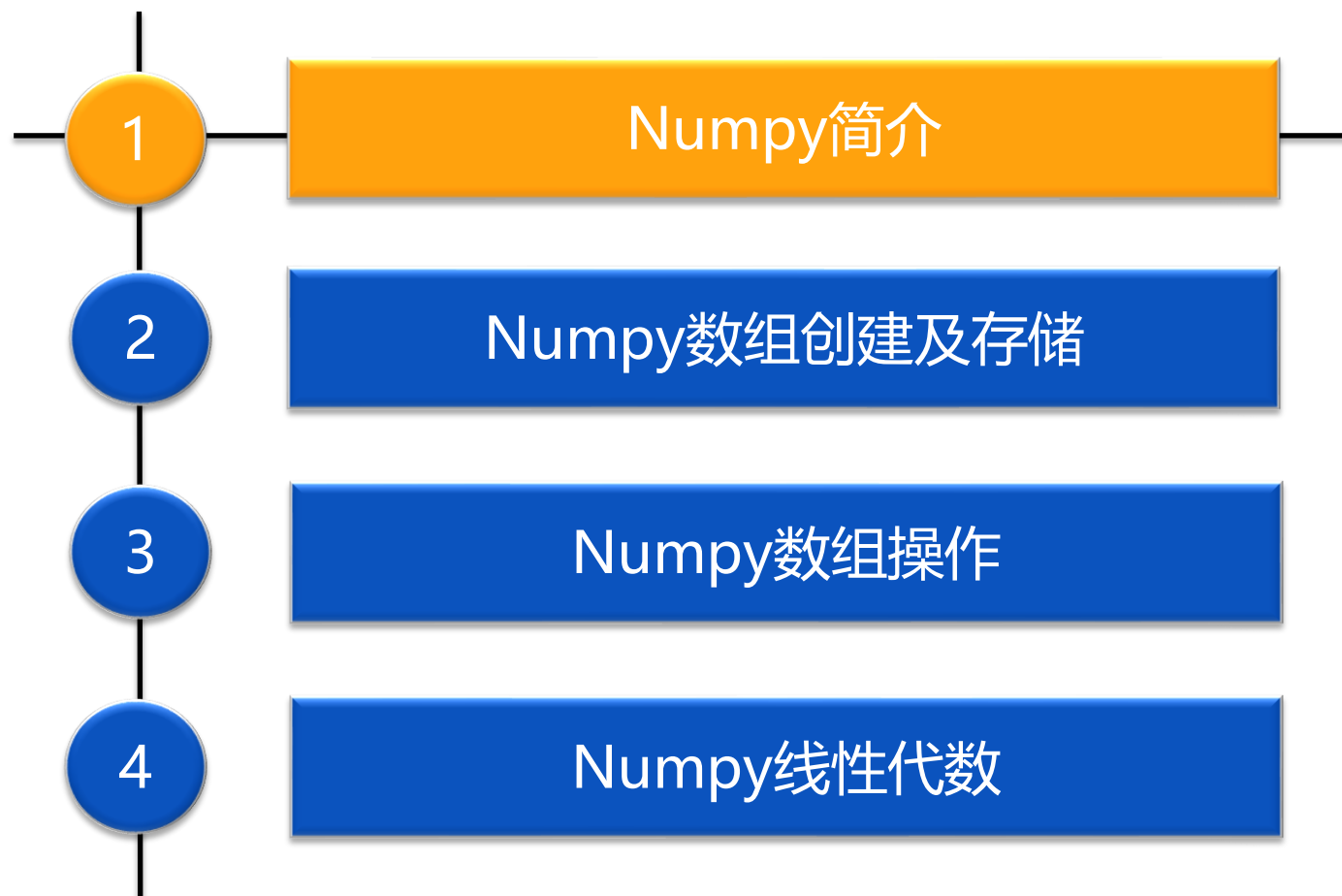




NumPy基础课程

王右雪

目录



A vertical line on the left side of the table of contents connects the four numbered circles. The first circle is orange, while the others are blue. Each circle is connected to a horizontal bar representing a chapter title.

1	Numpy简介
2	Numpy数组创建及存储
3	Numpy数组操作
4	Numpy线性代数

NumPy简介

NumPy(Numerical Python) 是 Python 语言的一个扩展程序库，支持大量的维度数组与矩阵运算，此外也针对数组运算提供大量的数学函数库。

- 一个强大的N维数组对象 ndarray
- 广播功能函数
- 整合 C/C++/Fortran 代码的工具
- 线性代数、傅里叶变换、随机数生成等功能

为什么要使用NumPy ?

