



Numpy

библиотека языка Python, ориентированная на работу с многомерными массивами и матрицами, с полиномами и с другими объектами



```
import numpy
import numpy as np
from numpy as np
```

Ndarray

основным объектом numpy является однородный многомерный массив элементов одного типа (называется `numpy.ndim`)



Некоторые атрибуты `ndarray`:

`ndarray.ndim` - число измерений (принято называть осями) массива

`ndarray.shape` - кортеж натуральных чисел, показывающих длину массива по каждой оси. Число элементов кортежа `shape` равно `ndim`

`ndarray.dtype` - тип элемента массива. numpy имеет собственные типы данных, например: `float32`, `complex64` и т. д.

`ndarray.itemsize` - размер каждого элемента в байтах

Создание массива



```
numpy.array(object, dtype=None, *, copy=True, order='K', subok=False, ndmin=0, like=None)
```

```
import numpy as np
```

```
a = np.array((2, -3, 1, 9))
```

```
print(type(a))
```

```
print(a)
```

```
print(a.shape)
```

```
print(a.dtype, \n)
```

Arrange



numpy.arange([start,]stop, [step,], dtype=None) - возвращает массив (numpy.ndarray) чисел, равномерно распределенных в заданном интервале.

start - начало интервала,

stop - конец интервала,

step - шаг,

dtype - тип данных.

```
a = np.arange(10)
```

```
print(a)
```

Linspace ()



numpy.linspace(start, stop, num=50, endpoint=True, retstep=False, dtype=None, axis=0) - возвращает массив (numpy.ndarray) N чисел, равномерно распределенных в заданном интервале.

start - начало интервала

stop - конец интервала

num-число элементов

endpoint - если True, то stop включается в массив

retstep - если True, то функция вернет кортеж вида(values, step)

a = np.linspace(0, 2, 5)

print(a)

Массив случайных чисел



np.random.rand(d0, d1,..., dn) - создает массив заданной размерности с числами от 0 до 1;

np.random.randint(low[, high, size, dtype]) - создает массив целых чисел из заданного диапазона и заданной размерности.

Размерность массива



```
import numpy as np
```

```
a = np.arange(12)
```

```
b = np.reshape(a,(2,6))
```

```
print('Изменение формы массива\n',b,'\n')
```

```
c = np.resize(a,(2,7))
```

```
print("Изменение формы массива при", 'несовпадении числа элементов\n',c,'\n')
```

Создание двумерного массива



```
a = np.array([[1, 2, 3], [9, 8, 7]], 'int64')
print(type(a))
print(a)
print (a.shape)
print (a.dtype,'\n\n')
```

Матрицы специального вида



```
a = np.empty((3, 2)) # Пустая матрица
b = np.identity(4) # Единичная матрица: 1 на главной диагонали, 0 -
остальные.
c = np.zeros((4, 5),int) # Нулевая матрица
d = np.ones((4,3), 'int64') # Матрица из единиц
f = np.full(3,3), 7) # Заполнение матрицы
```

Операции с матрицами



```
a = np.arange(1, 5).reshape(2, 2)
b = np.arange(5, 9).reshape(2, 2)
c_plus = a+b
c_minus = a-b
c_mult = a*b
c_power = a**b
c_num = a*5
```

Функции для работы с матрицами



```
c_multimat = np.dot(a, b) # матричное умножение
a0= np.arange(0, 4).reshape(2, 2)
a_div = a/a0 # деление
a_remainder = a%a0 # Остаток от деления
e = np.subtract(a,b) # Вычитание subtract
f = np.multiply(a,b) # Умножение поэлементное multiply
g = np.divide(a,b) # Деление divide
m = np.negative(a) # 'Смена знака
n = np.transpose(a) # Транспонирование
```

Математические функции



тригонометрические функции
функции округления
экспоненциальные и логарифмические функции
lcm(), gcd()
арифметические

Сортировки массивов



```
numpy.sort(a, axis=-1, kind=None, order=None)
kind:
quicksort
mergesort
heapsort
```

Другие возможности библиотеки



работа с масками

работа с разреженными матрицами

решение задач линейной алгебры