

```

import LT.box as B
import numpy as np

# Loop and Array Examples – Building on Class Work
# This combines the PDF examples with practical applications

# PART 1: Quick Array Definition (from PDF)
# a lot of times when you are dealing with a few data points, you don't
# have to read them from a data file. You can quickly define using the
# np.array() --
x=np.array([1,2,3,4]) # pay attention to the parenthesis outside,
# and the bracket inside
y=np.array([2.1,4.0,5.9,7.5])
deltay=np.array([0.1,0.1,0.2,0.2])
B.plot_exp(x,y,dy=deltay)

# Now you made a plot very quickly without creating a data file first and
# then reading from it

#####
#What if you have multiple data files which you want to analyze similarly
# Then you will benefit from using loops
# For example, we have three data files ( yellow.dat, etc) with similar data
#definitions We can do the following

filenames=np.array(['yellow.dat', 'green.dat', 'blue.dat'])
range_low=np.array([1,2,3])
range_high=np.array([8,9,10])

#since there are three data files, the argument to the function range() below
# should be range[0,3]
for fileindex in range(0,3):
    # Instead of range(0,3), you could also just use range(3)
    # the fileindex is an integer starting from 0, ending with 2
    # verify that by uncommenting the following statement
        #print ('fileindex is', fileindex)
    f = B.get_file(filenames[fileindex])
    #print (filenames[fileindex])
    A = B.get_data(f, 'A')
    #print (A)
    b = B.get_data(f, 'b')
    db = B.get_data(f, 'db')
    #B.pl.figure()      # this creates a new figure window for the plot
                        # if you comment the above out, then all graphs will
                        # will be in the same window

```

```

B.plot_exp(A, b, dy=db)
myrange=B.in_between(range_low[fileindex], range_high[fileindex],A)
fit = B.linefit(A[myrange], b[myrange], db[myrange])
B.plot_line(fit.xpl,fit.ypl)
#the above define the fitting window in termes of variable A, using
# the previously defined two arrays: range_low and range_high
#don't forget that the [fileindex] argument; see comments below

#What if you want to fit the graphs, but with a different range
# then you need to define the ranges (both lower and upper limits)
# in array as well, before the loop

# similarly, if you were to fit those data with different intial
# parameters, they should be pre-defined in arrays before the loop
# then access the corresponding array members in side the loop

# Notece all the statements in the loop should be indented; if there is no
# indent, then python assume the loop has ended

#If you did everything correctly, you shoudl see three graphs,
#almost straight lines, plotted on top of each other
# what if you want each graph to be plotted on a different canvas?

##### So far we addressed 1. quick defintion of arrays with few data points
##### 2. using loop to access multiple files and perform
##### similar analyses

##### Next we have a simple example of quickly assigning values to
#array members using loops
##### Let's say three arrays of 20 mebers (index from 0 the 19)
##### newx, newy, newdy
### there are different ways of doing this; blow is a simple version
### first you need to create the empty array,
newx=np.empty(20)      #if you use newx=np.array([]), and then try to access
                      # the members of newx[index],even if index is in range
                      # it won't work; since those array members dont' exist yet
newy=np.empty(20)

newdy=np.empty(20)

for array_index in range(20):
    newx[array_index]=array_index ### don't forget the [array_index]
                                  # part when accessing the array members
    # assume that newx is angle in radian
    newy[array_index]=np.cos(array_index)

    #newdy[array_index]=np.sqrt(newy[array_index])
    newdy[array_index]=0.2

B.pl.figure()

```

```

B.plot_exp(newx, newy, dy=newdy)

# Let's try to fit this graph with 8th order polynomial (even though
#we know it is basically y=cos(x)

r1=B.in_between(2,15,newx)
fit2 = B.polyfit(newx[r1], newy[r1], yerr=newdy[r1], order=8)
B.plot_line(fit2.xpl, fit2.ypl)

#you could comment out the line below if you just want to check
#if the 8th order polynomial worked
WiderX=np.linspace(0,20,100) #This linspace function creates an array with
#100 members from 0 to 20
WiderY=fit2.poly(WiderX)+1 #WE are accessing the fitting results/8th order
#polynomial functions from 3 lines above using fit2.poly()

# The "+1" above create an offset of 1 so you can separate the two lines

B.plot_line(WiderX, WiderY)
#You can see that the fit function overlaps with teh original fit2 line in
#the range of 2 to 15, where you defined the orginal fit2. However,
#it differs dramatically outside that range
# what did you learn from this exercise?
#Loops are faster than copy-pasting teh same code. Arrays let you handle
# different settings for each file automatically.
#Polynomial fits can look good but can looks bad outside the data range

```