- numpy的数组功能比Python自身的嵌套列表 (nested list structure)结构要**高效**的多（该结构也可以用来表示矩阵（matrix）），**支持大量的维度数组与矩阵运算**，此外也针对数组运算提供**大量的数学函数库**



Jim Hugunin

NumPy使用预备知识

- **NumPy 模块导入**:

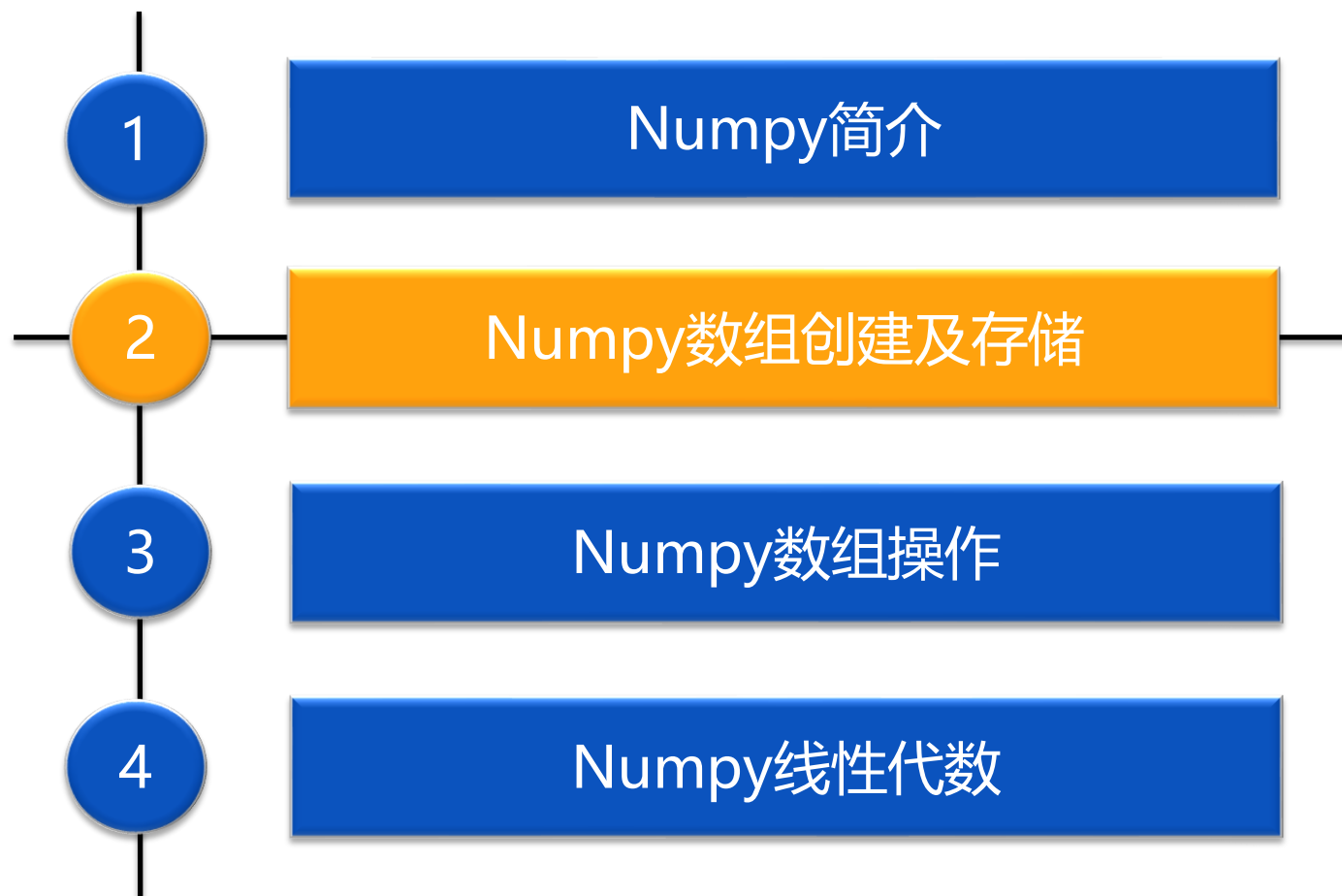
```
import numpy as np
```

```
from numpy import *
```

- **NumPy支持的数据类型**

类型	描述
bool	用一位存储的布尔类型（值为TRUE或FALSE）
int	由所在平台决定其精度的整数（一般为int32或int64）
float	浮点数，根据精度显示不同的长度
str	字符串数据
.....

