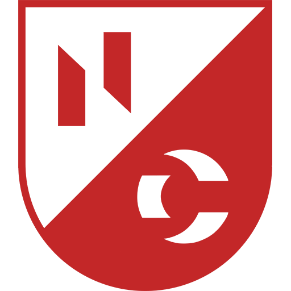
**Некоммерческое образовательное учреждение  
Учебно-научно-производственный комплекс**

**«Международный Университет Кыргызстана»**

**Среднее профессиональное образование**

**«Nomad» Колледж**



Программа «Программирование в компьютерных системах»

**АБДИЛАЗИЗ УУЛУ МУХАММЕДХИЛМИ**

**КУРСОВАЯ РАБОТА**

*На тему:* «*Исследование библиотеки NumPy»*

Руководитель курсовой работы:

преп.

\_\_\_\_\_\_\_\_\_\_\_ Уралбек уулу С.

“\_\_\_”\_\_\_\_\_\_\_\_\_\_\_\_2023 г.

**Бишкек 2024 г.**

**СОДЕРЖАНИЕ:**

[**ВВЕДЕНИЕ** 3](#_Toc7092964)

[**ГЛАВА 1. ИСТОРИЯ И РАЗВИТИЕ**](#_Toc7092965) 4

[1.1. ОСНОВНЫЕ ХАРАКТЕРИСТИКИ NamPy](#_Toc7092966) 6

[1.2. ПРЕИМУЩЕСТВО NamPy](#_Toc7092967) 13

[**ГЛАВА 2. ПРИМЕНЕНИЕ NamPy**](#_Toc7092968) 17

[2.1. ПРИМЕРЫ ИСПОЛЬЗОВАНИЯ 2](#_Toc7092969)2

[**ЗАКЛЮЧЕНИЕ**](#_Toc7092975) 26

[**СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ**](#_Toc7092976) 29

# **ВВЕДЕНИЕ**

NumPy, сокращение от Numerical Python, представляет собой мощную библиотеку для работы с многомерными массивами и выполнения вычислений в языке программирования Python. Разработанная с открытым исходным кодом, NumPy стала неотъемлемым компонентом в научных и инженерных приложениях, обеспечивая эффективные инструменты для обработки данных, матричных операций и статистического анализа.

Одной из ключевых особенностей NumPy является многомерный массив, или ndarray, который предоставляет эффективное хранение и манипуляции данными. Благодаря этим массивам, NumPy становится фундаментом для различных вычислительных задач, включая обработку изображений, анализ временных рядов, и решение уравнений линейной алгебры.

Библиотека также известна своими возможностями в области броадкастинга, что позволяет выполнять операции над массивами различных форм без необходимости явных циклов. Это делает код более лаконичным и улучшает его читаемость.

NumPy активно используется в области научных исследований, где его функционал поддерживает моделирование и анализ сложных систем. В области машинного обучения NumPy предоставляет инструменты для представления данных и выполнения операций, необходимых для обучения моделей.

Интеграция NumPy с другими библиотеками, такими как pandas и scikit-learn, делает её неотъемлемой частью стека инструментов для анализа данных на Python. С постоянными обновлениями и активным сообществом разработчиков, NumPy продолжает оставаться в центре внимания при решении сложных задач научных вычислений и анализа данных.

**ГЛАВА 1. ИСТОРИЯ И РАЗВИТИЕ**

История создания и развития библиотеки NumPy

NumPy, или Numerical Python, имеет свои корни в необходимости создания эффективного инструмента для научных вычислений и обработки массивов данных в языке программирования Python. Вот краткий обзор истории создания и развития этой библиотеки:

1. Зарождение (1995-2000):

В конце 1990-х годов, столкнувшись с ограничениями Python в области численных вычислений, стал расти спрос на инструмент, который мог бы обеспечить эффективную работу с массивами данных.

Travis Olliphant, физик и инженер, стал одним из ключевых деятелей в создании инструмента, который затем стал основой для NumPy.

2. Создание Numeric и Numarray (2000-2005):

В начале 2000-х годов были созданы Numeric и Numarray — первые попытки предоставить поддержку для массивов в Python.

Numeric, разработанный в основном Jim Hugunin, стал первой попыткой добавления массивов и вычислительных возможностей в Python.

