Python – Lektion 8
Numpy I

Rückblick

- Vererbung von Klassen
- Mehrfachvererbung von Klassen
- Testat I

Heutige Themen

- NumPy – Numerical Python

Was ist NumPy?

- Python-Bibliothek

  `import numpy as np`

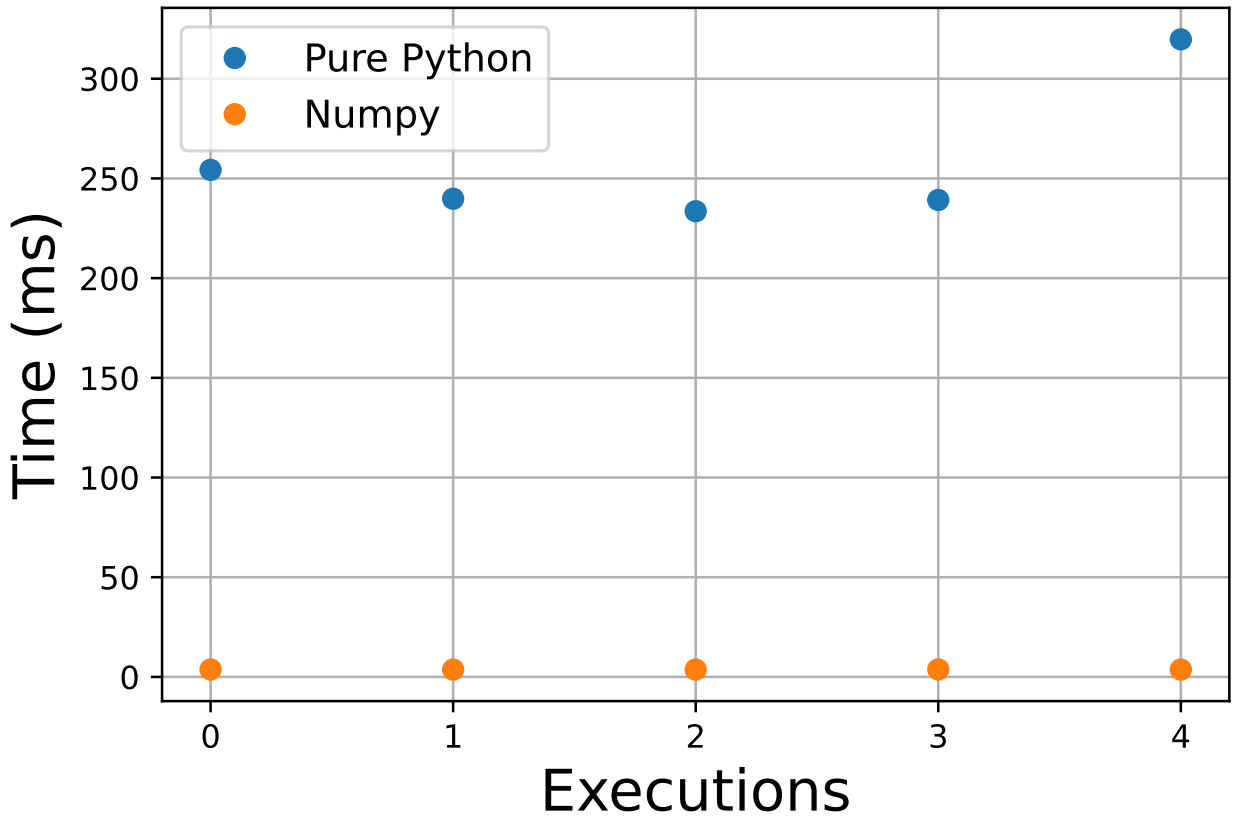
- Einfache Handhabung von Vektoren und Matrizen mit Hilfe mehrdimensionaler NumPy Arrays
- Funktionen für numerische Berechnungen
  - Grundlegende Operationen
  - Mathematische Funktionen (sin, cos, sqrt, exp, . . .)
  - Lineare Algebra
  - . . .
- Ähnlichkeit zu MATLAB®

  `https://docs.scipy.org/doc/numpy/user/numpy-for-matlab-users.html`

Vorteile der Verwendung von NumPy

Geschwindigkeit

Verwendung von in C implementierten Funktionen und Algorithmen



Vorteile der Verwendung von NumPy

Übersichtlichkeit

Mit Hilfe der NumPy Arrays können Array-basierte Operationen verwendet werden $\rightarrow$ keine `for`-Schleifen

- Umrechnung von Temperaturangaben in Celsius nach Fahrenheit mit "Standard" Python:

```
temp_C_list = [20.9, 21, 22.4, 23, 23.3]
temp_F_list = [val*9/5 + 32 for val in temp_C_list]
```

- Umrechnung von Temperaturangaben in Celsius nach Fahrenheit mit Numpy:

```
import numpy as np
temp_C_arr = np.array([20.9, 21, 22.4, 23, 23.3])
temp_F_arr = temp_C_arr*9/5 + 32
```

ndarray erzeugen

- n-dimensionales Array `ndarray`:

  Eine mehrdimensionale Sammlung von Elementen fester Grösse und desselben Typs

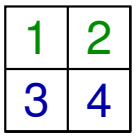
  `https://docs.scipy.org/doc/numpy/reference/generated/numpy.ndarray.html`

ndarray erzeugen

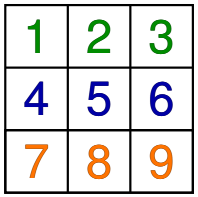
- `ndarray` erzeugen

```
arr1 = np.array([1,2])
```

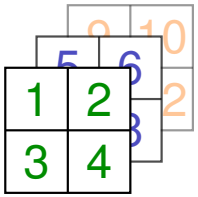

```
arr2 = np.array([[1,2], [3,4]])
```




```
arr3 = np.array([[1,2,3], [4,5,6], [7,8,9]])
```




```
arr4 = np.array([[[1,2],[3,4]],[[5,6],[7,8]],[[9,10],[11,12]]])
```



http://localhost:8888/notebooks/ndarray_erzeugen.ipynb

ndarray erzeugen

- Weitere Methoden, um Arrays zu erzeugen

Funktion	Resultat
`np.arange(0, 4)`	`array([0, 1, 2, 3])`
`np.ones((2,2))`	`array([[1., 1.], [1., 1.]])`
`np.ones_like(arr1)`	`array([1, 1, 1])`
`np.zeros((2,2))`	`array([[0., 0.], [0., 0.]])`
`np.zeros_like(arr1)`	`array([0, 0, 0])`
`np.full((2,2), 7.0)`	`array([[7., 7.], [7., 7.]])`
`np.full_like(arr1, 7)`	`array([7, 7, 7])`
`np.eye(2)`	`array([[1., 0.], [0., 1.]])`
`np.identity(2)`	`array([[1., 0.], [0., 1.]])`
`np.linspace(0, 1, 5)`	`array([0., 0.25, 0.5, 0.75, 1.])`
`np.logspace(0, 1, 4)`	`array([1., 2.1544, 4.6416, 10.])`
`np.random.randn(3)`	`array([0.7576, 0.0135, -0.8934])`
`np.random.randint(0,10,3)`	`array([0, 5, 4])`

