TRANSPORTA UN SAKARU INSTITŪTS



**Лабораторная работа №3:** «Методы решения нелинейного уравнения»

Выполнили: Михаил Красильников

Марк Беляев

Группа: 4601BV

Рига 2020

[**Содержание задания**](#_jvz7223kdjiv) **3**

[**Исходный код метод бисекции**](#_9c6ovmh67xas) **4**

[**Исходный код метод простых итераций**](#_u15ve5alxp4d) **5**

[**Результаты работы алгоритма**](#_fz3gox7curk1) **7**

[**Выводы**](#_6y5gdg22fcxp) **9**

# 

# Содержание задания

В данной лабораторной работе требуется реализовать два алгоритма решения нелинейного уравнения: метод бисекции и индивидуальный метод. Предусмотрено два способа реализации алгоритмов: с использованием скользящего окна или же с локально заданным интервалом. В зависимости от выбора реализации, будут меняться входные параметры метода. Во всех обоих случаях в качестве входного параметра требуется установить ввод точности решения (эпсилон).

# Исходный код метод бисекции

def f(x):

return x \* (2 \*\* (3 \* x))

iteration = 0

a = 1

b = 4

epsilon = 0.01

while abs(b - a) > epsilon:

iteration += 1

c = (a + b) / 2

if (f(a) \* f(c) > 0):

a = c

else:

b = c

print(str(iteration) + ' ' + str(abs(b-a)) + ' ' + str(c))

# Исходный код метод простых итераций

class Program

{

static void Main(string[] args)

{

string input;

double PreviousX;

int iterations = 0;

Console.WriteLine("Enter initial approximation");

input = Console.ReadLine();

double CurrentX = Convert.ToInt32(input);

Console.WriteLine($"X0 = {CurrentX}");

double dX = double.MaxValue;

while (Math.Abs(dX) > 1e-2)

{

PreviousX = CurrentX;

CurrentX = F2(PreviousX);

dX = CurrentX - PreviousX;

iterations++;

}

Console.WriteLine($"Iterations = {iterations}");

Console.WriteLine($"X = {CurrentX}");

}

static double F (double x)

{

return (x \* x + 4) / 5;

}

static double F2 (double x)

{

return 4 / (Math.Pow(Math.E, x / 2));

}

}

}

# Результаты работы алгоритма

Таблица с результатами (метод бисекции)

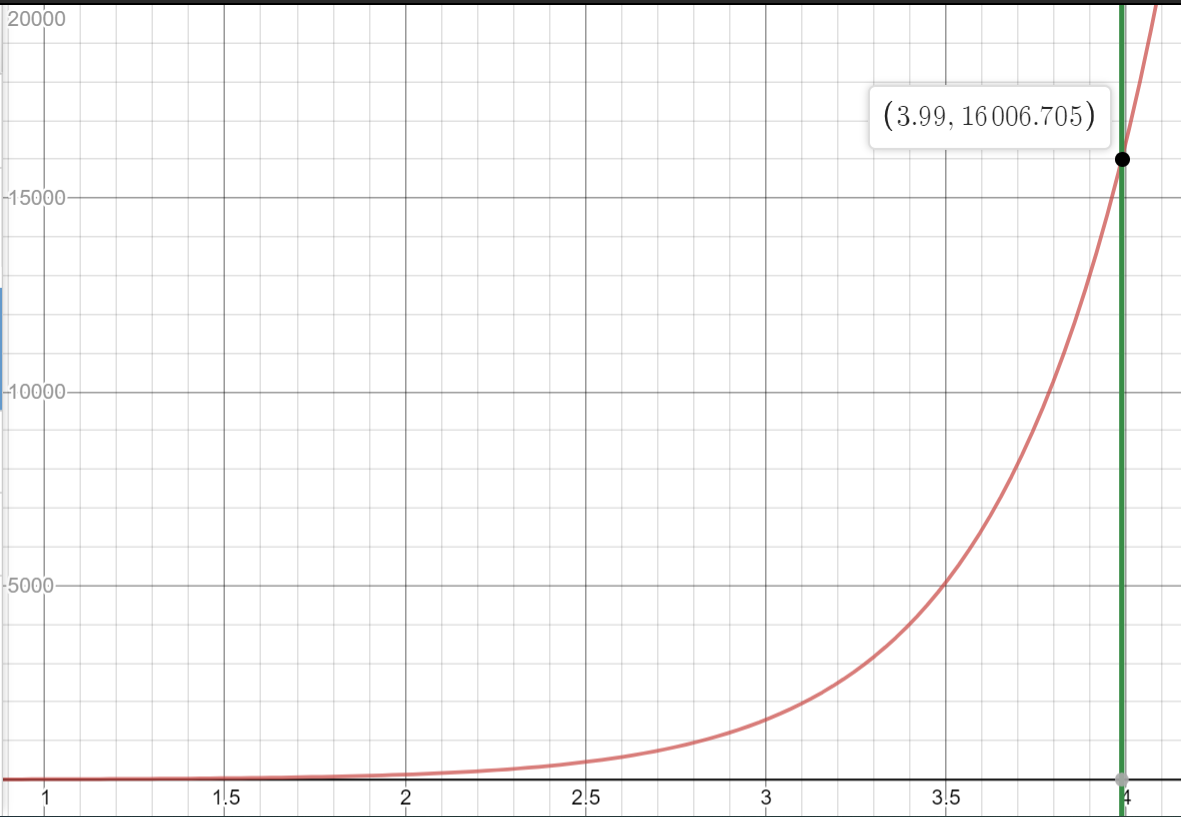
|  |  |  |  |
| --- | --- | --- | --- |
| **Номер функции** | **Точности** | **Найденнный корень** | **Число итераций** |
| **7** | **ε=0.01** | **x=3.9941** | **n=9** |
| **ε=0.0001** | **x=3.9999** | **n=15** |

