БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Факультет прикладной математики и информатики

Лабораторная работа №1

**Численное решение нелинейных уравнений**

**Выполнил:**

Соломевич Александр

2 курс 6 группа

**Преподаватель:**

Горбачева Ю.Н.

Минск, 2022

**Содержание**

Постановка задачи……………………………………………………………………………....3

Краткие теоретические сведения…………………………………………………………………………....4

Листинг программы……………………………………………….…………………….…..5

Результаты….……………………………………….….……………………….…7

Выводы…..……………………………..………………………………………...10

**Постановка задачи**

Написать программу, которая находит решение уравнения *f* (*x*) = 0 c точностью ε = 10-7 методами простой итерации, Ньютона,

Чебышева третьего порядка. Корень отделить сначала графически, затем с

помощью метода половинного деления с точностью ε = 0.1. Провести сравнительный анализ полученных результатов.

**Краткие теоретические сведения**

***Метод половинного деления:***

*x\** ∈ Δ = [a; b]

*a0 = a, b0 = b*

*k = 0,1,…*

*while bk – ak > 2ε:*

*xk = (ak + bk) / 2*

*if f(xk)f(ak) < 0: bk = xk  else:ak = xk*

***Метод простой итерации:***

*f(x) = 0, x\** ∈ Δ = [a; b]

*x = φ(x)*

*xk+1 = φ(x), k = 0,1,…*

*x0*

***Метод Ньютона:***

*f(x) = 0, x\** ∈ Δ = [a; b]

*f(x\*) = 0, f’(x\*) ≠ 0, x\* - простой*

*xk+1 = xk - f(xk) / f’(xk), k = 0,1,…*

*x0*

***Метод Чебышева 3-го порядка:***

*f(x) = 0, x\** ∈ Δ = [a; b]

*φ’(x) = 0, φ’’(x) = 0*

*xk+1 = xk - f(xk) / f’(xk) – (f’’(xk) \* f2(xk)) / 2( f’(xk2))3, k = 0,1,…*

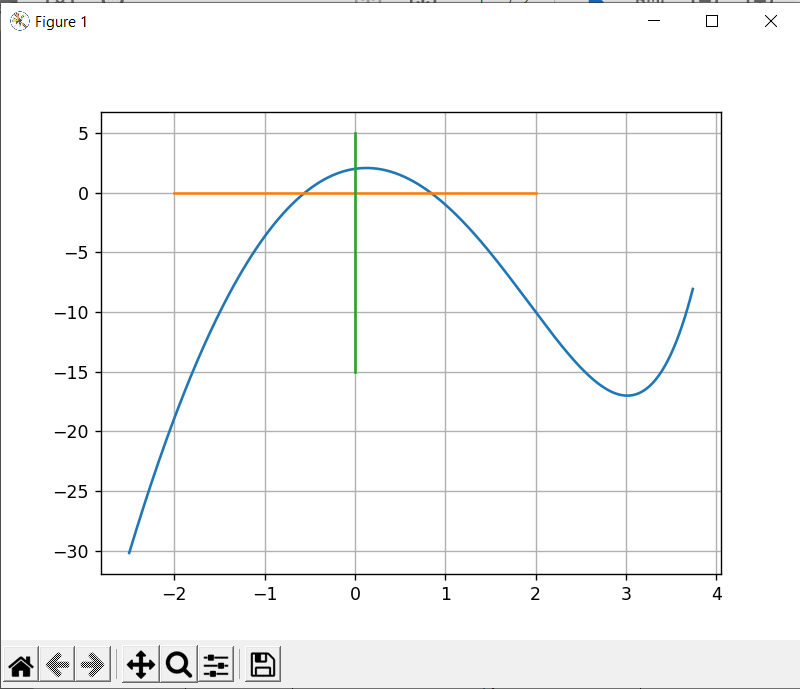
*x0*

**Листинг программы**

**main.py**

import numpy as np  
#import matplotlib.pyplot as plt  
  
e = 10 \*\* -7  
  
  
# x = np.arange(-2.5, 3.75, 0.01)  
# plt.plot(x, (3 \*\* x) - 5 \* (x \*\* 2) + 1, [-2, 2], [0, 0], [0, 0], [-15, 5])  
# plt.grid(True)  
# plt.show()  
  
def f(x):  
 return 3 \*\* x - 5 \* x \*\* 2 + 1  
  
  
a = -1  
b = 0  
k = 0  
  
while abs(b - a) > 0.2:  
 x = (a + b) / 2  
 if f(x) \* f(a) < 0:  
 b = x  
 else:  
 a = x  
 print("k = ", k)  
 print("a = ", a, " b = ", b)  
 print("f(a) = ", f(a), " f(b) = ", f(b))  
 print("(a + b) / 2 = ", (a + b) / 2)  
 print("b - a = ", b - a, '\n')  
 k += 1  
  
  
def phi(x):  
 return - np.sqrt((3 \*\* x + 1) / 5)  
  
