宇宙科学実習 I：python で遊ぼう (1)

今回は Ubuntu 環境で作業をしてください。

天文学のデータ解析等の分野で python は *de facto standard* な言語となりつつあります．今回は天文学者たちが開発した python パッケージである astropy と，グラフを描くパッケージ matplotlib で遊んでみましょう．

1 簡単な数値計算をしよう

1.1 astropy を読み込む

ここでは，天文学に現れる定数のモジュールを読み込んで簡単な計算をしてみましょう．端末から python を起動し，次のように入力します．

```
>>> from astropy import constants as const
```

これは,「astropy パッケージの constants というモジュールを，const という略称 (これは自分の好きな名前で構わない) で読み込みなさい」，という命令です．constants では，天文学に頻出の様々な定数の定義がなされています．このように python では様々なパッケージ，モジュールが開発&配布されていて，これらを読み込むことで楽に色々な仕事がこなせます．

早速どのような定数が使えるか見てみます．const.pc と入力してみます．

```
>>> const.pc
<<class 'astropy.constants.iau2015.IAU2015'> name='Parsec' value=3.0856775814671916e+16 uncertainty=
```

これは，距離の単位パーセクの定数です．const の部分でモジュールを，pc で特定の定数（ここではパーセク）を指定しています．このように，astropy.constants の定数は単なる数値ではなくいくつかの属性からなる "オブジェクト "であることがわかります．諸々の計算のために，単なる数値の部分だけ抜き出すと

```
>>> const.pc.value
3.0856775814671916e+16
```

のように，value の指定をします．標準ではこの値は SI 単位系での値となっていますが，cgs 単位系での値を使いたい場合は cgs を指定します．

```
>>> const.pc.cgs
<Quantity 3.08567758e+18 cm>
>>> const.pc.cgs.value
3.085677581467192e+18
```

どのような定数が定義されているかは，astropy.constants のドキュメントは http://docs.astropy.org/en/stable/const を参照してください．

1.2 ブラックホールシャドウの大きさ

第二回の問題を計算してみましょう．

例題

日米欧の共同研究グループが，史上初めてブラックホールの影（black hole shadow) の撮像に成功したという発表が 4/10 にありました．リングのみかけの直径は 42μas とすると (μ は 10^{-6}: マイクロ秒角)，影の実際の大きさは冥王星の軌道の大きさ（軌道長半径 40AU）を超えることを確認しなさい．ただし，M87 までの距離は 16.8Mpc とします（M(メガ)=10^6）．

この例題では，M87 までの距離に 21μas のタンジェントをかけて，1AU との比を取ることになります．このとき，角度の変換（角度秒→ラジアン）とタンジェントの計算には別の機能拡張モジュール「numpy」を読み込んで使うのが便利でしょう.[1] numpy を別名 np で読み込むとすると，円周率 π は np.pi，タンジェントは np.tan で呼び出せます．

```
>>> import numpy as np
>>> 16.8e6 * const.pc.value * np.tan(np.pi/180 * 21e-6/3600) / const.au.value
352.7999999972359
```

問：Sgr A*シャドウのみかけの大きさ

我々の銀河（天の川銀河）の中心には「いて座 A*(Sgr A *(star))」と呼ばれる強い電波を発する天体があります．これは質量が太陽の 400 万倍ほどもある巨大なブラックホールだと考えられています．Event Horizon Telescope はこの天体も観測しています．
質量 M のブラックホールシャドウのみかけの半径 α（角度:ラジアン）は次の近似式で表されます.[a]

$$\alpha = \frac{3\sqrt{3}}{2}\left(\frac{D}{\frac{2GM}{c^2}}\right)^{-1}. \tag{1}$$

ただし D はブラックホールまでの距離です．いま，Sgr A*までの距離 D が 8.5kpc，Sgr A*ブラックホールの質量 M を $4\times10^6 M_\odot$ としたとき，シャドウの大きさは何 μas になるか計算しなさい．ただし，このセクションでやったように，astropy の constants を const でインポートした場合，重力定数 G, 真空中の光の速さ c，太陽質量 $M_\odot$ の値はそれぞれ const.G.value, const.c.value，const.M_sun.value と書けます．

