NUMPY KUTUBXONASI VA UNING IMKONIYATLARI

Tojimamatov Israil Nurmamatovich
Farg‘ona davlat universiteti Amaliy matematika va informatika kafedrasi katta o’qituvchisi
E-mail: israiltojimamatov@gmail.com
Ergashboyev Muhammadrasul Ibrohim o‘g‘li
Farg‘ona davlat universiteti Amaliy matematika yo‘nalishi 3-bosqich 23.10-guruh talabasi
E-mail: muhammadrasulergashboyev20@gmail.com

ANNOTATSIYA: Ushbu fundamental ilmiy maqola Python dasturlash tilining NumPy (Numerical Python) kutubxonasining nazariy asoslari, asosiy imkoniyatlari va zamonaviy ilmiy hisoblashlardagi ahamiyatini keng qamrovli tahlil qilishga bag‘ishlangan. NumPy ilmiy hisoblashlar ekotizimining ajralmas qismi bo‘lib, u katta hajmdagi ma’lumotlar bilan samarali ishlash uchun asosiy ma’lumotlar tuzilmasi – **N-o‘lchovli massiv (ndarray)**ni taqdim etadi. Tadqiqotda ndarray ning ishlash tezligi, xususan, oddiy Python ro‘yxatlariga nisbatan vektorlashtirilgan operatsiyalarni qo‘llash orqali erishilgan ustunligi batafsil yoritiladi. Maqola, shuningdek, NumPy ning fundamental matematik funktsiyalarni, chiziqli algebra amallarini, Furye transformatsiyalarini va tasodifiy sonlarni generatsiya qilish mexanizmlarini amalga oshirishdagi markaziy rolini nazariy jihatdan asoslaydi. Tadqiqotimiz NumPy ning nafaqat mustaqil vosita, balki Pandas, SciPy va Scikit-learn kabi boshqa ilg‘or kutubxonalar uchun zamonaviy ma’lumotlar fanining poydevori ekanligini isbotlaydi.

Kalit So‘zlar: NumPy, N-o‘lchovli Massiv (ndarray), Ilmiy Hisoblashlar, Vektorlashtirish, Chiziqli Algebra, Ma’lumotlar Tahlili, Python, Algoritmik Samaradorlik.

АННОТАЦИЯ: Данная фундаментальная научная статья посвящена всестороннему анализу теоретических основ, ключевых возможностей и значимости библиотеки NumPy (Numerical Python) в языке программирования Python для современных научных вычислений. NumPy является неотъемлемой частью экосистемы научных вычислений, предоставляя базовую структуру данных – N-мерный массив (ndarray) – для эффективной работы с большими объемами данных. В исследовании подробно освещается скорость работы ndarray, в частности, его преимущество перед стандартными списками Python, достигаемое за счет использования векторизованных операций. Статья также теоретически обосновывает центральную роль NumPy в реализации фундаментальных математических функций, операций линейной алгебры, преобразований Фурье и механизмов генерации случайных чисел. Наше

исследование доказывает, что NumPy является не только самостоятельным инструментом, но и фундаментом современной науки о данных для других продвинутых библиотек, таких как Pandas, SciPy и Scikit-learn.

Ключевые Слова: NumPy, N-мерный Массив (ndarray), Научные Вычисления, Векторизация, Линейная Алгебра, Анализ Данных, Python, Алгоритмическая Эффективность.

ANNOTATION: This fundamental scholarly article is dedicated to a comprehensive analysis of the theoretical underpinnings, key capabilities, and significance of the NumPy (Numerical Python) library in the Python programming language for contemporary scientific computing. NumPy is an integral part of the scientific computing ecosystem, providing the core data structure – the N-dimensional array (ndarray) – for efficient manipulation of large datasets. The study thoroughly illuminates the operational speed of the ndarray, particularly its advantage over standard Python lists, achieved through the use of vectorized operations. The article also theoretically substantiates NumPy's central role in implementing fundamental mathematical functions, linear algebra operations, Fourier transformations, and random number generation mechanisms. Our research confirms that NumPy is not only a standalone tool but also the foundation of modern data science for other advanced libraries such as Pandas, SciPy, and Scikit-learn.

