Учреждение образования

«Брестский государственный университет имени А. С. Пушкина»

Физико-математический факультет

Кафедра прикладной математики и информатики

**КУРСОВОЙ ПРОЕКТ**

**РЕШЕНИЕ СИСТЕМ НЕЛИНЕЙНЫХ УРАВНЕНИЙ МЕТОДОМ НЬЮТОНА**

Хвалько Михаил Александрович,

студент 3 курса специальности

«Прикладная математика»

Кондратюк Александр Петрович ─

старший преподаватель кафедры

прикладной математики и информатики

Брест 2022

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 2](#_Toc121762645)

[**Глава 1. Общая характеристика метода** 3](#_Toc121762646)

[**1.1 Общие положения** 3](#_Toc121762647)

[**1.2 Постановка задачи для одномерного случая** 4](#_Toc121762648)

[**1.3 Постановка задачи для n-мерного случая** 5](#_Toc121762649)

[**1.4 Алгоритм метода Ньютона** 7](#_Toc121762650)

[**Глава 2. Прикладные средства для разработки** 8](#_Toc121762651)

[**2.1 Почему python** 8](#_Toc121762652)

[**2.2 SymPy** 8](#_Toc121762653)

[**Глава 3. Реализация Алгоритма** 10](#_Toc121762654)

[**3.1 Численный эксперимент** 13](#_Toc121762655)

[ЗАКЛЮЧЕНИЕ 15](#_Toc121762656)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 16](#_Toc121762657)

# ВВЕДЕНИЕ

Без математической грамотности невозможно успешное освоение методов решения задач по физике, химии, биологии и другим предметам. Весь комплекс естественных наук построен и развивается на базе математических знаний. Например, исследование ряда актуальных задач математической физики приводит к необходимости решения нелинейных уравнений. Решение нелинейных уравнений необходимо в нелинейной оптике, физике плазмы, теории сверхпроводимости и физике низких температур.

Многие прикладные задачи приводят к необходимости нахождения общего решения системы нелинейных уравнений. Общего аналитического решения системы нелинейных уравнений не найдено. Существуют лишь численные методы.

Следует отметить интересный факт о том, что любая система уравнений над действительными числами может быть представлена одним равносильным уравнением, если взять все уравнения в форме:

(1)

возвести их в квадрат и сложить.

Для численного решения применяются итерационные методы последовательных приближений (простой итерации) и метод Ньютона в различных модификациях. Итерационные процессы естественным образом обобщаются на случай системы нелинейных уравнений вида:

,

Обозначим через вектор неизвестных и определим вектор-функцию . Тогда система (1) записывается в виде уравнения: [2]

Объектом исследования являются методы решения нелинейных задач.

Предмет исследования – методы решения систем нелинейных уравнений.

Цель курсовой работы: реализовать метод Ньютона решения систем нелинейных уравнений на языке программирования и провести численный эксперимент, используя модельную задачу.

Вышесказанное позволило определить объект и предмет курсового исследования, сформулировать его цель и задачи.

Выбор языка программирования и дополнительных библиотек для реализации данного проекта. Для достижения поставленной цели решались следующие задачи:

1. Изучить общие характеристики метода Ньютона.
2. Выбрать средства для реализации алгоритма.
3. Проектирование и создание программной реализации алгоритма метода Ньютона.
4. Тестирование алгоритма на системах нелинейных уравнений.

**Глава 1. Общая характеристика метода**

**1.1 Общие положения**

Нелинейные уравнения можно разделить на два класса: алгебраические и трансцендентные. Алгебраическими называют уравнения, содержащие только алгебраические функции (целые, рациональные, иррациональные). В частности, многочлен является целой алгебраической функцией. Уравнения, содержащие другие функции (тригонометрические, показательные, логарифмические и др.) называются трансцендентными.

Методы решения нелинейных уравнений делятся на две группы: точные и итерационные.

