Московский авиационный институт
(национальный исследовательский университет)

Факультет информационных технологий и прикладной математики

Кафедра вычислительной математики и программирования

Лабораторная работа №2
Задание №1 часть II по курсу «Численные методы»
Вариант №12

Студент: С. М. Власова
Преподаватель: И. Э. Иванов
Группа: М8О-306Б
Дата:
Оценка:
Подпись:

Москва, 2020

Задание №2.1

Реализовать метод простой итерации решения нелинейных уравнений в виде программы, задавая в качестве входных данных точность вычислений. С использованием разработанного программного обеспечения найти положительный корень нелинейного уравнения (начальное приближение определить графически). Проанализировать зависимость погрешности вычислений от количества итераций.
Вариант: 12

$3^x - 5 \cdot x^2 + 1 = 0.$

1 Описание метода решения

Метод простой итерации. При использовании метода простой итерации уравнение $f(x) = 0$ заменяется эквивалентным уравнением с выделенным линейным членом

$$x = \phi(x)$$

Решение ищется путем построения последовательности

$$x^{(k+1)} = \phi(x^{(k)}), \qquad k = 0, 1, 2, \ldots$$

начиная с некоторого начального значения $x^{(0)}$. Если $\phi(x)$ — непрерывная функция, а $x^{(k)}$ $(k = 1, 2, \ldots)$ — сходящаяся последовательность, то значение $x^{(\star)} = \lim_{k\to\infty} x^{(k)}$ является решением эквивалентного уравнения.
Условия сходимости метода и оценка его погрешности определяются следующей теоремой.

Теорема 2. 1 *Пусть функция $\phi(x)$ определена и дифференцируема на отрезке $[a, b]$. Тогда, если выполняются условия*

1) $\phi(x) \in [a, b] \quad \forall x \in [a, b]$,

2) $\exists q : |\phi'(x)| \le q < 1 \quad \forall x \in [a, b]$,

то уравнение $x = \phi(x)$ имеет и притом единственный на $[a, b]$ корень $x^{(\star)}$; к этому корню сходится определяемая методом простой итерации последовательность $x^{(k)}$ $(k = 0, 1, 2, \ldots)$, начинающаяся с любого $x^{(0)} \in [a, b]$.
При этом справедливы оценки погрешности $(\forall k \in N)$:

$$|x^{(\star)} - x^{(k+1)}| \le \frac{q}{1-q} \cdot |x^{(k+1)} - x^{(k)}|$$

$$|x^{(\star)} - x^{(k+1)}| \le \frac{q^{k+1}}{1-q} \cdot |x^{(1)} - x^{(0)}|$$

2 Моя задача

Решить уравнение

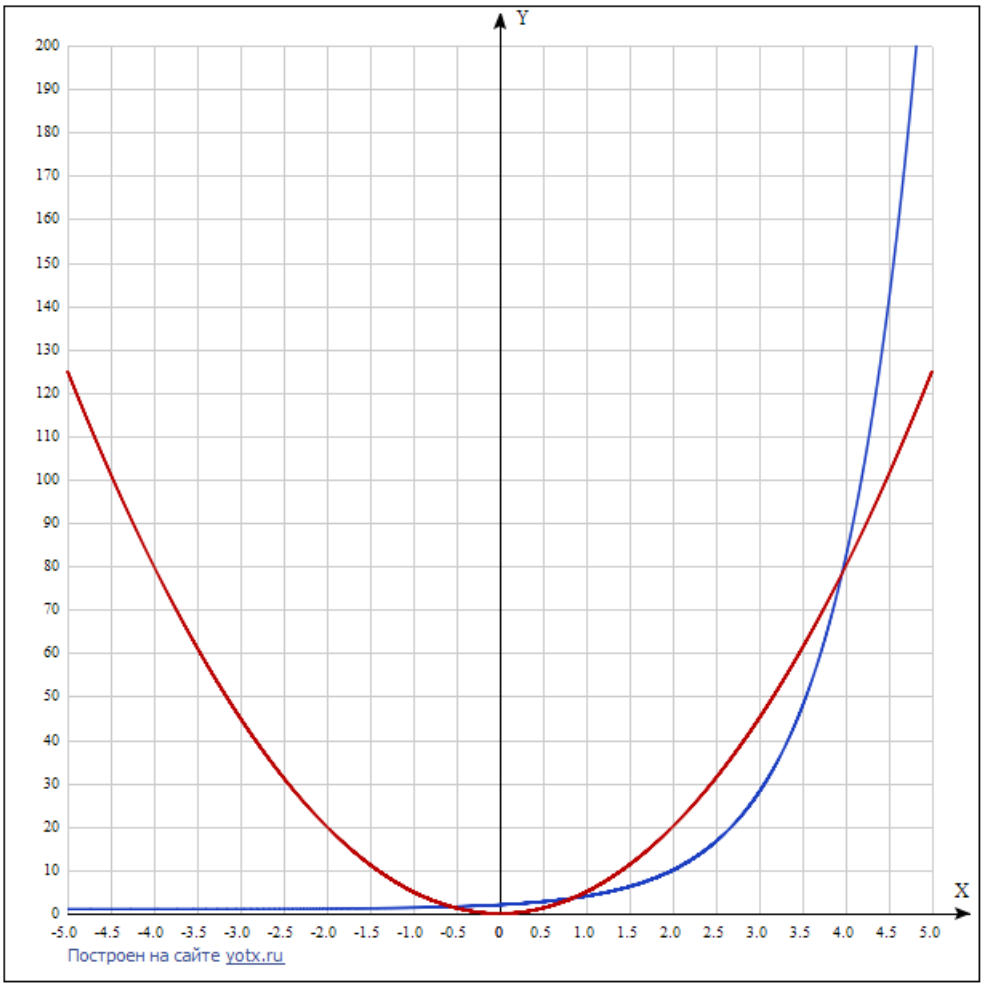
$$f(x) = 3^x - 5 \cdot x^2 + 1 = 0$$

с заданной точностью.

Решение:
Этап I. Локализация: для локализации корней применим графический способ. Преобразуем исходное уравнение к эквивалентному виду:

$$3^x + 1 = 5 \cdot x^2$$

Теперь построим графики функций $f_1(x) = 3^x + 1$ и $f_2(x) = 5 \cdot x^2$.



Графики функций пересекаются в трех точках, соответственно, уравнение имеет три

корня — два положительных и один отрицательный. Т.к. нас интересуют исключительно положительные корни, локализуем их.

$$0.5 < x_1^{(\star)} < 1$$

$$3.5 < x_2^{(\star)} < 4.5$$

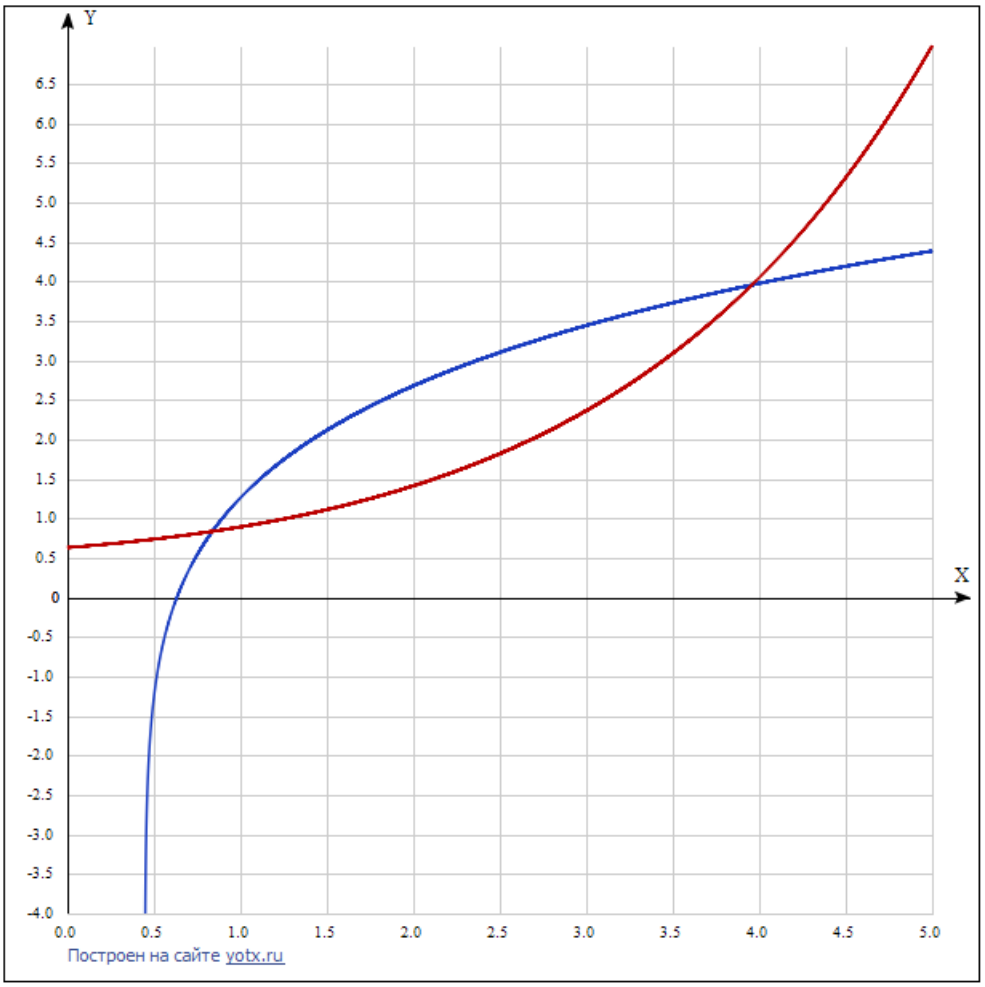
Этап II. Уточнение: для уточнения корней будем использовать метод простой итерации. Уравнение можно записать в виде

