

Inleiding tot Python

Brecht Baeten¹

¹KU Leuven, Technologie campus Diepenbeek,
e-mail: brecht.baeten@kuleuven.be

16 september 2016

Wat is Python?



- Progameertaal
- Zeer object georienteerd
- Packages voor wetenschappelijke toepassingen
- Gratis
- Verschillende GUI's beschikbaar, niet meegeleverd
- Z  r verscheiden toepassingsgebied (Wetenschappelijke berekeningen, Web toepassingen, GUIs, quick scripting)

Mac

- 2.7 standaard meegeleverd bij Mac OS X

Linux

- 2.7 standaard meegeleverd bij veel distributies

Windows

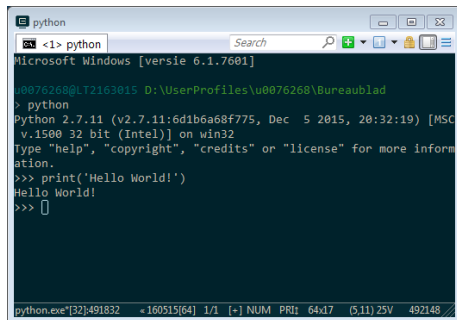
- Download een distributie (best 2.7 of 3.5) via <https://www.python.org/downloads/>
- Voeg Python toe aan het windows pad
- Download een degelijke text editor, bv. Notepad ++ <https://notepad-plus-plus.org/download/>
- Download een degelijke console, vb. ConEmu <http://sourceforge.net/projects/conemu/files/latest/download>

of

- Anaconda <https://www.continuum.io/downloads>
- WinPython <https://winpython.github.io/>
- Python(x,y) <http://python-xy.github.io/>

Interactieve console

- Open een console en typ "python"
- Commando's uitvoeren
- variabelen definiëren
- functies aanroepen
- Goed voor korte tests
- Sluiten met Ctrl + Z



```
python
C:\> python
Microsoft Windows [versie 6.1.7601]
u0076268@LT2163015 D:\UserProfiles\u0076268\Bureaublad
> python
Python 2.7.11 (v2.7.11:6d1b6a68f775, Dec 5 2015, 20:32:19) [MSC
v.1500 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more inform
ation.
>>> print('Hello World!')
Hello World!
>>> []
```

```
>>> print('Hello World!')
>>> x = 5
>>> x
>>> y = 4*x+x**2
>>> y
```

Scripts

- Opeenvolgende commando's opgeslagen in een .py bestand
- Aanroepen vanuit het command prompt
- Bestanden in de map waarin je python start zijn beschikbaar
- Met "-i" als argument start een interactieve sessie na het uitvoeren van het script

```
> python -i hello_world.py
```

Variablen

- Int
- Float
- List
- String
- Dictionary

```
>>> A = 1
>>> B = 3.572
>>> C = 'string'
>>> D = [1,2,3]
>>> D[0]
>>> D[-1]

>>> E = [1,'test',4,D]

>>> F = {'value':1, 3:'test', 'spam':'eggs'}
>>> F['spam']
```

Controle structuren

- for in :
- if : else:
- ...
- ":" en inspringen verplicht!

```
for i in range(10):  
    if i%2 == 0:  
        print('{} is even'.format(i))  
    else:  
        print('{} is oneven'.format(i))
```


Functies

- Groeperen van vaak gebruikte commando's
- Definieren voor aanroepen
- Documentatie, op te roepen via "**help**(digits2number)"

```
def digits2number(A,B,C,D):  
    """  
    returns a a number as if the arguments were  
        different digits in the number  
  
    Parameters:  
    A: float, hundreds  
    B: float, decades  
    C: float, units  
    D: float, tenths  
    """  
  
    return 100*A+10*B+C+0.1*D
```

Functies

- Functies kunnen in verschillende files (modules) worden gedefinieerd
- Functie gebruiken in een ander script kan met een "import" statement
- geïmporteerde modules hebben een eigen "namespace"

```
import functies
```

```
val = functies.digits2number(2,4,1,9)  
print(val)
```

of

```
from functies import *
```

```
val = digits2number(2,4,1,9)  
print(val)
```

Packages

- Folders kunnen beschouwd worden als packages door een bestand "`__init__.py`" toe te voegen
- Modules in een folder kunnen dan geïmporteerd worden
- Code in "`__init__.py`" wordt eerst uitgevoerd (vb submodules importeren)

```
.  
|-- lib  
    |-- __init__.py  
    |-- functies.py
```

```
>>> import lib.functies
```

