Lukas Müller

Programmieren für Studierende der Naturwissenschaften

V8 – Mehr Matplotlib und SciPy

Das Gute zuerst

- Die wichtigsten Sprachelemente kennt ihr nun bereits

- Auch die Grundlagen des Arbeitens mit Dateien haben wir besprochen
  - In anderen Programmiersprachen wird das anders aussehen!
- **ABER:**
  - Strukturen (Schleifen, Verzweigungen, Funktionen)
  - Prinzipien (Variablen, Datentypen, Zuweisung, Speicher)

  - …werdet ihr in anderen Sprachen auch wieder treffen (vielleicht mit etwas anderen Regeln)

Heute: Mehr Spielereien mit Daten

- Lineare Regression werden wir in diesem Semester nicht durchgehen

- Plotten von Bildern

- Nicht, weil genau das für euer Leben relevant ist
- SONDERN: Wie gehe ich vor, wenn ich etwas Neues versuchen möchte/muss?

Das Ausprobieren größere Testmengen generieren - Herangehensweise

- Künstlich Testdaten erzeugen, dazu
  - linear verteilte Werte erzeugen (np.linspace)
  - in eine lineare Funktion einsetzen (Parameter vorher festlegen)
  - Werte „stören“ (Rauschen addieren, als wären es echte Werte) (random oder gaussian)

Zufall?

numpy.random.randn

numpy.random.randn(*d0*, *d1*, ..., *dn*)

Return a sample (or samples) from the "standard normal" distribution.

If positive, int_like or int-convertible arguments are provided, randn generates an array of shape `(d0, d1, ..., dn)`, filled with random floats sampled from a univariate "normal" (Gaussian) distribution of mean 0 and variance 1 (if any of the d_i are floats, they are first converted to integers by truncation). A single float randomly sampled from the distribution is returned if no argument is provided.

This is a convenience function. If you want an interface that takes a tuple as the first argument, use numpy.random.standard_normal instead.

Parameters: **d0, d1, ..., dn** : *int, optional*
The dimensions of the returned array, should be all positive. If no argument is given a single Python float is returned.

Returns: **Z** : *ndarray or float*
A `(d0, d1, ..., dn)`-shaped array of floating-point samples from the standard normal distribution, or a single such float if no parameters were supplied.

See also:
random.standard_normal Similar, but takes a tuple as its argument.

Notes

For random samples from $N(\mu, \sigma^2)$, use:

```
sigma * np.random.randn(...) + mu
```

Examples

```
>>> np.random.randn()
2.1923875335537315 #random
```

Two-by-four array of samples from N(3, 6.25):

```
>>> 2.5 * np.random.randn(2, 4) + 3
array([[-4.49401501,  4.00950034, -1.81814867,  7.29718677],  #random
       [ 0.39924804,  4.68456316,  4.99394529,  4.84057254]]) #random
```

