Professur Technische Informatik
Prof. Dr. Wolfram Hardt

Introduction to Numpy Library

Exercise 1.6

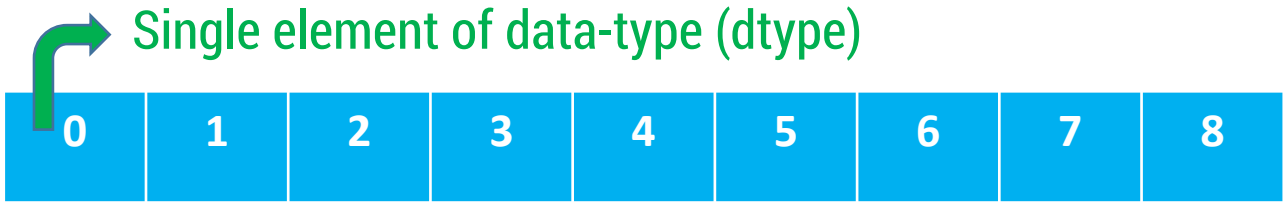
Shadi Saleh

Agenda

- Numpy: basics
- Array Shape
- Array Indexing and slicing
- Danger zone of array
- Image Arrays

What is Numpy

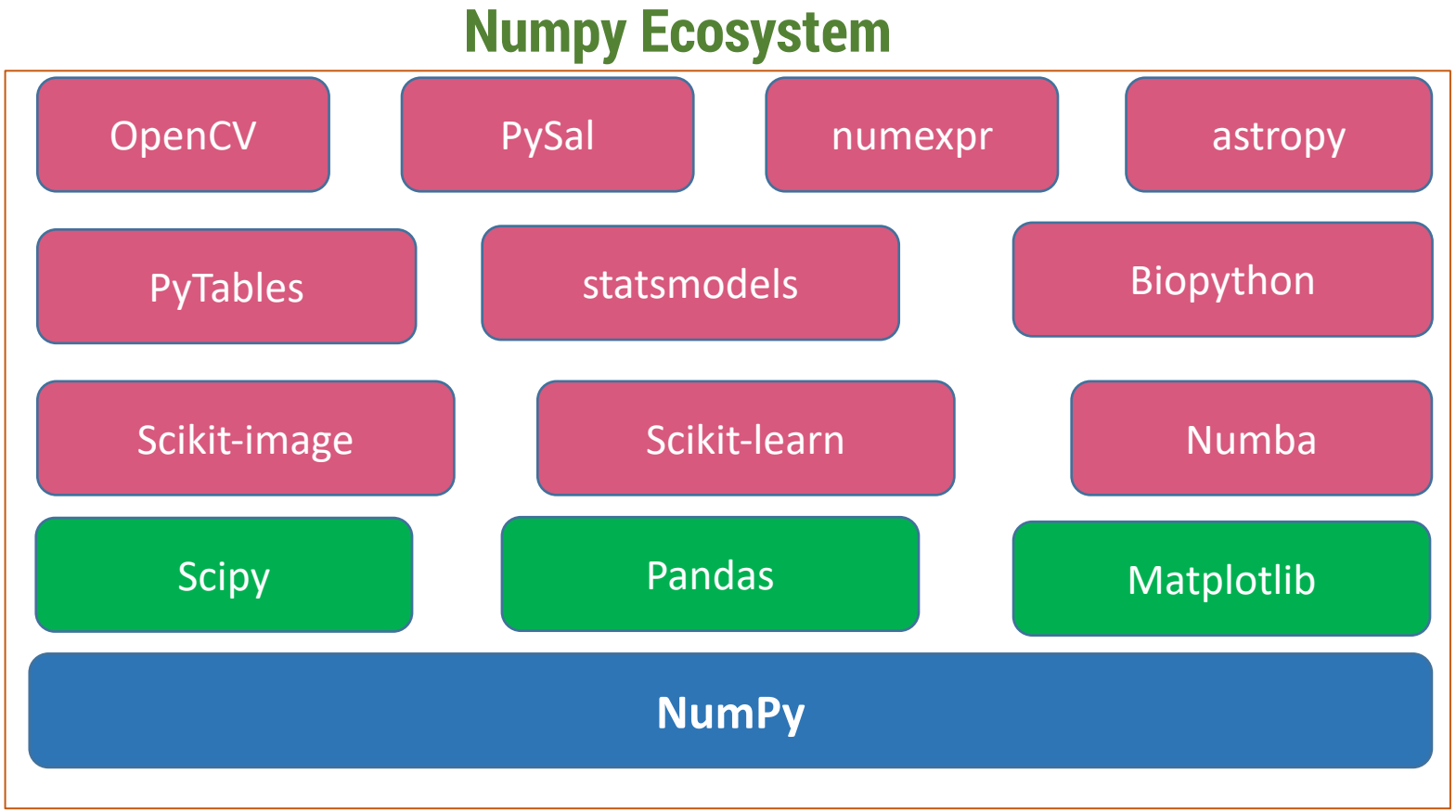
- NumPy is a Python C extension library for array-oriented computing
  - Efficient
  - In-memory
  - Homogeneous (but types can be algebraic)



