Федеральное государственное автономное образовательное учреждение высшего образования **«Национальный исследовательский университет ИТМО»**

Факультет Программной Инженерии и Компьютерной Техники

Лабораторная работа **№2**

**«Численное решение нелинейных уравнений и систем»**

по дисциплине «Вычислительная математика**»**

Вариант: **4**

**Выполнил:**

Кирячек Тимофей

Группа: Р3209

**Преподаватель:**

Наумова Надежда Александровна

Санкт-Петербург, 2025 г.

Цель работы: изучить численные методы решения нелинейных уравнений и их систем, найти корни заданного нелинейного уравнения/системы нелинейных уравнений, выполнить программную реализацию методов.

# 1. Вычислительная реализация задачи

# 1. Решение нелинейного уравнения

Изображение выглядит как линия, График, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.

Для определения интервалов изоляции корней данного уравнения можно воспользоваться методом интервалов знакопеременности. Для этого нужно найти значения функции на различных интервалах и определить знак функции на каждом из них.  
  
Из графика найдем приближенные значения корней:  
x ≈ -1.2, x ≈ 0.6, x ≈ 2.4

Теперь нужно разбить ось x на 4 интервала: (-∞, -1.2), (-1.2, 0.6), (0.6, 2.4) и (2.4, +∞). На каждом из этих интервалов нужно определить знак функции.

Для этого можем вычислить значения функции в произвольной точке каждого интервала. Например, для интервала (-∞, -1.2) можно выбрать x = -2, для интервала (-1.2, 0.6) x = 0, для интервала (0.6, 2.4) x = 2, и для интервала (2.4, +∞) x = 4.

Таким образом, получим следующие значения функции:

для x = -2: f(-2) = -9.8

для x = 0: f(0) = 1.76

для x = 2: f(2) = -1.8

для x = 4: f(4) = 27.52

Знаки функции на каждом интервале будут соответственно:

|  |  |  |  |
| --- | --- | --- | --- |
| (-∞, -1.2) | (-1.2, 0.6) | (0.6, 2.4) | (2.4, +∞) |
| - | + | - | + |

Таким образом, мы получаем два интервала изоляции корней уравнения:

(-1.5, 0), (0, 2) и (2, 3).

x1 ≈

x2 ≈

x3 ≈

Крайний правый корень – **Метод простой итерации**

Проверка **условия сходимости** метода на выбранном интервале:

На отрезке начального приближения [2, 3] функция определена, непрерывна и дифференцируема.

Изображение выглядит как линия, График, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки. Изображение выглядит как линия, График, диаграмма, Параллельный

Контент, сгенерированный ИИ, может содержать ошибки.

**итерационная последовательность сходится,** скорость сходимости низкая, критерий окончания итерационного процесса , x0 = 2

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | xk | xk+1 | f(xk+1) | │ xk+1- xk│ |
| 1 | 2 | 2.1318 | -1.40742 | 0.1318 |
| 2 | 2.1318 | 2.23463 | -0.98833 | 0.10283 |
| 3 | 2.23463 | 2.30698 | -0.63473 | 0,07235 |
| 4 | 2.30698 | 2.35345 | -0.37998 | 0,04647 |
| 5 | 2.35345 | 2.38127 | -0.21682 | 0,02782 |
| 6 | 2.38127 | 2.39714 | -0.12011 | 0,01587 |
| 7 | 2.39714 | 2.40593 | -0.06541 | 0,00879 |
| 8 | 2.40593 | 2.41072 | -0.03524 | 0,00479 |
| 9 | 2.41072 | **2.41330** | -0.01889 | 0,00258 |

Центральный корень – **Метод секущих**

Изображение выглядит как линия, График, диаграмма

Контент, сгенерированный ИИ, может содержать ошибки.Выберем , для этого построим график на интервале (0, 2):

Как видно из графика, на интервале нет ни одной точки, в которой обеспечивалась бы быстрая сходимость. Поэтому можно взять, например левую границу интервала

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | xk-1 | xk | xk+1 | f(xk+1) | │ xk+1- xk│ |
| 1 | 0 | 0.01 | 0.87181 | -0.75748 | 0.86181 |
| 2 | 0.01 | 0.87181 | 0.61040 | 0.062433 | 0.26141 |
| 3 | 0.87181 | 0.61040 | 0.63030 | -0.001069 | 0.01991 |
| 4 | 0.61040 | 0.63030 | **0.62997** | 0.0000013 | 0.00033 |

