VPython - symulacje fizyczne z grafiką 3D dla każdego

wykład 5

Dr hab. Adam Bzdak

Dlaczego NumPy ?

numpy9.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_4\numpy9.py

File Edit Format Run Options Windows Help

```
from __future__ import division
import numpy as np

a = np.array([1,2,3])

print a*10
print a/2
print a**2
print a + 10.5
print np.cos(a)
print a>0
```

Ln: 13 Col: 0

szybkie operacje na każdym elemencie

np.cos(a**)** – używamy funkcji z NumPy

```
>>>
[10 20 30]
[ 0.5  1.   1.5]
[1 4 9]
[ 11.5  12.5  13.5]
[ 0.54030231 -0.41614684 -0.9899925 ]
[ True  True  True]
```

Ln: 1 Col: 0

Dlaczego NumPy ?

numpy10.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_4\numpy10.py

File Edit Format Run Options Windows Help

```
from __future__ import division
import numpy as np

a = np.array([1, 2, 3])

b = np.array([5.0, 4.0, 3.0])

print a+b
print a*b
print b/a
print a>b
print (a-b)==0
```

Ln: 14 Col: 0

```
>>>
[ 6.  6.  6.]
[ 5.  8.  9.]
[ 5.  2.  1.]
[False False False]
[False False  True]
```

Ln: 1 Col: 0

Liczby pseudolosowe

random_1.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\random_1.py

File Edit Format Run Options Windows Help

```
import numpy as np


print np.random.normal(0,5, (2,3))
print
```

$\boldsymbol{\mu}, \boldsymbol{\sigma}$

$$\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

```
print np.random.exponential(0.5, (2,4))
print
```

$\boldsymbol{\beta}$

$$\exp(-x/\beta),\ x > 0$$

```
print np.random.poisson(5, 10)
```

$\boldsymbol{\lambda}$

$$\frac{\lambda^n}{n!}e^{-\lambda} \qquad \lambda = \langle n \rangle$$

Ln: 14 Col: 0

```
>>>
[[-0.53985692 -1.19592883  5.93510607]
 [-2.34318102 -1.10001634 -4.00327543]]

[[ 0.31203411  0.06056116  0.01186506  0.87519643]
 [ 0.22901289  0.30740062  0.56839502  0.76574675]]

[ 4  4  3  5 10  4  2  9  7  5]
```

Ln: 1 Col: 0

permutation, choice

random_2.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\random_2.py

File Edit Format Run Options Windows Help

```
import numpy as np

print np.random.permutation([1,10,100,1000])
print np.random.permutation(np.arange(1,11))
print np.random.permutation(['a','b','c'])
print ''

print np.random.choice(['a','b','c','d'], 2)
print np.random.choice(['a','b','c','d'], (2,10))
```

Ln: 11 Col: 0

```
>>>
[  10    1  100 1000]
[ 5  7  3  6  9  2  1  4 10  8]
['c' 'b' 'a']

['d' 'd']
[['b' 'c' 'a' 'd' 'd' 'b' 'a' 'd' 'a' 'a']
 ['a' 'd' 'b' 'd' 'b' 'd' 'a' 'd' 'd' 'a']]
```

Ln: 1 Col: 0

Arrays z różnymi typami

numpy42.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\numpy42.py

File Edit Format Run Options Windows Help

```
import numpy as np

a = np.array([1, 2, 3])
print a.dtype.name

b = np.array([1.0, 2.0, 3.0])
print b.dtype.name

c = a + b
print c
print c.dtype.name
```

Ln: 13 Col: 0

```
>>>
int32
float64
[ 2.  4.  6.]
float64
>>>
```

Ln: 1 Col: 0

końcowy obiekt jest bardziej ogólny (upcasting).

ostrożnie z +=, *=, itd

sum(axis=...)

numpy44.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\numpy44.py

File Edit Format Run Options Windows Help

```
import numpy as np

a = np.arange(0,12).reshape(3,4)

print a, '\n'

print a.sum()

print a.sum(axis=0)

print a.sum(axis=1)
```