3. Объединение усилий (2005):

Два развивающихся проекта, Numeric и Numarray, сталкивались с проблемой фрагментации. Для решения этой проблемы создатели NumPy объединили усилия для создания единого проекта.

4. Рождение NumPy (2006):

NumPy как самостоятельный проект официально родился в 2006 году.

NumPy стал преемником Numeric и Numarray, объединив в себе лучшие черты обеих библиотек.

5. Развитие и Распространение (после 2006):

Вслед за созданием NumPy, библиотека продолжала активно развиваться, привлекая внимание и участие сообщества разработчиков.

Постепенно NumPy стала основой для многих других библиотек, таких как pandas, scikit-learn и других, утвердив свою роль в научных вычислениях и анализе данных на Python.

6. Современность:

На сегодняшний день NumPy остается неотъемлемой частью экосистемы Python для научных вычислений.

Регулярные обновления и развитие продолжают улучшать функционал и производительность библиотеки.

Благодаря своему замечательному происхождению и постоянному развитию, NumPy продолжает оставаться фундаментальным инструментом для решения задач научных вычислений в Python.

NumPy прошла через несколько ключевых версий и вех в своем развитии, каждая из которых внесла существенные улучшения и нововведения. Вот несколько ключевых версий и вех в истории развития библиотеки:

NumPy 1.0 (2006):

Рождение NumPy как самостоятельного проекта после объединения проектов Numeric и Numarray.

Предоставление основных структур данных, таких как ndarray, и базовых функций для выполнения математических операций.

NumPy 1.3 (2009):

Внедрение инструмента броадкастинга, который значительно упрощает операции над массивами различных форм.

Улучшения производительности и стабильности.

NumPy 1.6 (2011):

Введение функции einsum для более удобных операций с многомерными массивами.

Повышение эффективности в использовании памяти и улучшения в работе с многомерными массивами.

NumPy 1.8 (2013):

Внедрение поддержки для многопоточности и распараллеливания вычислений.

Повышение производительности при использовании многоканальных данных.

NumPy 1.13 (2017):

Введение метода numpy.random.Generator для более управляемого и устойчивого генерирования случайных чисел.

Дополнительные улучшения производительности и стабильности.

NumPy 1.20 (2021):

Поддержка библиотеки Python 3.9.

Введение новых функций и оптимизаций, включая операции над типами данных с плавающей запятой.

NumPy 1.21 (2021):

Дополнительные улучшения производительности, в том числе оптимизации в операциях индексации и срезов.

Внедрение новых функций для улучшения удобства использования.

Каждая из этих версий принесла с собой значительные улучшения, отражающие эволюцию NumPy от базового инструмента для работы с массивами данных до мощной библиотеки для научных вычислений в Python. Регулярные обновления и инновации продолжают укреплять позиции NumPy в области научных и инженерных вычислений.

* 1. ***ОСНОВНЫЕ ХАРАКТЕРИСТИКИ NamPy***

Многомерные массивы (ndarrays) и их роль в хранении данных в NumPy

Многомерные массивы (ndarrays) являются одной из ключевых структур данных в библиотеке NumPy и играют важную роль в эффективной обработке и анализе данных. Вот почему ndarrays так важны и как они используются:

Определение ndarrays:

Многомерные массивы в NumPy представляют собой таблицы однотипных элементов, индексированных кортежами целых чисел.

ndarrays могут иметь произвольное число измерений, что делает их мощным инструментом для представления и обработки различных видов данных.

Хранение данных:

Многомерные массивы в NumPy эффективно хранят данные в памяти благодаря своей компактной структуре.

Они предоставляют единый интерфейс для работы с данными, независимо от их размерности.

Удобство индексации и срезов:

ndarrays поддерживают индексацию и срезы, что позволяет легко обращаться к конкретным элементам массива или выбирать подмассивы.

Индексация в NumPy начинается с нуля, и можно использовать отрицательные индексы для обратного обхода массива.

Операции между массивами:

NumPy предоставляет множество операций для выполнения математических действий над массивами, включая сложение, умножение, транспонирование и многое другое.

Эти операции выполняются поэлементно, что улучшает читаемость кода и обеспечивает высокую производительность.

Применение в науке о данных и машинном обучении:

Многомерные массивы широко используются в науке о данных для представления и анализа многомерных данных, таких как изображения и временные ряды.