Крайний левый корень – **Метод половинного деления**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | a | b | x | f(a) | f(b) | f(x) | |a – b| |
| 1 | -1.500 | 0.000 | -0.750 | -2.8675 | 1.75 | 1.775 | 1.500 |
| 2 | -1.500 | -0.750 | -1.125 | -2.8675 | 1.775 | 0.19414 | 0.750 |
| 3 | -1.500 | -1.125 | -1.3125 | -2.8675 | 0.19414 | -1.13181 | 0.375 |
| 4 | -1.3125 | -1.125 | -1.21875 | -1.1318 | 0.19414 | -0.42009 | 0.18750 |
| 5 | -1.21875 | -1.125 | -1.17188 | -0.42009 | 0.19414 | -0.10113 | 0,09375 |
| 6 | -1.17188 | -1.125 | -1.14844 | -0.10113 | 0.19414 | 0.049438 | 0,04688 |
| 7 | -1.17188 | -1.14844 | -1.16016 | -0.10113 | 0.04944 | -0.02511 | 0,02344 |
| 8 | -1.16016 | -1.14844 | -1.1543 | -0.02511 | 0.04944 | 0.012349 | -0.01172 |
| 9 | -1.16016 | -1.1543 | **-1.15723** | -0.02511 | 0.01235 | -0.00633 | -0.00586 |

# 2. Решение системы нелинейных уравнений

1. , Метод Ньютона

Изображение выглядит как линия, диаграмма, График

Контент, сгенерированный ИИ, может содержать ошибки.

Отметим, что решение системы уравнений являются точки пересечения эллипса и , следовательно, система имеет не более 2 различных решений.

Построим матрицу Якоби:

*, ,*

**Корень 1:** Шаг 1: Выбираем

Шаг 2. Решаем полученную систему.

Шаг 3. Вычисляем очередные приближения:

,

, , возвращаемся к шагу 1:

,

,

Таким образом, корень 1 найден: (0.5991, 0.5662)

Аналогично находим **другой корень**:

# 2. Программная реализация задачи

[**https://github.com/maxbarsukov/itmo/tree/master/4%20%D0%B2%D1%8B%D1%87%D0%BC%D0%B0%D1%82/%D0%BB%D0%B0%D0%B1%D0%BE%D1%80%D0%B0%D1%82%D0%BE%D1%80%D0%BD%D1%8B%D0%B5/lab2**](https://github.com/maxbarsukov/itmo/tree/master/4%20%D0%B2%D1%8B%D1%87%D0%BC%D0%B0%D1%82/%D0%BB%D0%B0%D0%B1%D0%BE%D1%80%D0%B0%D1%82%D0%BE%D1%80%D0%BD%D1%8B%D0%B5/lab2)

