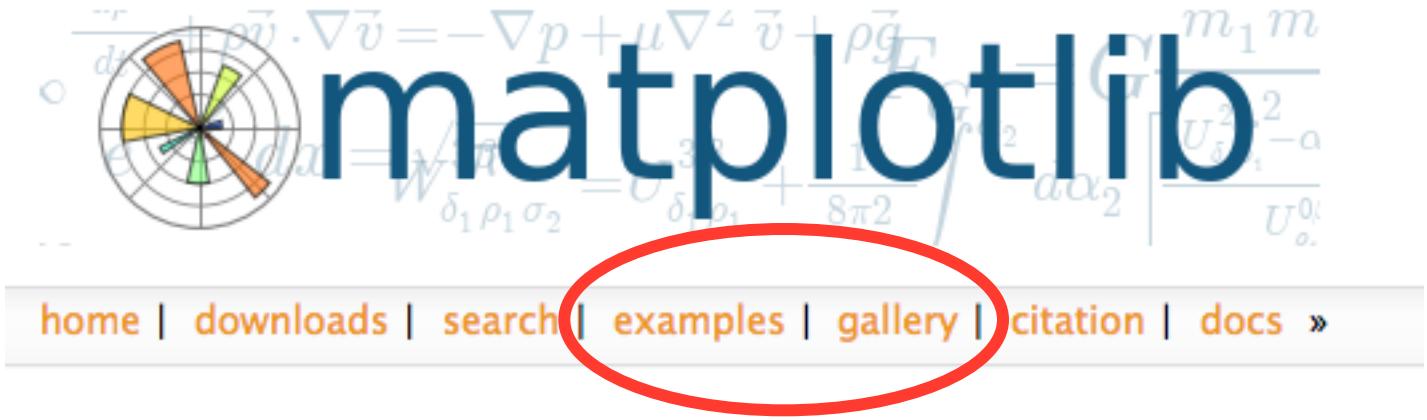
Technical University of Lodz
Institute of Electronics

Algorithms and Data Structures

9. Graphs, Images, Sounds

Łódź 2012

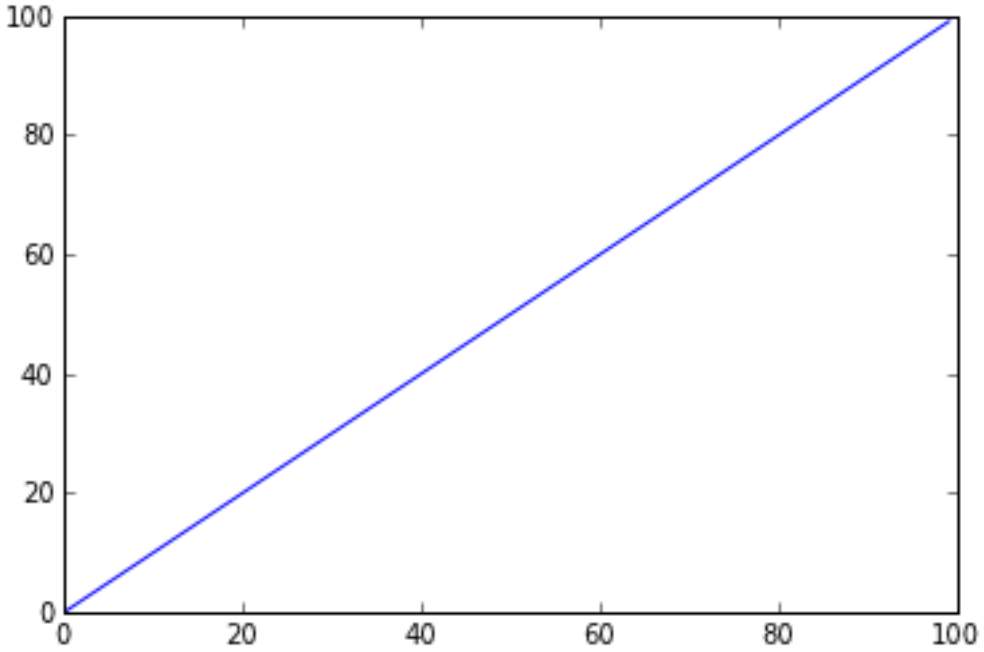
Graphs, Images



www.matplotlib.org

Matplotlib is a python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms.

