Министерство образования и науки РФ

Федеральное государственное автономное

образовательное учреждение высшего образования

«Национальный исследовательский университет ИТМО»

**факультет программной инженерии и компьютерной техники**

**ЛАБОРАТОРНАЯ РАБОТА №2**

по дисциплине

‘ВЫЧИСЛИТЕЛЬНАЯ МАТЕМАТИКА’

Вариант №1

*Выполнил:*

Студент группы P3207

Алферов Г. А.

*Преподаватель:*

Рыбаков С.Д.



Санкт-Петербург, 2025

**Цель работы**

Изучить численные методы решения нелинейных уравнений и реализовать три из них средствами программирования. Понять их сходства и различия.

**Ход работы**

**Часть 1.**

Графическое отделение корней:

Изображение выглядит как линия, текст, График, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

Интервалы изоляции корней:

Приближенные значения корней:

X ≈ -1.9, X ≈ -0.3, X ≈ 2.9

Интервалы на оси X:

(-∞, -1.9),(-1.9,-0,3),(-0,3,2.9),(2.9,+∞)

Вычисление знака функции на интервалах:

1 интервал: x = -2 f(-2) = -2.8 Знак: -

2 интервал: x = -1 f(-1) = 6.89 Знак: +

3 интервал: x = 0 f(0) = -3.72 Знак: -

4 интервал: x = 3 f(3) = 7.05 Знак: +

(-2,-1)(-1, 0) (2, 3)

Уточнение корней:

X1 ≈ -1.88

X2 ≈ -0.25

X3 ≈ 2.84

Правый корень – **метод половинного деления**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № шага | a | b | x | f(a) | f(b) | f(x) | |a-b| |
| 1 | 2.000 | 3.000 | 2.500 | -20.079 | 7.050 | -11.170 | 1.000 |
| 2 | 2.500 | 3.000 | 2.750 | -11.170 | 7.050 | -3.352 | 0.500 |
| 3 | 2.750 | 3.000 | 2.875 | -3.352 | 7.050 | 1.509 | 0.250 |
| 4 | 2.750 | 2.875 | 2.812 | -3.352 | 1.509 | -1.004 | 0.125 |
| 6 | 2.812 | 2.875 | 2.843 | -1.023 | 1.509 | 0.222 | 0.063 |
| 7 | 2.812 | 2.843 | 2.827 | -1.023 | 0.202 | -0.416 | 0.031 |
| 8 | 2.827 | 2.843 | 2.835 | -0.436 | 0.202 | -0.118 | 0.016 |
| 9 | 2.835 | 2.843 | 2.839 | -0.118 | 0.202 | 0.041 | 0.008 |

Изображение выглядит как Шрифт, текст, Графика, логотип

Контент, сгенерированный ИИ, может содержать ошибки.

Центральный корень – Метод простой итерации

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № итерации | Xk | Xk+1 | f(Xk+1) | | Xk+1- Xk | |
| 1 | -1.000 | -0.549 | 3.634 | 0.451 |
| 2 | -0.549 | -0.311 | 0.765 | 0.238 |
| 3 | -0.311 | -0.261 | 0.089 | 0.050 |
| 4 | -0.261 | **-0.255** | 0.009 | **0.005** |

Проверка сходимости:

Отрезок [-1, 0]

f ’(x) = 8.22x2 - 3.86x - 15.28

f ‘(a) = -3.2 < 0, f ‘(b) = -15.28 < 0

max(|f ‘(a)|, |f ‘(b)|) = 15.28

λ = 1/15.28

0.790

0

*Итерационная последовательность сходится*

Рабочая формула:

Изображение выглядит как Шрифт, текст, логотип, Графика

Контент, сгенерированный ИИ, может содержать ошибки.