3 wiersze, 4 kolumny

suma wszystkich elementów

suma z każdej kolumny

suma z każdego wiersza

Ln: 13 Col: 0

```
>>>
[[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]]
66
[12 15 18 21]
[ 6 22 38]
```

3

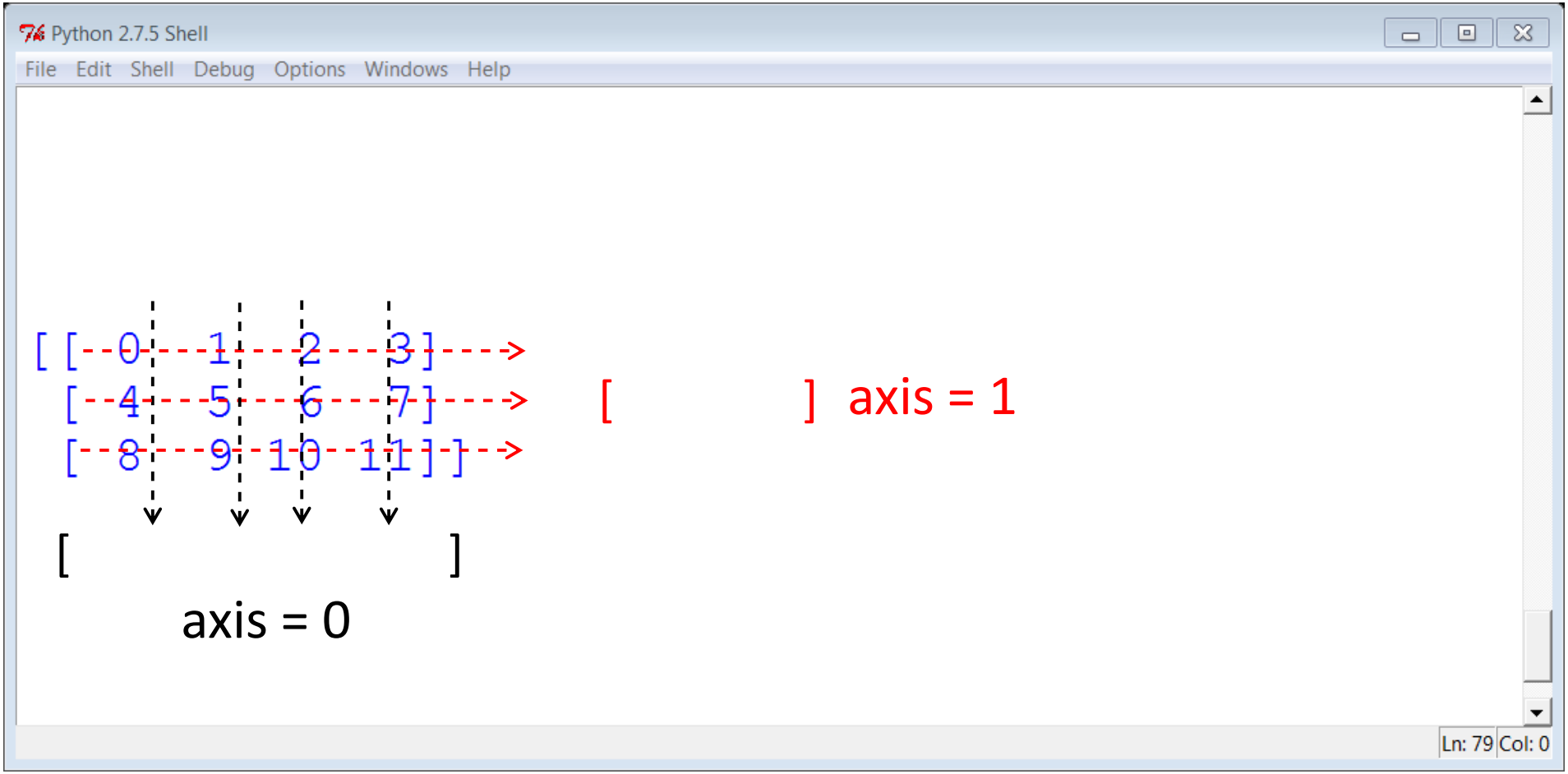
4

axis = 0

[ ]