В машинном обучении, где эффективная работа с матрицами и тензорами критична, ndarrays становятся основой для представления и обработки данных.

Броадкастинг:

Благодаря концепции броадкастинга, ndarrays позволяют выполнять операции над массивами различных размеров, что делает код более компактным и уменьшает необходимость в явных циклах.

Интеграция с другими библиотеками:

Многомерные массивы NumPy являются стандартом для представления данных во многих других библиотеках, таких как pandas, scikit-learn, и TensorFlow, обеспечивая совместимость и легкость интеграции.

В целом, многомерные массивы в NumPy играют фундаментальную роль в обработке данных, научных вычислениях и машинном обучении, предоставляя удобный и эффективный способ хранения и манипулирования многомерными данными в языке программирования Python.

NumPy предоставляет богатый набор операций линейной алгебры и математических функций, что делает библиотеку мощным инструментом для выполнения различных вычислений. Вот некоторые ключевые операции и функции:

1. Умножение матриц:

np.dot(A, B) или A @ B: Выполняет умножение двух матриц.

2. Транспонирование:

A.T: Возвращает транспонированную матрицу.

3. Решение систем линейных уравнений:

np.linalg.solve(A, b): Решает систему линейных уравнений Ax = b.

4. Нахождение обратной матрицы:

np.linalg.inv(A): Возвращает обратную матрицу A.

5. Определитель матрицы:

np.linalg.det(A): Вычисляет определитель матрицы A.

6. Собственные значения и векторы:

np.linalg.eig(A): Возвращает собственные значения и векторы матрицы A.

7. QR-разложение:

Q, R = np.linalg.qr(A): Возвращает QR-разложение матрицы A.

8. Сингулярное разложение (SVD):

U, S, V = np.linalg.svd(A): Возвращает сингулярное разложение матрицы A.

9. Базовые математические операции:

np.add(x, y): Сложение элементов двух массивов.

np.subtract(x, y): Вычитание элементов одного массива из другого.

np.multiply(x, y): Поэлементное умножение массивов.

np.divide(x, y): Поэлементное деление массивов.

np.exp(x): Возведение в экспоненту для каждого элемента массива.

np.log(x): Натуральный логарифм для каждого элемента массива.

np.sqrt(x): Квадратный корень для каждого элемента массива.

10. Тригонометрические и гиперболические функции:

np.sin(x), np.cos(x), np.tan(x): Синус, косинус, тангенс для каждого элемента массива.

np.sinh(x), np.cosh(x), np.tanh(x): Гиперболический синус, косинус, тангенс для каждого элемента массива.

Эти операции и функции предоставляют мощный инструментарий для выполнения различных вычислений, связанных с линейной алгеброй и математикой, и делают NumPy неотъемлемой частью инфраструктуры для научных и инженерных вычислений на языке Python.

NumPy (Numerical Python) является фундаментальной библиотекой для научных вычислений в языке программирования Python. Ее основные характеристики делают ее незаменимым инструментом для работы с массивами данных, выполнения математических операций и анализа данных. Давайте рассмотрим основные черты, которые делают NumPy таким важным компонентом в области научных вычислений.

1. Многомерные массивы (ndarrays):

Основной структурой данных в NumPy является многомерный массив, или ndarray. Эти массивы предоставляют эффективный и гибкий способ представления и манипулирования данными. Одномерные, двумерные и многомерные массивы могут хранить данные любого типа.

2. Броадкастинг:

Механизм броадкастинга в NumPy позволяет выполнять операции между массивами различных форм, автоматически расширяя их размерности. Это существенно улучшает читаемость кода и снижает необходимость явного повторения данных, делая код более компактным и понятным.

3. Универсальные функции (ufuncs):

NumPy предоставляет богатый набор универсальных функций, которые позволяют выполнять поэлементные операции над массивами. Эти функции векторизованы, что увеличивает производительность и упрощает код. Универсальные функции включают арифметические операции, тригонометрические функции, логарифмы, степени и многие другие.

4. Операции линейной алгебры:

NumPy предоставляет функции для выполнения широкого спектра операций линейной алгебры, включая умножение матриц, нахождение определителя, обратной матрицы, собственных значений и векторов, а также разложения матриц (QR, SVD).