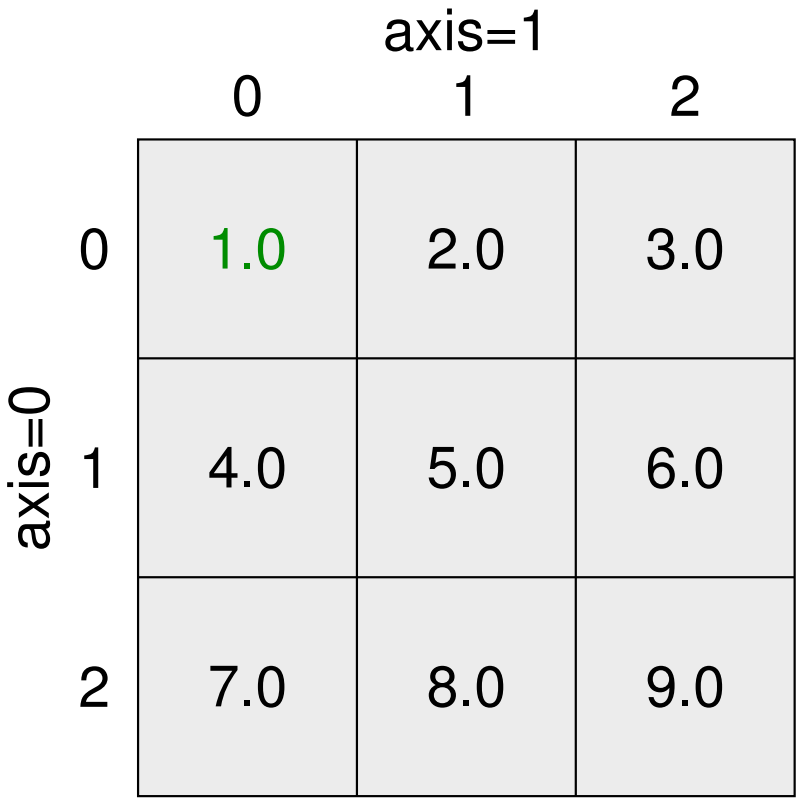
Indexierung

- Indexierung von 2D-Arrays

```
arr[axis0, axis1]
```

- Beispiele:

```
>>> arr[0, 0]
1.0
```



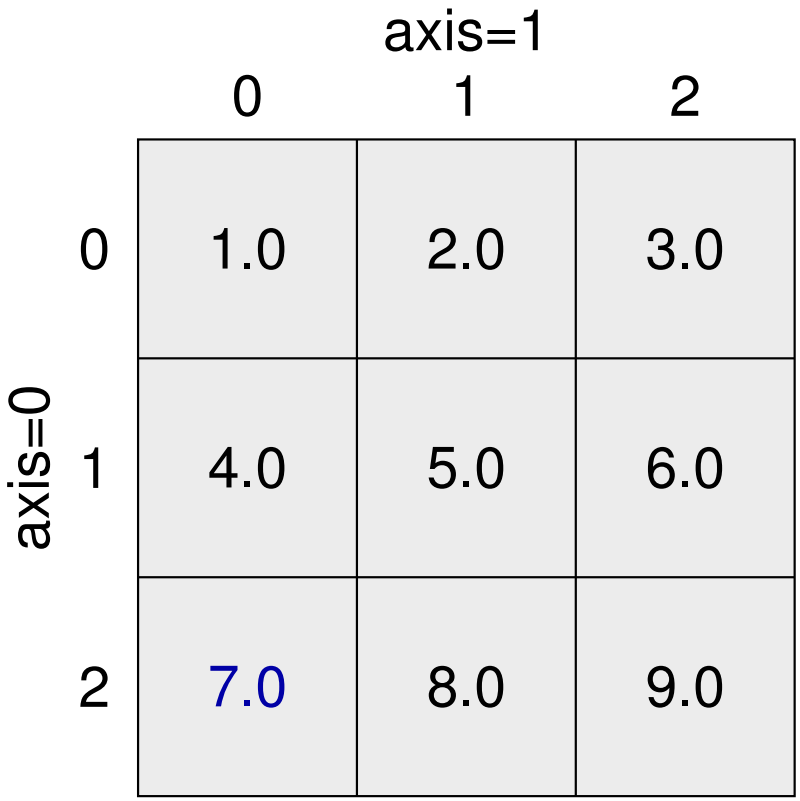
Indexierung

- Indexierung von 2D-Arrays

```
arr[axis0, axis1]
```

- Beispiele:

```
>>> arr[2, 0]
7.0
```



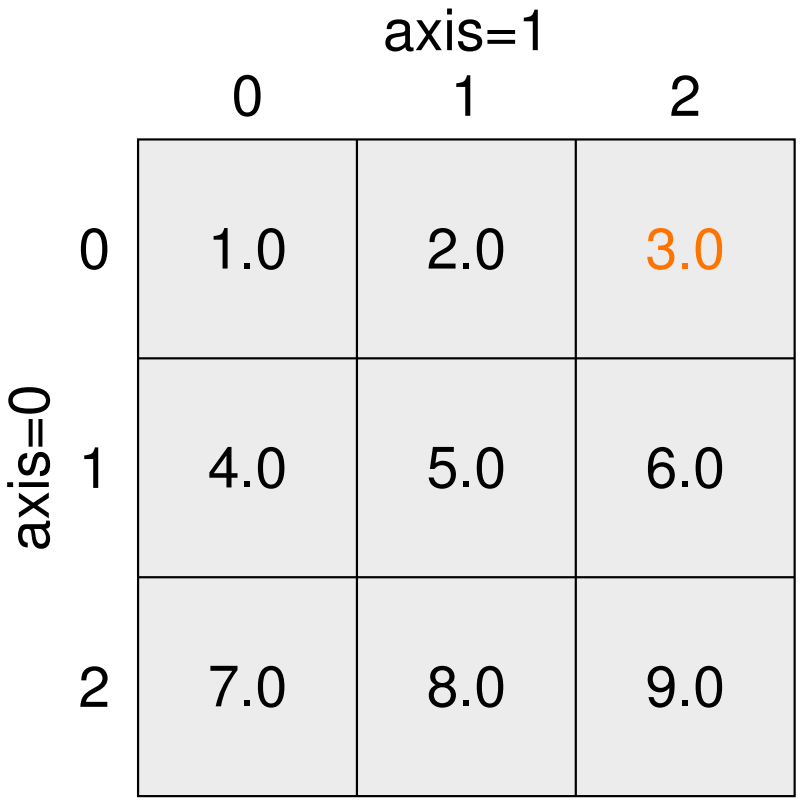
Indexierung

- Indexierung von 2D-Arrays

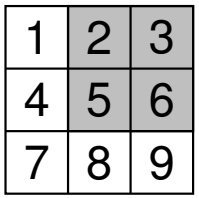
```
arr[axis0, axis1]
```

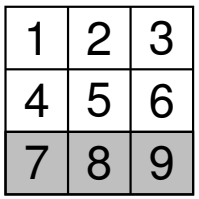
- Beispiele:

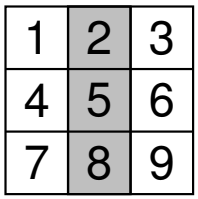
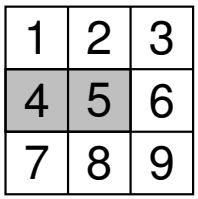
```
>>> arr[0, 2]
3.0
```



Slicing

arr	Ausdruck	Shape	Resultat
	`arr[:2, 1:]`	(2, 2)	`array([[2, 3], [5, 6]])`

	`arr[2]`
	`arr[2, :]`
	`arr[2:, :]`
		(3,)
		(3,)
		(1, 3)
			`array([7, 8, 9])`
			`array([7, 8, 9])`
			`array([[7, 8, 9]])`

	`arr[:, 2]` `arr[:, 2:]`	(3,) (3, 1)	`array([3, 6, 9])` `array([3, 6, 9])`
	`arr[1, :2]` `arr[1:2, :2]`	(2,) (1, 2)	`array([4, 5])` `array([[4, 5]])`

