Python Exam Guide

The exam is a guided analysis for the experiment of Stefan Boltzmann Law. You do not need to understand the physics of experiment. However, if you wish, the lab manual for that experiment is at http://wanda.fiu.edu/boeglinw/courses/Modern_lab_manual3/stefan_boltzmann.html (This could help you check whether your final result (the slope of the linear fits) are correct).

(*But if you want a short summary of the experiment, here it is: According to Stefan Balzmann law, the power (P) emitted by a black body (approximated by a tungsten wire) is proportional to T^4 (T being the temperature), and this experiment is to verify that. As it turns out although P is relatively easy to measure, T for a very hot wire is not. Fortunately, the wire resistivity (R) is not hard to measure, and T is usually a function of R. This function T(R) can be experimentally determined by fitting T vs R with some higher order polynomial). So, experimentally one can simply measure the Voltage/Current (giving you the resistance, then resistivity) of the very hot wire, then actually calculate the temperature from the already determined function of T(R). Then we can verify if the measured P is truly proportional to the calculated T^4.*)

You can use the sample programming (assignment 1/2), as well as various sources contained at http://wanda.fiu.edu/boeglinw/courses/Modern_lab_manual3/ python_practice.html. Internet/google search/ChatGPT is also allowed. Programs written by anyone else who has taken PHY3802L at FIU is not allowed.

Life Lines: You can request help for three items. You can ask “Is this correct or not?”. There is no penalty for that. However, if you ask about how to fix it in order to proceed, the points for that item will be fully deducted.

To avoid typos, copy variables names (other than typing it yourself), if specified, from the guide if you wish.

Also, make sure you when you “translate” the physics/math operations into code, avoid makings mistakes on math operation orders in subtraction and divisions.

0. If you prefer not to and did not download the zipped files, you could separately download the two data files: **W_resistivity.data** (resistivity data file); **Sb_Measurements.data** (referred as the experimental data file), save them to a directory where you will perform the analysis.

Please submit your .py file, and **Six** graphs (Note Graph 1, 3,4 are required, while Graph 2, 5 and 6 are extra points), then put all in **one PDF file for submission**.

Note: Item 1-8 sets up your program and determines T (temperature) as a function of R (resistivity)

1. Start a python script called SB_analysis.py and save it to the same directory as you downloaded data files.

2. Import the LabTools.box as B and numpy as np; 2pt

3. Load the resistivity data file and store the data in the following variables: The temperature (T) in the variable ***T*** the resistivity R in the variable ***rho,*** 10pt

4. Calculate the relative resistivity by dividing rho by its value at T = 300K, store the relative resistivity in a variable called ***rho_r (if you wrote: rho_r=rho/T or rho_r/300, there is a problem in reading comprehension)*** 5pt

5. Create a new figure using *B.pl.figure()*

6. Plot the values of ***T*** as a function of ***rho_r*** , 10pt (Graph 1)

7. Use polyfit to Fit a 7-th order polynomial to the temperature (***T***) as a function of the ***rho_r*** and store the fit in the variable ***p7,*** **then plot the fitted functions over the data points, in the same plot**. 10pt (Graph 1)

8. Label the axes and give this plot the title: *W temperature as a function of resistivity* 5pt (Graph 1)

   Note: Item 9 – 15 use results above to convert the measured resistivity to the temperateure of the hot wire

9. Load the other experimental data file and call it ***dexp***. Store the following data : the lamp current

   (I) in the variable ***I_l***, the lamp voltage (V) in the variable ***V_l***, and the thermopile signal (PmV) in the variable ***PmV*** and its uncertainty (sig_PmV) in the variable ***dPmV. Note the naming scheme where the "l" stands for lamp***10pt

10. Get the parameter Rw and store it in ***R_wire*** and the parameter R_Lc and store it in ***R_l_cold***. ***R_wire*** is the resistance of the wires connecting the lamp to the power supply and ***R_l_cold*** is the resistance of the cold lamp (meaning at 300K) .

    If you copy/pasted the numerical values from the data file, then you lose the 10 points here.