Graphs



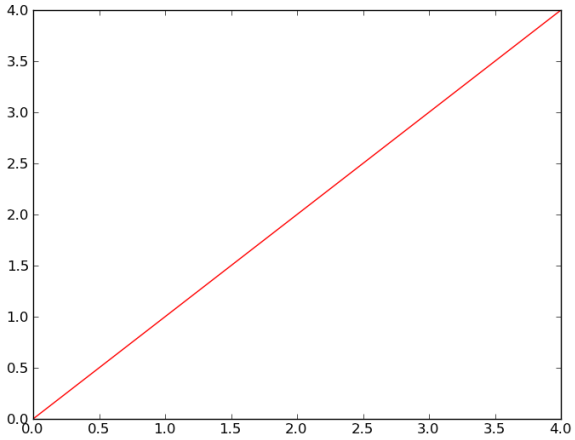
We use (import) two libraries: numpy and matplotlib

```
import numpy as np                  #arange(), sin(), cos(), sqrt(),...
import matplotlib.pyplot as plt     # figure(), plot(), subplot(), show(),…
```

Function y=x

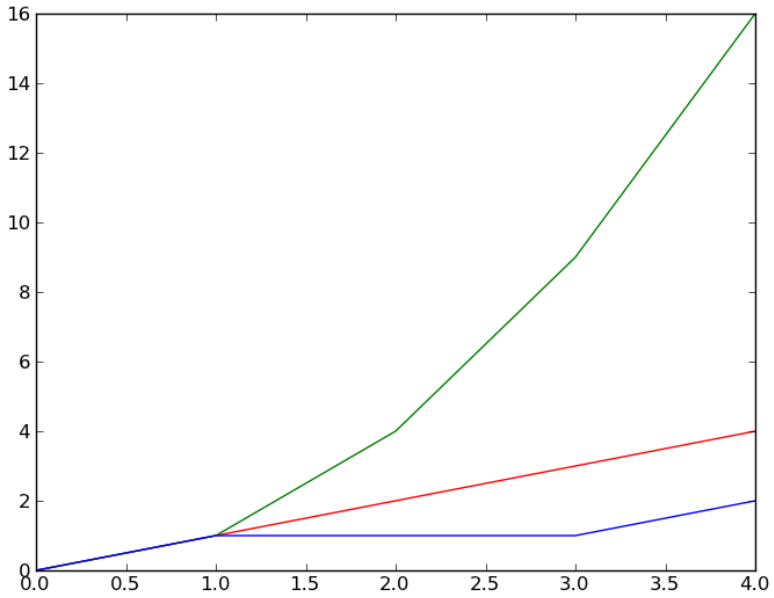
With the use of for loop (C, C++, Java)

```
import numpy as np                    #arange(), sin(), cos(), sqrt(),...
import matplotlib.pyplot as plt       #figure(), plot(), subplot(), show(),…


# define vectors for input values (x) and output values (y)
x = np.arange(5) #
y1 = np.zeros_like(x)

 for i in x:
     y1[i] = x[i]

plt.figure()
plt.plot(x,y1,'r')
plt.show() #one time, at the end of the script
```