Точные методы позволяют записать корни в виде некоторого конечного соотношения (формулы). Из школьного курса алгебры известны такие методы для решения тригонометрических, логарифмических, показательных, а также простейших алгебраических уравнений.

Как известно, многие уравнения и системы уравнений не имеют аналитических решений. В первую очередь это относится к большинству трансцендентных уравнений. Доказано также, что нельзя построить формулу, по которой можно было бы решить произвольное алгебраическое уравнение степени выше четвертой. Кроме того, в некоторых случаях уравнение содержит коэффициенты, известные лишь приблизительно, и, следовательно, сама задача о точном определении корней уравнения теряет смысл. Для их решения используются итерационные методы с заданной степенью точности.

Итерационный процесс состоит в последовательном уточнении начального приближения . Каждый такой шаг называется итерацией. В результате итераций находится последовательность приближенных значений корня . Если эти значения с увеличением числа итераций n приближаются к истинному значению корня, то говорят, что итерационный процесс сходится. [6]

## **1.2 Постановка задачи для одномерного случая**

Пусть дано уравнение

(2)

Найти корни этого уравнения с точностью .

Ограничимся обсуждением методов поиска лишь действительных корней, не затрагивая проблему корней комплексных.

Приближенное решение уравнения (1) обычно разбивается на два этапа:

1. Отделение корней, то есть установление промежутков, содержащих по одному корню.
2. Уточнение корней, то есть сужение отрезка, содержащего корень, до такой степени, что длина отрезка становится меньше требуемой точности.

На первом этапе как, как правило, применяется графический метод решения уравнений. Уточнение корней опирается на свойства непрерывных функций:

1. если функция непрерывна на отрезке и имеет на концах этого отрезка разные знаки , то на существует хотя бы один корень уравнения (2);
2. если функция непрерывна и монотонна не меняет знак на отрезке ) и , то на этом отрезке корень уравнения (1) единственный (рис. 1).

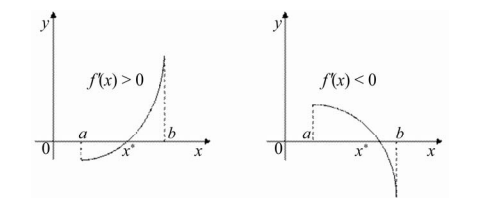


Рис.1. Существование единственного корня на отрезке

Решить уравнение (1) итерационным методом значит установить, имеет ли оно корни, сколько корней и найти значения корней с нужной точностью.

Всякое значение ζ, обращающее функцию в нуль, то есть такое, что

называется корнем уравнения (1) или нулем функции .[6]

## **1.3 Постановка задачи для n-мерного случая**

Каждая операция содержит свои уравнения, которые отражают её геометрический смысл. Уравнений может быть одно или несколько, они могут быть скалярными или векторными. Векторное уравнение является удобной записью нескольких скалярных уравнений для соответствующих компонент векторов. В общем случае выполнение операции сводится к решению некоторой системы скалярных нелинейных уравнений относительно параметров кривых и поверхностей. Задача определения корней системы нелинейных уравнений решается с помощью некоторого итерационного процесса, который шаг за шагом уточняет корни уравнения. Итерационный процесс предполагает, что и на первой итерации известно некоторое приближенное значение корней системы. Будем считать, что начальное приближение решения нам известно и что оно находится в области сходимости.

Пусть требуется найти значения n неизвестных , удовлетворяющих n алгебраическим уравнениям:

где ­­­– непрерывно дифференцируемые в некоторой окрестности начального приближения функции. Метод Ньютона является наиболее распространённым методом решения системы нелинейных уравнений. Основным преимуществом является его быстрота и надёжность процесса сходимости. Основным недостатком является то, что он требует большого объёма вычислений и промежуточной информации на каждом итерационном шаге. Предположим, что после итераций нам известно -e приближение решения , которое отличается от точного решения на величины , .