→ `ndim` bleibt erhalten, falls bei jeder `axis` ein ":" steht.

`http://localhost:8888/notebooks/indexierung_und_slicing.ipynb`

Mathematische Funktionen

- NumPy beinhaltet viele mathematische Funktionen:
  `https://docs.scipy.org/doc/numpy/reference/routines.math.html`
  - `np.sin()`
  - `np.cos()`
  - `np.exp()`
  - `np.cumsum()`
  - `...`
- Diese Funktionen operieren über das gesamte Array

```
>>> t = np.linspace(1, 3, 5)
array([1, 1.5, 2, 2.5, 3])
>>> np.exp(t)
array([2.718,  4.481,  7.389 , 12.182, 20.085])
>>> np.cumsum(t)
array([1,  2.5,  4.5,  7, 10])
>>> np.mean(t)
2.0
```

Lineare Algebra

- Liste der Funktionen:
  `https://docs.scipy.org/doc/numpy/reference/routines.linalg.html`
- Matrix $\boldsymbol{M}$ mit Vektor $\boldsymbol{v}$ multiplizieren
- Matrix transponieren $\boldsymbol{M}^T$
- Matrix invertieren $\boldsymbol{M}^{-1}$
- Lineares Gleichungssystem $\boldsymbol{Ax} = \boldsymbol{b}$ lösen

`http://localhost:8888/notebooks/linalg.ipynb`