5. Индексация и срезы:

NumPy обладает мощными инструментами для индексации и срезов массивов. Индексация может осуществляться с использованием целочисленных значений, логических условий и даже функций. Это обеспечивает гибкость и удобство в выборе определенных элементов массива.

6. Генерация случайных чисел:

Библиотека включает модуль numpy.random, предоставляющий разнообразные функции для генерации случайных чисел. Это необходимо для симуляций, статистических вычислений и других областей, где случайность играет важную роль.

7. Функции изменения формы массивов:

NumPy предоставляет функции для изменения формы массивов, такие как reshape(), flatten(), resize(), что упрощает работу с данными различных размерностей и форм.

8. Интеграция с другими библиотеками:

NumPy легко интегрируется с другими библиотеками для научных вычислений, такими как SciPy, pandas и Matplotlib. Эта интеграция создает мощную среду для работы с данными, анализа и визуализации результатов.

9. Поддержка различных типов данных:

Библиотека NumPy поддерживает различные типы данных, включая целочисленные, числа с плавающей точкой, комплексные числа и другие. Это позволяет эффективно работать с различными видами данных.

10. Обработка ошибок и управление памятью:

NumPy обеспечивает удобный механизм обработки ошибок и эффективное управление памятью, что особенно важно при работе с большими объемами данных.

11. Богатая документация и активное сообщество:

NumPy обладает обширной документацией, включающей примеры кода и руководства, что облегчает изучение и использование библиотеки. Активное сообщество разработчиков также гарантирует поддержку и постоянное развитие библиотеки.

12. Поддержка параллельных вычислений:

NumPy предоставляет возможность использования многопоточности и параллельных вычислений, что может повысить производительность при выполнении некоторых операций.

13. Возможности для машинного обучения и искусственного интеллекта:

Благодаря своей эффективности и удобству использования, NumPy широко применяется в области машинного обучения и искусственного интеллекта. Многие библиотеки и фреймворки, такие как TensorFlow и PyTorch, используют NumPy как основу для манипуляции данными.

* 1. [***ПРЕИМУЩЕСТВО NamPy***](#_Toc7092967)

NumPy, библиотека для научных вычислений в языке программирования Python, предоставляет множество преимуществ, делая ее неотъемлемым инструментом для исследований, анализа данных и разработки моделей в области науки и инженерии. Рассмотрим ключевые преимущества NumPy, которые обеспечивают ее выдающуюся популярность и широкое использование.

1. Многомерные массивы (ndarrays):

Эффективное Хранение Данных: NumPy предоставляет многомерные массивы, или ndarrays, которые обеспечивают эффективное и оптимизированное хранение больших объемов данных. Это особенно важно для работы с изображениями, временными рядами, трехмерными данными и другими сложными структурами.

Быстрый Доступ и Манипуляции: Массивы NumPy обеспечивают быстрый и простой доступ к элементам, а также возможность проведения различных операций, таких как изменение формы, срезы и индексация. Это делает NumPy мощным инструментом для обработки данных.

2. Броадкастинг:

Удобные Операции с Разными Формами: Механизм броадкастинга в NumPy упрощает операции между массивами различных форм, позволяя автоматически транслировать массивы для выполнения операций. Это делает код более читаемым и компактным.

Экономия Памяти и Времени: Благодаря броадкастингу, нет необходимости явно копировать данные или изменять форму массивов перед выполнением операций. Это уменьшает объем занимаемой памяти и повышает производительность.

3. Универсальные функции (ufuncs):

Векторизованные Операции: Универсальные функции NumPy (ufuncs) обеспечивают векторизованные операции, что позволяет эффективно выполнять операции над целыми массивами данных, без использования явных циклов.

Широкий Набор Математических Функций: NumPy предоставляет обширный набор математических функций, включая тригонометрические, логарифмические, статистические и другие. Это позволяет легко выполнять сложные математические операции.

4. Операции Линейной Алгебры:

Эффективные Расчеты: NumPy предоставляет встроенные функции для выполнения операций линейной алгебры, таких как умножение матриц, нахождение определителя, собственных значений и векторов. Это существенно упрощает решение линейных систем уравнений и других задач.

Интеграция с BLAS и LAPACK: NumPy интегрируется с библиотеками BLAS (Basic Linear Algebra Subprograms) и LAPACK (Linear Algebra PACKage), что обеспечивает оптимизированные и высокопроизводительные вычисления.