Таблица с результатами (метод простых итераций)

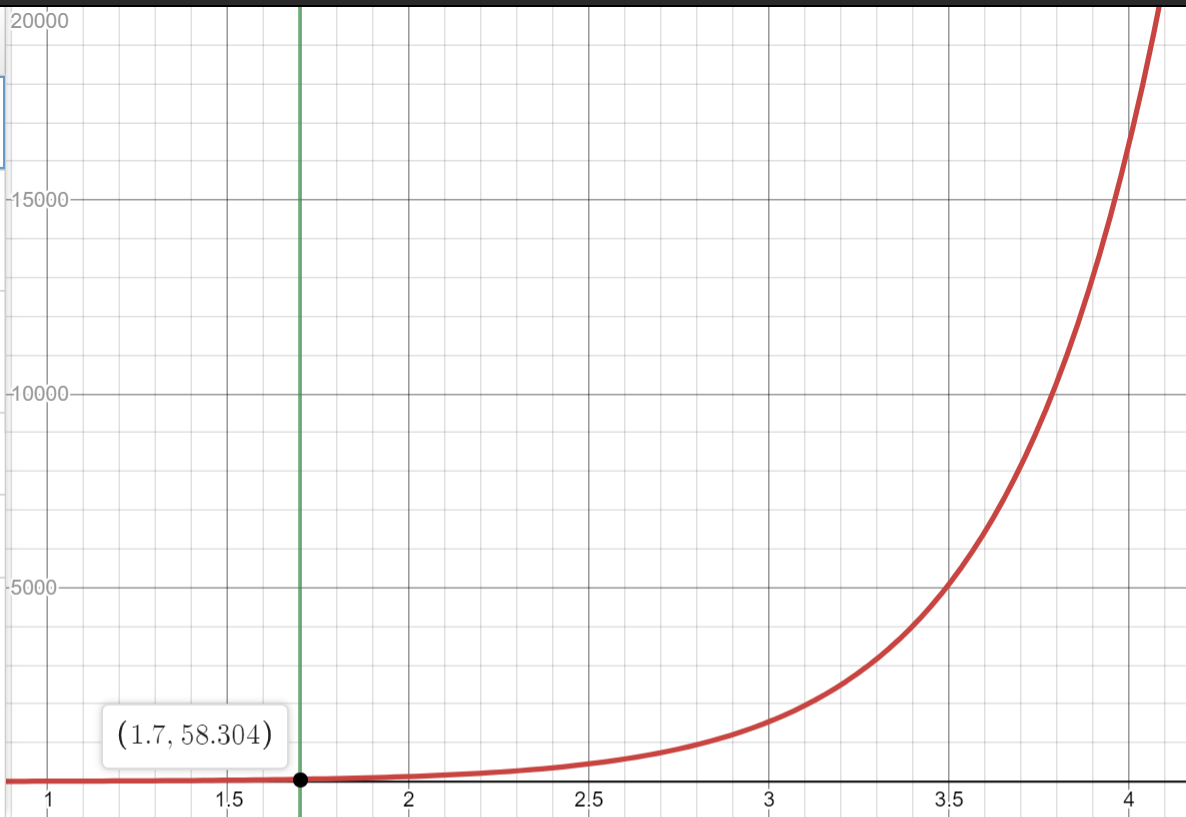
|  |  |  |  |
| --- | --- | --- | --- |
| **Номер функции** | **Точности** | **Найденнный корень** | **Число итераций** |
| **7** | **ε=0.01** | **x=1.7006** | **n=37** |
| **ε=0.0001** | **x=1.7052** | **n=66** |

График функции на заданном участке

Метод бисекции



Метод простых итераций



# Выводы

В методе бисекций требуется большее число вхождений для поиска корня, чем в методе простых итераций. Самая низкая скорость вычислений по сравнению с другими методами решения нелинейных уравнений является главным недостатком метода деления отрезка пополам. Метод бисекции – это метод с линейной сходимостью (порядок скорости сходимости р=1). Он абсолютно устойчив и позволяет оценить время расчета. За счет того,что метод относится к интервальным,остановка поиска произойдет, когда модуль разности концов интервала станет меньше заданной точности.

Сравнение двух алгоритмов: