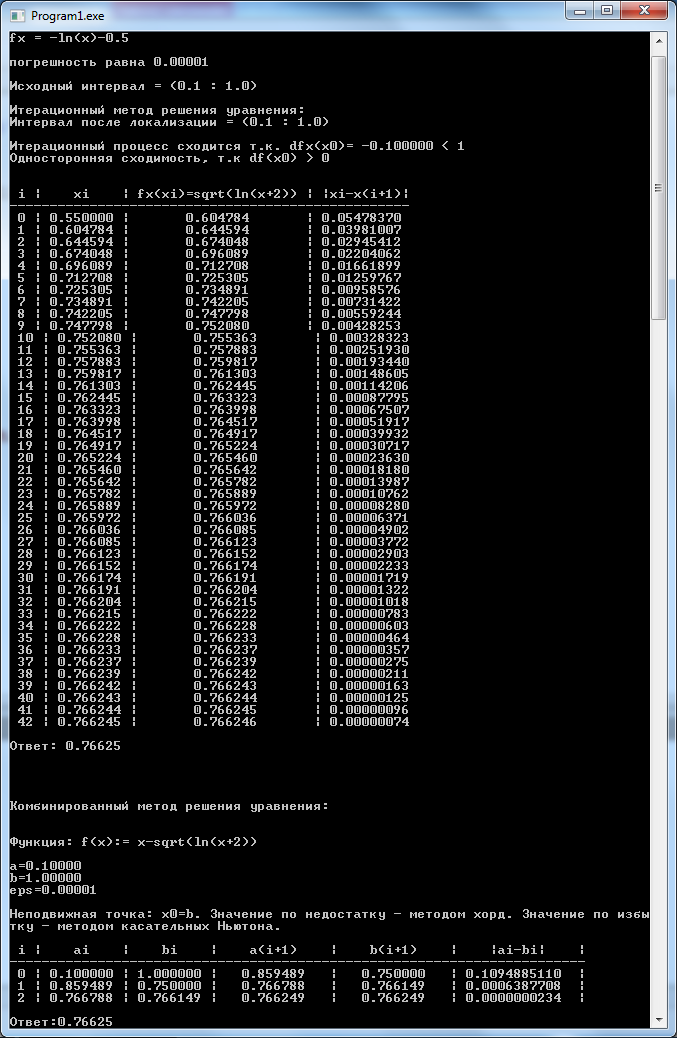
Листинг программы представлен в приложении А. 

Рисунок 4 – скриншот выполнения программы, реализующей уточнение корней способом простых итераций и комбинированным методом

**2.4 Результат проверки выполнения программ**

Для точного построения графика была использована математическая система WolframAlpha. Изображение графика и результаты проверки представлены на рисунке 5.

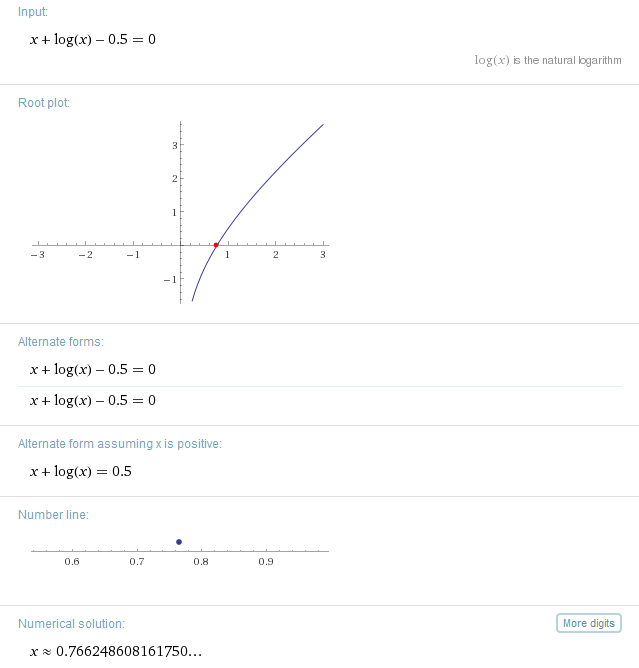


Рисунок 5 – Изображение графика и результаты проверки

Приложение А

Исходный код программы

**program** Lab1;

**uses** crt;

**const**

{Интервал изоляции корня}

a=0.1;

b=1.0;

eps=0.00001; //погрешность

h=eps/10;

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

**begin**

f:= x+ln(x)-0.5;

**end**;

**function** fx(x:real):real; //функция вида x=f(x);

**begin**

fx:=x-0.1\*f(x);

**end**;

**function** df(x:real):real; //производная функции f

**begin**

df:= 1+1/x;

**end**;

**function** dfx(x:real):real; //производная функции fx

**begin**

dfx:= 1-0.1\*df(x);

**end**;

**function** ddfx(x:real):real; //производная второго порядка

**begin**

ddfx:= 1/sqr(x);