$$x = \sqrt{\frac{3^x + 1}{5}}$$

или

$$x = \log_3 (5 \cdot x^2 - 1)$$

Построим соответствующие этим функциям графики.



Определим, какая из этих функций будет приемлемой для первого корня.

1. $\phi(x) \in [0.5, 1] \quad \forall x \in [0.5, 1]$
2. $|\phi'(x)| < q \leq 1 \quad \forall x \in [0.5, 1]$

По рисунку видно, что первое условие теоремы выполняется на интервале $[0.5, 1]$ для функции, отмеченной красным — $\phi_1(x) = \sqrt{\frac{3^x+1}{5}}$.
Для этой же функции $\forall x \in [0.5, 1] \quad \phi_1'(x) = \frac{3^x}{2 \cdot \sqrt{(3^x+1) \cdot 5}} < q \leq 1$ и $x_1^{(0)} = \frac{1+0.5}{2} = 0.75$.

Аналогично, со вторым корнем. Для него условия теоремы выполняются на интервале $[3.5, 4.5]$ для функции, отмеченной синим — $\phi_2(x) = \log_3(5 \cdot x^2 - 1)$.
Для этой же функции $\forall x \in [3.5, 4.5] \quad \phi_2'(x) = \frac{10 \cdot x}{(5 \cdot x^2 - 1) \cdot \log(3)} < q \leq 1$ и $x_2^{(0)} = \frac{3.5+4.5}{2} = 4$.

Таким образом, мы подобрали эквивалентные функции для обоих корней, при которых условия сходимости метода простой итерации выполняются.

3 Протокол

Входные данные через командную строку я задаю значение точности ε.
Выходные данные я вывожу на экран.
Скриншот консоли:
$\varepsilon = 0.1$.

```
(base) vlasochka@vlasochka-VPCSB11FX:~/Документы/ЧМ/Lab2$ g++ Iter2.1.cpp -o iter2.1
(base) vlasochka@vlasochka-VPCSB11FX:~/Документы/ЧМ/Lab2$ ./iter2.1
Введите точность вычислений: 0.1
q = 0.368486
1-й положительный корень нелинейного уравнения, найденный м-м итераций: 0.809877
Число итераций: 1
q = 0.52877
2-й положительный корень нелинейного уравнения, найденный м-м итераций: 3.97724
Число итераций: 1
(base) vlasochka@vlasochka-VPCSB11FX:~/Документы/ЧМ/Lab2$
```

$\varepsilon = 0.01$.

```
(base) vlasochka@vlasochka-VPCSB11FX:~/Документы/ЧМ/Lab2$ ./iter2.1
Введите точность вычислений: 0.01
q = 0.368486
1-й положительный корень нелинейного уравнения, найденный м-м итераций: 0.83494
Число итераций: 3
q = 0.52877
2-й положительный корень нелинейного уравнения, найденный м-м итераций: 3.96184
Число итераций: 3
(base) vlasochka@vlasochka-VPCSB11FX:~/Документы/ЧМ/Lab2$
```

$\varepsilon = 0.0001$.

```
(base) vlasochka@vlasochka-VPCSB11FX:~/Документы/ЧМ/Lab2$ ./iter2.1
Введите точность вычислений: 0.0001
q = 0.368486
1-й положительный корень нелинейного уравнения, найденный м-м итераций: 0.837906
Число итераций: 7
q = 0.52877
2-й положительный корень нелинейного уравнения, найденный м-м итераций: 3.95763
Число итераций: 9
(base) vlasochka@vlasochka-VPCSB11FX:~/Документы/ЧМ/Lab2$
```

Можно заметить, что сходимость метода простой итерации немного хуже сходимости метода Ньютона. При $\varepsilon = 0.0001$ положительные корни исходного уравнения следующие

$$x_1^{(\star)} \approx 0.837906$$

$$x_2^{(\star)} \approx 3.95763$$

Алгоритм находит их за 7 и 9 итераций.