Ln: 12 Col: 4

axis=0, axis=1


Python 2.7.5 Shell
File Edit Shell Debug Options Windows Help
[[0 1 2 3]
[4 5 6 7]
[8 9 10 11]]
[ ]
axis = 0
[ ] axis = 1
Ln: 79 Col: 0

min(axis=...), max(axis=...)

numpy45.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\numpy45.py

File Edit Format Run Options Windows Help

```
import numpy as np

a = np.arange(0,12).reshape(3,4)

print a, '\n'

print a.min()
print a.min(axis=0)
print a.min(axis=1)
```

najmniejszy element
min z każdej kolumny
min z każdego wiersza

Ln: 11 Col: 0

```
>>>
[[ 0  1  2  3]
 [ 4  5  6  7]
 [ 8  9 10 11]]

0
[0 1 2 3]
[0 4 8]
```

Inny sposób:
np.min(a)
np.min(a, axis=0)
np.min(a, axis=1)

Ln: 1 Col: 0

np.sort(axis=...)

numpy46_b.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\numpy46_b.py

File Edit Format Run Options Windows Help

```
import numpy as np

a = np.array([[8,4,1], [2,1,0], [3,13,2]])
print a, '\n'

print np.sort(a, axis=0)
print ''

print np.sort(a, axis=1)
```

sortuje każdą kolumnę

sortuje każdy wiersz

Ln: 11 Col: 0

```
[[ 8  4  1]
 [ 2  1  0]
 [ 3 13  2]]

[[ 2  1  0]
 [ 3  4  1]
 [ 8 13  2]]

[[ 1  4  8]
 [ 0  1  2]
 [ 2  3 13]]
```

a.sort(axis=...)
zmienia sam obiekt

Ln: 1 Col: 0

identity()

numpy47.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\numpy47.py

File Edit Format Run Options Windows Help

```
import numpy as np

a = np.identity(4)
print a, '\n'

a = np.identity(7, dtype=np.int)
print a
```

Ln: 9 Col: 0

```
[[ 1.  0.  0.  0.]
 [ 0.  1.  0.  0.]
 [ 0.  0.  1.  0.]
 [ 0.  0.  0.  1.]]

[[1 0 0 0 0 0 0]
 [0 1 0 0 0 0 0]
 [0 0 1 0 0 0 0]
 [0 0 0 1 0 0 0]
 [0 0 0 0 1 0 0]
 [0 0 0 0 0 1 0]
 [0 0 0 0 0 0 1]]
>>>
```

Ln: 17 Col: 4

za duży array

numpy8.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\numpy8.py

File Edit Format Run Options Windows Help

```
import numpy as np

# np.set_printoptions(threshold='nan')

a = np.arange(10000)
print a.reshape(100,100)
```

Ln: 8 Col: 0

```
[[    0     1     2 ...,    97    98    99]
 [  100   101   102 ...,   197   198   199]
 [  200   201   202 ...,   297   298   299]
 ...,
 [9700 9701 9702 ..., 9797 9798 9799]
 [9800 9801 9802 ..., 9897 9898 9899]
 [9900 9901 9902 ..., 9997 9998 9999]]
```

Ln: 1 Col: 0

Jeśli obiekt jest za duży to NumPy pokazuje tylko *rogi*.
Można to wyłączyć poprzez set_printoptions.

linspace

numpy7.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\numpy7.py

File Edit Format Run Options Windows Help

```
import numpy as np

x = np.linspace(0, 1, 6)
print x
```