Keywords: NumPy, N-dimensional Array (ndarray), Scientific Computing, Vectorization, Linear Algebra, Data Analysis, Python, Algorithmic Efficiency.

Kirish

Zamonaviy fan va muhandislik sohalarida, jumladan, ma’lumotlar fanida, mashinani o‘rganishda va statistik tahlilda, yuqori samarali hisoblashlarga bo‘lgan talab tobora ortib bormoqda. Python dasturlash tili o‘zining sintaktik soddaligi va kengayish imkoniyatlari tufayli bu sohalarda dominant tilga aylandi. Bu muvaffaqiyatning asosi, shubhasiz, NumPy (Numerical Python) kutubxonasi bilan bog‘liqdir.

NumPy ning paydo bo‘lishi Python tilida ilmiy hisoblashlar sohasida inqilobiy o‘zgarish yasadi. Uning tarixi, asosan, Jim Xyuginning Numeric kutubxonasi va Trey Bidlerning Numarray kutubxonalarining integratsiyasi va evolyutsiyasi bilan bog‘liq bo‘lib, bu jarayon Travis Olifant tomonidan amalga oshirildi va hozirgi kundagi yagona, kuchli platformaga asos solindi.

NumPy ning asosiy maqsadi — katta hajmdagi ma’lumotlarni samarali saqlash va ular ustida tezkor operatsiyalarni bajarish uchun C/C++ tillarida yozilgan optimallashtirilgan yadro funksiyalarini Python interfeysi orqali taqdim etish. Bu kombinatsiya Pythonning ishlab chiqish tezligini C ning bajarish tezligi bilan birlashtiradi.

Ushbu ilmiy maqolaning maqsadi NumPy kutubxonasining asosiy ma’lumotlar tuzilmasini, ya’ni N-o‘lchovli massiv (ndarray) ni nazariy tahlil qilish, uning operatsiyalarni vektorlashtirish mexanizmini tushuntirish va chiziqli algebra kabi yuqori darajadagi matematik imkoniyatlarini ilmiy asoslashdan iborat. Tadqiqotimiz NumPy ning nafaqat algoritmlarni tezlashtirishda, balki ilmiy tadqiqotlarning nazariy asosini mustahkamlashdagi strategik ahamiyatini ham ta’kidlaydi.

Asosiy Qism

NumPy kutubxonasining butun kuchi uning markaziy ma’lumotlar tuzilmasi – N-o‘lchovli massiv (ndarray) da mujassamlashgan. Bu massivlar Pythonning standart ro‘yxatlaridan tubdan farq qiladi va ilmiy hisoblashlar uchun ulkan nazariy ustunlikni taqdim etadi.

ndarray – bu ma’lumotlarni bir xil turdagi elementlar majmuasi sifatida saqlaydigan, bir xil xotira blokida joylashgan va strukturasi jihatidan tartiblangan ma’lumotlar to‘plamidir. Standart Python ro‘yxatlaridan farqli o‘laroq, ndarray ning barcha elementlari bir xil turga (masalan, float oltmish to‘rt yoki butun son o‘ttiz ikki) ega bo‘lishi shart. Bu xususiyat ma’lumotlarning xotirada uzluksiz joylashishini ta’minlaydi. Uzluksiz joylashuv esa ma’lumotlarga kirishni optimallashtiradi va xotiradan foydalanishning referensiyalar (havolalar) o‘rniga to‘g‘ridan-to‘g‘ri qiymatlar orqali amalga oshirilishini ta’minlaydi.

Bu anatomik ustunlik ikki jihatdan muhim: Birinchidan, xotiradan samarali foydalanish: Har bir element uchun xotirada ma’lum bir bayt miqdori qat’iy ajratilgan bo‘ladi, bu esa xotiraning isrofgarchiligini kamaytiradi. Ikkinchidan, tezkor kirish: Ma’lumotlar uzluksiz joylashganligi sababli, protsessorning kash (cache) mexanizmi ularga tezroq kirish imkoniyatini beradi, bu esa hisoblash tezligini keskin oshiradi.