**end**;

**function** StaticPoint(x:real):Boolean;

**begin**

**if** f(x)\*ddfx(x)>0 **then** StaticPoint:=true **else** StaticPoint:=false;

**end**;

**procedure** FindRes; //уточнение корня простым итерационным методом

**var** q,x0,x1,a1,b1:real; i:integer;

**begin**

i:=0;

a1:=a; b1:=b;

WriteLn('Итерационный метод решения уравнения:');

WriteLn('Интервал после локализации = (',a1:0:1,' : ',b1:0:1,')'); WriteLn;

**if** (abs(dfx(a1))<1) **then begin**

WriteLn('Итерационный процесс сходится т.к. dfx(x0)= ',dfx(a1):0:6,' < 1');

**if** (abs(dfx(a1))<0) **then begin**

q:=1;

WriteLn('Двусторонняя сходимость, т.к. df(x0) < 0');

WriteLn;

**end**;

**if** (abs(dfx(a1))>0) **then begin**

**if** dfx(a1)>dfx(b1) **then** q:=dfx(a1) **else** q:= dfx(b1);

WriteLn('Односторонняя сходимость, т.к df(x0) > 0');

WriteLn;

**end**;

**end else begin** WriteLn('Итерационный процесс расходится'); **exit**; **end**;

{вывод результатов}

WriteLn;

WriteLn(' i | xi | fx(xi)=sqrt(ln(x+2)) | |xi-x(i+1)|');

WriteLn('--------------------------------------------------');

x1:=(a1+b1)/2;

**repeat**

x0:=x1;

write(' ',i,' | ',x0:0:6);

x1:=fx(x0);

write(' | ',fx(x0):0:6);

WriteLn(' | ',abs(x1-x0):0:8);

i:=i+1;

**until** abs(x1-x0)<(q\*h);

WriteLn;

Writeln('Ответ: ',x1:0:5);

WriteLn; WriteLn;

**end**;

**procedure** CombMeth; //комбинированный метод

**var** a0,a1,b0,b1,static:Real; i:Integer;

**begin**

i:=0;

WriteLn;

WriteLn('Комбинированный метод решения уравнения:');

WriteLn; WriteLn;

WriteLn('Функция: f(x):= x-sqrt(ln(x+2))');

WriteLn;

WriteLn('a=',a:0:5);

WriteLn('b=',b:0:5);

WriteLn('eps=',eps:0:5);

WriteLn;

a1:=a; b1:=b;

**if** StaticPoint(b) **then** WriteLn('Неподвижная точка: x0=b. Значение по недостатку - методом хорд. Значение по избытку - методом касательных Ньютона.')

**else if** StaticPoint(b) **then** WriteLn('Неподвижная точка: x0=a. Значение по недостатку - методом касательных Ньютона. Значение по избытку - методом хорд.')

**else**

**begin**

WriteLn('Ошибка!');

**Exit**;

**end**;

{вывод результатов}

WriteLn;

WriteLn(' i | ai | bi | a(i+1) | b(i+1) | |ai-bi| |');

WriteLn('------------------------------------------------------------------------');

**repeat**

a0:=a1;

b0:=b1;

**if** StaticPoint(b) **then begin**

a1:=a0-(f(a0)\*(b0-a0)/(f(b0)-f(a0)));

b1:=b0-(f(b0)/df(b0));

static:=a1;

**end**;

**if** StaticPoint(a) **then begin**

b1:=b0-(f(b0)\*(a0-b0)/(f(a0)-f(b0)));

a1:=a0-(f(a0)/df(a0));

static:=b1;

**end**;

WriteLn(' ',i,' | ',a0:0:6,' | ',b0:0:6,' | ',a1:0:6,' | ',b1:0:6,' | ',abs(a1-b1):0:10,' |');

i:=i+1;

**until** abs(a1-b1)<h;

WriteLn;

WriteLn('Ответ:', a1:0:5);

**end**;

**var** x:Real; ex:Boolean;

**begin**

WriteLn('f = x+ln(x)-0.5'); WriteLn;

WriteLn('fx = -ln(x)-0.5'); WriteLn;

WriteLn('погрешность равна 0.00001'); WriteLn;

WriteLn('Исходный интервал = (',a:0:1,' : ',b:0:1,')'); WriteLn;

ex:=false;

**if** f(a)\*f(b)<0 **then begin**//как минимум один корень

ex:=true;

FindRes; //уточнение корня методом простых итераций

Readln;

CombMeth;//комбинированный метод

Readln;

**end else begin** //нахождение корня если их четное количество

x:=a;

**while** (f(x)<=0) **or** (x>=b) **do** x:=x+h;

**if** x>=b **then** WriteLn('Корни на данном промежутке отсутствуют') **else** ex:=true;

**if** ex **then begin**

FindRes; //уточнение корня методом простых итераций

Readln;

CombMeth;//комбинированный метод

Readln;

**end**;

**end**;

**end**.