5. Индексация и Срезы:

Гибкий и Удобный Доступ к Данным: NumPy обладает мощными средствами индексации и срезов, позволяя легко и гибко выбирать нужные части данных. Это особенно важно при работе с большими наборами данных.

Эффективное Использование Памяти: Использование срезов NumPy позволяет создавать представления массивов без копирования данных. Это способствует экономии памяти при работе с большими объемами данных.

6. Генерация Случайных Чисел:

Гибкий Генератор Случайных Чисел: Модуль numpy.random предоставляет множество функций для генерации случайных чисел различных распределений. Это необходимо для статистических вычислений, машинного обучения и других приложений.

Воспроизводимость Результатов: NumPy обеспечивает возможность установки начального состояния генератора случайных чисел, что позволяет воспроизводить результаты экспериментов.

7. Функции для Изменения Формы Массивов:

Гибкость в Манипуляциях с Формой: NumPy предоставляет функции для изменения формы массивов, такие как reshape(), flatten(), resize(). Это упрощает работу с данными различных размерностей и форм.

Производительность и Эффективность: Операции изменения формы выполняются без копирования данных, что повышает производительность и уменьшает использование памяти.

8. Интеграция с Другими Библиотеками:

Широкие Возможности Интеграции: NumPy легко интегрируется с другими библиотеками для научных вычислений, такими как SciPy, pandas, Matplotlib и многими другими. Эта совместимость обеспечивает комплексный инструментарий для анализа данных и визуализации результатов.

Поддержка Научных Проектов: Множество научных проектов и библиотек используют NumPy в качестве основы для обработки данных, что обеспечивает ее широкую поддержку в научном сообществе.

9. Поддержка Различных Типов Данных:

Работа с Разнообразными Типами Данных: NumPy поддерживает различные типы данных, включая целочисленные, числа с плавающей точкой, комплексные числа и другие. Это позволяет эффективно работать с разнообразными видами данных.

Настройка Точности: Возможность задания точности чисел позволяет балансировать требования к точности и использование памяти в зависимости от конкретной задачи.

10. Обработка Ошибок и Управление Памятью:

Эффективная Обработка Ошибок: NumPy предоставляет средства для обработки ошибок при выполнении операций, что обеспечивает безопасность и надежность кода.

Управление Памятью: NumPy умело управляет памятью, позволяя эффективно использовать ресурсы системы. Возможности оптимизации памяти важны при работе с массивами больших размеров.

11. Богатая Документация и Активное Сообщество:

Развернутая Документация: NumPy предоставляет обширную документацию с примерами кода, руководствами и объяснениями основных концепций. Это значительно облегчает процесс обучения и использования библиотеки.

Активное Сообщество: Сообщество разработчиков NumPy активно поддерживает и развивает библиотеку, обеспечивая обновления, исправления ошибок и новые функциональности.

12. Поддержка Параллельных Вычислений:

Возможности Многозадачности: NumPy предоставляет функциональность для использования многозадачности и параллельных вычислений, что повышает производительность в многозадачных сценариях.

Совместимость с Инструментами для Параллельных Вычислений: NumPy интегрируется с инструментами для параллельных вычислений, такими как Dask и Numba, что улучшает масштабируемость и производительность.

13. Поддержка Машинного Обучения и Искусственного Интеллекта:

Интеграция с Машинным Обучением: NumPy является основой для многих библиотек и фреймворков машинного обучения, таких как TensorFlow и scikit-learn. Это обеспечивает эффективную манипуляцию данными и обработку признаков.

Применение в Искусственном Интеллекте: NumPy широко используется в проектах и исследованиях по искусственному интеллекту благодаря своей эффективности и удобству в использовании.

**ГЛАВА 2. ПРИМЕНЕНИЕ NamPy**

В данной главе рассмотрим практические примеры и области применения библиотеки NumPy в научных вычислениях, обработке данных и разработке алгоритмов.

2.1. Матричные Операции и Линейная Алгебра:

NumPy предоставляет мощные инструменты для выполнения матричных операций и операций линейной алгебры. Рассмотрим пример умножения двух матриц:

import numpy as np

# Создаем две матрицы

matrix\_a = np.array([[1, 2], [3, 4]])

matrix\_b = np.array([[5, 6], [7, 8]])

# Умножение матриц

result = np.dot(matrix\_a, matrix\_b)

print(result)

Этот код выведет результат умножения матриц matrix\_a и matrix\_b. Такие операции широко используются в областях, требующих анализа данных, например, в машинном обучении.

