**ДОМАШНЯЯ РАБОТА №1**

«Численные методы решения нелинейных уравнений»

**Выполнил:**

студент 2 курса 13 группы кафедры ТП.

Петров Андрей Александрович

**Вариант 8.**

**Задание:** Дана функция 𝑓(𝑥) = 𝑐𝑜𝑠(𝑛𝑥 − (2𝑛 + 1)) + 𝑛𝑥, где 𝒏– номер студента в списке подгруппы.

Необходимо выполнить следующее:

• Отделить все корни уравнения 𝑓(𝑥) = 0.

• Сузить отрезки отделённости корней до размера 10−2 с помощью метода бисекций.

• Решить с точностью 𝜀 = 10−5 указанное уравнение методом простых итераций.

• Решить с точностью 𝜀 = 10−5 указанное уравнение методом Ньютона.

Сравнить полученные в последних двух пунктах результаты.

**Ход работы:**

𝑓(𝑥) = 𝑐𝑜𝑠(8𝑥 – 17) + 8𝑥

**Отделим все корни уравнения 𝑓(𝑥) = 0.**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| x | -∞ | -2 | -1 | 0 | 1 | 2 | +∞ |
| y | - | - | - | - | + | + | + |

Проанализировав функцию, можно сделать вывод, что на промежутке [0; 1] возможно имеется хотя бы один корень.

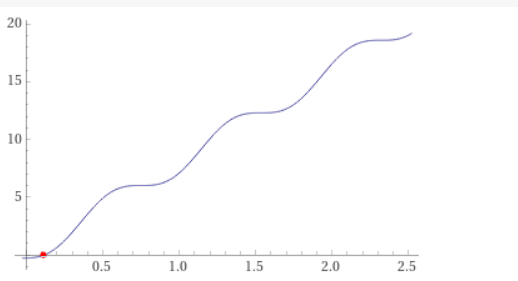
Рассмотрим [0; 1] и возьмем производную от нашей функции.

𝑓’(𝑥) = -8 \* sin(8𝑥 – 17) + 8

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| x | 0 | 0.25 | 0.5 | 0.75 | 1 |
| y' | + | + | + | + | + |

Так как наша производная 𝑓’(x) на промежутке [0; 1] принимает положительные значения, то функция 𝑓(𝑥) на промежутке [0; 1] постоянно возрастает, что позволяем нам сделать вывод о наличии только одного корня.

График функции 𝑓(𝑥) = 𝑐𝑜𝑠(8𝑥 – 17) + 8𝑥 (из Wolfram Alpha).



**Сузить отрезки отделённости корней до размера 10−2 с помощью метода бисекций**

Напишем программу, которая на будет сужать промежуток [0,1] до отрезка отдалённости корней размером 10-2, на котором находится наш корень.

let ***a*** = 0.0;  
let ***b*** = 1.0;  
  
***console***.log(`Start segment: [a = ${***a***}, b = ${***b***}]`)  
  
while (***Math***.abs(***a*** - ***b***) > 2e-2) {  
 let x = (***a*** + ***b***) / 2;  
 ***console***.log(`\tSegment: [a = ${***a***}, b = ${***b***}]`);  
 (f(x) \* f(***a***)) < 0 ? ***b*** = x : ***a*** = x;  
}  
  
***console***.log(`Final segment: [a = ${***a***}, b = ${***b***}]`)

Результат работы программы:

Изображение выглядит как текст

Автоматически созданное описание

**Решить с точностью 𝜀 = 10−5 указанное уравнение методом простых итераций.**

Выразим x из функции 𝑓(𝑥) = 𝑐𝑜𝑠(8𝑥 – 17) + 8𝑥

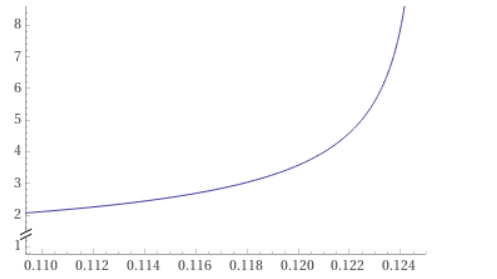
𝑓(x) = 0 ⇒ x = φ(x),   
 Выразим х двумя способами:

φ1(x) = -cos(8x - 17) / 8  
 φ2(x) = (arccos(-8x) + 17) / 8

Докажем, что производные этих функций на промежутке [0,1] меньше 1 (условие сходимости).

φ'1(x) = -1 ≤ sin(8x - 17) ≤ 1  
 φ'2(x) = 1 /

График φ'2(x) = 1 / на промежутке [0.109375; 0.125]



Исходя из графика, полученного в Wolfram Alpha, можно предположить, что при φ2(x) метод простых итераций не сойдется.

Напишем программу, которая реализует метод простых итераций.

let phi = (x) => -***Math***.cos(8.0 \* x - 17.0) / 8.0;  
let ***xCurrent*** = (0.109375 + 0.125) / 2;  
let ***xPrevious*** = 0.109375;  
let ***iteration*** = 0;  
  
while (***Math***.abs(***xCurrent*** - ***xPrevious***) > 1e-5) {  
 ***xPrevious*** = ***xCurrent***;  
 ***xCurrent*** = phi(***xCurrent***);  
 ***iteration***++;  
}  
***console***.log(`Count iteration: ${***iteration***}; x = ${***xCurrent***}`)

Результат выполнения программы:  


**Решить с точностью 𝜀 = 10−5 указанное уравнение методом Ньютона.**

Напишем программу, которая реализует решение уравнения методом Ньютона:

Код программы:

let f = (x) => ***Math***.cos(8 \* x - 17) + 8 \* x;  
let fDerivative = (x) => -8 \* ***Math***.sin(8 \* x - 17) + 8;  
  
let xCurrent = (0.109375 + 0.125) / 2;  
let xPrevious = 0.109375;  
let iteration = 0;  
  
while (***Math***.abs(xCurrent - xPrevious) > 1e-5) {  
 iteration++;  
 xPrevious = xCurrent;  
 xCurrent -= f(xPrevious) / fDerivative(xPrevious);  
}  
***console***.log(`Count iteration: ${iteration}; x = ${xCurrent}`)

Результат выполнения программы:



**Сравнить полученные в последних двух пунктах результаты.**

|  |  |
| --- | --- |
| Решение Wolfram Alpha: | x ≈ 0.11724634095943301740 |
| Метод простых итераций: | x ≈ 0.1172438862436553 |
| Метод Ньютона: | x ≈ 0.11724634095943533 |

Исходя из полученных результатов работы методов Ньютона и простых итераций, реализованных мной, а также результата, полученного с помощью Wolfram Alpha, можно сделать вывод, что метод Ньютона точнее метода простых итераций и выполняет программу за меньшее количество итераций.