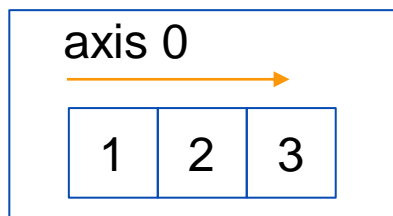
目录



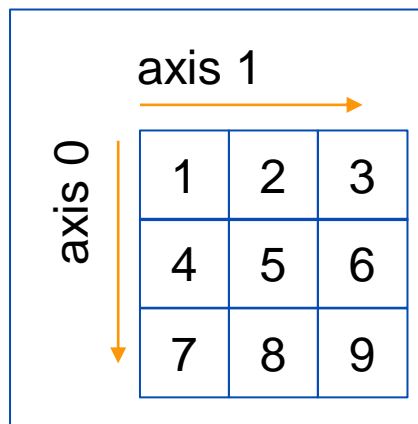
数组相关知识

ndarray是一个(通常是固定大小的)**多维**容器，包含**相同类型**的项。

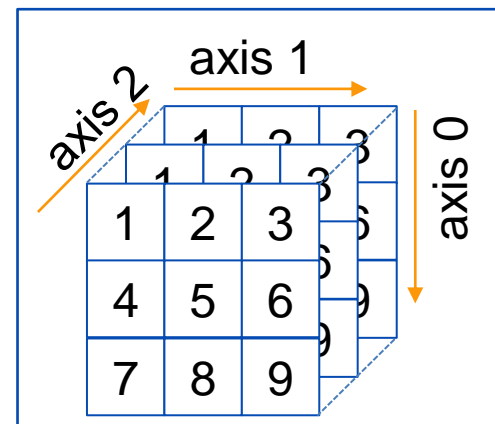
一维数组



二维数组



多维数组



数组函数

创建数组通常使用array函数

array函数详解

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

- object*: 接收array。表示想要创建的数组。无默认
- dtype*: 接收data-type。表示数组所需的数据类型。如果未给定，则选择保存对象所需的最小类型。默认为None
- ndmin*: 接收int。指定生成数组应该具有的最小维数。默认为None

查看数组基本信息

函数	描述	函数	描述
size	数组的大小	shape	数组的形状
ndim	数组最大维度	dtype	数据类型

数组生成函数

arange函数

numpy.arange(*start, stop, step, dtype=None*)

Start: 可忽略不写, 默认从0开始;起始值

stop: 结束值; 生成的元素不包括结束值。

step: 可忽略不写, 默认步长为1; 步长

dtype: 默认为None, 设置显示元素的数据类型

linspace 和 logspace 函数

numpy.linspace (*start, stop, num=50, endpoint=True, retstep=False, dtype=None, axis=0*)

numpy.logspace (*start, stop, num=50, endpoint=True, base=10.0, dtype=None, axis=0*)

endpoint: True则包含stop; False则不包含stop

restep: 如果为True则结果会给出数据间隔

base: 函数的底

$\log_{base} \text{linspace}(start, end, num, endpoint)$

简便的数组生成函数

函数名	描述
ones	根据给定形状和数据类型生成全1数组
ones_like	根据所给的数组生成一个形状一样的全1数组
zeros	根据给定形状和数据类型生成全0数组
zeros_like	根据所给的数组生成一个形状一样的全0数组
empty	根据给定形状生成一个没有初始化的空数组
empty_like	根据所给的数组生成一个形状一样但没有初始化数值的空数组
full	根据给定形状和数据类型生成指定数值的数组
full_like	根据所给的数组生成一个形状一样但内容是指定数值的数组
eye,identity	生成一个 $N \times N$ 特征矩阵（对角线是1，其他是0）
diag	将一个方针对角元素作为一维数组返回，或者将一位数组转换成一个方针，并在非对角线上有零点

Numpy随机数模块

numpy.random模块填补了Python内建的random模块的不足，可以高效生成随机数组

numpy.random.randint(*low*, *high=None*, *size=None*, *dtype='l'*)

low: 生成的数值最低要大于等于low, (high = None时, 生成的数值要在[0, low)区间内)

high: 如果使用这个值, 则生成的数值在[low, high)区间

size: 输出随机数的尺寸, 比如size = (m * n * k)则输出同规模即m * n * k个随机数。默认是None的, 仅仅返回满足要求的单一随机数

无约束条件下生成随机数

np.random.random(*n*)

np.random.random((*m*, *n*))