4 Исходный код

Листинг 1: Метод простой итерации

```
#include<iostream>
#include<map>
#include<math.h>
#include<vector>

double Phi1(double x)
{
    double val = pow((pow(3,x) + 1)/5, 0.5);
    return val;
}

double DPhi1(double x)
{
    double val = pow(3,x)*log(3)/(2*pow(5,0.5)*pow( pow(3,x) + 1, 0.5 ));
    return val;
}

double Phi2(double x)
{
    double val = log(5*pow(x,2)-1)/log(3);
    return val;
}

double DPhi2(double x)
{
    double val = 10*x/( (5*pow(x,2) - 1)*log(3));
    return val;
}

int InitPhi(std::pair<double, double>& p, double eps)
{
    int func_id = 0;
    while(func_id == 0)
    {
        if( Phi1(p.first) >= p.first && Phi1(p.first) <= p.second && Phi1(p.second) >= p.first
            && Phi1(p.second) <= p.second )
            func_id = 1;
        else if( Phi2(p.first) >= p.first && Phi2(p.first) <= p.second && Phi2(p.second) >= p.
            first && Phi2(p.second) <= p.second )
```

```
38              func_id = 2;
39          else
40          {
41              p.first += eps;
42              p.second -= eps;
43          }
44      }
45      return func_id;
46  }
47
48  int IterMethod(std::vector<double>& X, double q, double eps, int m)
49  {
50      int it_count = 0;
51      double kaf = q/(1-q);
52      do
53      {
54          X[0] = X[1];
55          if(m == 1)
56              X[1] = Phi1(X[0]);
57          else
58              X[1] = Phi2(X[0]);
59          it_count++;
60      }
61      while( kaf*fabs(X[1]-X[0]) >= eps);
62      return it_count;
63  }
64
65  int main()
66  {
67      double eps, q;
68      int it_count;
69      int i = 0;
70
71      std::vector<double> X(2);
72      std::vector<std::pair<double, double>> ab(2);
73      ab[0] = std::make_pair(0.5, 1);
74      ab[1] = std::make_pair(3.5, 4.5);
75
76      std::cin >> eps;
77      for(int i = 0; i < 2; i++)
78      {
79          it_count = 0;
80          if( InitPhi(ab[i], eps) == 1)
81          {
```

```
82              q = std::max(fabs(DPhi1(ab[i].first)), fabs(DPhi1(ab[i].second)));
83
84              while(q >= 1)
85              {
86                  ab[i].first += eps;
87                  ab[i].second -= eps;
88                  q = std::max(fabs(DPhi1(ab[i].first)), fabs(DPhi1(ab[i].second)));
89              }
90              X[1] = (ab[i].first + ab[i].second)/2;
91              it_count = IterMethod(X, q, eps, 1);
92          }
93          else
94          {
95              q = std::max(fabs(DPhi2(ab[i].first)), fabs(DPhi2(ab[i].second)));
96
97              while(q >= 1)
98              {
99                  ab[i].first += eps;
100                 ab[i].second -= eps;
101                 q = std::max(fabs(DPhi2(ab[i].first)), fabs(DPhi2(ab[i].second)));
102             }
103             X[1] = (ab[i].first + ab[i].second)/2;
104             it_count = IterMethod(X, q, eps, 2);
105         }
106         std::cout << "q = " << q << std::endl;
107         std::cout << i+1 << "-st(nd) positive solution of a nonlinear equation found by method
                of iterations: " << X[1] << std::endl;
108         std::cout << "The number of iterations: " << it_count << std::endl;
109     }
110     return 0;
111 }
```

5 Выводы

Выполнив вторую часть первого задания второй лабораторной работы, я познакомилась с методом простой итерации, который решает задачу уточнения корней нелинейных уравнений на заданном интервале с заданной точностью. Этот метод является итерационным и базируется на последовательном приближении, согласно итерационной формуле. Функция представляется в эквивалентном виде таким образом, чтобы метод сходился. Для этого необходимо, чтобы выполнялся ряд условий. Одно из этих условий — непрерывность эквивалентной функции на заданном интервале. Ее область определения и область значений должна ограничиваться этим интервалом. Также для сходимости метода необходимо, чтобы производная этой функции по модулю не превосходила 1. Если эти условия выполняются, метод строит последовательность точек, пределом которых является решение исходного уравнения.