Погрешности неизвестны и подлежат определению. Введем обозначения: — для совокупности неизвестных на -м приближении, … , ) , — для функций на -м приближении. Разложим левую часть каждого уравнения системы (3) в ряд Тейлора по степеням в окрестности -го приближения

Δ+ Δ+…+ Δ+

+ + (4)

Левую часть (4) заменим нулем, так как … , ) = 0, а в правой части отбросим величины второго порядка малости и выше по , .

Тогда система (3) заменится системой уравнений:

являющейся линейной системой относительно погрешностей , .

Решив систему линейный алгебраических уравнений (5), вычислим следующее приближение для искомого решения:

Каждое приближение будет отличаться от точного решения, так как мы линеаризовали систему уравнений. Погрешности на каждом предложении будут становиться всё меньше и меньше. Мы не будем останавливаться на условиях и скорости сходимости метода Ньютона. Скажем только, что если начальное приближение выбрано достаточно хорошо и матрица системы линейных уравнений (5) на каждой итерации хорошо обусловлена и имеет обратную матрицу, то метод Ньютона сходится к единственному в данной окрестности решению

Метод Ньютона имеет квадратичную сходимость.

Это сравнительно быстрая сходимость – после очередной итерации погрешность каждой неизвестной уменьшается примерно на один или два порядка. Итерационный процесс решения системы нелинейных уравнений (3) заканчивается, когда на очередной итерации погрешности всех неизвестных становятся меньше заданной величины :[3]

для всех

Перейдем к матричному виду и рассмотрим алгоритм.

## **1.4 Алгоритм метода Ньютона**

Пусть

Формула для нахождения решения:

где

Так как процесс вычисления обратной матрицы является трудоемким, преобразуем (7) следующим образом

где – поправка к текущему приближению

Умножим последнее выражение слева на матрицу Якоби :

В результате получена система линейных алгебраических уравнений относительно поправки . После её определения вычисляется следующее приближение

Алгоритм метода ньютона для решения нелинейных систем

1. Задать начальное приближение и малое положительное число ε(точность). Положить k = 0.
2. Решить систему линейных алгебраических уравнений относительно оправки

**3.** Вычислить следующее приближение:

**4.** Если = , процесс закончить и положить . Если , то положить k = k+1 и перейти к пункту 2.

Недостатком метода Ньютона является необходимость вычисления производных на каждом шаге.[1]

# **Глава 2. Прикладные средства для разработки**

Для реализации алгоритма существует множество языков программирования. В своем проекте я решил использовать Python

## **2.1 Почему python**

Python — язык программирования общего назначения, который используют во многих областях IT-индустрии. Его основные свойства:

1. Понятность кода. Синтаксическая особенность Python – выделение блоков кода отступами, что значительно упрощает зрительное восприятие программ, написанных на этом языке.
2. Интерпретируемость Программы, написанные на языке программирования Python, не переводятся в машинный код, а сразу выполняются программой-интерпретатором. Это позволяет запускать код на любой платформе с установленным заранее интерпретатором
3. Объектно ориентированность. Python – это язык, созданный согласно парадигме объектно-ориентированного программирования (ООП). В ней основными являются понятия объекта и класса. Классы – это специальные типы данных, объекты – экземпляры классов. То есть любое значение является объектом конкретного класса. В Python вы можете не только использовать уже существующие классы, но и создавать свои собственные
4. Динамическая типизация. В отличие от С-подобных языков программирования, в Python переменные связываются с типом в момент присваивания в них конкретных значений

Python является самым популярным языком программирования в рейтинге TIOBE за август 2022 года. Его применяют повсеместно: в аналитике данных, тестировании и разработке игр. Кроме того, Python нашел широкое применение среди ученых благодаря своей простоте. [4]

## **2.2 SymPy**

SymPy — это библиотека Python для выполнения символьных вычислений. Это система компьютерной алгебры, которая может выступать как отдельное приложение, так и в качестве библиотеки для других приложений. Поработать с ней онлайн можно на [**https://live.sympy.org/**](https://live.sympy.org/). Поскольку это чистая библиотека Python, ее можно использовать даже в интерактивном режиме.

