```
In [26]: # practiging fitting plots with various functionsimport numpy
as np

In [27]: import numpy as np

In [28]: import LT.box as B

In [28]:

In [29]: file_name = 'examples_data.py' #set name of data file

In [30]: f = B.get_file(file_name) #open and read the file

In [31]: # get the data

In [32]: # current

In [32]:

In [33]: #extract the data from the file and save them into their
respective variables

In [34]: A = B.get_data(f,'A')

In [35]: b = B.get_data(f,'b')

In [36]: db = B.get_data(f, 'db')

In [37]: C = B.get_data(f,'C')

In [38]: D = B.get_data(f, 'D')

In [39]: # The following examples fitl to fit fits C vs A, using linear
fit,

In [40]: #polynomial fits, in either the whole ranges (fit1, fit2) or a
subrange (fit 3, 4)

In [41]: B.plot_exp(A, C, db) #plot the data with error bars (db)
Out[41]: <ErrorbarContainer object of 3 artists>
```

Important
Figures are displayed in the Plots pane by default. To make them also appear inline in the console, you need to uncheck "Mute inline plotting" under the options menu of Plots.

```
In [42]: B.pl.show() #display the plot

In [42]:

In [43]: #You should uncomment the next line to see what it does to the
x-axi

In [44]: #Can you set the y-axis title now?

In [45]: B.pl.xlabel("x (unit)") # sets the x-axis label
Out[45]: Text(0.5, 0, 'x (unit)')

In [46]: B.pl.ylabel("y (unit)")
Out[46]: Text(0, 0.5, 'y (unit)')

In [46]:

In [47]: fit1 = B.linefit(A, C, db) #does the linear fit of the entire
dataset
chisq/dof =  1.7887470965978935
offset  =  -8.886820603907685  +/-  0.24439543707385852
slope =  9.463587921847273  +/-  0.1290421055659006

In [48]: B.plot_line(fit1.xpl, fit1.ypl) #plots the fitted line
Out[48]: [<matplotlib.lines.Line2D at 0x15f60a110>]

In [48]:

In [49]: #the following two lines selecting the ranges

In [50]: r1 = B.in_between(4.0, 18.0, C) #. in window should be changed
to in _between

In [51]: r2 = B.in_between(2.0, 12.0, C) #creates a variable with the
data points of C

In [52]: #between 1 and 12

In [52]:

In [53]: #only fit the selected ranges as specified by r1 or r2

In [54]: fit3 = B.linefit(A[r1], C[r1], db[r1]) #linear fit using data
from range 1
chisq/dof =  1.2245662034703695
offset  =  -9.81841745266039  +/-  0.375043497129335
slope =  9.892720858233144  +/-  0.18371099586622422

In [55]: B.plot_line(fit3.xpl, fit3.ypl) #plot it
Out[55]: [<matplotlib.lines.Line2D at 0x15f73e050>]
```

```
In [55]:

In [56]: #the fit below use second order-- defined by the "2" in the
argument

In [57]: #polynomial; you should change 2 to other integers and see how
the

In [58]: #fits are different

In [59]: fit4 = B.polyfit(A[r2], C[r2], db[r2], 2)
chisq/dof =  0.7757575757575722
parameter [ 0 ] =  -2.9259259259254775  +/-  2.029820013068078
parameter [ 1 ] =  2.1548821548819577  +/-  2.5188754443260843
parameter [ 2 ] =  2.104377104377827  +/-  0.7555453817525889

In [60]: B.plot_line(fit4.xpl, fit4.ypl)
Out[60]: [<matplotlib.lines.Line2D at 0x15f7f09d0>]

In [60]:

In [61]: #for polynomial 5 (example) within range r2

In [62]: fit5 = B.polyfit(A[r2], C[r2], db[r2], 5)
chisq/dof =  0.6414918427924733
parameter [ 0 ] =  415.9043355282652  +/-  399.0864134652094
parameter [ 1 ] =  -1363.0645292471067  +/-  1266.7850482018996
parameter [ 2 ] =  1749.8791172516821  +/-  1586.3094883400727
parameter [ 3 ] =  -1098.668405712301  +/-  979.9029609901401
parameter [ 4 ] =  339.3010443124799  +/-  298.73046810568184
parameter [ 5 ] =  -41.218238158485356  +/-  35.97382499319897

In [63]: B.plot_line(fit5.xpl, fit5.ypl)
Out[63]: [<matplotlib.lines.Line2D at 0x1680d0ed0>]

In [63]:

In [63]:

In [64]: speed = fit4.parameters[1].value #for speed = fit4.par[1]

In [65]: #coefficient value (slope)

In [65]:

In [66]: d_speed = fit4.parameters[1].err #or d_speed = fit4.sig_par[1]

In [67]: #uncertainty value in the coefficient

In [67]:
```

```
In [68]: #printing out the fitting parameters with proper significant
digits

In [69]: print("\nspeed is %3E +/- %3E m/s \n" % (speed, d_speed))

speed is 2.154882E+00 +/- 2.518875E+00 m/s


In [70]: # Print results in scientific notation

In [70]:

In [70]:

In [70]:

In [71]:
```