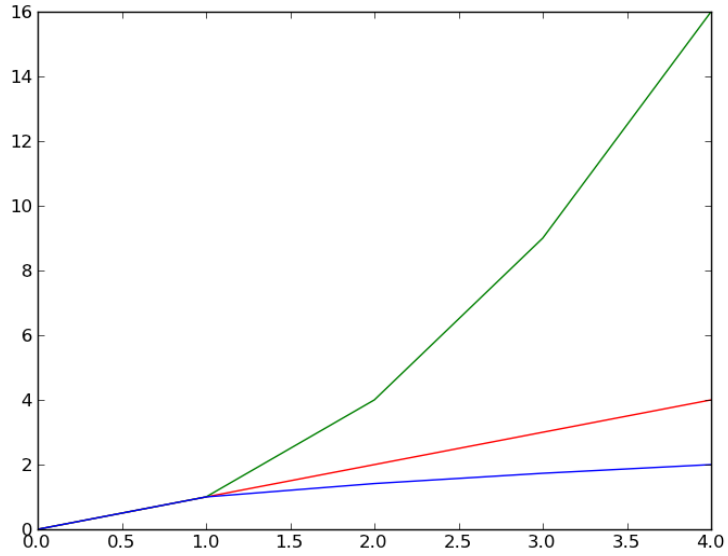
Functions y=x, $y=x^2$, $y=\sqrt{x}$

```
import numpy as np
import matplotlib.pyplot as plt

# define vectors for input values (x) and output values (y)
x = np.arange(5)
y1 = np.zeros_like(x)
y2 = np.zeros_like(y1)
y3 = np.zeros_like(y1)

for i in x:
        y1[i] = x[i]
        y2[i] = x[i]**2
        y3[i] = np.sqrt(x[i])

plt.figure()
plt.plot(x,y1,'r')
plt.plot(x,y2,'g')
plt.plot(x,y3,'b')
plt.show() #one time, at the end of the script
```



Functions y=x, $y=x^2$, $y=\sqrt{x}$

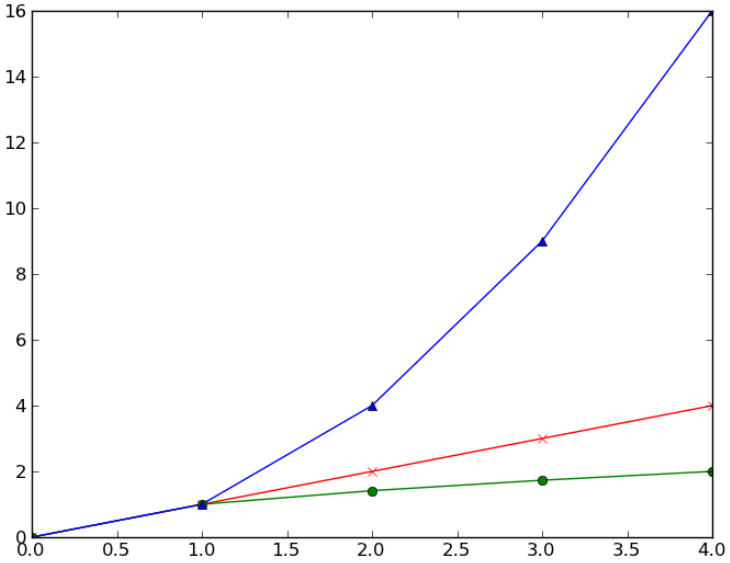
```
import numpy as np
import matplotlib.pyplot as plt

# define vectors for input values (x) and output values (y)
x = np.arange(5)
y1 = np.zeros_like(x)
y2 = np.zeros(x.shape, dtype=np.float)
y3 = np.zeros(x.shape, dtype=np.float)

for i in x:
        y1[i] = x[i]
        y2[i] = x[i]**2
        y3[i] = np.sqrt(x[i])

plt.figure()
plt.plot(x,y1,'r')
plt.plot(x,y2,'g')
plt.plot(x,y3,'b')
plt.show() #one time, at the end of the script
```



Functions y=x, $y=x^2$, $y=\sqrt{x}$

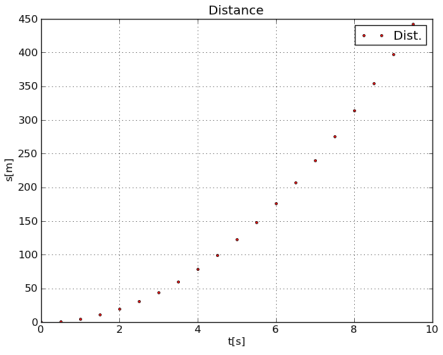
Matrix notation (Python, Matlab)