В SymPy есть разные функции, которые применяются в сфере символьных вычислений, математического анализа, алгебры, дискретной математики, квантовой физики и так далее. SymPy может представлять результат в разных форматах: LaTeX, MathML и так далее. Распространяется библиотека по лицензии New BSD. Первыми эту библиотеку выпустили разработчики Ondřej Čertík и Aaron Meurer в 2007 году. Текущая актуальная версия библиотеки — 1.6.2.[5]

Вот где применяется SymPy:

* Многочлены
* Математический анализ
* Дискретная математика
* Матрицы
* Геометрия
* Построение графиков
* Физика
* Статистика
* Комбинаторика

**Глава 3. Реализация Алгоритма**

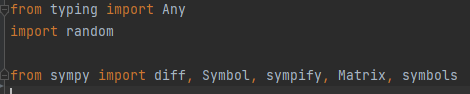


Рис. 2 подключим нужные модули

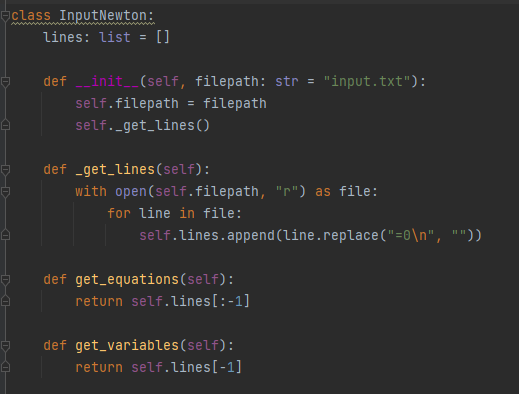


Рис.3 в классе InputNewton будем задавать начальную систему и переменные из файла

Пример, как может выглядеть система:

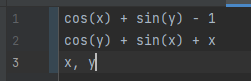


Рис. 4 начальная система нелинейных уравнений

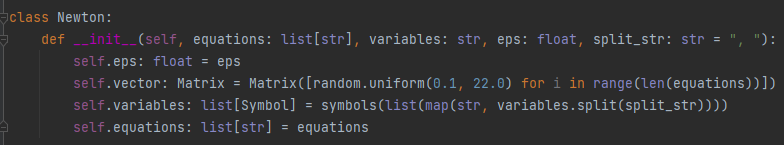


Рис. 5. Запись параметров в класс.

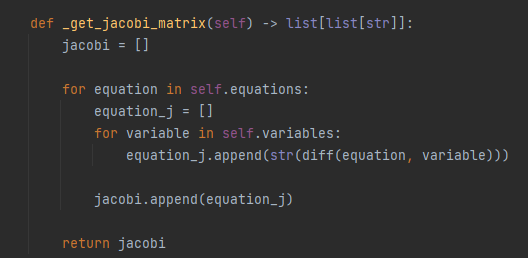


Рис. 6 Рассчитаем матрицу Якоби

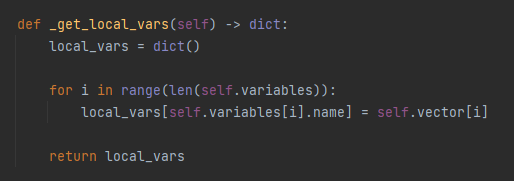


Рис. 7 Получение словаря ключей переменных и значения текущего приближения

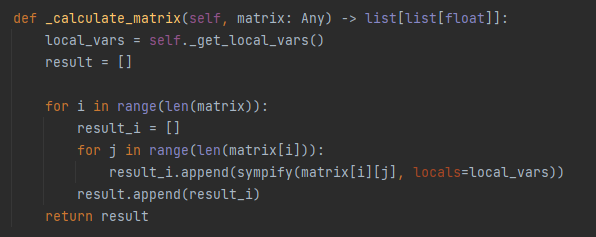


Рис. 8 функция принимает в себя приближение

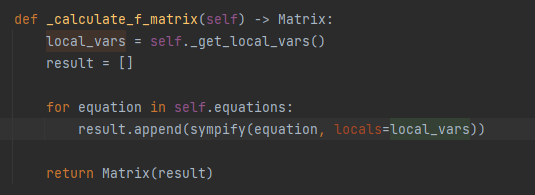


Рис. 9 Подставляет текущее приближение в начальную функцию F(x)