Левый корень – метод секущих

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № итерации | Xk-1 | Xk | Xk+1 | f(Xk+1) | | Xk+1- Xk | |
| 1 | -2.000 | -1.000 | -1.711 | 3.049 | 0.711 |
| 2 | -1.000 | -1.711 | -2.275 | -11.227 | 0.564 |
| 3 | -1.711 | -2.275 | -1.831 | 0.956 | 0.443 |
| 4 | -2.275 | -1.831 | -1.866 | 0.264 | 0.035 |
| 5 | -1.831 | -1.866 | -1.879 | -0.011 | 0.013 |
| 6 | -1.866 | -1.879 | **-1.878** | 0.000004 | **0.00002** |

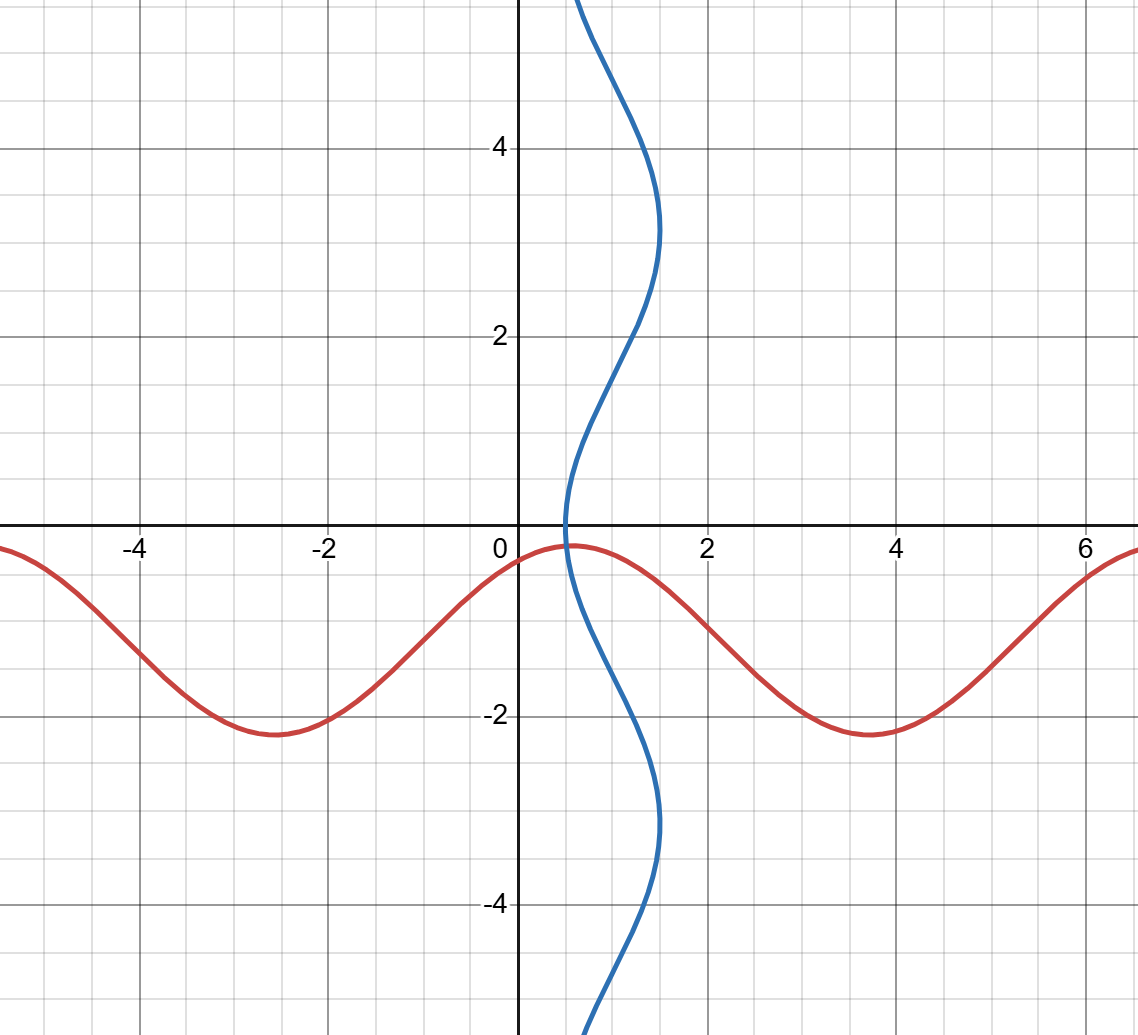
Рабочая формула:

Изображение выглядит как текст, Шрифт, линия, число

Контент, сгенерированный ИИ, может содержать ошибки.