*随机浮点数, 浮点数范围 (0,1)生成服从均匀分布的随机数

Numpy随机数

函数	描述
random	无约束条件下生成随机数
rand	从均匀分布中抽取样本
randn	从均值0方差1的正态分布中抽取样本
binomial	产生二项分布的随机数。
normal	产生正态（高斯）分布的随机数。
beta	产生beta分布的随机数。
chisquare	产生卡方分布的随机数。
gamma	产生gamma分布的随机数。
uniform	产生在[0,1)中均匀分布的随机数。
seed	确定随机数生成器的种子。
permutation	返回一个序列的随机排列或返回一个随机排列的范围。
shuffle	对一个序列进行随机排序。

Numpy数据的读写

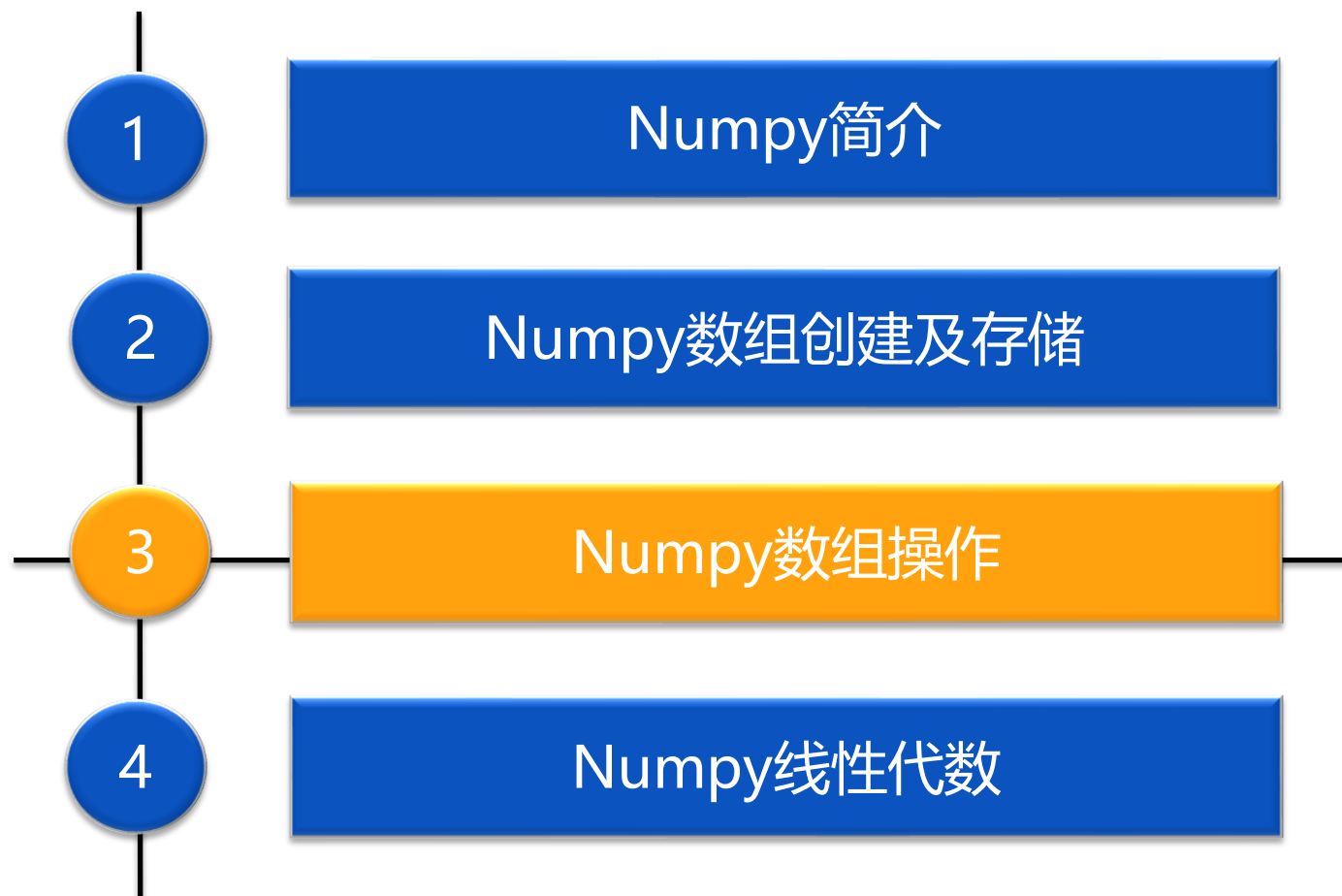
NumPy文件读写主要有二进制的文件读写和文件列表形式的数据读写两种形式

- save函数是以二进制的格式保存数据。 **np.save("../tmp/save_arr",arr)**
- load函数是从二进制的文件中读取数据。 **np.load("../tmp/save_arr.npy")**
- savez函数可以将多个数组保存到一个文件中。 **np.savez('../tmp/savez_arr',arr1,arr2)**
*存储时可以省略扩展名，但读取时不能省略扩展名。