Externe packages

- Numpy - Algebra
- Matplotlib - plots
- Scipy - Algemene wetenschappelijke functies

Installatie

- Voeg het "pip" pad ("`pythonfolder/Scripts`") toe aan het windows pad
- Installatie van python packages via "`pip install numpy`"

Of

- Distributie downloaden via de project website en installeren

```
>>> import numpy as np
>>> a = np.zeros(10)
```

Werken met numpy

- prealloceren met "np.zeros", "np.linspace",...
- "len()", "Array.shape"
- indexeren met "[]", ":", vanaf 0 tot -1

```
x = np.zeros(10)
for i in range(len(x)):
    x[i] = 5*i-2

z = np.linspace(4,8,20)

y = np.zeros( (10,4) )
for i in range(y.shape[0]):
    for j in range(y.shape[1]):
        y[i,j] = 4*i+3*j-2

y[4,1:]
```

Plotten

- "figure", "plot", "xlabel", "legend"

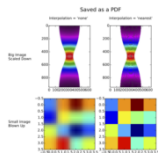
```
x = np.linspace(0, 2 * np.pi, 50)
s = np.sin(x)
c = scipy.integrate.cumtrapz(s, x)

plt.rc('text', usetex=True)
plt.rc('font', family='serif', size=8)
plt.rc('figure', autolayout=True)

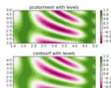
plt.figure(figsize=(10/2.54, 7/2.54))
plt.plot(x, s, '-s', label=r'sinus')
plt.plot(x[1:], c, label=r'$\int$ sinus$(x)$ d$x$')
plt.xlabel(r'$x$ (rad)')
plt.ylabel(r'$y$')
plt.legend(numpoints=1)

plt.savefig('sinus_cosinus.pdf')
plt.savefig('sinus_cosinus.png')
plt.show()
```

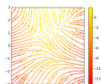
Matplotlib gallery (<http://matplotlib.org/gallery.html>)



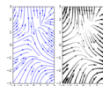
interpolation_none_vs_nearest



pcolormesh_levels



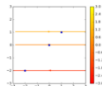
streamplot_demo_features



streamplot_demo_features



streamplot_demo_masking



streamplot_demo_start_points

Pie and polar charts



pie_demo_features



pie_demo_features



polar_bar_demo



polar_scatter_demo

Bestanden inlezen

- ascii: via file handler "with **open**('file.txt','r') as f:"
- Volledige file in één keer als string "data = f.read()"
- Lijn per lijn "for line in f:"
- "\t", "\n"
- Speciale packages voor speciale formaten (csv, openpyxl,...)

```
import csv

coord = []
with open('data.csv') as f:
    reader = csv.DictReader(f, delimiter='\t')
    for row in reader:
        coord.append( (row['x'],row['y'],row['y'],) )

for c in coord:
    print( c )
```


Een project structureren

- Gebruik functies
- Maak binnen functies gebruik van sub-functies indien nuttig
- Geef functies een betekenisvolle naam
- Groepeer functies die bij elkaar horen in een map en voeg deze map toe aan het Matlab pad
- Don't Repeat Yourself (DRY)
- Gebruik betekenisvolle namen voor variabelen
- Documenteer alles

Een project structureren

Voorbeeld folderstructuur:

```
mijnProject
|-- data
|   |-- mijndata.csv
|
|-- lib
|   |-- __init__.py
|   |
|   |-- data
|       |-- __init__.py
|       |-- data_inlezen.py
|       |-- data_naar_coordinaten.py
|       |
|   |-- plot
|       |-- __init__.py
|       |-- plot_coordinaten.m
|
|-- main.py
|-- readme
```

Een project structureren

Voorbeeld main.py:

```
# main.py
# dit script leest data in, vertaalt deze in coördinaten
# en maakt een plot

# functies importeren
import lib

# data inlezen en bewerken
data = lib.data.data_inlezen('data/mijndata.csv')
[x,y,z] = lib.data.data_naar_coördinaten(data)

# plotten
lib.plot.plot_coördinaten(x,y,z)
```

© ⓘ 2016 Brecht Baeten

Dit werk is gelicenseerd onder de licentie Creative Commons Naamsvermelding-GelijkDelen 4.0 Internationaal. Ga naar <http://creativecommons.org/licenses/by-sa/4.0/> om een kopie van de licentie te kunnen lezen.

De bron van dit document en alle tekeningen zijn beschikbaar op <https://github.com/BrechtBa/inleiding-tot-matlab>