- NumPy is suited to many applications
  - Image processing
  - Signal processing
  - Linear algebra
  - A plethora of others

Numpy supports

- NumPy is the foundation of the python scientific stack



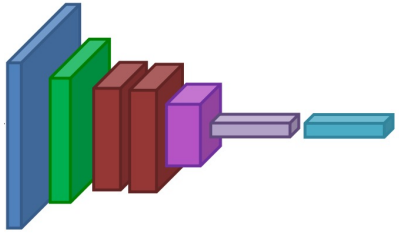
Arrays

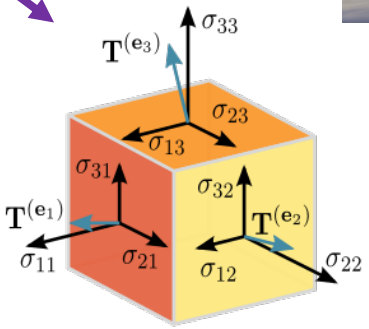
Structured lists of numbers.

- Vectors
- Matrices
- Images
- Tensors
- ConvNets

$$\begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}$$

$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} \end{bmatrix}$$





Arrays, Basic Properties

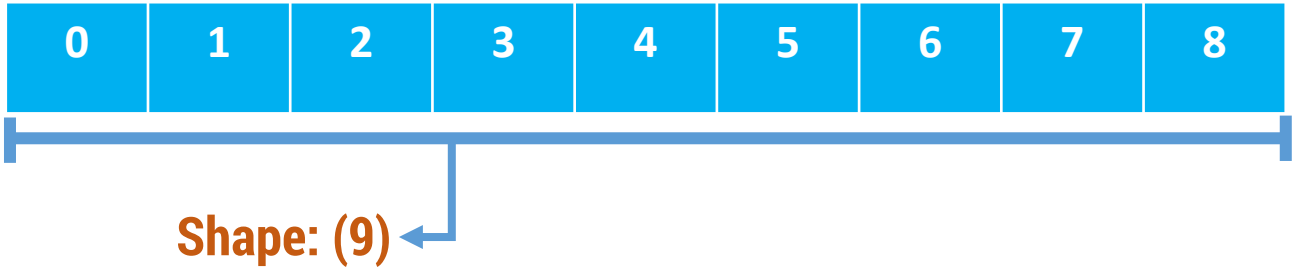
- Arrays can have any number of dimensions, including zero (a scalar).
- Arrays are typed: np.uint8, np.int64, np.float32, np.float64
- Arrays are dense. Each element of the array exists and has the same type.

```
import numpy as np
a = np.array( [[1,2,3],  [4,5,6]],  dtype=np.float32 )
print (a.ndim, a.shape, a.dtype)
```