[a] Synge J.L., *Mon. Not. Roy. Astron. Soc.*, **131**, 463 (1966)

2 matplotlib でグラフを描こう

つぎに，python のグラフィカルな開発環境である spyder をターミナルから立ち上げてください．第一回目で試したように，左のウィンドウに python のプログラム（スクリプト）を書いて実行できます．

[1] numpy は数値計算を行うためのライブラリです．ベクトル，行列計算や様々な数学的関数が定義されています．

ここでは，第一回目で描いた月平均太陽黒点数の変化をプロットしましょう．まず，ターミナル上で，no1 ディレクトリにあるデータファイル SN_m_tot_V2.0.csv を現在のディレクトリにコピーしておきます．

```
$ cp ../no1/SN_m_tot_V2.0.csv .
```

python にはいくつかのグラフ描画用ライブラリが存在しますが，matplotlib は比較的使われる頻度が高いものの一つです．簡単なデータプロットには，matplotlib の pyplot というモジュールをインポートして使いましょう．

```
from matplotlib import pyplot as plt
```

ここでは pyplot を plt という略称でインポートしています．

pyplot の２次元プロットでは，x 軸と y 軸に対応する数の配列（リスト）を指定して，対応する成分を座標とする点を座標平面にプロットします．次の例を spyder に入力して実行してみなさい (注意：各行とも１列目に空白を開けずに書くこと).[2]

```
from matplotlib import pyplot as plt
xlist = [0.0, 0.1, 0.5, 0.7, 1.0]
ylist = [1.0, 0.8, 0.5, 0.6, 0.9]
plt.scatter(xlist,ylist)
plt.show()
```

データ数５点の散布図が得られました．ここで，

```
xlist = [0.0, 0.1, 0.5, 0.7, 1.0]
ylist = [1.0, 0.8, 0.5, 0.6, 0.9]
```

の２行は，データ点の x, y 座標を与えるリスト xlist]，ylist を定義しています．plt.scatter(xlist,ylist) は xlist を横軸に，ylist を縦軸にして散布図を描く命令です（しかしこの段階では画面に絵を描いていない）．plt.show() は，画面に絵を出力しろ，という命令です

また，点を結んで曲線を描きたい場合は scatter の代わりに plot を使います．

```
plt.plot(xlist, ylist)
```

軸のラベルは

```
plt.xlabel('X axis')
plt.ylabel('Y axis')
```

などと，軸ラベルの文字列を引用符でくくって指定します．

問

上の例で，scatter を plot に変えて実行してみなさい．また，点データと曲線を両方描くにはどうしたらいいでしょうか．実際にやってみなさい．

[2]実行初回は保存するファイルの名前を尋ねてきますので，場所は no4 ディレクトリを指定して，testplot.py などと好きな名前で保存してください．

2.1 CSV ファイルのデータをプロットする

SN_m_tot_V2.0.csv は，セミコロンで列を区切られた CSV 形式になっています．この 3 列目（年），4 列目（黒点数）のデータを読み込んでプロットをしましょう．

CSV ファイルのプロット

```
#####
from matplotlib import pyplot as plt
#####

# reading data file and create list
datrows = []
exfile = open('SN_m_tot_V2.0.csv', 'r')
line = exfile.readline()
while line:
       ltmp = line.rstrip()
       datrows.append(ltmp.split(';'))
       line = exfile.readline()
exfile.close

## creating 1D lists of x and y
xlist = []
ylist = []
##
for irow in datrows:
    xlist.append(float(irow[2]))
    ylist.append(float(irow[3]))

## print the combination of x & y
plt.plot(xlist,ylist)
plt.xlabel('year')
plt.ylabel('Number of sunspots')
plt.show()
```