****

**Результаты выполнения программы при различных исходных данных:**

|  |
| --- |
| Выберите тип программы:  1: Нелинейное уравнение  2: Система нелинейных уравнений  3: Выход  Введите номер типа: 1  Выберите уравнение:  1: -1.38\*x^3 - 5.42\*x^2 + 2.57\*x + 10.95  2: x^3 - 1.89\*x^2 - 2\*x + 1.76  3: x/2 - 2\*(x + 2.39)^(1/3)  4: -x/2 + e^x + 5\*sin(x)  Введите номер уравнения: 1  Выберите метод:  1: Метод половинного деления  2: Метод хорд  3: Метод простой итерации  4: Метод Ньютона  Введите номер метода: 2  Введите имя файла для загрузки исходных данных и интервала или пустую строку, чтобы ввести вручную:  Введите левую границу интервала: -4  Введите правую границу интервала: -1.5  Введите погрешность вычисления: 0.000001  Введите имя файла для вывода результата или пустую строку, чтобы вывести в консоль:  Процесс решения:  1: a = -4.000, b = -1.908, x = -3.254, f(a) = 2.270, f(b) = -4.098, f(x)=-7.253338075903418, |x\_k+1 - x\_k| = 1.3463753767240685  2: a = -4.000, b = -3.254, x = -3.822, f(a) = 2.270, f(b) = -7.253, f(x)=-0.9961797791033895, |x\_k+1 - x\_k| = 0.568022551124693  3: a = -4.000, b = -3.822, x = -3.876, f(a) = 2.270, f(b) = -0.996, f(x)=-0.07183806668107628, |x\_k+1 - x\_k| = 0.05421895603697724  4: a = -4.000, b = -3.876, x = -3.880, f(a) = 2.270, f(b) = -0.072, f(x)=-0.004903874657289364, |x\_k+1 - x\_k| = 0.0037899812480461925  5: a = -4.000, b = -3.880, x = -3.880, f(a) = 2.270, f(b) = -0.005, f(x)=-0.0003334833824535366, |x\_k+1 - x\_k| = 0.0002581574086821803  6: a = -4.000, b = -3.880, x = -3.881, f(a) = 2.270, f(b) = -0.000, f(x)=-2.267235988462346e-05, |x\_k+1 - x\_k| = 1.7553172981354948e-05  7: a = -4.000, b = -3.881, x = -3.881, f(a) = 2.270, f(b) = -0.000, f(x)=-1.541386730252725e-06, |x\_k+1 - x\_k| = 1.1933664496588392e-06  8: a = -4.000, b = -3.881, x = -3.881, f(a) = 2.270, f(b) = -0.000, f(x)=-1.047914928165028e-07, |x\_k+1 - x\_k| = 8.113129679188091e-08  Результат:  Найденный корень уравнения: -3.880518  Значение функции в корне: -1.047914928165028e-07  Число итераций: 8  Еще раз? [y/n] |
| Выберите тип программы:  1: Нелинейное уравнение  2: Система нелинейных уравнений  3: Выход  Введите номер типа: 2  Выберите систему уравнений:  1: x^2 + y^2 - 1, x^2 - y - 0.5  Введите номер системы: 1  Введите начальные приближения x0, y0: 0 0  Введите погрешность вычисления: 0.01  0. x1=1.0, x2=-0.5, xnext=(1.0, -0.5), |xk+1 - xk|=1.118033988749895  1. x1=0.8660254037844386, x2=0.5, xnext=(0.8660254037844386, 0.5), |xk+1 - xk|=1.0089346819448337  2. x1=0.8660254037844386, x2=0.2499999999999999, xnext=(0.8660254037844386, 0.2499999999999999), |xk+1 - xk|=0.2500000000000001  3. x1=0.9682458365518543, x2=0.2499999999999999, xnext=(0.9682458365518543, 0.2499999999999999), |xk+1 - xk|=0.10222043276741566  4. x1=0.9682458365518543, x2=0.4375000000000001, xnext=(0.9682458365518543, 0.4375000000000001), |xk+1 - xk|=0.18750000000000022  5. x1=0.8992184106211348, x2=0.4375000000000001, xnext=(0.8992184106211348, 0.4375000000000001), |xk+1 - xk|=0.06902742593071942  6. x1=0.8992184106211348, x2=0.3085937499999999, xnext=(0.8992184106211348, 0.3085937499999999), |xk+1 - xk|=0.12890625000000022  7. x1=0.9511939326241193, x2=0.3085937499999999, xnext=(0.9511939326241193, 0.3085937499999999), |xk+1 - xk|=0.0519755220029845  8. x1=0.9511939326241193, x2=0.4047698974609377, xnext=(0.9511939326241193, 0.4047698974609377), |xk+1 - xk|=0.09617614746093783  9. x1=0.9144185748930639, x2=0.4047698974609377, xnext=(0.9144185748930639, 0.4047698974609377), |xk+1 - xk|=0.03677535773105545  10. x1=0.9144185748930639, x2=0.3361613301094619, xnext=(0.9144185748930639, 0.3361613301094619), |xk+1 - xk|=0.0686085673514758  11. x1=0.9418044171371449, x2=0.3361613301094619, xnext=(0.9418044171371449, 0.3361613301094619), |xk+1 - xk|=0.027385842244081027  12. x1=0.9418044171371449, x2=0.38699556013903724, xnext=(0.9418044171371449, 0.38699556013903724), |xk+1 - xk|=0.05083423002957532  13. x1=0.9220815779705572, x2=0.38699556013903724, xnext=(0.9220815779705572, 0.38699556013903724), |xk+1 - xk|=0.01972283916658768  14. x1=0.9220815779705572, x2=0.3502344364326728, xnext=(0.9220815779705572, 0.3502344364326728), |xk+1 - xk|=0.03676112370636442  15. x1=0.936662073288274, x2=0.3502344364326728, xnext=(0.936662073288274, 0.3502344364326728), |xk+1 - xk|=0.014580495317716768  16. x1=0.936662073288274, x2=0.3773358395366879, xnext=(0.936662073288274, 0.3773358395366879), |xk+1 - xk|=0.027101403104015098  17. x1=0.9260764893901275, x2=0.3773358395366879, xnext=(0.9260764893901275, 0.3773358395366879), |xk+1 - xk|=0.010585583898146456  18. x1=0.9260764893901275, x2=0.35761766420114305, xnext=(0.9260764893901275, 0.35761766420114305), |xk+1 - xk|=0.019718175335544874  19. x1=0.9338680882497905, x2=0.35761766420114305, xnext=(0.9338680882497905, 0.35761766420114305), |xk+1 - xk|=0.007791598859662963  20. x1=0.9338680882497905, x2=0.3721096062513185, xnext=(0.9338680882497905, 0.3721096062513185), |xk+1 - xk|=0.014491942050175455  21. x1=0.9281887959545131, x2=0.3721096062513185, xnext=(0.9281887959545131, 0.3721096062513185), |xk+1 - xk|=0.005679292295277416  22. x1=0.9281887959545131, x2=0.3615344409354887, xnext=(0.9281887959545131, 0.3615344409354887), |xk+1 - xk|=0.010575165315829804  Неизвестные: x = 0.92819, y = 0.36153  Количество итераций: 22  Невязка: -0.0077584070819749495, 0.0 |

# Вывод

В ходе выполнения лабораторной работы были изучены численные методы решения нелинейных уравнений и систем нелинейных уравнений с использованием Python. В результате работы были найдены корни заданных уравнений и систем с использованием различных численных методов, а также были построены графики функций для полного представления исследуемых интервалов.