读取文本格式的数据

- savetxt函数是将数组写到某种分隔符隔开的文本文件中。 **np.savetxt("../tmp/arr.txt", arr, fmt="%d", delimiter=",")**
- loadtxt函数执行的是把文件加载到一个二维数组中 **np.loadtxt("../tmp/arr.txt",delimiter=",")**
- genfromtxt函数面向的是结构化数组和缺失数据。 **np.genfromtxt("../tmp/arr.txt", delimiter = ",")**

目录



数组索引与切片

一维数组

单个元素: `arr[j]`

连续多个元素: `arr[j1:j2]`

分散多个元素: `arr[[j1, j2, j3]]`

正向步长取值: `arr[j1:j2:t]` * $j_1 < j_2$

反向步长取值: `arr[j1:j2:-t]` * $j_1 > j_2$

从末尾索引: `arr[-1]` * 正向检索规则对负向同样适用

多维数组

单行操作

单个元素: `arr[i,j]`

连续多个元素: `arr[i, j1:j2]`

分散多个元素: `arr[i, [j1, j2, j3]]`

正向步长取值: `arr[j1:j2:t]` * $j_1 < j_2$

反向步长取值: `arr[j1:j2:-t]` * $j_1 > j_2$

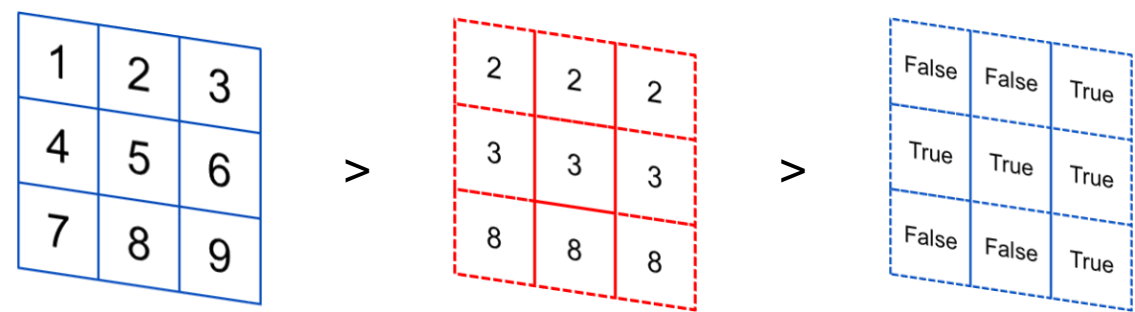
多行操作

整列操作: `arr[i,:]`

整行操作: `arr[:,i]`

布尔数组

对两个任意形状大小的数组进行比较运算可以获得一个相同形状的布尔数组。



根据布尔值的特性，可以对布尔数组做多种操作。

函数	描述
sum	返回所有元素的和
any	检验数组里有没有true,如果有一个ture,则返回true
all	检验数组是不是全都是true,只有在全都是true的情况下才返回true