NumPy ning asosiy nazariy afzalligi bu vektorlashtirish (Vectorization) tamoyilidir. Vektorlashtirish – bu massivning barcha elementlari ustida bir vaqtning o‘zida operatsiyalarni bajarish usulidir. Bu, Pythonning standart tsikllaridan (masalan, for tsikli) foydalanishdan tubdan farq qiladi. Python tsikllari ma’lumotlar ustida operatsiyalarni elementma-element bajaradi, bu esa har bir iteratsiyada Python interpretatorining overhead (ortiqcha yuk) bilan ishlashini talab qiladi.

Vektorlashtirilgan operatsiyalar esa bevosita past darajadagi C yoki Fortran kutubxonalari (masalan, BLAS va LAPACK) orqali amalga oshiriladi. Bu protsessorning SIMD (Single Instruction, Multiple Data - Yagona Qo‘llanma, Ko‘p Ma’lumot) kabi maxsus buyruqlaridan foydalanish imkonini beradi. Natijada, bitta Python kodi butun bir massiv ustida tezkor hisoblashni amalga oshiradi, bu esa ishlash tezligini nazariy jihatdan o‘nlab, ba’zan yuzlab marta oshiradi.

Broadcasting (eshittirish) – bu har xil shakldagi (o‘lchamdagi) massivlar ustida arifmetik operatsiyalarni bajarish imkonini beruvchi kuchli nazariy qoida. NumPy ning bu mexanizmi kichik massivni yoki skalyar qiymatni katta massivning shakliga

(o‘lchamiga) mos keladigan darajada virtual ravishda takrorlash orqali ishlaydi. Bu takrorlash xotirada haqiqiy nusxa olishni talab qilmaydi, faqat mantiqiy darajada amalga oshiriladi.

Broadcasting ning asosiy tamoyili: ikki massivning o‘lchamlari (shape) o‘ng tomondan chapga qarab taqqoslanadi. Agar o‘lchamlar mos kelsa yoki ulardan biri birga teng bo‘lsa, operatsiya bajariladi. Bu mexanizm dasturchiga katta hajmdagi ma’lumotlar ustida qisqa va o‘qilishi oson kod yozish imkonini beradi, chunki qo‘shimcha tsikllar va shartli tekshiruvlarga ehtiyoj qolmaydi.

NumPy kutubxonasi nafaqat ma’lumotlarni samarali saqlaydi, balki yuqori darajadagi matematik hisoblashlar uchun keng va chuqur funksional imkoniyatlarni taqdim etadi. Bu imkoniyatlar ilmiy tadqiqotlarning deyarli barcha jabhalarida asosiy hisoblanadi.

NumPy dagi universal funksiyalar (ufunc) massivlar ustida tezkor operatsiyalarni bajarish uchun mo‘ljallangan funksiyalar to‘plamidir. Bu funksiyalar (masalan, trigonometrik funksiyalar, eksponent, logarifm) elementma-element asosda ishlaydi va yuqorida muhokama qilingan vektorlashtirish tamoyilidan to‘liq foydalanadi. Ufunc larning nazariy ahamiyati shundaki, ular ma’lumot turini (dtype) mustaqil ravishda boshqarish imkonini beradi, bu esa hisoblashda yuqori aniqlik va tezlikni ta’minlaydi. Ular ma’lumotlar tahlilida katta massivlarga murakkab matematik transformatsiyalarni qo‘llash uchun eng tezkor yechimdir.

Chiziqli algebra ilmiy hisoblashlar va mashinani o‘rganishning markaziy asosi hisoblanadi. NumPy kutubxonasi ushbu operatsiyalarni amalga oshirish uchun LAPACK va BLAS kabi sanoat standartidagi optimallashtirilgan past darajadagi kutubxonalarga asoslangan kuchli modulni taqdim etadi.

NumPy yordamida matritsalarni ko‘paytirish, matritsaning teskarisini topish, determinantni hisoblash, xos qiymatlarni (eigenvalues) va xos vektorlarni (eigenvectors) topish kabi murakkab amallar samarali bajariladi. Ayniqsa, ikki matritsaning ko‘paytmasi kabi hisoblash talab qiladigan operatsiyalar vektorlashtirilgan yondashuv orqali oddiy Python tsikllariga qaraganda juda tez amalga oshiriladi. Mashinani o‘rganishda, masalan, neyron tarmoqlarning og‘irliklarini hisoblashda, bu tezlik hal qiluvchi ahamiyatga ega.