2.2. Обработка и Анализ Данных:

NumPy облегчает работу с данными, предоставляя удобные методы для обработки и анализа. Рассмотрим пример вычисления среднего значения и стандартного отклонения:

import numpy as np

# Создаем массив данных

data = np.array([1, 2, 3, 4, 5])

# Вычисляем среднее и стандартное отклонение

mean\_value = np.mean(data)

std\_dev = np.std(data)

print("Среднее значение:", mean\_value)

print("Стандартное отклонение:", std\_dev)

Такие вычисления часто используются в статистике и анализе данных.

2.3. Генерация Случайных Чисел:

NumPy обеспечивает генерацию случайных чисел, что полезно при создании синтетических данных или в задачах стохастического моделирования:

import numpy as np

# Генерируем случайное число от 0 до 1

random\_number = np.random.rand()

print("Случайное число:", random\_number)

2.4. Обработка Изображений:

В области обработки изображений NumPy позволяет эффективно работать с пиксельными данными:

import numpy as np

import matplotlib.pyplot as plt

# Загружаем изображение

image = plt.imread('example\_image.jpg')

# Преобразуем изображение в оттенки серого

gray\_image = np.dot(image[...,:3], [0.2989, 0.5870, 0.1140])

# Выводим изображение

plt.imshow(gray\_image, cmap=plt.get\_cmap('gray'))

plt.show()

NumPy упрощает манипуляции с пикселями, обеспечивая удобный доступ к цветовым каналам.

2.5. Интеграция с Pandas:

NumPy хорошо интегрируется с библиотекой Pandas, расширяя возможности анализа данных. Например, конвертация Pandas DataFrame в массив NumPy:

import pandas as pd

import numpy as np

# Создаем DataFrame

data = {'A': [1, 2, 3], 'B': [4, 5, 6]}

df = pd.DataFrame(data)

# Преобразуем DataFrame в массив NumPy

numpy\_array = df.to\_numpy()

print(numpy\_array)

2.6. Вычисления с Временными Рядами:

NumPy предоставляет функциональность для работы с временными рядами, что полезно в финансовых и экономических приложениях:

import numpy as np

import matplotlib.pyplot as plt

# Генерация временного ряда

time = np.arange(0, 10, 0.1)

sin\_wave = np.sin(time)

# Визуализация временного ряда

plt.plot(time, sin\_wave)

plt.xlabel('Время')

plt.ylabel('Значение')

plt.title('Синусоидальный временной ряд')

plt.show()

Этот пример демонстрирует создание и визуализацию синусоидального временного ряда.

В данной главе мы рассмотрели лишь несколько примеров применения библиотеки NumPy. Эта библиотека предоставляет обширный инструментарий для решения различных задач в области научных вычислений, анализа данных и машинного обучения, что делает ее неотъемлемым компонентом в мире программирования на Python.

***2.1. ПРИМЕРЫ ИСПОЛЬЗОВАНИЯ***

1. Создание Массива:

import numpy as np

# Создание одномерного массива

array\_1d = np.array([1, 2, 3, 4, 5])

# Создание двумерного массива

array\_2d = np.array([[1, 2, 3], [4, 5, 6]])

print("Одномерный массив:")

print(array\_1d)

print("\nДвумерный массив:")

print(array\_2d)

2. Операции с Массивами:

import numpy as np

# Сложение двух массивов

array\_a = np.array([1, 2, 3])

array\_b = np.array([4, 5, 6])

result\_addition = array\_a + array\_b

# Умножение массива на число

result\_multiply = array\_a \* 2

print("Сложение массивов:")

print(result\_addition)

print("\nУмножение массива на число:")

print(result\_multiply)

3. Индексация и Срезы:

import numpy as np

# Индексация одномерного массива

array\_1d = np.array([1, 2, 3, 4, 5])

print("Элемент с индексом 2:", array\_1d[2])

# Срез двумерного массива

array\_2d = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])

print("Вторая строка, первые два элемента:", array\_2d[1, :2])