6 liczb od 0 do 1

Ln: 8 Col: 0

```
>>>
[ 0.   0.2  0.4  0.6  0.8  1. ]
>>>
```

Ln: 1 Col: 0

Iteracja

iterating_1.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\iterating_1.py

File Edit Format Run Options Windows Help

```
import numpy as np

a = np.arange(20)**2
a = a.reshape(5,4)
print a, '\n'

for i in a:
    print i
```

iteracja po każdym rzędzie

Ln: 10 Col: 0

```
[[  0   1   4   9]
 [ 16  25  36  49]
 [ 64  81 100 121]
 [144 169 196 225]
 [256 289 324 361]]

[0 1 4 9]
[16 25 36 49]
[ 64  81 100 121]
[144 169 196 225]
[256 289 324 361]
```

Ln: 16 Col: 4

Iteracja, flat

iterating_2.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\iterating_2.py

File Edit Format Run Options Windows Help

```
import numpy as np

a = np.arange(20)**2
a = a.reshape(5,4)
print a, '\n'

for i in a.flat:
    print i,
```

iteracja po każdym elemencie

to samo z

```
for i in a:
    for el in i:
        print el,
```

Ln: 10 Col: 0

```
>>>
[[  0   1   4   9]
 [ 16  25  36  49]
 [ 64  81 100 121]
 [144 169 196 225]
 [256 289 324 361]]

0 1 4 9 16 25 36 49 64 81 100 121 144 169 196 225 256 289 324
361
```

Ln: 12 Col: 4

```
+---+---+---+---+---+---+
| P | y | t | h | o | n |
+---+---+---+---+---+---+

  0   1   2   3   4   5

 -6  -5  -4  -3  -2  -1
```

s[2:5] = ’tho’

włączone wyłączone

Cięcia (slicing)

slicing_1.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\slicing_1.py

File Edit Format Run Options Windows Help

```
import numpy as np

a = np.arange(5)**2
print a, '\n'

print a[1], a[-2]

print a[2:4]

print a[:3]

print a[3:]

print a[:]
```

a[: 3] = a[0 : 3]

a[3 :] = a[3 : len(a)] (do ostatniego elementu)

a[:] = a[0 : len(a)]

Ln: 16 Col: 0

```
[ 0  1  4  9 16]

1 9
[4 9]
[0 1 4]
[ 9 16]
[ 0  1  4  9 16]
```

Ln: 1 Col: 0

slicing

slicing_2.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\slicing_2.py

File Edit Format Run Options Windows Help

```
import numpy as np

a = np.arange(7)**2
print a, '\n'

print a[2:6:2]

print a[ : :3]

print a[ : :-1]

a[2:6] = 0
print a
```

co drugi element

a[: : 3] = a[0 : len(a) : 3] (co trzeci element)

do tyłu

Ln: 15 Col: 0

```
[ 0  1  4  9 16 25 36]

[ 4 16]
[ 0  9 36]
[36 25 16  9  4  1  0]
[ 0  1  0  0  0  0 36]
```

Ln: 1 Col: 0

slicing

slicing_3.py - C:\Users\Wysokie Energie\Desktop\VPytho...

File Edit Format Run Options Windows Help

```
import numpy as np

a = np.arange(20)**2
a = a.reshape(5,4)

print a, '\n'
```

2 wiersz, 3 kolumna

```
print a[2, 3], '\n'
```

wiersze, kolumny

```
print a[0:3, 0:2], '\n'
```

wiersze, wszystkie kolumny

```
print a[0:3, :]
```

Ln: 17 Col: 0

Python 2.7.5 Shell

File Edit Shell Debug Options Windows Help

```
>>>
[[  0   1   4   9]
 [ 16  25  36  49]
 [ 64  81 100 121]
 [144 169 196 225]
 [256 289 324 361]]

121

[[ 0  1]
 [16 25]
 [64 81]]

[[  0   1   4   9]
 [ 16  25  36  49]
 [ 64  81 100 121]]
>>>
```

Ln: 20 Col: 4

copy

copy_1.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\copy_1.py

File Edit Format Run Options Windows Help

```
import numpy as np

a = np.random.random_integers(0,10,6)
print a, '\n'

b = a

b[0] = 0

print a
```