利用布尔数组切片

布尔索引是把数组中布尔值为True的相应行、列或元素抽取了出来（注意：布尔值的长度必须和想要切片的维度或轴的长度一致）

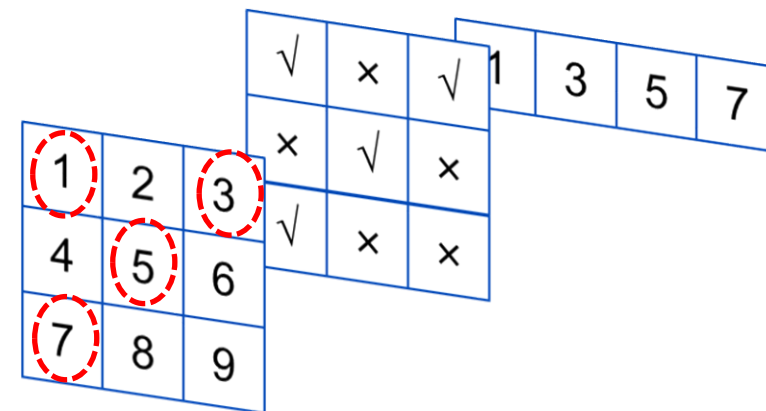
一维数组

一维数组中的每个元素的布尔型数值对一个与一维数组有着同样行数或列数的数组进行符合匹配

多维数组

提取行列：将行(列)对应布尔值进行符合匹配

提取元素：将需要提取的元素对应的布尔值数组匹配提取即可。提取后的结果返回一个一维数组



Numpy数组计算

Numpy可以直接进行算数操作，比如两个数组相加可以直接使用arr1 + arr2，两个数组对应元素相乘可以直接使用arr1*arr2。

此外，数组还可以调用通用函数（又称ufunc，一种在数组中进行逐元组操作的函数）实现更高级的功能。

一元通用函数

函数		函数	
abs\fabs	逐元素地计算整数,浮点数或复数的绝对值	rint	将元素保留到整数部，并保持dtype
sqrt	计算每个元素的平方根	modf	分别返回整数和小数部的数组
square	计算每个元素的平方	isnan	返回数组中的元素是否是一个NaN（布尔）
exp	计算每个元素的自然指数值 e^x	cos\sin\tan \cosh\sinh\ tanh\arccos \arcsin\arctan	三角函数：余弦、正弦、正切、双曲余弦、双曲正弦、双曲正切、反余弦、反正弦、反正切
log,log2	自然对数、对数2为底		
sign	计算每个元素的符号值（1:正，0:0，-1:负）		
ceil	计算每个元素的最高整数值		
floor	计算每个元素的最小整数值	logical_not	对数组的元素按位取反

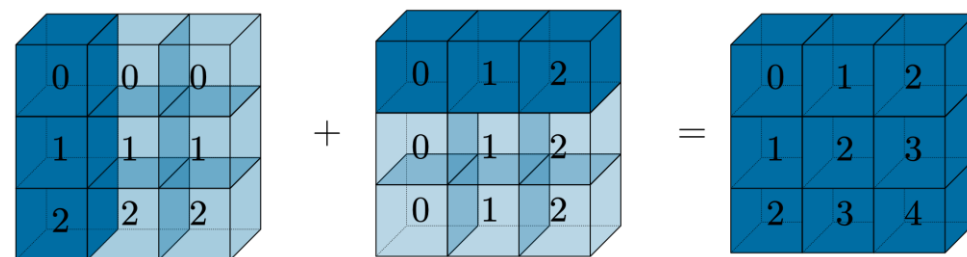
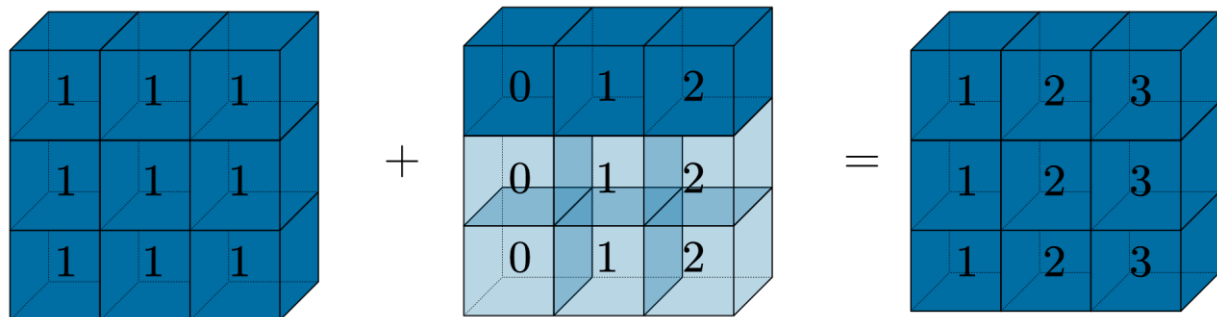
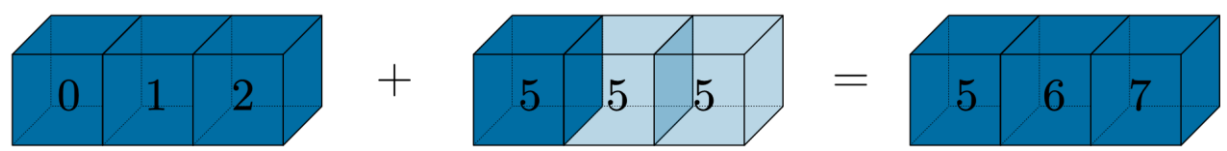
二元通用函数

函数	描述
add	将数组的对应元素相加
subtract	在第二个数组中，将第一个数组中包含的元素去除
multiply	将数组的对应元素相乘
divide/floor_divide	除或整除（放弃余数）
power	将第二个数组的元素作为第一个数组的幂次方
maximum/fmax	逐个元素计算最大值，fmax忽略NaN
minimum/fmin	逐个元素计算最大值，fmin忽略NaN
mod	按元素的求模计算