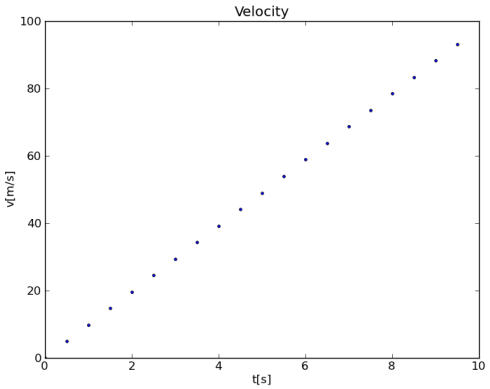
```
import numpy as np
import matplotlib.pyplot as plt

# define vectors for input values (x)
# and output values (y)
x = np.arange(5)

# Vectorization
y1 = x
y2 = np.sqrt(x)
y3 = x**2

plt.figure()
plt.plot(x,y1,'r', x,y2,'g', x, y3,'b') # draw lines
plt.plot(x,y1,'rx', x,y2,'go', x, y3,'b^') # draw markers
plt.show()
```



Graph description

```
t=np.arange(0,10,0.5)
a=9.81 # acceleration
v=a*t # velocity
s=(a*t**2)/2. # distance

plt.figure('Uniform acceleration')
# plot velocity in discrete time instances
plt.plot(t,v,'.')
plt.title('Velocity')
plt.xlabel('t[s]')
plt.ylabel('v[m/s]')

plt.figure("Figure with distance")
# plot distance in discrete time instances
plt.plot (t,s,'.r', label='Dist.')
plt.title('Distance'); plt.xlabel('t[s]'); plt.ylabel('s[m]')
plt.grid()
plt.legend()
plt.show()
```

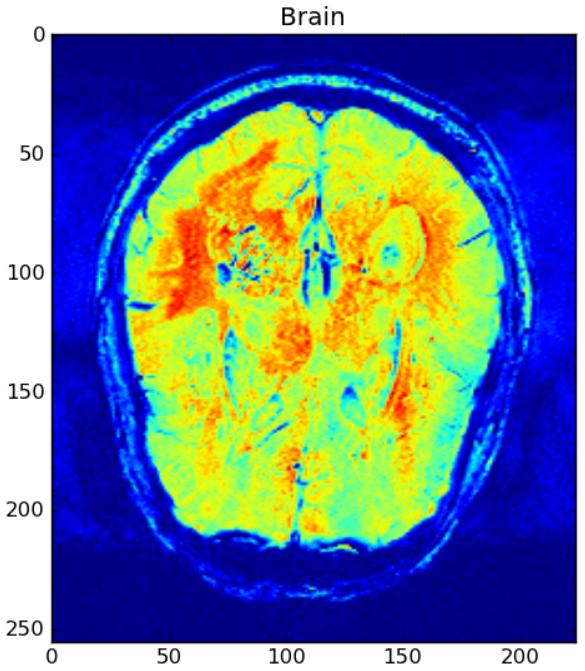




Images

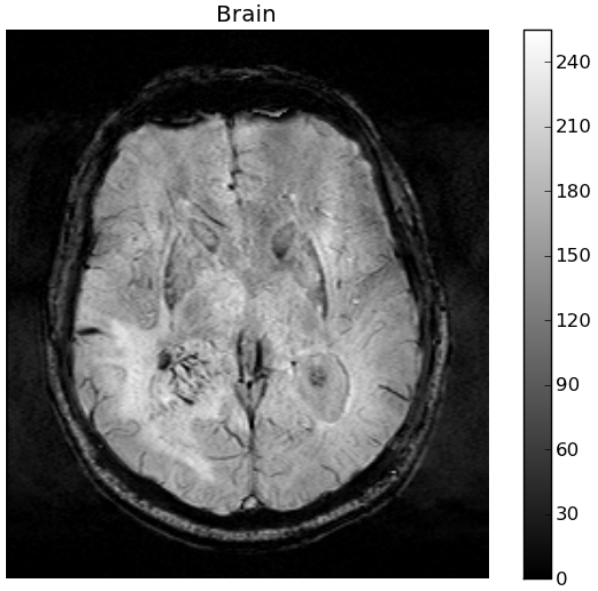
Load image from file and display it

```
import matplotlib.pyplot as plt

img1 = plt.imread(„brain1.bmp”)

plt.figure()
plt.imshow(img1)
plt.title(„Brain”)
plt.show()
```