NumPy, shuningdek, Furye transformatsiyalarini (xususan, Tez Furye Transformatsiyasini - FFT) amalga oshirish uchun maxsus modulni o‘z ichiga oladi. Furye transformatsiyasi signallarni qayta ishlash, tasvirni tahlil qilish va chastotali tahlillarda keng qo‘llaniladi. NumPy ning bu imkoniyati signallarni vaqt domenidan chastota domeniga tezkor o‘tkazish imkonini beradi.

Bundan tashqari, tasodifiy sonlar generatsiyasi NumPy ning muhim komponentidir. Ushbu mexanizm Monte-Karlo simulyatsiyalari, statistik modellashtirish va mashinani o‘rganish modellarini (og‘irliklarni boshlang‘ich

qiymatlarga keltirish) yaratishda zarurdir. NumPy turli ehtimollik taqsimotlaridan (masalan, normal, bir xil, binomial) yuqori sifatli va tezkor tasodifiy sonlarni generatsiya qilish imkoniyatini taqdim etadi.

NumPy ning asosiy ma’lumotlar tuzilmasi bo‘lgan ndarray nafaqat matematik operatsiyalarni tezlashtiradi, balki ma’lumotlar fanining boshqa asosiy vositalari bilan bevosita va samarali integratsiya qilishning yagona platformasini yaratadi.

NumPy massivlarida ma’lumotlarga kirish va ularni tanlash (selektsiya) uchun oddiy Python ro‘yxatlaridan ancha ilg‘or bo‘lgan maxsus indeksatsiya mexanizmlari mavjud. Bunga boolean maskalash va butun sonli massiv indeksatsiyasi kiradi. Boolean maskalash mexanizmi massivning bir xil o‘lchamli boolean (mantiqiy) massivi yordamida ma’lum shartlarga javob beradigan elementlarni tezkor tanlash imkonini beradi. Bu usul, an’anaviy dasturlashda talab qilinadigan shartli tsikllardan qochib, to‘g‘ridan-to‘g‘ri ma’lumotni filtrlaydi va vektorlashtirish tamoyilini saqlab qoladi. Butun sonli massiv indeksatsiyasi esa massivdan tartibsiz, ixtiyoriy joylashgan elementlar yoki satrlarni bir vaqtning o‘zida tanlab olishga imkon beradi, bu esa tasodifiy namunalar olish yoki ma’lumotlarni qayta tartiblash uchun juda qulaydir.

NumPy ning muhim nazariy ahamiyati shundaki, u butun Python ilmiy ekotizimi uchun universal ma’lumot almashish formati bo‘lib xizmat qiladi. Pandas kutubxonasining markaziy ma’lumotlar tuzilmalari bo‘lgan Series va DataFrame lar o‘zlarining ichki xotira vakilligida bevosita NumPy ndarraylaridan foydalanadi. Bu shuni anglatadiki, Pandas ma’lumotlar to‘plamlarini saqlashning qulay interfeysini taklif qilsa-da, har qanday hisoblash yoki statistik operatsiyalar bajarilganda, ular oxir-oqibat NumPy ning yuqori tezlikdagi vektorlashtirilgan funksiyalari orqali bajariladi.

SciPy (Scientific Python) kutubxonasi NumPy ustiga qurilgan bo‘lib, u optimallashtirish, integratsiya, signallarni qayta ishlash va ehtimollik taqsimotlari kabi yanada murakkab ilmiy va muhandislik masalalarini hal qilish uchun algoritmlarni taqdim etadi. SciPy dagi har qanday funksiya ma’lumotlarni qabul qilish va natijalarni qaytarish uchun ndarray formatini talab qiladi, bu esa ikkala kutubxonaning uzviy va samarali ishlashini ta’minlaydi.

NumPy ning imkoniyatlari ko‘plab ilmiy va texnologik sohalarda fundamental yechim sifatida qo‘llaniladi. Mashinani o‘rganish (Machine Learning)da NumPy ning ahamiyati beqiyosdir. Har qanday murakkab model, masalan, neyron tarmoqlar, o‘zining og‘irliklari (weights) va xususiyat (feature) vektorlarini ndarray formatida saqlaydi. Mashinani o‘rganishning asosiy operatsiyalari – gradientlarni hisoblash, xato (loss) funksiyasini optimallashtirish va teskari tarqalish (backpropagation) jarayonlari – matritsali ko‘paytirish kabi chiziqli algebra amallariga asoslanadi. NumPy ning ushbu operatsiyalarni yuqori tezlikda bajarish qobiliyati mashinani o‘rganish modellarini katta ma’lumotlar to‘plamlarida samarali o‘qitish imkonini beradi.

Signallarni qayta ishlashda (Signal Processing) NumPy ning Tez Furye Transformatsiyasi (FFT) imkoniyatlari hal qiluvchi rol o‘ynaydi. Tovush, radiolokatsiya yoki telekommunikatsiya signallari ndarray sifatida ifodalanadi, keyin esa ular ustida signallarni filtrlash, shovqinni yo‘qotish va chastotali tahlil qilish uchun FFT kabi operatsiyalar qo‘llaniladi.

Tasvirni qayta ishlash sohasida ham har bir tasvir piksellar qiymatlarining ikki yoki uch o‘lchovli NumPy massivi sifatida ifodalanadi. Tasvirga filtrlar (masalan, chekkalarni aniqlash) qo‘llash, ranglarni o‘zgartirish yoki tasvirlarni manipulyatsiya qilish kabi barcha amallar vektorlashtirilgan massiv operatsiyalari orqali juda tez amalga oshiriladi.

Statistik tahlil va modellashtirishda katta ma’lumotlar to‘plamlarining statistik xususiyatlarini (o‘rtacha qiymat, dispersiya, standart og‘ish, korrelyatsiya) hisoblash uchun NumPy ning vektorlashtirilgan statistik funksiyalaridan foydalaniladi. Bu hisoblashlarni an’anaviy usullar bilan bajarishdan ko‘ra sezilarli darajada tezroqdir. Bundan tashqari, Monte-Karlo simulyatsiyalari kabi murakkab simulyatsiya usullari NumPy ning yuqori sifatli tasodifiy sonlar generatsiyasi mexanizmlariga tayanadi. Bu mexanizm ko‘p sonli ehtimoliy senariylarni tezkor yaratish va ularning natijalarini statistik tahlil qilish imkonini beradi.

NumPy ning algoritmlarni bajarishdagi yuqori samaradorligi uning ichki tuzilmasi bilan chambarchas bog‘liq. NumPy massivlari Python ro‘yxatlariga nisbatan ishlash tezligining nazariy ustunligi ikki omilga asoslanadi: past darajadagi implementatsiya va ma’lumotlarning bir xil turi. C tilida yozilgan yadro funksiyalari va ularning optimallashtirilgan C, Fortran kutubxonalariga tayanishi Python Global Interpreter Lock (GIL) ning cheklovlarini chetlab o‘tishga imkon beradi, bu esa ba’zi operatsiyalarda ko‘p treda ishlash (multithreading) samaradorligiga yaqinlashishni ta’minlaydi. Ma’lumotlarning bir xil turga ega bo‘lishi esa tipni tekshirish (type checking) jarayonini bartaraf etadi va protsessorning vektorli qayta ishlashdan to‘liq foydalanishini ta’minlaydi.

NumPy ning xotiradan samarali foydalanishining muhim mexanizmi bu "ko‘rinishlar" (Views) tushunchasidir. Massivdan ma’lumotlarning kichik qismini kesib olish (slicing) operatsiyasida NumPy odatda ma’lumotlarning haqiqiy nusxasini yaratmaydi. Buning o‘rniga, u asl massivning xotira blokiga ishora qiluvchi yangi ko‘rinish yaratadi. Bu yondashuv ma’lumotlarni manipulyatsiya qilishda xotira sarfini minimal darajada ushlab turishga va yuqori samaradorlikka erishishga imkon beradi.

NumPy ning kelajakdagi tadqiqot yo‘nalishlari parallellashtirish va heterogen hisoblashlarga (GPU, TPU) qaratilgan. Dask va CuPy kabi kutubxonalar NumPy ga o‘xshash interfeysni taqdim etish orqali bu cheklovlarni yengib o‘tishga harakat qilmoqda, ammo NumPy ning o‘zi ham past darajadagi yadro funksiyalarini zamonaviy apparat arxitekturalariga moslashtirish ustida ishlamoqda. Bu esa Python

ilmiy hisoblashlarining kelajakda ham eng yuqori darajada samarali bo‘lishini ta’minlaydi.

NumPy ning xotiradan samarali foydalanishi ko‘rinishlar (Views) tushunchasiga asoslanadi. Oddiy Python ro‘yxatlaridan farqli o‘laroq, massivni kesib olish (slicing) yoki qayta shakllantirish (reshaping) kabi ko‘plab operatsiyalar ma’lumotlarning haqiqiy nusxasini yaratmaydi. Buning o‘rniga, yangi massiv obyekti asl massivning xotira blokiga ishora qiluvchi, lekin o‘zining "qadamlar" (strides) va shakl (shape) parametrlariga ega bo‘lgan ko‘rinishni yaratadi.

Strides parametri massivning xotirada joylashgan har bir o‘lchovi bo‘yicha keyingi elementga o‘tish uchun qancha bayt o‘tkazib yuborish kerakligini belgilaydi. Ko‘rinishlar tufayli massivning elementlarini almashtirish, o‘lchamlarini o‘zgartirish yoki hatto transponirlash (transpose) kabi amallar deyarli lahzada amalga oshiriladi, chunki xotirada katta ma’lumotlar to‘plamini ko‘chirishga hojat qolmaydi.

NumPy dtype (data type) tizimi Pythonning standart tiplaridan ancha keng va murakkabdir. Bu tizim har bir massiv elementining xotirada qancha joy egallashini va undagi ma’lumotlarning aniq turini (masalan, float 16, float 32, float 64, kompleks sonlar) qat’iy belgilash imkonini beradi. Bu qat’iylik uchta muhim afzallikni beradi:

Birinchidan, Xotirani optimallashtirish: Katta massivlarda kerakli aniqlikka qarab kichikroq tiplarni (masalan, float 32) tanlash orqali xotira sarfini keskin kamaytirish mumkin.

Ikkinchidan, Apparat bilan moslashuv: Ma’lumotlar tipini aniq belgilash protsessorning SIMD (Single Instruction, Multiple Data) kabi vektorli qayta ishlash buyruqlaridan to‘liq foydalanishini ta’minlaydi.

Uchinchidan, Strukturaviy massivlar: NumPy strukturaviy (structured) massivlarni yaratish imkonini beradi, bu esa massivning har bir elementini turli tiplardagi maydonlar to‘plami (masalan, har bir elementda ism (string) va yosh (integer)) sifatida saqlashga imkon beradi. Bu ma’lumotlar bazasi yozuvlarini yoki murakkab ilmiy ma’lumotlar to‘plamini samarali boshqarishda juda muhimdir.

NumPy dagi Universal Funksiyalar (ufunc) ning ishlash tezligi C tilida yozilganligi bilan belgilanadi. Biroq, NumPy ning imkoniyati shundaki, foydalanuvchilar o‘zlari ham yangi custom ufunc larni yaratishlari mumkin. Bu Cython yoki Numba kabi vositalar yordamida Python kodini yuqori samarali mashina kodiga kompilyatsiya qilish orqali amalga oshiriladi. Ushbu texnologik yondashuv Python tilida yozilgan algoritmlarning C/Fortran tezligida ishlashini ta’minlaydi, bu esa maxsus ilmiy vazifalar uchun juda muhimdir.

NumPy ning o‘zi toza Python darajasida Global Interpreter Lock (GIL) tufayli ko‘p yadroli protsessorlarning to‘liq parallellashtirish imkoniyatini cheklaydi. Lekin NumPy ning ko‘plab ichki funksiyalari (masalan, matritsani ko‘paytirish) past darajadagi kutubxonalarga tayanadi va ular GIL ni bo‘shatadi.

