Daten im Textformat in Numpy Arrays wandeln und plotten

jean-claude.feltes@education.lu

Problemstellung

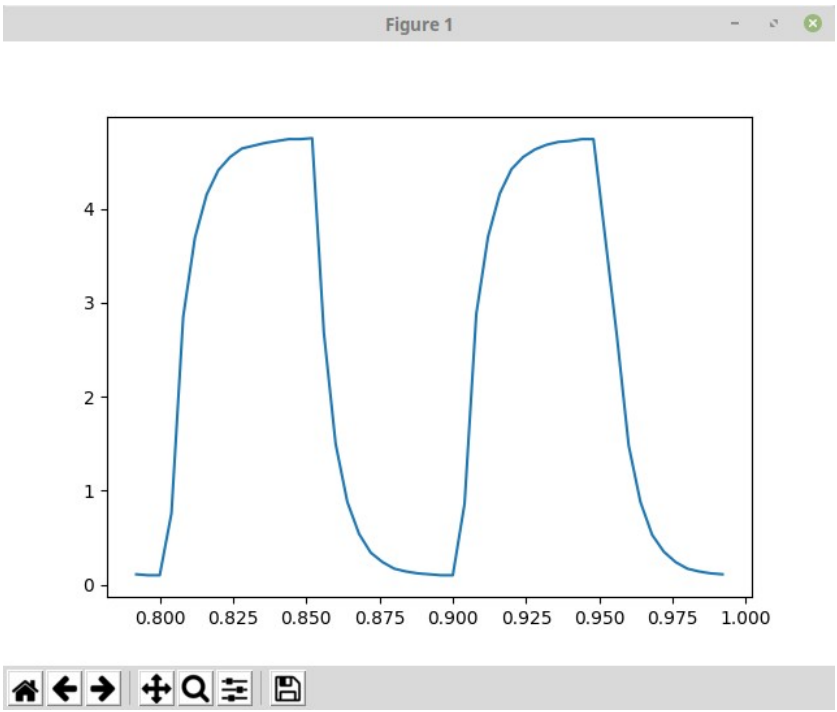
Es liegen z.B. Messwerte in einem tabellarischen Textformat vor, bei dem jede Spalte eine Serie von Werten eines Sensors darstellt:

```
s="""
0.792   0.01    0.11    0.23    0.20
0.796   0.01    0.10    0.20    0.20
0.800   0.01    0.10    0.20    0.20
...
0.988   0.01    0.12    0.23    0.23
0.992   0.01    0.11    0.23    0.20
"""
```

(die meisten Zeilen wurden der Übersicht halber weggelassen)

Als Trennungszeichen zwischen den Spalten dient ein Tabulator „\t“ oder ein oder mehrere Whitespaces.

Diese String-Daten sollen in ein Numpy-Array umgewandelt und geplottet werden, z.B. die dritte Spalte als Funktion der ersten:



Achtung: die Spalten werden, wie üblich in Python, beginnend mit 0 indexiert, so dass z.B. die dritte Spalte den Index 2 hat.

Datenumwandlung mit numpy.loadtxt

Numpy bietet die Funktionen loadtxt und genfromtxt

Beispiel:

```
import numpy as np
from io import StringIO

testdata=StringIO(s)
data=np.loadtxt(testdata)

x=data[:,0]
y=data[:,2]

from matplotlib import pyplot as plt
plt.plot(x,y)
plt.show()
```

Im Beispiel liegen die Daten als String vor. Da die Funktion loadtxt() ein Datei-Objekt erwartet, wird der String mit StringIO() in ein dateiartiges Objekt umgewandelt.

Wenn die Daten in einer Datei stehen, kann die loadtxt-Funktion direkt auf diese angewendet werden.

Die loadtxt-funktion kann auch mit **Kommentaren** umgehen, dafür wird defaultmässig das **„#“**-Zeichen angenommen (dies kann aber geändert werden).

Auch **fehlende Datenzeilen sind kein Problem**.

Die Anzahl der Spalten muss allerdings überall im String die gleiche sein.

Weitere Infos:

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

Mit zusätzlichen Parametern können andere Trennzeichen für die Spalten, andere Kommentarzeichen usw. festgelegt werden, und es können Zeilen zu Beginn übersprungen werden, wenn es eine Titelzeile gibt.

Weiter kann die Option **usecols = (0,2)** interessant sein, wenn nur die Spalten 0 und 2 benutzt werden sollen. Zeitlich scheint diese Option keinen Vorteil zu bieten, aber sie ignoriert alles was rechts von der Spalte 2 steht. Dies könnte bei inhomogenen Strings mit einer variablen Spaltenzahl interessant sein:

```
def plot2(s, xcol, ycol):

    testdata=StringIO(s)
    data=np.loadtxt(testdata, usecols=(xcol,ycol))

    x=data[:,0]
    y=data[:,1]

    plt.plot(x,y)
```

```
        plt.show()

    plot2(s,0,2)
```

Mit dieser Methode wird „Datenmüll“ rechts von Spalte 2 ignoriert.

Datenumwandlung mit numpy.genfromtxt

Die Funktionen genfromtxt kann defaultmässig genau wie loadtxt angewendet werden.

Interessant ist sie vor allem bei heterogenen Daten, wie zum Beispiel:

```
    s="""
    0.792   0.01    0.11    0.23    0.20
    0.924   5.08    4.55    4.43    4.32
    0.93
    0.932   5.09    4.68    4.61    4.49
    ...
    0.954   5
    0.956   0.01    2.64    3.37    3.58
    ...
    0.992   0.01    0.11    0.23    0.20
    """
```

Dies würde bei der loadtxt-Funktion Fehler verursachen (bei genfromtxt mit den Default-Einstellungen ebenfalls), aber mit

```
    data = np.genfromtxt(testdata, invalid_raise=False)
```

kann der Datenstring korrekt umgewandelt werden.

Dabei werden Warnungen auf der Konsole ausgegeben:

```
test_numpy_loadtxt.py:102: ConversionWarning: Some errors were detected !
    Line #37 (got 1 columns instead of 5)
    Line #44 (got 2 columns instead of 5)
  data = np.genfromtxt(testdata, invalid_raise=False)
```

Dem Array fehlen einfach die Werte, die zu den entsprechenden Zeilen gehören.
Es kann aber normal verarbeitet werden.

Weitere Informationen finden sich hier:

https://docs.scipy.org/doc/numpy-1.13.0/user/basics.io.genfromtxt.html#defining-the-input

Fazit

Mit der **numpy.genfromtxt** kann ein Datenstring (oder Daten in einer Datei) problemlos umgewandelt werden. Bei inhomogenen Daten ist diese Funktion der numpy.loadtxt – Funktion vorzuziehen, wenn die Daten mit Sicherheit homogen sind, kann man auch die loadtxt – Funktion anwenden, die laut Dokumentation einen Geschwindigkeitsvorteil bieten soll (bei kleinen Datenmengen ist dieser aber irrelevant).