Images

Colormap and origin changes, display colorbar and no axis




```
import matplotlib.pyplot as plt

img1 = plt.imread(„brain1.bmp”)

plt.figure()
plt.imshow(img1,cmap=plt.cm.gray,origin='lower')
plt.title(„Brain”)
plt.axis('off')
plt.colorbar()
plt.show()
```

Print additional information about loaded image:

```
print type(img1), img1.shape, img1.dtype, img1.min(), img1.max()
```

Images

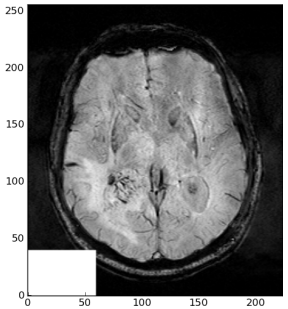
Print values of first 10 rows and 15 columns with the use of **for loop**

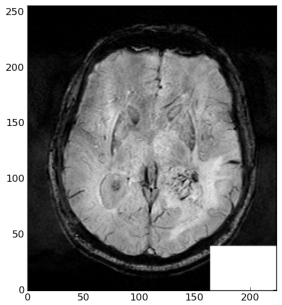
```
for row in range(10):
    for col in range(15):
        print img1[row,col],
    print
```

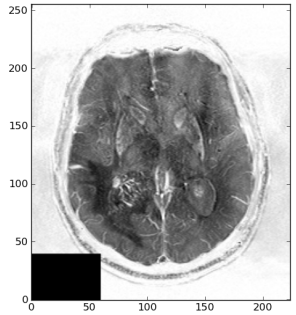
```
[0, 0, 0, 0, 0, 0, 2, 1, 0, 0, 0, 0, 0, 0, 0]
[0, 2, 0, 0, 2, 2, 2, 0, 0, 1, 1, 1, 0, 1, 0]
[0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0]
[0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1]
[0, 0, 0, 2, 2, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0]
[0, 1, 0, 0, 0, 0, 2, 1, 2, 2, 1, 0, 0, 0, 1]
[0, 2, 2, 0, 0, 0, 0, 0, 0, 3, 1, 3, 2, 0, 2]
[0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0]
[0, 1, 2, 2, 0, 1, 2, 0, 1, 0, 1, 2, 1, 0, 0]
[0, 1, 3, 2, 0, 0, 2, 3, 2, 1, 0, 0, 0, 0, 1]
```

TASKS (for loop):

1. Set the values of first 40 rows and 60 columns as 255. Display the image
2. Display mirrored image (in x direction)
3. Invert brightness of the image (255 – value of each pixel)







Images

The same tasks with the use of **matrix operation:**

print values of first 10 rows and 15 columns - Matrix style

```
print img1[:10,:15]
```

Set the values of first 10 rows and 15 columns as 255. Display the image

```
img1[:10,:15] = 255
```

Display mirrored image (in x direction)