Biroq, zamonaviy ma’lumotlar fanida katta ma’lumotlar to‘plamlarida massiv hisoblashni talab qiladigan vazifalar uchun Dask kabi kutubxonalar NumPy massivlarini avtomatik ravishda kichik bloklarga ajratadi. Bu bloklar bir vaqtning o‘zida bir nechta protsessor yadrolarida yoki hatto taqsimlangan klasterlarda qayta ishlanadi. Bu mexanizm NumPy ning massiv tuzilmasidan foydalangan holda tarqatilgan hisoblashlarni samarali amalga oshirishga imkon beradi, bu esa Big Data muhitlarida ishlash uchun muhimdir.

GPU (Graphics Processing Unit) hisoblashlarining mashhurligi ortishi bilan, NumPy massivlarini bevosita GPU da qayta ishlashga imkon beruvchi CuPy kabi kutubxonalar rivojlandi. CuPy deyarli NumPy kabi interfeysga ega, lekin u ishlash uchun CUDA texnologiyasidan foydalanadi. Bu esa massiv operatsiyalarining tezligini, ayniqsa chuqur o‘rganish va katta matritsali hisoblashlarda, keskin oshiradi. NumPy ning universal ndarray spetsifikatsiyasi bu kabi ilg‘or va heterogen hisoblash muhitlariga o‘tish uchun bevosita yo‘l ochib berdi.

Xulosa

Tadqiqot natijalari shuni ko‘rsatadiki, NumPy ning markaziy ma’lumotlar tuzilmasi bo‘lgan N-o‘lchovli massiv (ndarray) ma’lumotlarni uzluksiz xotirada joylashtirish, bir xil tipni ta’minlash va C/Fortran kabi past darajadagi tillarda optimallashtirilgan yadro funksiyalaridan foydalanish orqali Pythonning standart ma’lumotlar tuzilmalariga nisbatan mislsiz algoritmic samaradorlikni taqdim etadi.

Vektorlashtirish va broadcasting mexanizmlari dastur kodining soddaligini saqlagan holda, operatsiyalarni tezkor bajarishni ta’minlaydigan nazariy tamoyillardir. NumPy ning ufunc funksiyalari, chiziqli algebra imkoniyatlari va Furye transformatsiyalari kabi matematik apparatlarni o‘zida mujassam etishi, uni ma’lumotlar tahlili, mashinani o‘rganish, signallarni qayta ishlash va statistik simulyatsiyalar uchun asosiy poydevorga aylantirgan.

Xulosa qilib aytganda, NumPy nafaqat bir kutubxona, balki Python tilida yuqori samarali ilmiy tadqiqotlar o‘tkazish uchun paradigma siljishini ifodalaydi. Uning SciPy va Pandas kabi boshqa ilg‘or vositalar bilan uzviy integratsiyasi, ilmiy jamoatchilikka murakkab nazariy muammolarni amaliy jihatdan hal qilish uchun zarur bo‘lgan mustahkam va tezkor vositani taqdim etadi.

Foydalanilgan Adabiyotlar

1. Oliphant T E. A Guide to NumPy. NumPy bo‘yicha qo‘llanma. Trelgol Publishing.
2. Peterson P, J. The NumPy Array: A Structure for Efficient Scientific Computation. NumPy massivi: samarali ilmiy hisoblash uchun tuzilma. *Computing in Science & Engineering* jurnali.
3. Harris C R, Millman K J, van der Walt S J, et al. Array programming with NumPy. NumPy bilan massiv dasturlash. *Nature* jurnali.

4. Van Der Walt S, Colbert S C, Varoquaux G. The NumPy Array: A Building Block for Scientific Computing in Python. NumPy massivi: Pythonda ilmiy hisoblash uchun asosiy qurilish bloki. *Computing in Science & Engineering* jurnali.
5. McKinney W. Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython. Ma’lumotlar tahlili uchun Python: Pandas, NumPy va IPython bilan ma’lumotlarni saralash. O'Reilly Media.
6. Hunter J D. Matplotlib: A 2D graphics environment. Matplotlib: ikki o‘lchovli grafik muhiti. *Computing in Science & Engineering* jurnali.
7. Jones E, Oliphant T E, Peterson P et al. SciPy: Open source scientific tools for Python. SciPy: Python uchun ochiq manbali ilmiy vositalar. Python Rasmiy Hujjatlari.