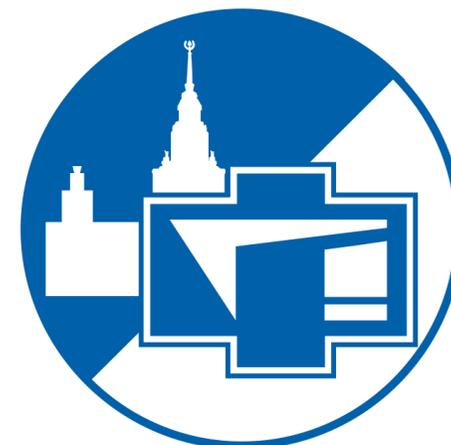


Нитру и компания

Змея выползает на тропу производительности



Скорость – как сделать больше за то же время



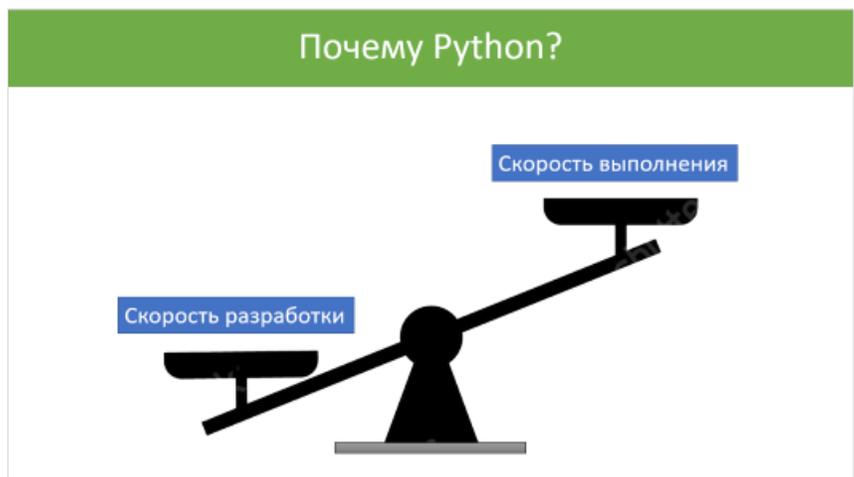
Оптимизация алгоритмов

- Обоснованные и необоснованные предположения

Параллелизация обработки

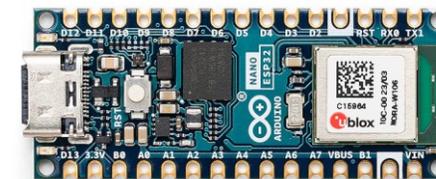
- Внутри программы
- Средствами ОС/сторонних программ

Сторонние библиотеки



Примеры применения в современной физике

- Обработка (в том числе статистическая) данных, их визуализация
- Численное моделирование – в задачах, менее масштабных, чем high performance computing
 - И даже тут Python хорош для прототипирования
- Символьные вычисления (но не дотягивает до лидеров отрасли)
- Автоматизация эксперимента – например, работа с USB устройствами (датчики etc)
 - MicroPython есть даже для Arduino



Стандартная библиотека

random

Генерация случайных чисел

`random.randrange(stop)`
`random.randrange(start, stop[, step])`
`random.randint(a, b)`
`random.choice(seq)`
`random.shuffle(x)`
`random.uniform(a, b)`
`random.random()`
`random.choices(population, weights=None, *, cum_weights=None, k=1)`

```
>>> random() # Random float: 0.0 <= x < 1.0
0.37444887375646646
>>> uniform(2.5, 10.4) # Random float: 2.5 <= x <= 10.4
3.1800140873117523
>>> expovariate(1 / 5) # Interval between arrivals averaging 5 seconds
5.148957171865801
>>> randrange(10) # Integer from 0 to 9 inclusive
7
>>> randrange(0, 101, 2) # Even integer from 0 to 100 inclusive
26
>>> choice(['win', 'lose', 'draw']) # Single random element from a sequence
'draw'
>>> deck = "ace two three four".split()
>>> shuffle(deck) # Shuffle a list
>>> deck
['four', 'two', 'ace', 'three']
>>> sample([10, 20, 30, 40, 50], k=4) # Four samples without replacement
[40, 10, 30, 50]
```

<https://docs.python.org/3/library/random.html>

numpy

math

```
module 'math' (built-in)
This module provides access to the mathematical functions
defined by the C standard.

e = 2.718281828459045
inf = inf
nan = nan
pi = 3.141592653589793
tau = 6.283185307179586
acos = def acos(x, /): Return the arc cosine (measured in radians) of x.
acosh = def acosh(x, /): Return the inverse hyperbolic cosine of x.
asin = def asin(x, /): Return the arc sine (measured in radians) of x.
asinh = def asinh(x, /): Return the inverse hyperbolic sine of x.
atan = def atan(x, /): Return the arc tangent (measured in radians) of x.
atan2 = def atan2(y, x, /): Return the arc tangent (measured in radians) of y/x.
atanh = def atanh(x, /): Return the inverse hyperbolic tangent of x.
ceil = def ceil(x, /): Return the ceiling of x as an Integral.
comb = def comb(n, k, /): Number of ways to choose k items from n items without repetition and without order.
copysign = def copysign(x, y, /): Return a float with the magnitude (absolute value) of x but the sign of y.
cos = def cos(x, /): Return the cosine of x (measured in radians).
cosh = def cosh(x, /): Return the hyperbolic cosine of x.
degrees = def degrees(x, /): Convert angle x from radians to degrees.
dist = def dist(p, q, /): Return the Euclidean distance between two points p and q.
erf = def erf(x, /): Error function at x.
erfc = def erfc(x, /): Complementary error function at x.
exp = def exp(x, /): Return e raised to the power of x.
expm1 = def expm1(x, /): Return exp(x)-1.
fabs = def fabs(x, /): Return the absolute value of the float x.
factorial = def factorial(x, /): Find x!.
floor = def floor(x, /): Return the floor of x as an Integral.
fmod = def fmod(x, y, /): Return fmod(x, y), according to platform C.
frexp = def frexp(x, /): Return the mantissa and exponent of x, as pair (m, e).
fsum = def fsum(seq, /): Return an accurate floating point sum of values in the iterable seq.
gamma = def gamma(x, /): Gamma function at x.
gcd = def gcd(*integers): Greatest Common Divisor.
hypot = def hypot(*coordinates): Return the value
isclose(a, b, *, rel_tol=1e-09, abs_tol=0.0): Determine whether two floating point numbers are close in value.
isfinite = def isfinite(x, /): Return True if x is neither a positive or negative infinity, and false otherwise.
isinf = def isinf(x, /): Return True if x is a positive or negative infinity, and false otherwise.
isnan = def isnan(x, /): Return True if x is a NaN (not a number), and false otherwise.
```

```
1 import math
2
3 a = math.sin(3.141592653589793)
4 print(a)
5
6 b = math.tan(math.pi)
7 print(b)
8
9 print(math.hypot(3, 4))
Last executed at 2025-02-16 00:21:21 in 5ms
1.2246467991473532e-16
-1.2246467991473532e-16
5.0
```

statistics

Функции для работы со статистическими данными

<code>mean()</code>	Arithmetic mean ("average") of data.
<code>fmean()</code>	Fast, floating-point arithmetic mean, with optional weighting.
<code>geometric_mean()</code>	Geometric mean of data.
<code>harmonic_mean()</code>	Harmonic mean of data.
<code>kde()</code>	Estimate the probability density distribution of the data.
<code>kde_random()</code>	Random sampling from the PDF generated by <code>kde()</code> .
<code>median()</code>	Median (middle value) of data.
<code>median_low()</code>	Low median of data.
<code>median_high()</code>	High median of data.
<code>median_grouped()</code>	Median (50th percentile) of grouped data.
<code>mode()</code>	Single mode (most common value) of discrete or nominal data.
<code>multimode()</code>	List of modes (most common values) of discrete or nominal data.
<code>quantiles()</code>	Divide data into intervals with equal probability.

<code>covariance()</code>	Sample covariance for two variables.
<code>correlation()</code>	Pearson and Spearman's correlation coefficients.
<code>linear_regression()</code>	Slope and intercept for simple linear regression.

<code>pstdev()</code>	Population standard deviation of data.
<code>pvariance()</code>	Population variance of data.
<code>stdev()</code>	Sample standard deviation of data.
<code>variance()</code>	Sample variance of data.

<https://docs.python.org/3/library/statistics.html>

numpy

Хранение данных (в человекочитаемом виде)

Хранение данных в **одновременно** машино- и человекочитаемом виде

- json – структурированные данные
- csv – табличные данные

```
total_time,total_steps,current_phase,phase_steps,phase_time,current_temp,atoms,defects,mg_seed,ions,gases,nc
3.0228034,100,0,0,0.0228034,130,106,1457,[[1,11638946771510751024],[1,1,76,122],2
3.0324315,146,0,0,0.0324315,130,106,1467,[[1,11638946771510751024],[1,1,80,118],2
3.0456919,215,0,0,0.0456919,130,107,1477,[[1,11638946771510751024],[1,1,75,123],2
3.0667131,316,0,0,0.0667131,130,107,1477,[[1,11638946771510751024],[1,1,74,124],2
3.0968543,404,0,0,0.0968543,130,109,1487,[[1,11638946771510751024],[1,1,74,124],2
3.140541,681,0,0,0.140541,130,104,1467,[[1,11638946771510751024],[1,1,81,117],2
3.204465,999,0,0,0.204465,130,107,1477,[[1,11638946771510751024],[1,1,73,125],2
3.296198,1467,0,0,0.296198,130,107,1477,[[1,11638946771510751024],[1,1,75,123],2
3.436191,2154,0,0,0.436191,130,109,1607,[[1,11638946771510751024],[1,1,110,88],2
3.631582,3162,0,0,0.631582,130,109,1877,[[1,11638946771510751024],[1,1,116,82],2
3.926893,4641,0,0,0.926893,130,100,1827,[[1,11638946771510751024],[1,1,121,77],2
1.34828,6812,0,0,1.34828,130,102,9,[[1,11638946771510751024],[1,1,1,96,17,124],3
2.64889,9999,0,0,2.64889,130,104,87,887,[[1,11638946771510751024],[1,2,1,195],2
3.50574,14677,0,0,3.50574,130,130,51,52,1427,[[1,11638946771510751024],[3,1,1,99,107],2
2.55376,21544,0,0,2.55376,130,110,57,64,657,[[1,11638946771510751024],[1,1,1,46,144],3
14.9027,31627,0,0,14.9027,130,110,60,61,65,1507,[[1,11638946771510751024],[1,1,1,3,90,100],3
```

<https://docs.python.org/3/library/csv.html>
<https://docs.python.org/3/library/json.html>

```
name: example_run
atoms: 100
barriers: barriers_adaptive4
barriers_heights: 2 eV
E1: 0.235
E2: 0.295
E3: 0.241
E4: 0.378
initial_atoms_count: 10 # atoms
defects_count: 0 # atoms
```

yaml

pandas

numpy

Numpy



Numpy = Numerical Python (численный Python)

Numpy – это фундаментальная библиотека для научных вычислений с Python. Она предоставляет интерфейс для работы с многомерными массивами (ndarray, N-dimensional array), а также с многими другими производными объектами (включая массивы с масками и матрицы), а также большое число разнообразных функций, осуществляющих быстрые операции над ними: математические, логические, манипулирующие формой массивов, осуществляющие сортировку, выбор, ввод/вывод, различные преобразования, базовую линейную алгебру, базовые статистические операции, генерацию случайных объектов и многое другое.

- Numpy стремится к эффективности за счет работы 'in-memory'.
- Критически важно: циклы for в Python очень медленные.
- Реализована на C и Fortran, вычисления векторизованы.

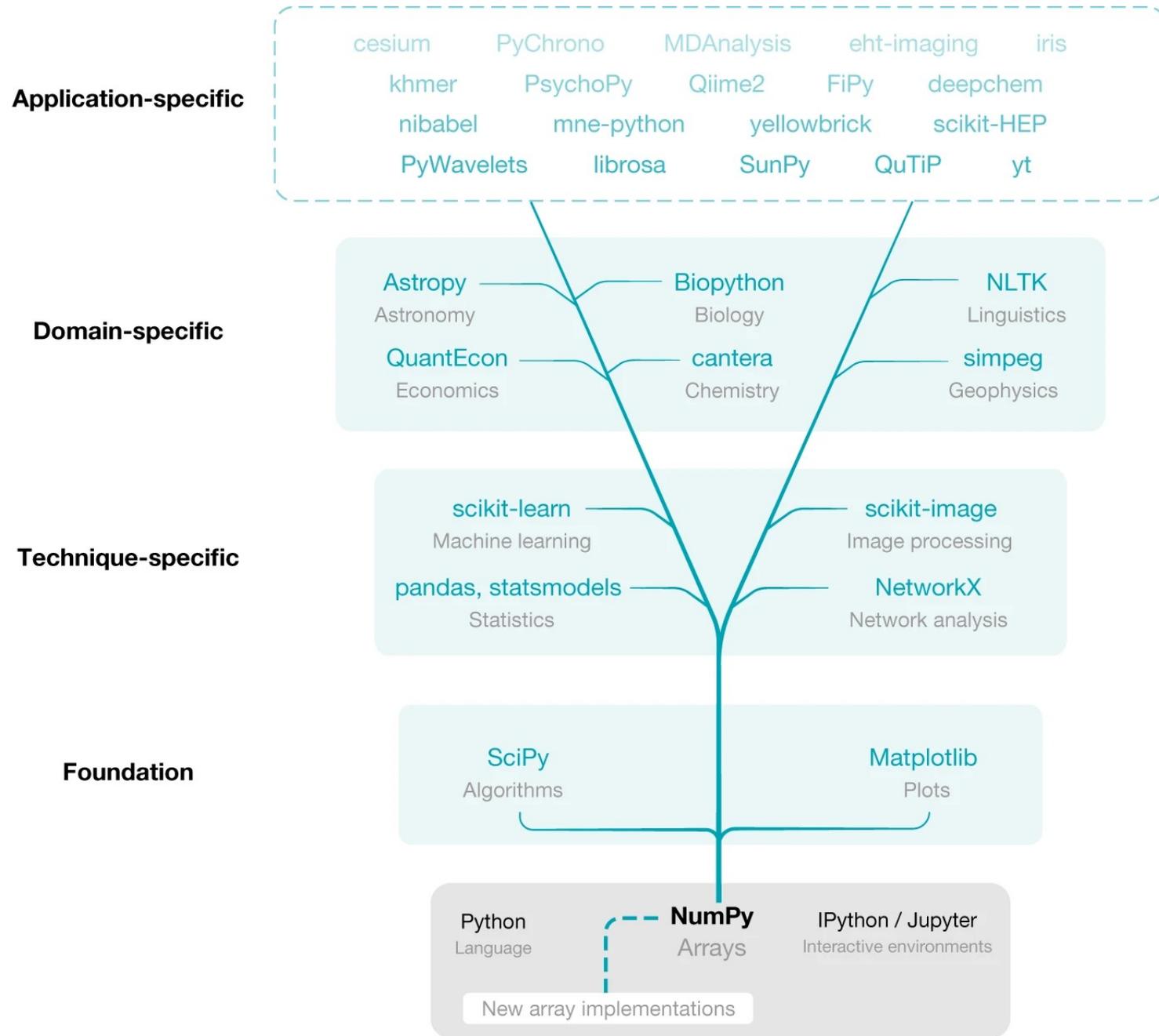
Numpy – основание научной экосистемы Python

Используется почти во всех областях науки и технологии

Универсальный стандарт

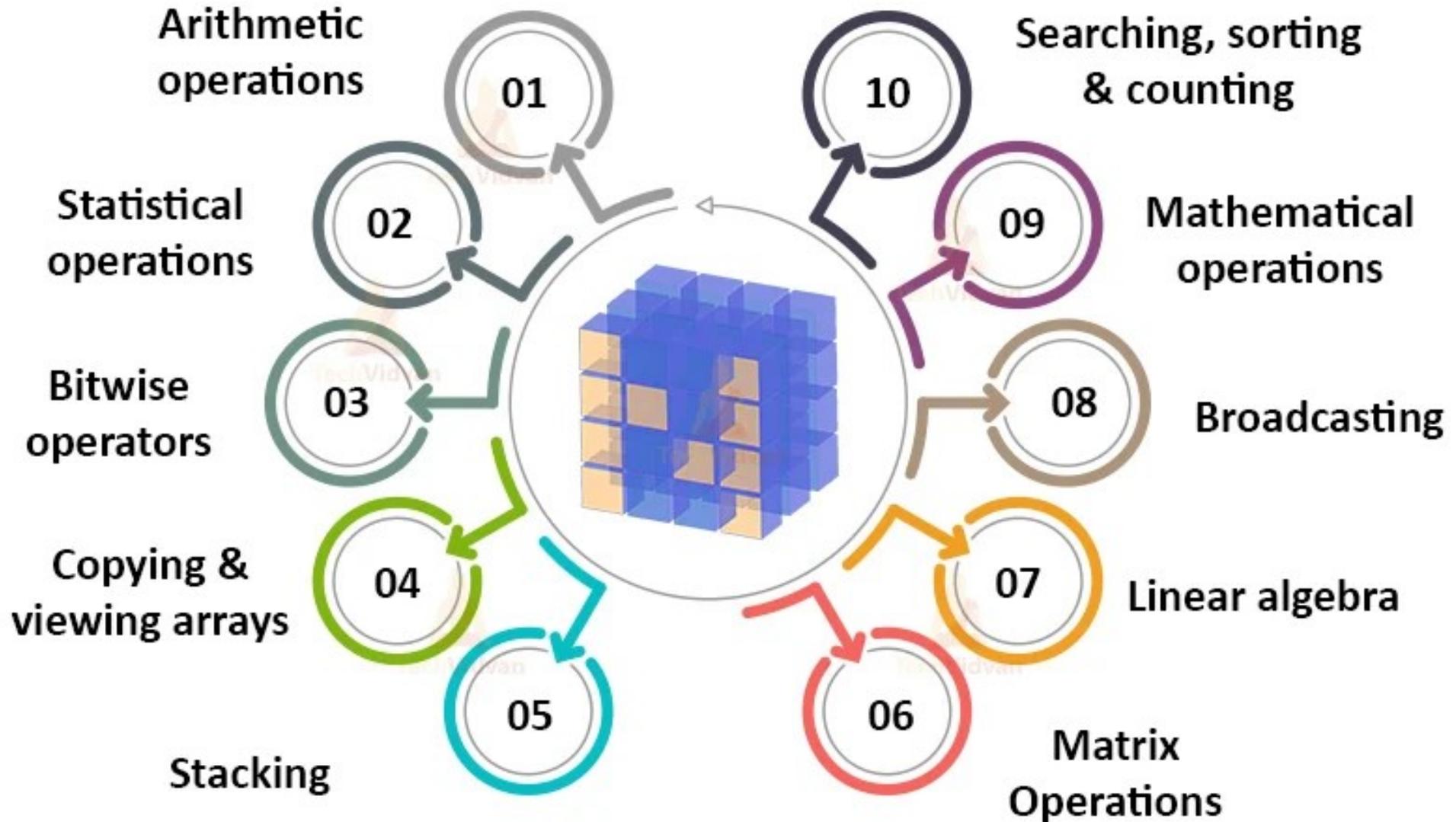
Предоставляет основу для создания других научных пакетов

- Науки о Земле: GeoPandas, MovingPandas, Xarray
- Космос: SpacePy, AstroPy, SunPy
- Scientific Domain: SciPy, SimPy, Shapely
- Data Science: Pandas, Dask
- Machine Learning: Scikit-Learn, TensorFlow
- Визуализация: Matplotlib, Cartopy, Seaborn, HoloViews



NumPy API ——— Array Protocols - - - -

Uses of NumPy



Базовые идеи

ndarray - это гомогенный объект, n-мерный массив + методы для работы с ним: это структура (данные + метаданные).

Индексация как обычно: срезы и индексы.

Индексация при помощи скалярных координат, масок или массивов координат, при этом еще и дешево.

Эффективная параллелизация операций за счет векторизации.

Транслирование массивов

Редукция вдоль одной и более осей.

Документация

<https://numpy.org/doc/stable/user/index.html>

<https://numpy.org/doc/stable/user/quickstart.html>

https://numpy.org/doc/stable/user/absolute_beginners.html

Fundamentals and usage

[NumPy fundamentals](#)

[Array creation](#)

[Indexing on `ndarrays`](#)

[I/O with NumPy](#)

[Data types](#)

[Broadcasting](#)

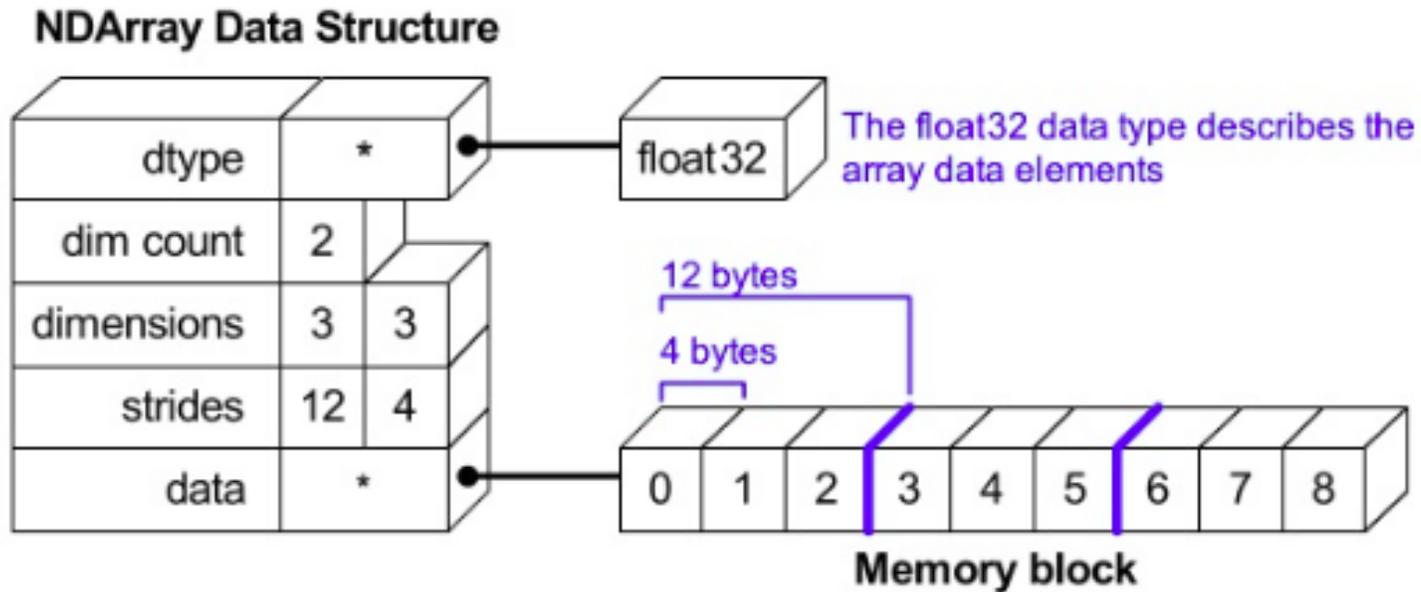
[Copies and views](#)

[Working with Arrays of Strings And Bytes](#)

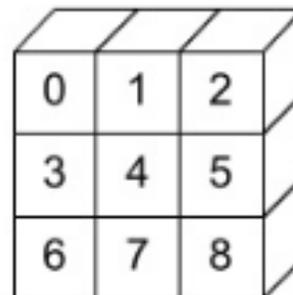
[Structured arrays](#)

[Universal functions \(`ufunc`\) basics](#)

Что такое ndarray?



Python View :



Преимущества
Изменяя метаданные, можно менять интерпретацию данных, порядок следования данных без использования промежуточного буфера.

C-order vs F-order

How the array is represented in Numpy

How the array is stored in memory

Row Major
Order (C)
(default in NumPy)



Column Major
Order (Fortran)



Начало работы

```
import numpy as np
```

```
a = [[1, 2, 0], [3, 8, 7]]
```

```
b = np.array(a, dtype=float)  
print(b, b.dtype, b.shape, b.size)
```

```
[[1.  2.  0.]  
 [3.  8.  7.]] float64 (2, 3) 6
```

Создание массивов с нуля

numpy.array

numpy.empty

numpy.ones

numpy.zeros

numpy.full

numpy.empty_like

numpy.ones_like

numpy.zeros_like

numpy.full_like

numpy.eye

numpy.identity

numpy.arange

numpy.linspace

numpy.logspace

numpy.geomspace

numpy.meshgrid

numpy.diag

numpy.tri

numpy.tril

numpy.triu

```
>>> np.empty([2, 2], dtype=int)
array([[ -1073741821, -1067949133],
       [  496041986,   19249760]])
```

```
>>> np.ones((5,), dtype=int)
array([1, 1, 1, 1, 1])
```

```
>>> np.zeros((2, 1))
array([[ 0.],
       [ 0.]])
```

```
>>> np.full((2, 2), np.inf)
array([[inf, inf],
       [inf, inf]])
>>> np.full((2, 2), 10)
array([[10, 10],
       [10, 10]])
```

```
>>> np.full((2, 2), [1, 2])
array([[1, 2],
       [1, 2]])
```

```
>>> np.eye(3, k=1)
array([[0.,  1.,  0.],
       [0.,  0.,  1.],
       [0.,  0.,  0.]])
```

```
>>> y = np.arange(3, dtype=float)
>>> y
array([0., 1., 2.])
>>> np.ones_like(y)
array([1., 1., 1.] )
```

```
>>> np.arange(3)
array([0, 1, 2])
>>> np.arange(3.0)
array([ 0.,  1.,  2.])
>>> np.arange(3,7)
array([3, 4, 5, 6])
>>> np.arange(3,7,2)
array([3, 5])
```

```
>>> np.linspace(2.0, 3.0, num=5)
array([2.   , 2.25, 2.5   , 2.75, 3.   ])
```

```
>>> x = np.arange(9).reshape((3,3))
>>> x
array([[0, 1, 2],
       [3, 4, 5],
       [6, 7, 8]])
```

```
>>> np.diag(x)
array([0, 4, 8])
```

`numpy.ones(shape, dtype=None, order='C', *, device=None, like=None)`

Чтение данных в массив

`numpy.array`

`numpy.copy`

`numpy.frombuffer`

`numpy.from_dlpack`

`numpy.fromfile`

`numpy.fromfunction`

`numpy.fromiter`

`numpy.fromstring`

`numpy.loadtxt`

`numpy.load`

`numpy.save`

```
>>> x = np.array([1, 2, 3])
>>> y = x
>>> z = np.copy(x)

>>> x[0] = 10
>>> x[0] == y[0]
True
>>> x[0] == z[0]
False
```

```
>>> np.fromfunction(lambda i, j: i + j, (3, 3), dtype=int)
array([[0, 1, 2],
       [1, 2, 3],
       [2, 3, 4]])
```

```
>>> np.fromstring('1 2', dtype=int, sep=' ')
array([1, 2])
>>> np.fromstring('1, 2', dtype=int, sep=',')
array([1, 2])
```

```
>>> import numpy as np
>>> from io import StringIO
>>> c = StringIO("0 1\n2 3")
>>> np.loadtxt(c)
array([[0., 1.],
       [2., 3.]])
```

```
>>> np.save('/tmp/123', np.array([[1, 2, 3], [4, 5, 6]]))
>>> np.load('/tmp/123.npy')
array([[1, 2, 3],
       [4, 5, 6]])
```

Простейшие операции с данными

Базовые операции



Изменение формы

Операции типа транспонирования

Изменение числа измерений

Изменение типа массива

Объединение массивов

Деление массивов на части

Повторение массивов

Добавление/удаление элементов

Перестановка элементов

`copyto` (dst, src[, casting, where])

`ndim` (a)

`shape` (a)

`size` (a[, axis])

Доступ по индексам etc

```
>>> a[0,3:5]  
array( [3,4] )
```

```
>>> a[4:, 4:]  
array( [ 28, 29],  
       [ 34, 35] )
```

```
>>> a[:, 2]  
array( [2, 8, 14, 20, 26, 32] )
```

```
>>> a[2::2, ::2]  
array( [ 12, 14, 16],  
       [ 24, 26, 28] )
```

0	1	2	3	4	5
6	7	8	9	10	11
12	13	14	15	16	17
18	19	20	21	22	23
24	25	26	27	28	29
30	31	32	33	34	35

Изменение формы массива

data

1
2
3
4
5
6

data.reshape(2,3)

1	2	3
4	5	6

Diagram illustrating the reshape operation `data.reshape(2,3)`. The original 1D array is reshaped into a 2x3 matrix. The first row contains elements 1, 2, and 3. The second row contains elements 4, 5, and 6. A red vertical bracket on the left indicates the height is 2, and a purple horizontal bracket at the bottom indicates the width is 3.

data.reshape(3,2)

1	2
3	4
5	6

Diagram illustrating the reshape operation `data.reshape(3,2)`. The original 1D array is reshaped into a 3x2 matrix. The first row contains elements 1 and 2. The second row contains elements 3 and 4. The third row contains elements 5 and 6. A red vertical bracket on the left indicates the height is 3, and a purple horizontal bracket at the bottom indicates the width is 2.

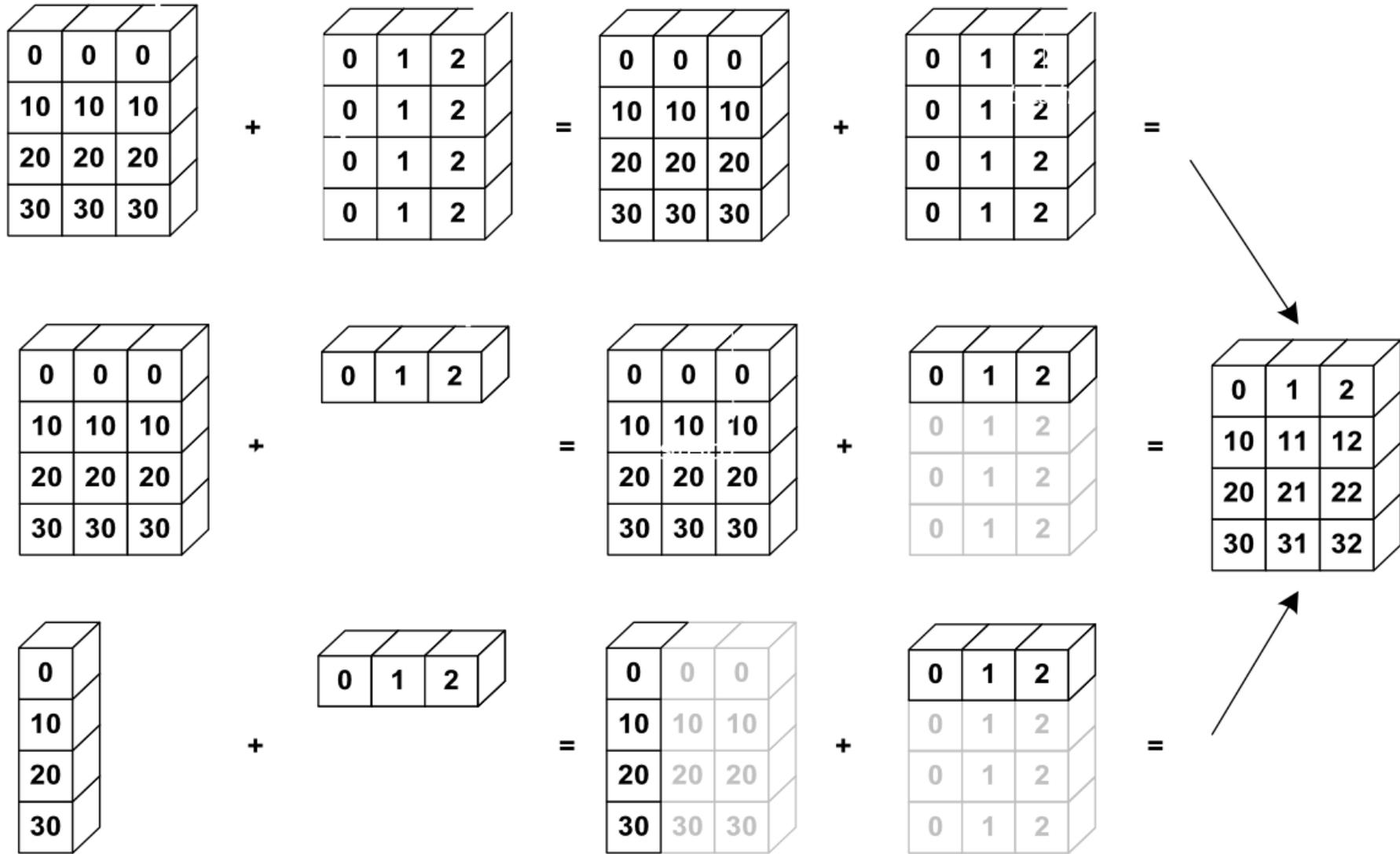
Транслирование массивов (broadcasting)

Если не все входные массивы имеют одинаковое число измерений, то к тем, у кого размерность ниже, к их форме спереди добавляется единица, пока размерности всех массивов не станут одинаковыми.

Одномерные массивы длиной, равной 1, вдоль некоторой оси ведут себя так, как если бы они имели вдоль этой оси такой же размер, как и другие массивы, входящие в выражение. Каждое значение в массиве вдоль этой оси будет равно тому единственному, которое там было изначально.

Если эти два правила не выполняются, генерируется ошибка.

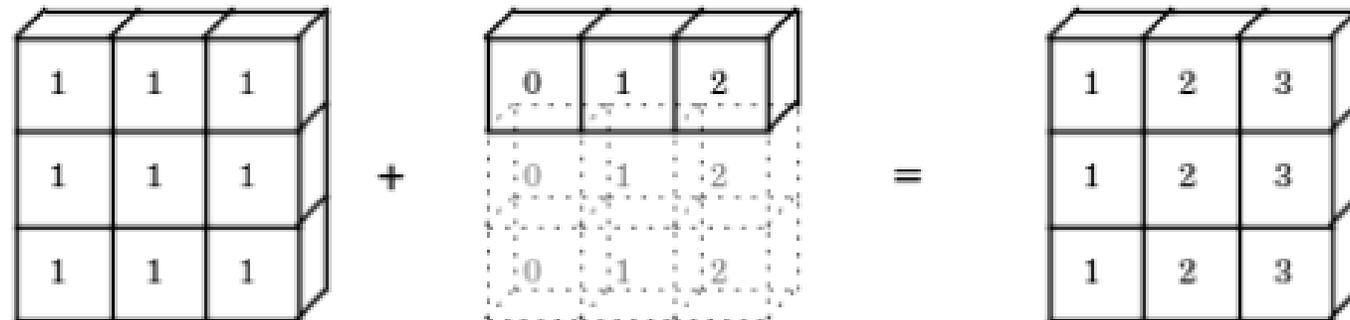
Транслирование массивов (broadcasting)



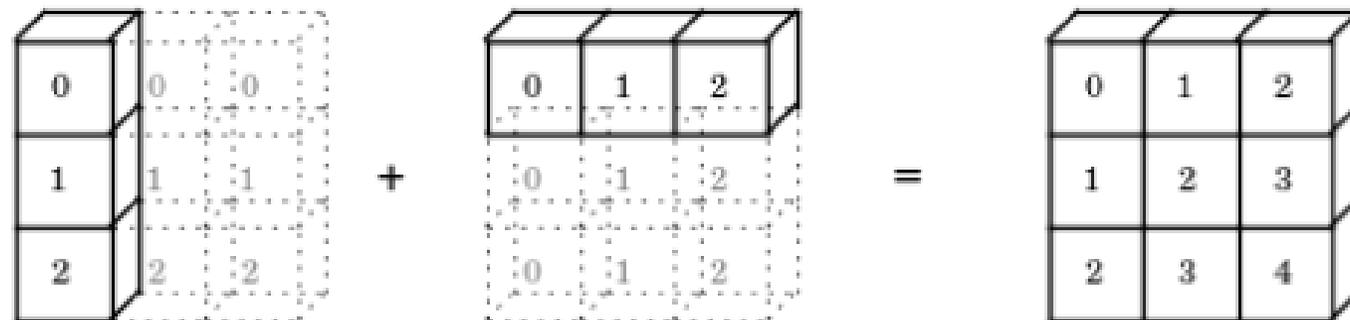
`np.arange(3) + 5`



`np.ones((3, 3)) + np.arange(3)`

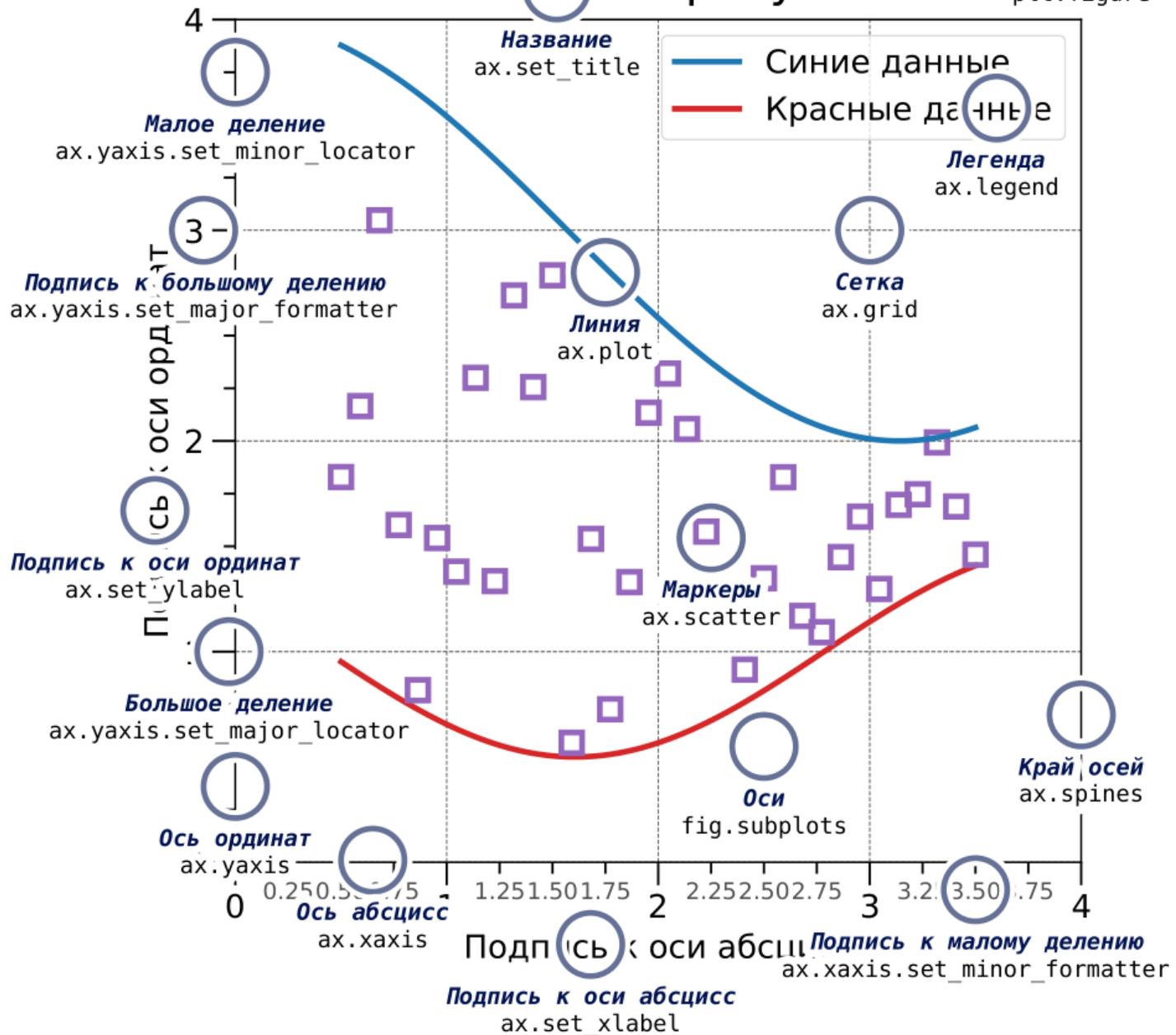


`np.arange(3).reshape((3, 1)) + np.arange(3)`



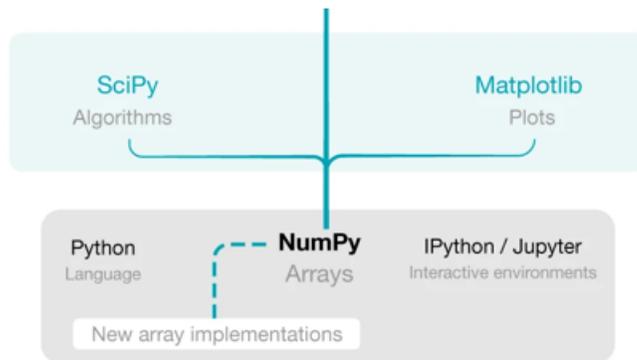
Аналогия рисунка

Рисунок
plt.figure



Scipy

<https://docs.scipy.org/doc/scipy/tutorial/index.html>



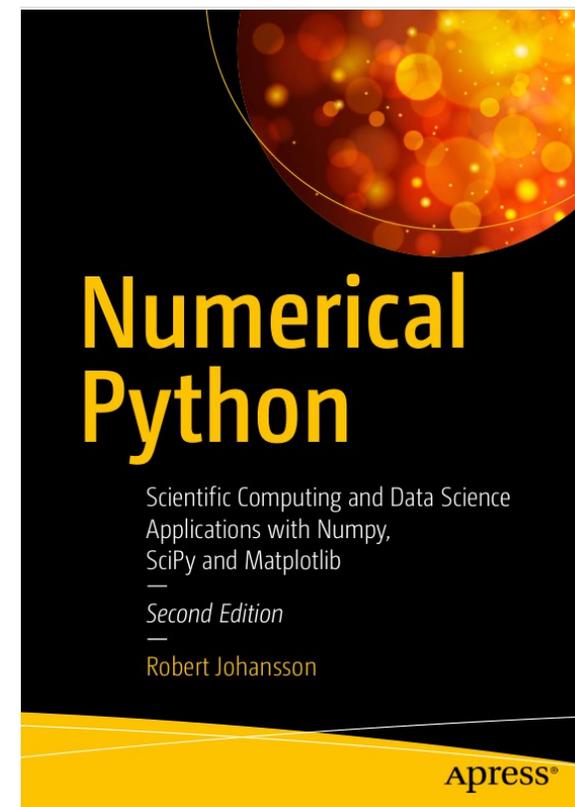
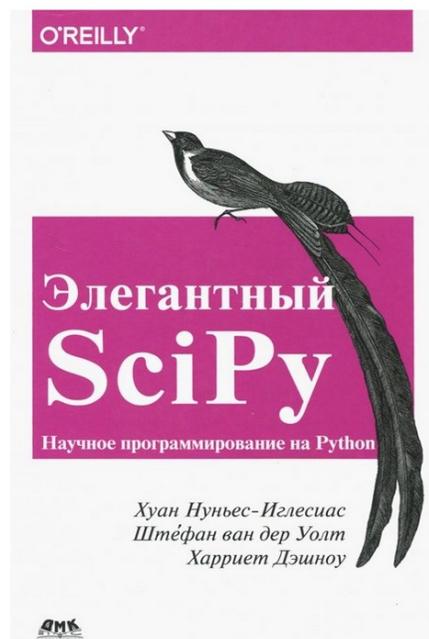
Subpackage	Description and User Guide
cluster	Clustering algorithms
constants	Physical and mathematical constants
differentiate	Finite difference differentiation tools
fft	Fourier Transforms (scipy.fft)
fftpack	Fast Fourier Transform routines (legacy)
integrate	Integration (scipy.integrate)
interpolate	Interpolation (scipy.interpolate)
io	File IO (scipy.io)
linalg	Linear Algebra (scipy.linalg)
ndimage	Multidimensional Image Processing (scipy.ndimage)
odr	Orthogonal distance regression
optimize	Optimization (scipy.optimize)
signal	Signal Processing (scipy.signal)
sparse	Sparse Arrays (scipy.sparse)
spatial	Spatial Data Structures and Algorithms (scipy.spatial)
special	Special Functions (scipy.special)
stats	Statistics (scipy.stats)

Дальнейшее чтение

Маккинни, У. *Python и анализ данных* 3 издание. 536 с. isbn: 978-6-01810-347-6 (ДМК Пресс, 2023).

Нуньес-Иглесиас, Х., Уолт, Ш., Дэшноу, Х. *Элегантный SciPy*. 266 с. isbn: 978-5-97060-600-1 (ДМК Пресс, 2018).

Johansson, R. *Numerical Python: Scientific Computing and Data Science Applications with Numpy, SciPy and Matplotlib* isbn: 9781484242469. <http://dx.doi.org/10.1007/978-1-4842-4246-9> (Apress, 2019).



<https://mathstat.dal.ca/~brown/sound/python/Numerical%20Python.pdf>