（例題コードの解説）
まず，#で始まる行はコメントなので，計算機は読み飛ばします．

６行目では，datarows という ”空（から）”のリストを定義しておきます．この空のリストにあとから要素を付け加えていくことにします．

７列目は，データファイルを open 関数で開きます．’r’ は，ファイルを読み取り（read only）モードで開く指定です．この開いたファイルをプログラム内では exifile という名前のオブジェクトとして扱います．

８列目は exfile から１行読み込みます．（ ）の中には読み込みオプションが指定できますが，いまは何も指定しなくて良いです．読み込んだ行は line というオブジェクトに格納します．

９列目から１２列目で，ファイルの内容を１行１行読んで記憶していきます．ここでは while という構文を使っています．これは，条件が満たされる間は繰り返すという指定の構文です．ここのように，「while line :」などとすると，line が存在する限り（読み込んだ行が存在している限り）繰り返しを続けます．「:」を忘れないでください．また，while やすぐ下の for のような構文で，繰り返す命令はタブ（tab）キーで１段下げて書くのが python の文法です．

１０列目では，行のオブジェクト line から改行記号を取り去って，テンポラリーなオブジェクト ltmp を作っています．これは，python ではファイルの各行の最後につく改行記号も一緒に読み込んでしまうためです．これはデータとして必要ないので rstrip で取り去ってしまいます．

> python では，各オブジェクトにメソッドという関数が付随していて，それを指定して様々な作業をおこなうことができます．オブジェクト line の末尾から改行記号を取り去るには，オブジェクト名に続けて.rstrip() と書きます．この結果を新たなオブジェクト変数 ltmp に仮に代入しています．

１１列目では，ltmp オブジェクトで，括弧内にしていした区切り文字を取り去るメソッド split を適用し（ltmp.split(’;’)），その結果を予め用意しておいた datrows というリスト（箱のようなもの）に入れていきます．このとき，リストオブジェクト datrows のメソッド append を使っています．

１２列目は，データファイルからもう一列読み込んで（第８列参照）繰り返しに戻ります．

１３列目はファイルを閉じています．開いたファイルは，使い終わったら close メソッドで閉じておきましょう．

１６，１７列は別の空のリスト xlist，ylist を x，y 座標を格納する配列として用意しておきます．

１９，２０，２１列は上で生成した配列 datrows から，対応する列を抜き出して xlist，ylist に付け加えていきます（append メソッド），ここでは，２次元配列 datrows の１行１行を読んで，必要な列の要素を付け加えるために，for 構文を使っています．「for i in datrows: 」と書くと，datrows から１行抜き出して irow とし，それが尽きるまで繰り返します．

２０列では，まず irow の第２列目の文字（python では，ファイルから読み込んだものは文字として扱われます（「0.75」は文字「0」「.」「7」「5」の列と認識）） irow[2] を float 関数で数値に変換します．その結果をリストオブジェクト xlist の append メソッドによってリストに加えていきます．ここで注意したいのは，python では配列の番号は 0 から始まるということです．

2.2 python での関数定義

python で自作の関数を定義するには次のように行います．

```
def Teff(D,A):
    import numpy as np
    from astropy import constants as const
    return (const.L_sun.value * (1-A)/(16 * const.sigma_sb.value \
        * np.pi * (D* const.au.value)**2))**0.25
```

最初の行は関数 Teff を定義する def コマンドで，引数 D，A を書いておきます．以下 tab で段下げをして関数の定義をしていきます．まず２，３行目では関数内で numpy と astropy のモジュールを使うので，その宣言をしておきます．４行目の return 以降の値が関数値として返されます．なお，４行目最後の「\」は，長い行を折り返すためにつけています（つまり，５行目は４行目末から続いている）．

課題：内惑星の放射平衡温度のプロット

データファイル planets.dat は，１列目に惑星名，２列目に AU 単位の公転軌道半径，３列目にアルベド，４列目に表面温度の観測値の平均（K）を書いた CSV ファイルです．これを読み込み，pyplot の scatter 機能を使って，横軸に距離（AU）縦軸に温度のグラフを描きなさい．また，各惑星の放射平衡温度を重ねてプロットしてみなさい．