**Часть 2.**

, Метод простой итерации

****

Проверка сходимости:

Условие: ∣∂φ1/∂x∣+∣∂φ1/∂y∣<1 и ∣∂φ2/∂x∣+∣∂φ2/∂y∣<1

0 + |sin(y)/2| < 1 и |cos(x + 1)| + 0 < 1

Это условие выполняется.

Итерации:

Начальное приближение x0 = 0, y0 = 0

Формула: xk+1 =1−cos(yk) / 2, yk+1=sin(xk+1) − 1.2

1.

X1 = 1 – cos(0)/2 = 0.500

Y1 = sin(1) – 1.2 = -0.358

Проверка точности: X |0.500-0| = 0.500 > 0.01 и Y |-0.358-0| > 0.01

2.

X2 = 1 – cos(-0.358)/2 = 0.532

Y2 = sin(1.5) – 1.2 = -0.202

Проверка точности: X |0.532-0.500| = 0.032 > 0.01 и Y |-0.202+0.358| = 0.156 > 0.01

3.

X3 = 1 – cos(-0.202)/2 = 0.510

Y3 = sin(1.532) – 1.2 = -0.200

Проверка точности: X |0.510-0.532| = 0.022 > 0.01 и Y |-0.200+0.202| = 0.002 < 0.01

4.

X3 = 1 – cos(-0.200)/2 = 0.509

Y3 = sin(1.510) – 1.2 = -0.200

Проверка точности: X |0.509-0.510| = 0.001 < 0.01 и Y |-0.200+0.200| = 0 < 0.01

Решение системы с точностью до 0.01:

X≈0.509, Y≈-0.200

**Код используемых методов.**

**Метод Хорд**

import additional\_functions.validation  
import properties.properties  
import tasks.task  
  
  
def calc(e, a, b, f):  
 func = tasks.task.get\_function(f)  
 for n in range(properties.properties.max\_iter):  
 x\_n = (a \* func(b) - b \* func(a)) / (func(b) - func(a))  
 if abs(x\_n - b) < e or abs(func(x\_n)) < e:  
 return x\_n, func(x\_n), n + 1  
 elif additional\_functions.validation.has\_root(f, x\_n, a):  
 b = x\_n  
 else:  
 a = x\_n  
 return 0, 0, 1000

**Метод Ньютона**

import additional\_functions.derivative  
import properties.properties  
import tasks.task  
  
  
def calc\_newton(f, x, e):  
 func = tasks.task.get\_function(f)  
 df = additional\_functions.derivative.fabric\_df(func)  
  
 x0 = x  
 for i in range(properties.properties.max\_iter):  
 fx = func(x0)  
 dfx = df(x0)  
 if dfx == 0:  
 print("Производная равна нулю!")  
 return 0, 0, 0  
 x\_next = x0 - fx/dfx  
  
 if abs(x\_next - x0) <= e:  
 return x\_next, func(x\_next), i + 1  
  
 x0 = x\_next  
  
 return 0, 0, properties.properties.max\_iter

**Метод простых итераций**

import numpy as np  
import additional\_functions.derivative  
import additional\_functions.validation  
import properties.properties  
import tasks.task  
  
  
def phi\_calc(func, lambda\_):  
 return lambda x: x + lambda\_ \* func(x)  
  