  
print("MPI: ")  
xZero = -0.5625  
accuracy = abs(phi(xZero) - xZero)  
x = xZero  
k = 0  
while accuracy >= e:  
 temp = x  
 x = phi(x)  
 accuracy = abs(x - temp)  
 print("k = ", k)  
 print("x = ", x)  
 print("x[k] - x[k - 1] = ", accuracy)  
 k += 1  
  
  
def df(x):  
 return np.log(3) \* 3 \*\* x - 10 \* x  
  
  
print("\nNewton Method: ")  
xZero = -0.5625  
accuracy = abs(f(xZero) / df(xZero))  
x = xZero  
k = 0  
  
while accuracy >= e:  
 temp = x  
 x = x - f(x) / df(x)  
 accuracy = abs(x - temp)  
 print("k = ", k)  
 print("x = ", x)  
 print("x[k] - x[k - 1] = ", accuracy)  
 k += 1  
  
def d2f(x):  
 return np.log(3) \*\* 2 \* 3 \*\* x - 10  
  
  
print("\nСhebyshev Third Order Method: ")  
xZero = -0.5625  
accuracy = abs(f(xZero) / df(xZero) + (d2f(xZero) \* f(xZero) \*\* 2) / (2 \* df(xZero \*\* 2) \*\* 3))  
x = xZero  
k = 0  
  
while accuracy >= e:  
 temp = x  
 x = x - f(x) / df(x) - (d2f(x) \* f(x) \*\* 2) / (2 \* df(x \*\* 2) \*\* 3)  
 accuracy = abs(x - temp)  
 print("k = ", k)  
 print("x = ", x)  
 print("x[k] - x[k - 1] = ", accuracy)  
 k += 1

**Результаты:**

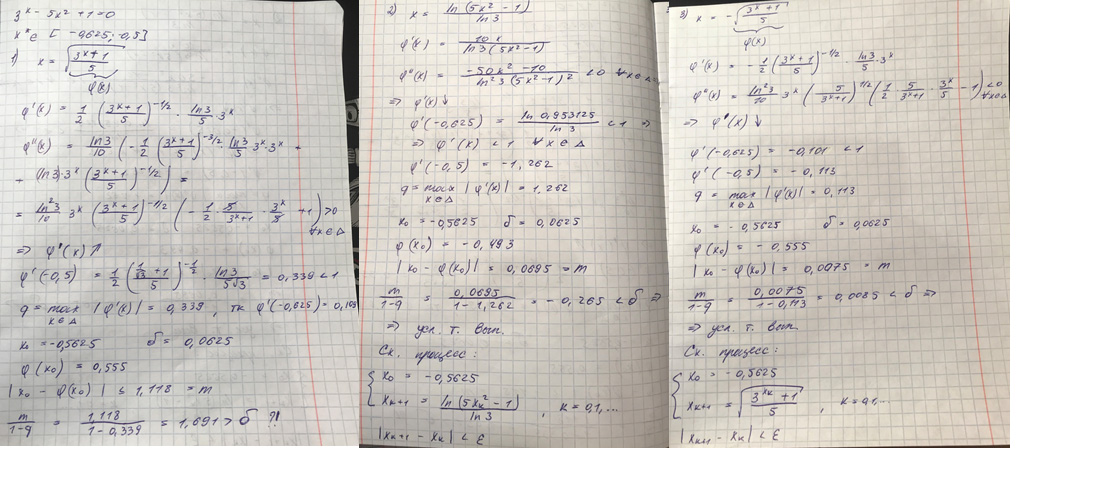


Исходя из графика в качестве изначального отрезка можем выбрать [-1; 0].