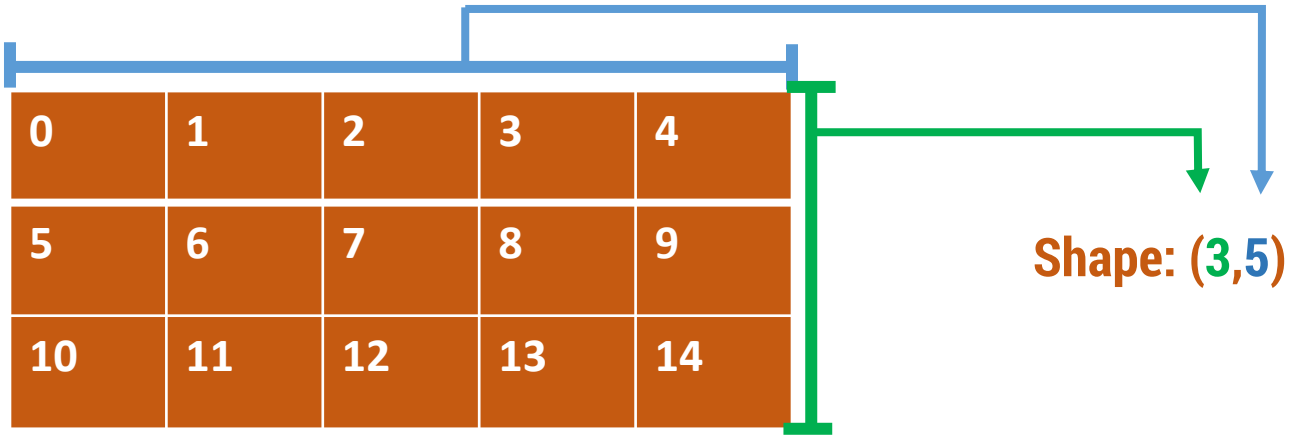
Output: `2 (2, 3) float32`

Array shape

- One dimensional arrays have a 1-tuple for their shape

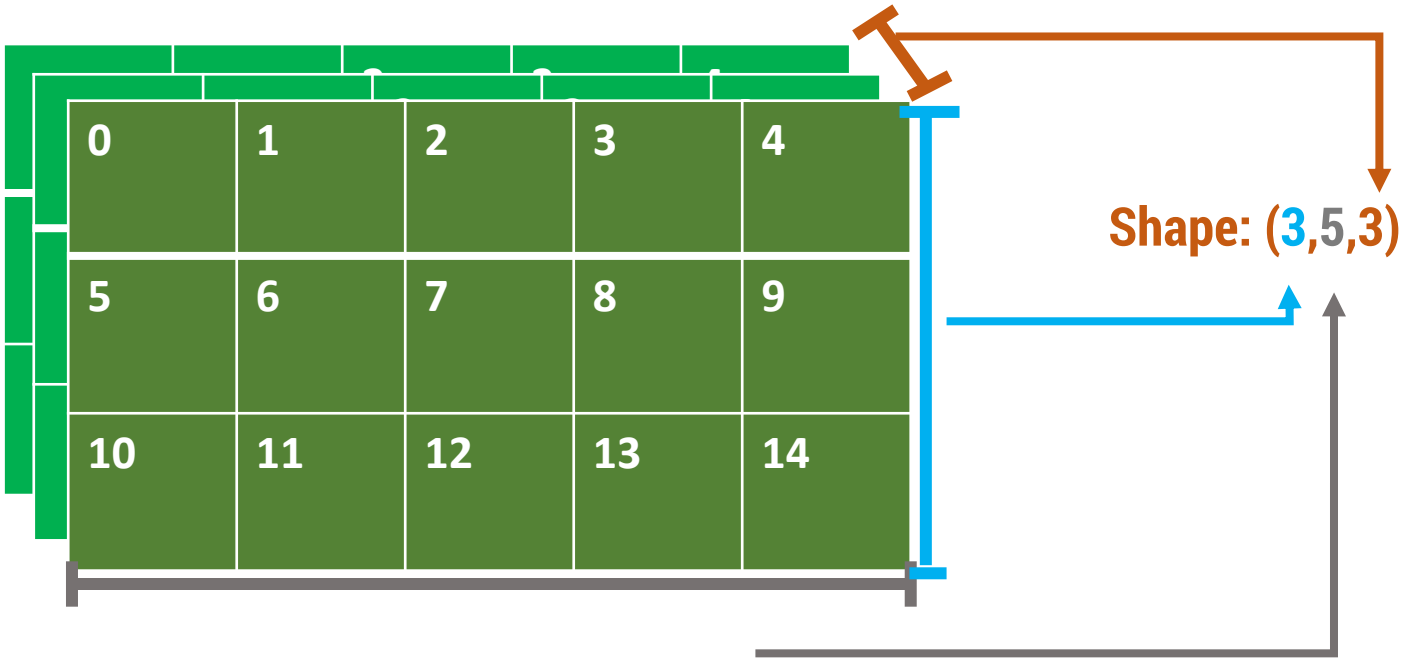


- Two dimensional arrays have a 2-tuple



Multi dimensional

- According to requirements multi dimensions are possible



Arrays, creation

- **np.ones, np.zeros**
- np.arange
- np.concatenate
- np.zeros_like, np.ones_like
- np.random.random

```
import numpy as np
a = np.ones (( 2 , 5 ),  dtype = np.float32 )
print (a)

Output:
[[1. 1. 1. 1. 1.]
 [1. 1. 1. 1. 1.]]
```

```
import numpy as np
a = np.zeros (( 3 , 5 ),  dtype = np.float32 )
print (a)

Output:
[[0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]]
```

Arrays, creation

- np.ones, np.zeros
- **np.arange**
- np.concatenate
- np.zeros_like, np.ones_like
- np.random.random

```
import numpy as np
a = np.arange ( 12 , 19 )
print (a)
```