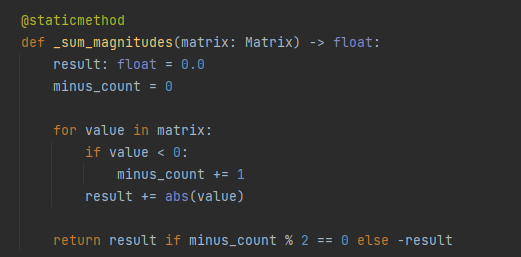


Рис. 10 Функция принимает в себя матрицу и суммирует в одну переменную все собственные значения матрицы по модулю. Если количество отрицательных элементов матрицы нечетное, то результат умножается на (-1)

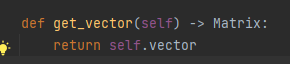


Рис.11 Функция возвращает текущее приближение

Зададим начальное приближение и точность

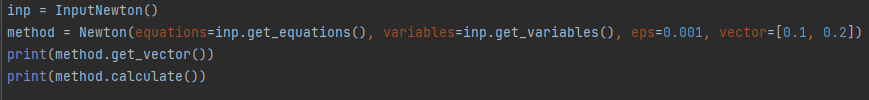


Рис. 12 точность eps = 0.001 и начальное приближение

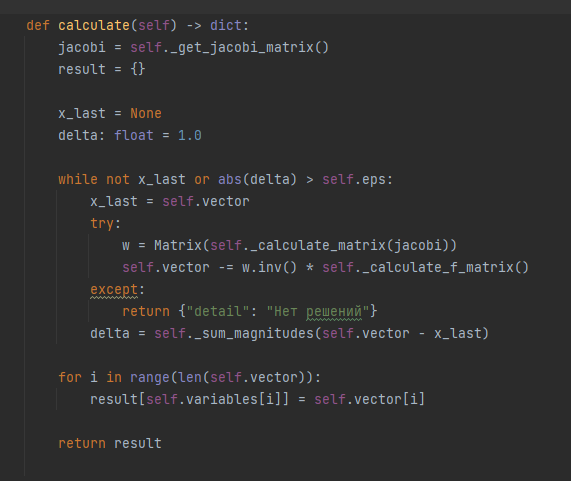


Рис. 13 Наконец реализуем алгоритм и обработаем исключение о невозможности найти решение

## **3.1 Численный эксперимент**

Воспользуемся системой (8) и зададим начальное приближение и точность: пусть и . Выполним наш код и получим результат:



Рис.5 Получение результата, где Matrix есть начальное приближение

Для проверки воспользуемся математическим сервисом для построения графиков desmos.com:

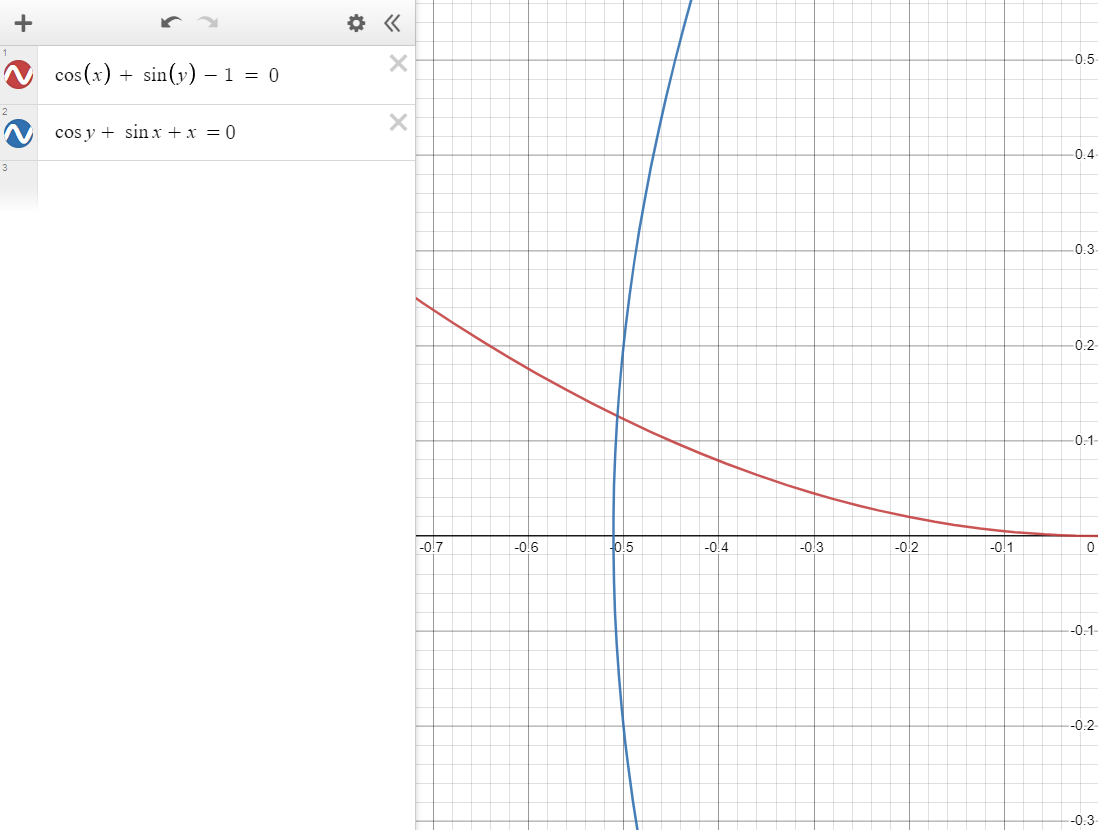


Рис. 6 Графическое отображение уравнений системы (8)

Получим точку пересечения

# ЗАКЛЮЧЕНИЕ

Цель курсового проекта – изучить методы решения нелинейных уравнений, выбрать наиболее актуальный и апробировать его в опытно-экспериментальной среде В ходе теоретического изучения различных источников были уточнены основные этапы разработки алгоритма с использованием языка Python и математической библиотеки SymPy.

На основе полученных данных был спроектирован и разработан алгоритм, который позволил пользователю получить решение нелинейной системы при заданном начальном приближении и точности. В результате проведения численного эксперимента удалось получить решение заданной нелинейной системы и проверить его графически, то есть поставленные в курсовом проекте задачи были выполнены.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Методы решения систем нелинейных уравнений. [Электронный ресурс]. – Режим доступа: <http://mathhelpplanet.com/static.php?p=metody-resheniya-sistem-nelineynykh-uravneniy>. – Дата доступа: 01.12.2022.
2. Численные методы решения систем нелинейных уравнений. [Электронный ресурс]. – Режим доступа: <https://habr.com/ru/post/419453/>. – Дата доступа: 01.12.2022.
3. Голованов, Н.Н. Геометрическое моделирование / Н.Н. Голованов. – Издательство Физико-математической литературы, 2002. – 215 c.
4. Преимущества и недостатки Python. [Электронный ресурс]. – Режим доступа: <https://skysmart.ru/articles/programming/preimushestva-i-nedostatki-python>. – Дата доступа: 02.12.2022.
5. Математическая библиотека Python SymPy. [Электронный ресурс]. – Режим доступа: <https://pythonru.com/biblioteki/sympy-v-python> – Дата доступа: 02.12.2022.
6. Потапова, Н. Н. Решение нелинейных уравнений. Методические указания к лабораторной работе / Н.Н. Потапова, О.М. Забродина – Волгоградский государственный архитектурно-строительный университет, 2013. – 4 с.