```
# fichier créé avec python 3.9.1

##### AVERTISSEMENT #####
# dans python, la protection des types est encore très primitive...
# je décline toute responsabilité associée à ce défaut regrettable !
#########################

import numpy as np

import MinimiLib_18 as mn # qui fait lui même appel à myFCN

# pour le calcul du chi2, les points expérimentaux sont dans myFCN

# noms des paramètres (utilisé dans les résultats pour mieux les reconnaitre)
# parN devrait être array(range(30), str)...
# ...mais la gestion simpliste des types par python permet de simplifier
parN = ["p","e","th0"]

# valeurs des paramètres
# par devrait être array(range(30), float) mais on simplifie de même
par = [2., 0.9, 3.]

# estimation des pas d'optimisation ;
# (si zéro : le paramètre est traité comme une constante et n'est pas optimisé)
# dPar devrait être array(range(30), float) mais on simplifie de même
dPar = [0.1, 0.1, 0.1]

nnPar = 3
# il faut indiquer sérieusement le nombre de paramètres utilisés (les suivants
# sont ignorés)... d'autant plus qu'on a simplifié !

stepp = 1.0
epsii = 0.0001
impp = 0 # on fait comme on veut, mais on laisse ainsi minimi choisir au mieux
errorss = True
minList = mn.minimi(nnPar,par,parN,dPar,stepp,epsii,impp,errorss)
print(minList)

the_p = 3.395417943915725 # on récupère les valeurs ajustées
the_e = 0.8380131544868554
the_th0 = 2.9219166032837487

import myFCN as mf # c'est là qu'on a placé les points
import matplotlib.pyplot as plt

thp = [i/200 for i in range(1257)] # on prépare la courbe
rp = [mf.fonc(3,[the_p,the_e,the_th0],i/200) for i in range(1257)]

ax = plt.axes(projection='polar')

plt.plot(thp, rp, color='black',
```

```
        zorder = 0) # tracé bidon pour installer le graphique dessous

# en polaire, les barres de errorbar sont mal dessinées
for ip in range(mf.NPts):
    rBth = [mf.xPts[ip], mf.xPts[ip]]
    rBr = [mf.yPts[ip]+mf.dyPts[ip], mf.yPts[ip]-mf.dyPts[ip]]
    plt.plot(rBth, rBr, color='green',
            zorder = 1)
    thBth = [mf.xPts[ip]+mf.dxPts[ip]*(1 - ith/50) for ith in range(101)]
    thBr = [mf.yPts[ip] for ith in range(101)]
    plt.plot(thBth, thBr, color='green',
            zorder = 1)

plt.plot(mf.xPts, mf.yPts, 'o', linestyle='', markersize=3.6, color='black',
        zorder = 2)

plt.plot(thp, rp, color='black',
        zorder = 3) # retracé pour que ce soit dessus

ax.set_rmax(25) # doit être imposé après les points car ils recadrent
ax.set_rticks([5.,10.,15.,20.])
ax.set_rlabel_position(315)
plt.show()
```