4. Операции Линейной Алгебры:

import numpy as np

# Умножение матриц

matrix\_a = np.array([[1, 2], [3, 4]])

matrix\_b = np.array([[5, 6], [7, 8]])

result\_matrix\_multiply = np.dot(matrix\_a, matrix\_b)

# Нахождение обратной матрицы

matrix\_inverse = np.linalg.inv(matrix\_a)

print("Умножение матриц:")

print(result\_matrix\_multiply)

print("\nОбратная матрица:")

print(matrix\_inverse)

5. Генерация Случайных Чисел:

import numpy as np

# Генерация случайных чисел от 0 до 1

random\_numbers = np.random.rand(3, 3)

print("Случайные числа:")

print(random\_numbers)

6. Математические Функции:

import numpy as np

# Вычисление синуса для каждого элемента массива

array = np.array([0, np.pi/2, np.pi])

result\_sin = np.sin(array)

print("Синус каждого элемента:")

print(result\_sin)

Эти примеры демонстрируют основные возможности NumPy, такие как создание массивов, математические операции, операции линейной алгебры и генерация случайных чисел.

**ЗАКЛЮЧЕНИЕ**

NumPy является одной из ключевых библиотек для научных вычислений и анализа данных в языке программирования Python. Вот основные выводы о роли и значимости NumPy:

Эффективные Массивы:

NumPy предоставляет мощные и эффективные многомерные массивы (ndarrays), которые обеспечивают удобное хранение и манипуляции данными.

Массивы NumPy эффективны в использовании памяти и обеспечивают быстрый доступ к элементам, что делает их идеальными для обработки больших объемов данных.

Операции Линейной Алгебры:

NumPy предоставляет богатый функционал для выполнения операций линейной алгебры, таких как умножение матриц, нахождение обратной матрицы, собственные значения и многое другое.

Интеграция с библиотеками BLAS и LAPACK обеспечивает высокую производительность при выполнении сложных математических операций.

Броадкастинг:

Механизм броадкастинга в NumPy упрощает операции между массивами различных форм, делая код более читаемым и компактным.

Броадкастинг позволяет выполнять операции без явного изменения формы массивов, что экономит память и повышает производительность.

Универсальные Функции:

NumPy предоставляет векторизованные операции, или универсальные функции (ufuncs), что позволяет эффективно выполнять операции над целыми массивами данных.

Обширный набор математических функций упрощает выполнение различных вычислений.

Обработка Изображений и Видео:

NumPy широко используется в области обработки изображений и видео, обеспечивая эффективные методы работы с пиксельными данными.

Возможности интеграции с библиотеками для компьютерного зрения делают NumPy важным инструментом в области компьютерного зрения и обработки изображений.

Интеграция с Pandas и Другими Библиотеками:

NumPy легко интегрируется с другими популярными библиотеками, такими как Pandas, SciPy, Matplotlib и многими другими.

Использование NumPy в комбинации с этими библиотеками обеспечивает широкий спектр функциональности для анализа данных, статистики и визуализации.

Поддержка Временных Рядов и Статистических Вычислений:

NumPy предоставляет удобные инструменты для работы с временными рядами, что важно в финансовой аналитике и экономических исследованиях.

Вычисления среднего, стандартного отклонения и других статистических параметров делают NumPy незаменимым для анализа данных.

Роль в Машинном Обучении и Искусственном Интеллекте:

NumPy является основой для многих библиотек и фреймворков машинного обучения, таких как TensorFlow и scikit-learn.

Его эффективность и удобство в использовании делают NumPy популярным среди исследователей и инженеров в области искусственного интеллекта.

NumPy занимает центральное место в экосистеме Python для научных вычислений и анализа данных. Его мощные инструменты, эффективные структуры данных и широкий функционал делают его незаменимым инструментом для исследователей, разработчиков и специалистов в области данных. Роль NumPy в развитии Python в области научных вычислений трудно переоценить, и его вклад продолжает расширяться в сферах исследований, разработки и инноваций.

[**СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ**](file:///C:\Users\User\Downloads\курсовая%20работа.docx#_Toc7092976)

1. <https://habr.com/ru/articles/352678/>
2. <https://pythonworld.ru/numpy/1.html>
3. <https://blog.skillfactory.ru/glossary/numpy/>
4. <https://ru.m.wikipedia.org/wiki/NumPy>
5. <https://habr.com/ru/articles/353416/>