广播规则

NumPy的广播遵循一组严格的规则，设定这组规则是为了巨顶两个数组间的操作

- **规则1**：如果两个数组的维度数不相同，那么小维度数组的形状将会在最左边补1
- **规则2**：如果两个数组的形状在任何一个维度上都不匹配，那么数组的形状会沿着维度为1的维度扩展以匹配另一个数组的形状
- **规则3**：如果两个数组的形状在仍何一个维度上都不匹配并且没有仍何一个维度等于1，那么会引发异常



左上: `np.arange(3)+5`

左下: `np.ones((3,3))+ np.arange(3)`

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

Numpy排序与唯一值

Numpy可以使用sort方法按照位置进行排序

直接排序

- sort函数是最常用的排序方法。 **numpy.sort(arr)** *该方法不支持反向排序
- sort函数也可以指定一个axis参数，使得sort函数可以沿着指定轴对数据集进行排序。axis=1为沿横轴排序； axis=0为沿纵轴排序。 **numpy.sort(arr, axis)**

间接排序

- argsort函数返回值为重新排序值的下标。 **numpy.argsort(arr)**
- lexsort函数返回值是按照最后一个传入数据排序的。 **numpy.lexsort((a,b,c))**

Numpy去重与重复数据

去重

通过unique函数可以找出数组中的唯一值并返回已排序的结果。

重复

➤ tile函数主要有两个参数，参数“A”指定重复的数组，参数“*reps*”指定重复的次数。

np.tile(A, reps)

➤ repeat函数主要有三个参数，参数“a”是需要重复的数组元素，参数“repeats”是重复次数，参数“axis”指定沿着哪个轴进行重复，axis = 0表示按行进行元素重复；axis = 1表示按列进行元素重复。

numpy.repeat(a, repeats, axis=None)

这两个函数的主要区别在于，tile函数是对数组进行重复操作，repeat函数是对数组中的每个元素进行重复操作。

NumPy统计函数

使用聚合函数可以直接对数组进行计算，当axis=0时，表示沿着纵轴计算。当axis=1时，表示沿着横轴计算。默认时计算一个总值。

函数	说明
sum	计算数组的和
mean	计算数组均值
std	计算数组标准差
var	计算数组方差
min	计算数组最小值
max	计算数组最大值
argmin	返回数组最小元素的索引
argmax	返回数组最大元素的索引
cumsum	计算所有元素的累积和
cumprod	计算所有元素的累积积