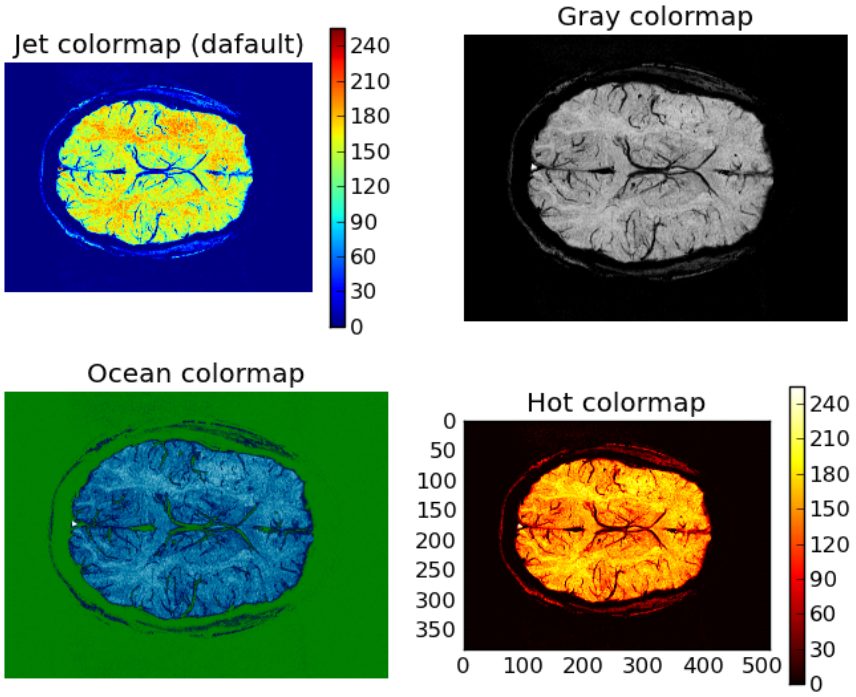
```
img1[:,::-1]
```

Invert brightness of the image

```
img1=255 - img1
```

Subplot

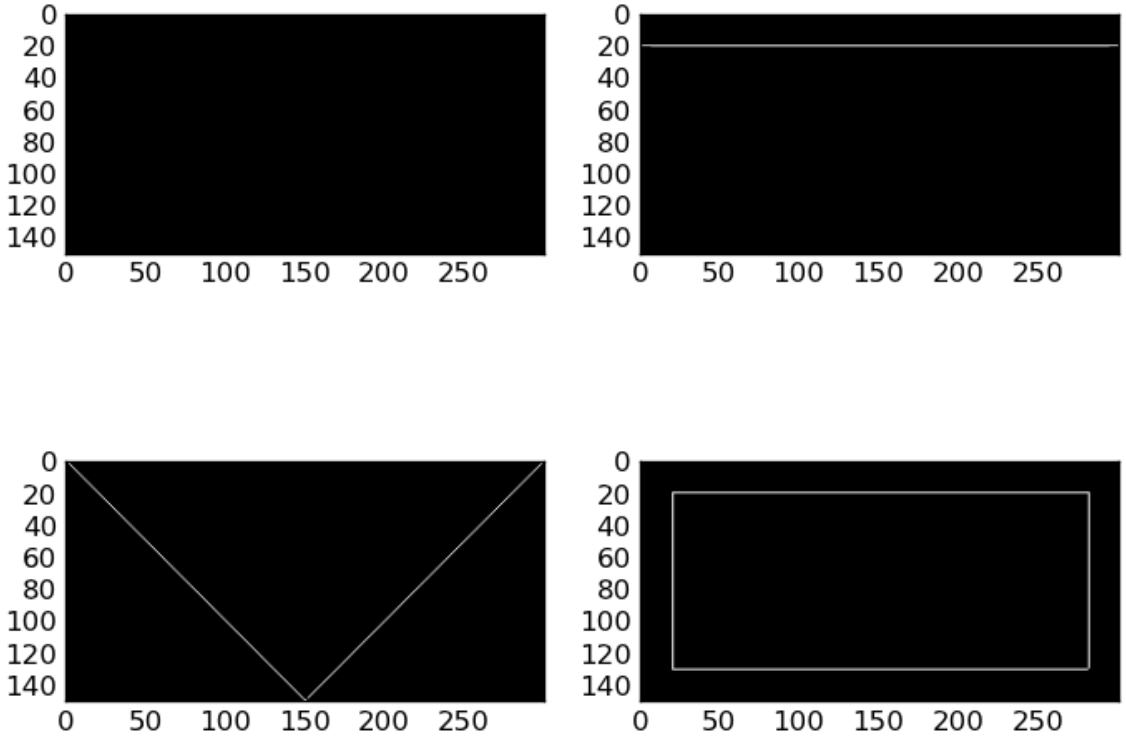
```
img1 = imread('brain.bmp')
plt.figure()
plt.subplot(2,2,1)
plt.imshow(img1)
plt.title('Jet colormap (dafault)')
plt.colorbar()

plt.subplot(2,2,2)
plt.imshow(img1,cmap='gray')
plt.title('Gray colormap')

plt.subplot(2,2,3)
plt.imshow(img1,cmap='ocean')
plt.title('Ocean colormap')

plt.subplot(2,2,4)
plt.imshow(img1, cmap='hot')
plt.title('Hot colormap')
plt.colorbar()
plt.show()
```