Ln: 12 Col: 0

a i b wskazują na ten sam obiekt

to samo z np. b = a[2 : 4] (view)

```
>>>
[6 2 9 6 1 5]

[0 2 9 6 1 5]
>>>
```

Ln: 1 Col: 0

copy

copy_3.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\copy_3.py

File Edit Format Run Options Windows Help

```
import numpy as np

a = np.random.random_integers(0,10,5)
print a, '\n'

b = a.copy()
b[:] = 0

print a, b
```

tworzy się nowy obiekt

Ln: 11 Col: 0

```
>>>
[ 7  5 10  9  6]

[ 7  5 10  9  6] [0 0 0 0 0]
>>>
```

Ln: 1 Col: 0

newaxis

newaxis.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\newaxis.py

File Edit Format Run Options Windows Help

```
import numpy as np

a = np.random.random_integers(1,10,5)
print a, '\n'

b = a[ : ,np.newaxis]
print b
```

powstaje wektor

Ln: 9 Col: 0

```
>>>
[ 5  8  3 10 10]

[[ 5]
 [ 8]
 [ 3]
 [10]
 [10]]
>>>
```

Ln: 1 Col: 0

trik z boolean

trick_boolean.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\trick_boolean.py

File Edit Format Run Options Windows Help

```python
import numpy as np

a = np.random.random_integers(0,10,(2,8))
print a, '\n'

b = (a >= 5)
print b, '\n'

a[b] = 0
print a
```

Ln: 12 Col: 0

```
>>>
[[ 3  4  5  0  7 10 10  7]
 [ 5  6  6  0  6  5  5  1]]

[[False False  True False  True  True  True  True]
 [ True  True  True False  True  True  True False]]

[[3 4 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 1]]
```

Ln: 1 Col: 0

zaokrąglanie, round

round.py - C:\Users\Wysokie Energie\Desktop\VPython\lec_5\round.py

File Edit Format Run Options Windows Help

```
import numpy as np

a = np.random.uniform(0,10,(2,4))
print a, '\n'

print np.round(a, 3)
```

Ln: 8 Col: 0

```
>>>
[[ 0.50074407  9.26755549  0.20833275  7.43238577]
 [ 6.42303532  0.8144777   9.92471464  0.97883386]]

[[ 0.501  9.268  0.208  7.432]
 [ 6.423  0.814  9.925  0.979]]
>>>
```

Ln: 1 Col: 0

Proszę zobaczyć
https://docs.scipy.org/doc/numpy/reference/routines.html
funkcje NumPy kategoriami

File Edit View History Bookmarks Tools Help

Routines — NumPy v1.16 Manu

https://docs.scipy.org/doc/numpy/reference/routines.html 110%

- Binary Format Description
- Linear algebra (`numpy.linalg`)
  - Matrix and vector products
  - Decompositions
  - Matrix eigenvalues
  - Norms and other numbers
  - Solving equations and inverting matrices
  - Exceptions
  - Linear algebra on several matrices at once
- Logic functions
  - Truth value testing
  - Array contents
  - Array type testing
  - Logical operations
  - Comparison
- Masked array operations
  - Constants
  - Creation
  - Inspecting the array
  - Manipulating a MaskedArray
  - Operations on masks
  - Conversion operations
  - Masked arrays arithmetics
- Mathematical functions
  - Trigonometric functions
  - Hyperbolic functions
  - Rounding
  - Sums, products, differences

Proszę zobaczyć:
https://scipy.github.io/old-wiki/pages/Numpy_Example_List_With_Doc.html

Bardzo dużo różnych funkcji

File Edit View History Bookmarks Tools Help

Numpy_Example_List_With_Do...

scipy.github.io/old-wiki/pages/Numpy_Example_List_With_Doc.html

402. var()
403. vdot()
404. vectorize()
405. view()
406. vonmises()
407. vsplit()
408. vstack()
409. weibull()
410. where()
411. who()
412. zeros()
413. zeros_like()

...

```
>>> from numpy import *
>>> a = arange(12)
>>> a = a.reshape(3,2,2)
>>> print a
[[[ 0  1]
 [ 2  3]]
 [[ 4  5]
 [ 6  7]]
 [[ 8  9]
```