通过函数改变数组的形状

改变数组形状





`numpy.reshape (arr, newshape,)`

`arr.reshape (newshape)`

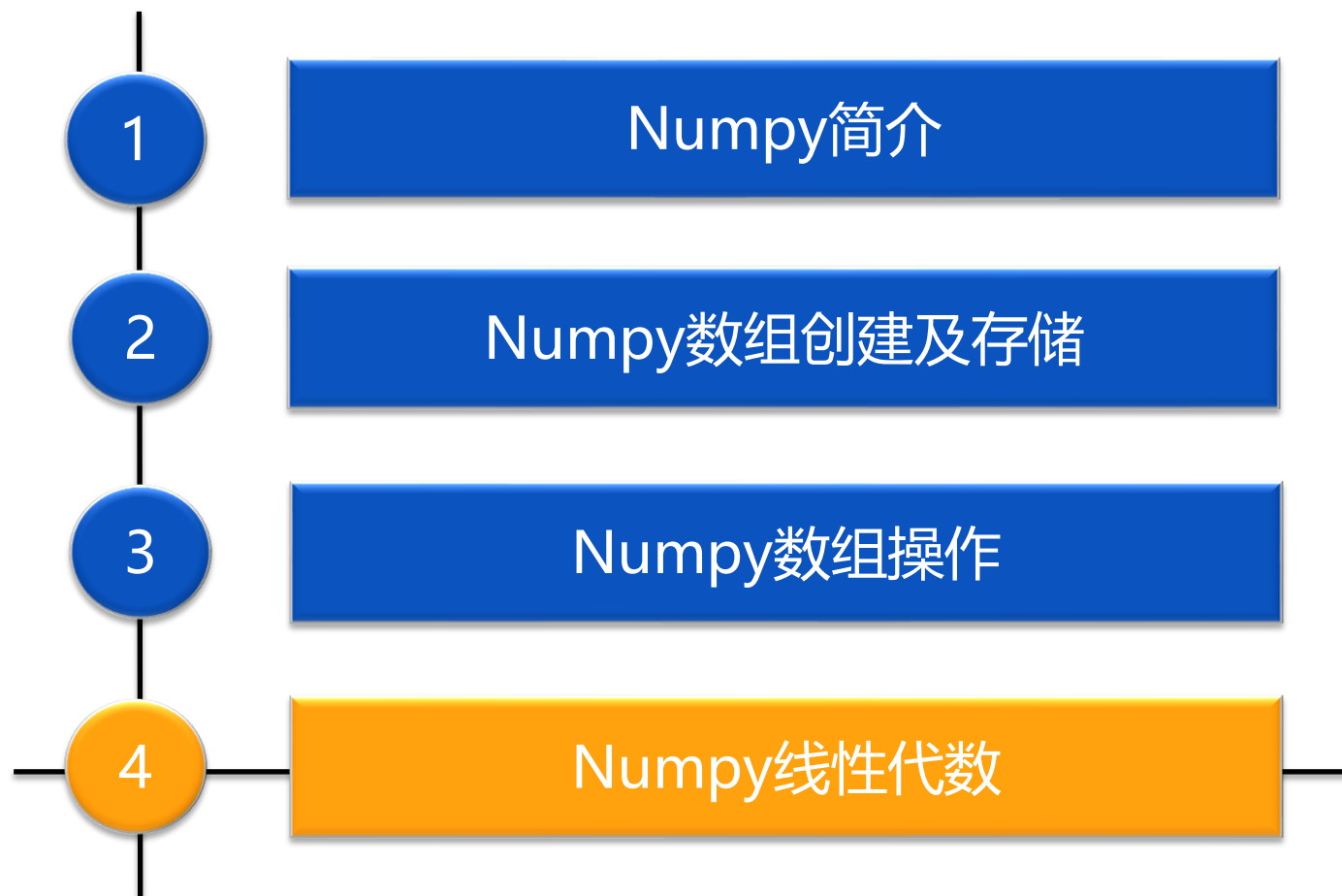
newshape: 新数组形状

`numpy.expand_dims(dim_index)` # 在指定维度上增加一个维度

展平数组

arr. ravel ()		arr. flatten ()	arr. flatten ('F') *按列展开
效果	方向	函数	
组合	横向	<code>hstack((arr1,arr2))</code> 、 <code>concatenate((arr1,arr2),axis = 1)</code>	 横向组合
	纵向	<code>vstack((arr1,arr2))</code> 、 <code>concatenate((arr1,arr2),axis = 0)</code>	 纵向组合
分割	横向	<code>vsplit(arr, n)</code> 、 <code>split(arr, n, axis=0)</code>	 横向分割
	纵向	<code>hsplit(arr, n)</code> 、 <code>split(arr, n, axis=1)</code>	 纵向分割

目录



NumPy基础：矩阵计算

- 矩阵与数相乘： `matr1*n`
- 矩阵相加减： `matr1±matr2`
- 矩阵相乘： `matr1*matr2`
- 矩阵对应元素相乘： `numpy.multiply(matr1,matr2)`

矩阵特有属性：

属性	说明
T	返回自身的转置
H	返回自身的共轭转置
I	返回自身的逆矩阵
A	返回自身数据的2维数组的一个视图

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \times n = \begin{pmatrix} a_{11} \times n & a_{12} \times n \\ a_{21} \times n & a_{22} \times n \end{pmatrix}$$

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \pm \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} a_{11} \pm b_{11} & a_{12} \pm b_{12} \\ a_{21} \pm b_{21} & a_{22} \pm b_{22} \end{pmatrix}$$

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \times \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} c_{11} & c_{12} \\ c_{21} & c_{22} \end{pmatrix}$$

$$c_{11} = a_{11} \times b_{11} + a_{12} \times b_{21}$$

$$c_{12} = a_{11} \times b_{12} + a_{12} \times b_{22}$$

$$c_{21} = a_{21} \times b_{11} + a_{22} \times b_{21}$$

$$c_{22} = a_{21} \times b_{12} + a_{22} \times b_{22}$$

$$\text{multiply}\left(\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix}, \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix}\right) = \begin{pmatrix} a_{11} \times b_{11} & a_{12} \times b_{12} \\ a_{21} \times b_{11} & a_{22} \times b_{12} \end{pmatrix}$$

Numpy线性函数

线性代数，比如矩阵乘法、分解、行列式等方阵数学，是所有数组类库的重要组成部分。numpy的linalg拥有一个矩阵分解的标准函数集。

函数	描述
dot	矩阵点乘
trace	计算对角元素和
det	计算矩阵行列式
eig	计算方阵的特征值和特征向量
inv	计算仿真的逆矩阵
qr	计算矩阵的QR分解
svd	计算矩阵的奇异值分解
solve	求解x的线性系统 $Ax=B$ ，其中A是矩阵
lstsq	计算 $Ax=B$ 的最小二乘解



Thank you!