Output:
[12 13 14 15 16 17 18]

Arrays, creation

- np.ones, np.zeros
- np.arange
- **np.concatenate**
- np.zeros_like, np.ones_like
- np.random.random

```
import numpy as np
a = np.ones (( 3, 1 ))
b = np.zeros (( 3, 2 ))

np.concatenate ( [a,b],   axis=1)


Output:
array(
        [[1., 0., 0.],
         [1., 0., 0.],
         [1., 0., 0.]]
)
```

Arrays, creation

- np.ones, np.zeros
- np.arange
- np.concatenate
- **np.zeros_like, np.ones_like**
- np.random.random

```
import numpy as np
a = np.ones (( 2 , 2, 3 ))
b = np.zeros_like(a)
print (b.shape)
```

Output:
(2, 2, 3)

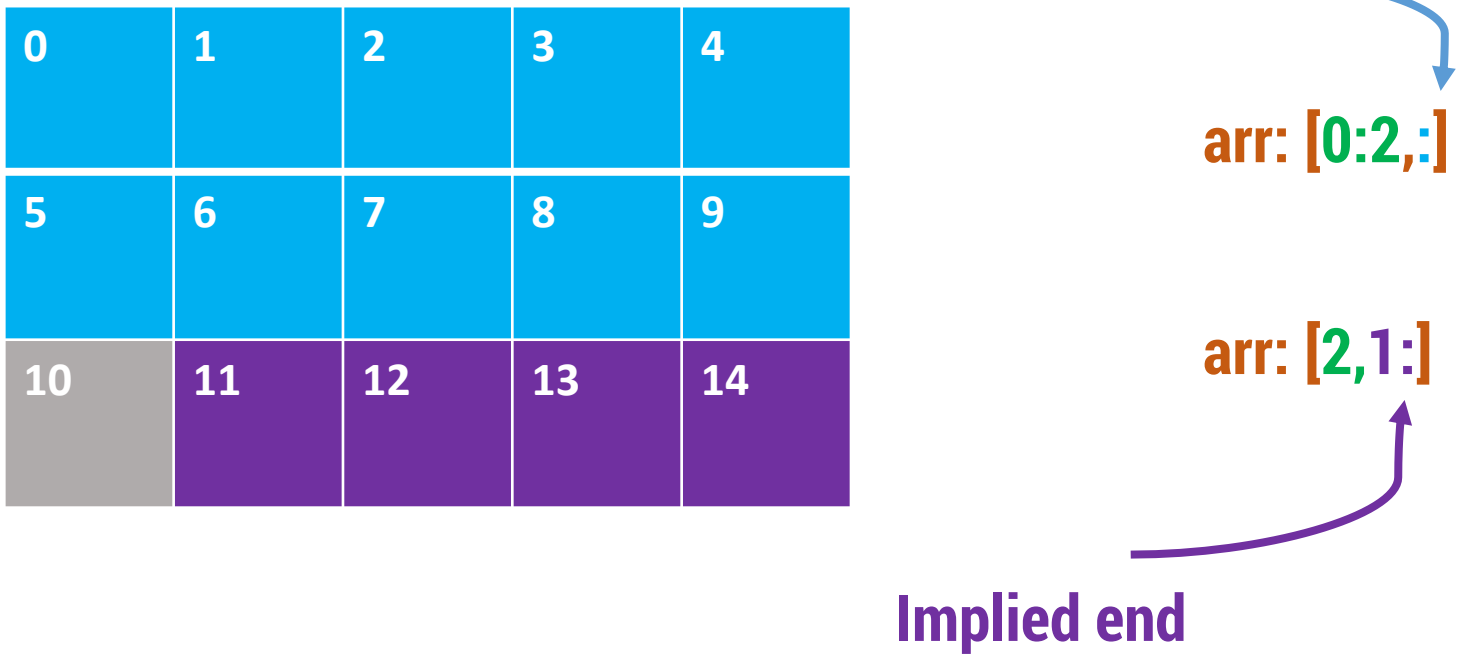
Arrays, creation

- np.ones, np.zeros
- np.arange
- np.concatenate
- np.zeros_like, np.ones_like
- **np.random.random**

```
import numpy as np
a = np.random.random ((  4,  2 ))
print (a)


Output:
[[0.00740995 0.42532528]
 [0.48785954 0.3486584 ]
 [0.06314456 0.66190268]
 [0.82574986 0.46058244]]
```