  
def calc(f, a, b, e):  
 func = tasks.task.get\_function(f)  
 df = additional\_functions.derivative.fabric\_df(func)  
 max\_df = max(abs(df(a)), abs(df(b)))  
 if max\_df == 0:  
 print('Производная равна нулю на интервале')  
 return 0, 0, 0  
 if df(a) > 0 and df(b) > 0:  
 lambda\_ = -1 / max\_df  
 elif df(a) < 0 and df(b) < 0:  
 lambda\_ = 1 / max\_df  
 else:  
 print("Не выполняется условие для получения lambda")  
 return 0, 0, 0  
  
 phi = phi\_calc(func, lambda\_)  
  
 dhi = additional\_functions.derivative.d\_phi(df, lambda\_)  
  
 for x in np.linspace(a, b, 10):  
 if abs(dhi(x)) >= 1:  
 print(f"Условие сходимости нарушено на интервале")  
 return 0, 0, 0  
  
 x0 = (a + b) / 2  
 for i in range(properties.properties.max\_iter):  
 x\_next = phi(x0)  
  
 if abs(x\_next - x0) <= e:  
 return x\_next, func(x\_next), i + 1  
 x0 = x\_next  
  
 return 0, 0, properties.properties.max\_iter

**Метод Ньютона для системы**

import additional\_functions.derivative  
import numpy as np  
  
import properties.properties  
import tasks.task  
  
  
def build\_jac(f, x, y):  
 df1\_dx = additional\_functions.derivative.partial\_d(f, x, y, 'x', 0)  
 df1\_dy = additional\_functions.derivative.partial\_d(f, x, y, 'y', 0)  
 df2\_dx = additional\_functions.derivative.partial\_d(f, x, y, 'x', 1)  
 df2\_dy = additional\_functions.derivative.partial\_d(f, x, y, 'y', 1)  
  
 J = np.array([[df1\_dx, df1\_dy], [df2\_dx, df2\_dy]])  
  
 return J  
  
  
def newton\_system\_calc(f, x, y, e):  
 x0, y0 = x, y  
 system = tasks.task.get\_system(f)  
 print("№ | x0 | y0 | x\_n | y\_n | |x\_n - x0| | |y\_n - y0|")  
 for i in range(properties.properties.max\_iter):  
 f1\_val = system[0](x0, y0)  
 f2\_val = system[1](x0, y0)  
  
 J = build\_jac(f, x0, y0)  
  
 if np.abs(np.linalg.det(J)) < 1e-12:  
 print("Матрица близка к 0!")  
 return 0, 0, 0, 0, 0  
  
 rhs = -np.array([f1\_val, f2\_val])  
 dx, dy = np.linalg.solve(J, rhs)  
  
 x\_n = x0 + dx  
 y\_n = y0 + dy  
  
 print(  
 f"{i + 1} | {x0:.6f} | {y0:.6f} | {x\_n:.6f} | {y\_n:.6f} | {abs(x\_n - x0):.6f} | {abs(y\_n - y0):.6f}")  
  
 if abs(x\_n - x0) <= e and abs(y\_n - y0) <= e:  
 return x\_n, y\_n, i + 1, abs(x\_n - x0), abs(y\_n - y0)  
  
 x0 = x\_n  
 y0 = y\_n  
  
 print("Метод не сошелся за максимальное кол-во итераций")  
 return 0, 0, properties.properties.max\_iter, 0, 0

https://github.com/wrakelft/CompMathITMO/tree/main/lab2

**Результат работы программы.**

**Изображение выглядит как текст, снимок экрана

Контент, сгенерированный ИИ, может содержать ошибки.**

**Изображение выглядит как линия, диаграмма, График, Параллельный

Контент, сгенерированный ИИ, может содержать ошибки.**

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Изображение выглядит как диаграмма, линия, текст, График

Контент, сгенерированный ИИ, может содержать ошибки.

**Выводы**

Во время выполнения лабораторной работы я познакомился с методами решения нелинейных уравнений. Применил их для ручного вычисления значений и реализовал методы Хорд, Ньютона и простой итерации на Python.