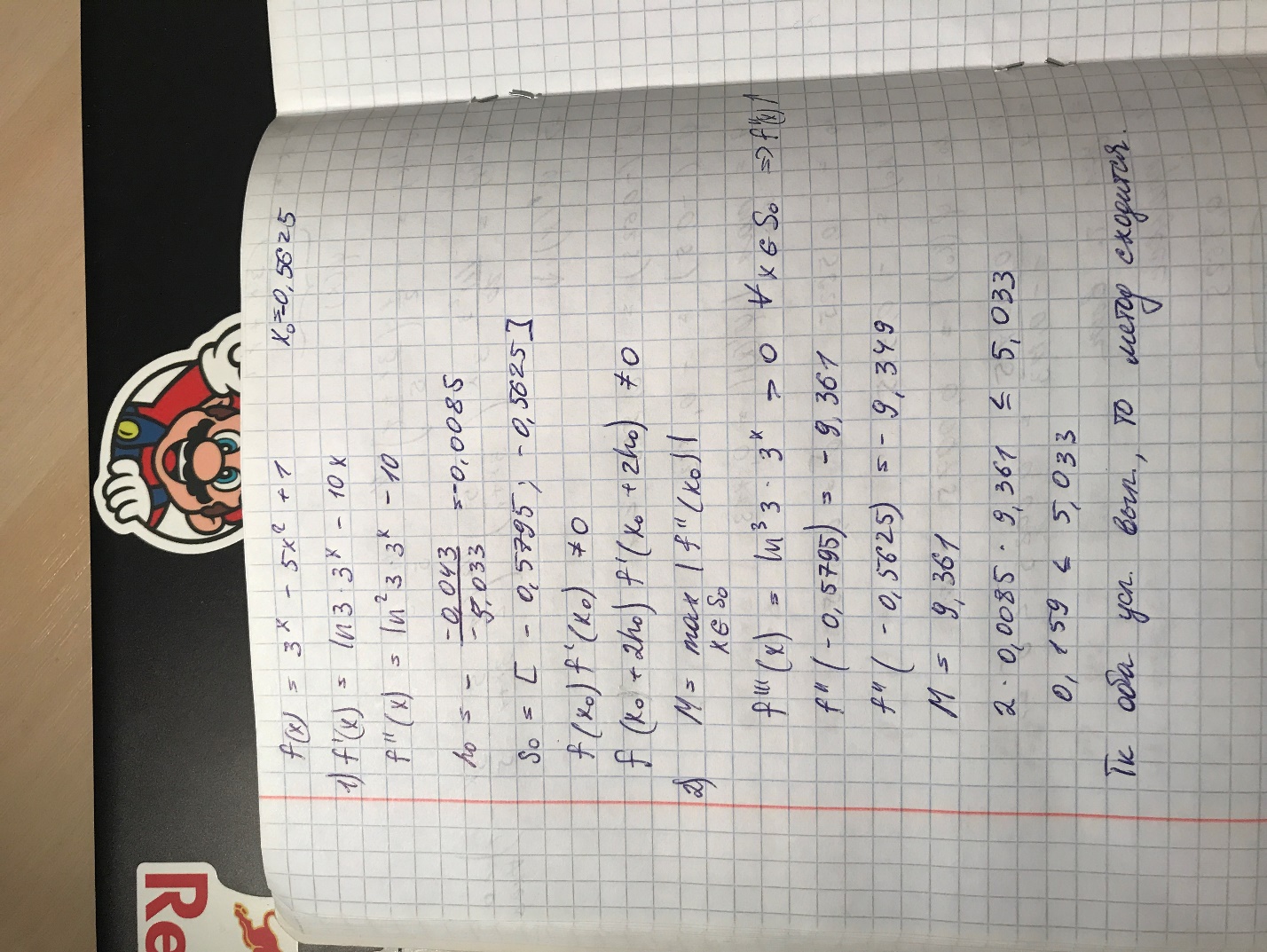
Метод половинного деления:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *k* | *ak* | *bk* | *f(ak)* | *f(bk)* | *(ak + bk) / 2* | *bk - ak* |
| *0* | *-1* | *-0.5* | *-3.(6)* | *0.327* | *-0.75* | *0.5* |
| *1* | *-0.75* | *-0.5* | *-1.374* | *0.327* | *-0.625* | *0.25* |
| *2* | *-0.625* | *-0.5* | *-0.45* | *0.327* | *-0.5625* | *0.125* |

Проверка условий теоремы о сходимости метода простой итерации:

****

Проверка условий теоремы о сходимости метода Ньютона:



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *Номер итерации k* | *Метод простой итерации* | | *Метод Ньютона* | | *Метод Чебышева 3-го порядка* | |
| *xk* | |*xk – xk-1*| | *xk* | |*xk – xk-1*| | *xk* | |*xk – xk-1*| |
| *0* | *-0.554804089875* | *0.007695910124551* | *-0.5555847643322* | *0.006915235667765* | *-0.5576599881744* | *0.004840011825575* |
| *1* | *-0.5556284154958* | *0.0008243256203570* | *-0.5555484368110* | *3.632752115612092e-05* | *-0.5557586855107* | *0.001901302663679* |
| *2* | *-0.5555398452279* | *8.8570267894994e-05* | *-0.5555484358088* | *1.0022198626558065e-09* | *-0.5555505847305* | *0.0002081007801638* |
| *3* | *-0.555549358556* | *9.513328241839147e-06* |  |  | *-0.5555484360341* | *2.148696481851786e-06* |
| *4* | *-0.5555483366934* | *1.021862697125897e-06* |  |  | *-0.5555484358088* | *2.2523993781220497e-10* |
| *5* | *-0.5555484464551* | *1.0976172926113748e-07* |  |  |  |  |
| *6* | *-0.5555484346653* | *1.1789883513912969e-08* |  |  |  |  |

**Выводы:**

Наиболее быстрая сходимость наблюдалась при использовании метода Ньютона, который сошелся за 3 итерации при заданной точности. Методу Чебышева 3-го порядка понадобилось 5 итераций для того, чтобы сойтись. Метод простой итерации оказался самым медленным и сошелся за 7 итераций.