Indexing and slicing



Indices

- NumPy array indices can also take an optional stride

0	1	2	3	4
5	6	7	8	9
10	11	12	13	14

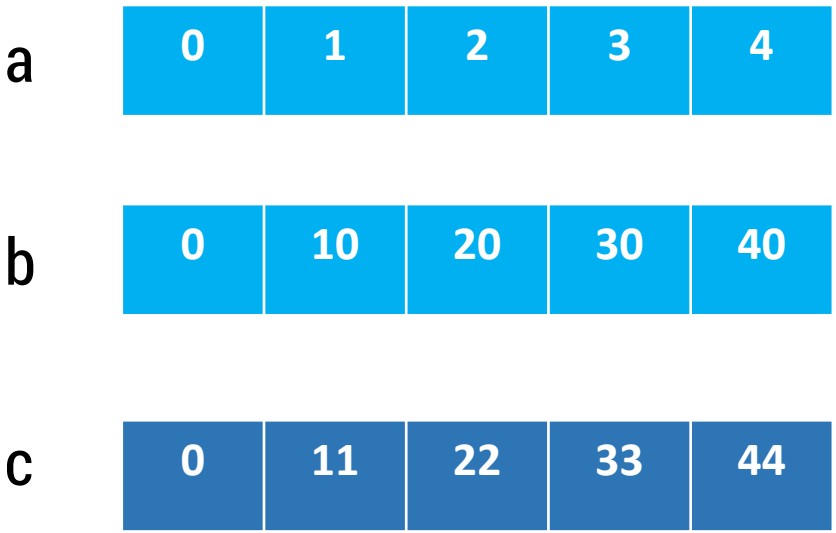
arr: [:,::2]

0	1	2	3	4
5	6	7	8	9
10	11	12	13	14

arr: [::2,::3]

Universal functions

- NumPy ufuncs are functions that operate element-wise on one or more arrays



Universal functions

- NumPy has many built-in ufuncs

operation	Symbols
Comparison	<, <=, ==, !=, >=, >
arithmetic	+, -, *, /, reciprocal, square
Exponential	exp, expm1, exp2, log, log10, log1p, log2, power, sqrt
Trigonometric	sin, cos, tan, acsin, arccos, atctan
Hyperbolic	sinh, cosh, tanh, acsinh, arccosh, atctanh
Bitwise operation	&, | , ~, ^, left_shift, right_shift
predicates	isfinite, isinf, isnan, signbit

Arrays, danger zone

- Must be dense, no holes.
- Must be one type
- Cannot combine arrays of different shape

```
import numpy as np
np.ones([5,6]) + np.ones([8,3])
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-316-c6c9d1b49ace> in <module>
      1 import numpy as np
      2
----> 3 np.ones([5,6]) + np.ones([8,3])

ValueError: operands could not be broadcast together with shapes (5,6) (8,3)
```

Return values

- Numpy functions return either **views** or **copies**.
- Views share data with the original array, like references in Java/C++. Altering entries of a view, changes the same entries in the original.
- The numpy documentation says which functions return views or copies
- Np.copy, np.view make explicit copies and views.

Saving and loading arrays

```
np.savez('data.npz', a=a)
data = np.load('data.npz')
a = data['a']
```

1. NPZ files can hold multiple arrays
2. np.savez_compressed similar.

Image arrays

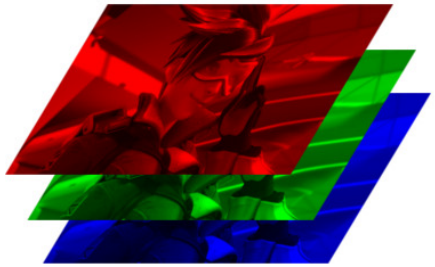
Images are 3D arrays: width, height, and channels

Common image formats:

- height x width x RGB (band-interleaved)
- height x width (band-sequential)

Gotchas:

- Channels may also be BGR (OpenCV does this)
- May be [width x height], not [height x width]



Saving and Loading Images

SciPy: skimage.io.imread,skimage.io.imsave

height x width x RGB

PIL / Pillow: PIL.Image.open, Image.save

width x height x RGB

OpenCV: cv2.imread, cv2.imwrite

height x width x BGR

Exercise 1

- Write a NumPy program to create an element-wise comparison (greater, greater_equal, less and less_equal) of two given arrays

Exercise 2

- Write a NumPy program to create an array of the integers from 30 to70

Thank you for your attention

Shadi Saleh

Professur Technische Informatik
Fakultät für Informatik

Technische Universität Chemnitz
Straße der Nationen 62
09111 Chemnitz
Germany

shadi.saleh@informatik.tu-chemnitz.de