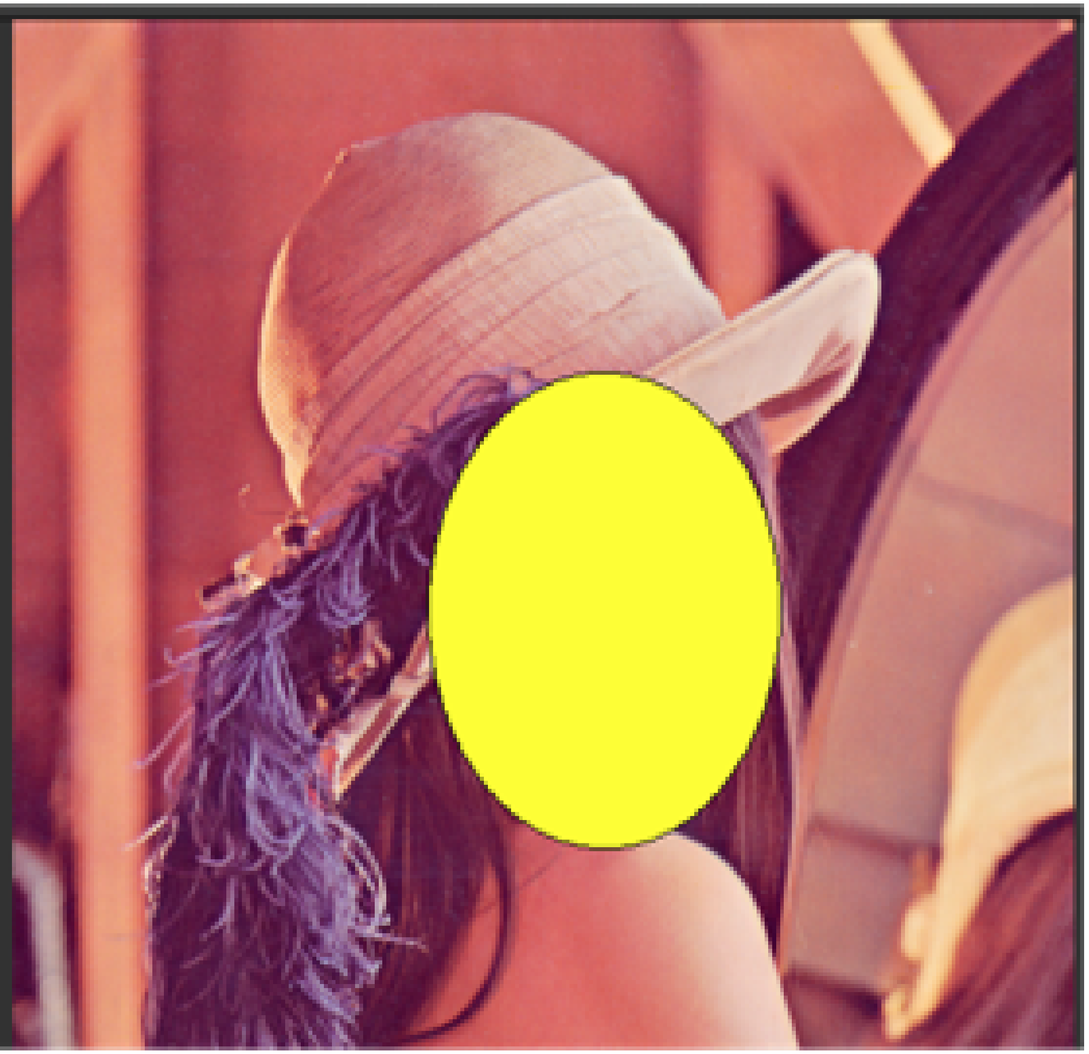
Noch ein Beispiel - Bilder

- Wie kann ich in Python mit Bildern arbeiten?
- Dinge, die man möglicherweise mit Bildern tun möchte:
  - Sie ausgeben (anschauen)
  - Das Histogramm (Helligkeitsverteilung) anschauen (und vielleicht sogar verändern)?
  - Sie filtern
  - Die Fouriertransformation berechnen (Frequenzen)?
- Wie fängt man an, wenn man gar nichts darüber weiß?
  - Suchen und zu Beginn einfachste Beispiele ausprobieren
  - Beispiele modifizieren und komplizierter gestalten
  - Eigene Aufgaben aus der Praxis stellen und lösen

```
2   # -*- coding: utf-8 -*-
3   """
4   Created on
5
6   @author: alexanderwolodkin
7   """
8
9   from PIL import Image
10  img = Image.open("lena.png")
11  # print()
12  # print(img)
13  # r, g, b = img.split()
14  # print(r)
15
16  #testR = Image.merge("RGB",
17   #                   (g,b,r))
18
19  img = img.rotate(45, expand=True)
20  img.show()
21
```

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on

@author: alexanderwolodkin
"""

from PIL import Image
from PIL import ImageEnhance



img = Image.open("lena.png")
img.show()

enhancer = ImageEnhance.Contrast(im
enhancer.enhance(0.9).show()
```

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on

@author: alexanderwolodkin
"""

from PIL import Image
# from PIL import ImageEnhance
from PIL import ImageDraw

img = Image.open("lena.png")
# img.show()

test = ImageDraw.Draw(img)
test.ellipse((200,170,370,400),
             'yellow','black')

img.show()
```




#!/usr/bin/env python3
-*- coding: utf-8 -*-
"""
Created on
@author: alexanderwolodkin
"""
from PIL import Image
# from PIL import ImageEnhance
from PIL import ImageDraw
img = Image.open("lena.png")
width, height = img.size
for x in range(int(width/2)):
for y in range(height):
r, g, b = img.getpixel((x,
r, g, b = r, int(g*1.5), b
img.putpixel((x,y), (r, g,
for x in range(int(width/2), width)
for y in range(height):
r, g, b = img.getpixel((x,
r, g, b = r, g, int(b*1.5)
img.putpixel((x,y), (r, g,
img.show()
Suchen
ase -- a string containing
SCII uppercase letters
; containing all ASCII deci
octdigits -- a string
all ASCII punctuation
sidered printable
thon/temp.py', wdir=

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on

@author: alexanderwolodkin
"""

from PIL import Image
# from PIL import ImageEnhance
from PIL import ImageDraw

img = Image.open("lena.png")

width, height = img.size
for x in range(width):
    for y in range(height):
        r, g, b = img.getpixel((x,
        grey = int((r + g + b) / 3)
        img.putpixel((x,y), (grey,
img.show()
```

```
ase -- a string containing
SCII uppercase letters
g containing all ASCII deci
octdigits -- a string
all ASCII punctuation
sidered printable

base

/thon/temp.py', wdir=
Users/alexanderwolodkin/Documents/Python')
```

Zusammenfassend

- Neue Sachen ausprobieren kann langwierig sein:
  - Problemstellung → Nachschlagen
  - Minimalbeispiele → Fehler → Korrektur → wiederhole...

- Dokumentation lesen zu können ist wichtig, um Syntax und Funktion von einzelnen Befehlen zu verstehen!

- Mit kleinen Beispielen anfangen und von dort versuchen zu abstrahieren!