Images

TASKS (for loop and matrix style):

Create images as follows:



Sound

Playing wav file from Python in MS Windows 7 OS

```
import winsound
winsound.PlaySound("scale.wav", winsound.SND_ALIAS)
```

Sound

Loading files into Python

```
from scipy.io.wavfile import read as read_wav
sampling_rate, data = read_wav('scale.wav')
print sampling_rate
print data
```

Saving file to wav file format

```
from scipy.io.wavfile import write as write_wav
write_wav('inv_scale.wav',sampling_rate,data[::-1])
winsound.PlaySound("inv_scale.wav", winsound.SND_ALIAS)
```

Infinite Impuls Response (IIR) Filter

IIR systems have an impulse response function that is non-zero over an infinite length of time

```
from scipy.signal import lfilter

r = 0.7 # reverberation coefficient
N = 1000 # time delay in samples
a = np.zeros(N)
a[0] = 1
a[-1]=-r
b = [1]

from scipy.io.wavfile import read as read_wav
sampling_rate, data = read_wav('male_voice.wav')

y1= lfilter(b,a,data)
y1= y1.astype('int16')
y1 -= y1.min()
y1 *=60

from scipy.io.wavfile import write as write_wav
write_wav('male_voice_filtered.wav',sampling_rate,y1)

# Play saved wave files
winsound.PlaySound("male_voice.wav", winsound.SND_ALIAS)
winsound.PlaySound("male_voice_filtered.wav", winsound.SND_ALIAS)
```

Sound

Remove every second value (sample) from vector data

```
sampling_rate, data = read_wav('male_voice.wav')
data1 = data[::2]
write_wav('male_voice_1.wav',sampling_rate,data1)
```

```
# Play saved wave files
winsound.PlaySound("male_voice.wav", winsound.SND_ALIAS)
winsound.PlaySound("male_voice_1.wav", winsound.SND_ALIAS)
```

Sound

Double each probe in the vector data

```
sampling_rate, data = read_wav('male_voice.wav')
print data.shape, data.dtype

data2 = np.zeros((data.shape[0]*2),dtype=np.uint8)
print data2.shape, data2.dtype

data2[::2]=data
data2[1::2]=data
write_wav('male_voice_2.wav',sampling_rate,data2)



# Play saved wave files
winsound.PlaySound("male_voice.wav", winsound.SND_ALIAS)
winsound.PlaySound("male_voice_2.wav", winsound.SND_ALIAS)
```

Sound

Increase values of data (multiply by 2)

```
sampling_rate, data = read_wav('male_voice.wav')
print data.dtype, data.max()


data3=data*2
print data3.dtype, data3.max()
write_wav('male_voice_3.wav',sampling_rate,data3)



# Play saved wave files
winsound.PlaySound("male_voice.wav", winsound.SND_ALIAS)
winsound.PlaySound("male_voice_3.wav", winsound.SND_ALIAS)
```

Sound

Changing the sampling rate

```
sampling_rate, data = read_wav('male_voice.wav')
print sampling_rate

write_wav('male_voice_5.wav',sampling_rate/2,data)
write_wav('male_voice_6.wav',sampling_rate*2,data)
```

```
# Play saved wave files
winsound.PlaySound("male_voice.wav", winsound.SND_ALIAS)
winsound.PlaySound("male_voice_5.wav", winsound.SND_ALIAS)
winsound.PlaySound("male_voice_6.wav", winsound.SND_ALIAS)
```

Literature

Brian Heinold, Introduction to Programming Using Python, Mount St. Mary's University, 2012 (*http://faculty.msmary.edu/heinold/python.html*).

Brad Dayley, Python Phrasebook: Essential Code and Commands, SAMS Publishing, 2007 (dostępne też tłumaczenie: B. Dayley, Python. Rozmówki, Helion, 2007).

Mark J. Johnson, A Concise Introduction to Programming in Python, CRC Press, 2012.

Paul Barry & David Griffiths, Head First Programming, O'REILLY Media Inc, 2009