    10pt (62 point so far item 1-10)

11. From the cold lamp resistance subtract twice the wire resistance and store the result in ***R_l_cold_corr*** 5pt

12. From the measured lamp voltage and lamp current, calculate the lamp resistance (**yes, you do need to some basics physics here**) and store the result in the variable ***R_l*** 5pt

13. Correct the lamp resistance by subtracting twice the wire resistance and store the result in the variable ***R_l_corr***. 5pt

14. Calculate the ratio of the corrected lamp resistance***(R_l_corr)*** to its valueat 300K, i.e. the resistance of the cold lamp (***R_l_cold_corr***), and store this result in the variable ***R_l_rel*** (lamp relative resistivity***)***5pt

15. (Assuming that this ratio is dominated by the resistivity of tungsten) Use your previous fitted function (p7: Temperature as a function of relative resistivity) and calculate the corresponding lamp temperature, using ***R_l_rel*** as the variable, and store the results in the variable ***T_L. (You can simply use the p7.poly() function, or using all those parameters from p7)*** 10pt

The above (item 15) is a crucial step to finish the later analysis, if you wish to be helped, you could; But the 10pt would be deducted.

15. a. (Extra 5pt) Create a new figure. Graph the calculated T_L vs R_l_rel and plot the same function (p7) as in item 7, with data points in **blue triangle markers**. The graph and the function should be on the same figure. This will help you make sure your calculation is actually correct. Note you can use any online resources and you might need it here. Label y-axis as Calculated W temperature, and x-axis as lamp relative resistivity. (Graph 2)

Note: Item 16- investigate the relationship of power emission and the temperature , and verify if it obeys the Stefan-Boltzmann law

16. The Stefan-Boltzmann law states that the total power emitted by a black body is: $P = \sigma T^4$ where σ is a constant and T is the temperature of the black body. Taking the logarithm of both sides of this equation gives us : $\ln(P) = \ln(\sigma) + 4\ln(T)$. Plotting $\ln(P)$ as a function of $\ln(T)$ should therefore be a line.

To test this, calculate the (natural) logarithm (using *np.log*) of the power measured by the thermopile (variable ***PmV*** and its uncertainty ***dPmV***). Store the logarithm of the power in the variable ***logP***. Using error propagation calculate the uncertainty of ln(***PmV***) given the uncertainty or ***PmV*** is ***dPmV*** and store the result in the variable ***dlogP*** 20pt

17. Calculate the logarithm of the calculated lamp temperature(***T_L***) and store in the variable ***logT_L*** ("***L***" here stands for lamp) 10pt

18. Create a new figure using *B.pl.figure()* , then plot ***logP*** as a function of ***logT_L*** including the errors (d***logP)***10pt (Graph 3)

19. Fit a line to ***logP*** (including its errors) as a function of ***logT_L*** and store the fit in the variable ***logf*** 10pt

20. Plot the fitted line over the data points (as in item 18).5pt (85 pt item 11-20, and 5 extra pt) (Graph 3)

21. Label all axes appropriately and add the title: Test of the Stefan-Boltzmann Law 5pt (Graph 3)

22. **Print the fitted slope and its uncertainty** (such as "My slope is 5.23 +- 0.35 versus the expected value of ") – do not copy/paste any numerical values -- with 2 digits after the decimal point **on the same figure** (as in item 18 and 20). Do not print it in the console. Here you need to know what is the expected slope from this guide document. 5pt (Graph 3)

23. Create a new figure and show the same graph as in item 18. (5pt) (Graph 4)

24. Create three empty arrays (**tmin, slopes, dslopes**). Each array should have 8 elements. (5pt)

Now we want to perform the same linear fits as in item 19, but in 8 different starting point/ranges, and store the fitting parameters for the slopes and their uncertainties in the array **slopes and dslopes,** and graph all 8 fitting lines on the same data points and graph**.**

Note: item 25 -31 , totally 52 points, should be done using the for loop. If you obtained the same graph 4 by copying/pasting the same code 8 times, you will receive a 30 point deduction.

25. Start a **for loop** by using an integer variable **index** (or named anything else you prefer) ranging from **0 to 8:** item 26-29 will be executed 8 times because of the loop. Obviously you are not allowed to just copy/paste them 8 times. (5pt)

26. define the starting point of your fitting range tmin[index] by adding 200*index to the minimum of the T_L array. You might find T_L.min() useful here (5pt)

27. Selecting the graphed data points (as in item 18 and 23) using a masked array, i.e., B.in_window() or B.in_between() depending on the version of your software—so the calculated temperature ***T_L*** is larger than tmin[index], and up to ***T_L.***max(). Call this selection array ***sel_ht*** **10pt**

28. Make another linear fit as in item 19, but only for the selected range right above. Store the new fit in the variable ***logf1.*** 10pt (Graph 4)

29. Store the fitting parameter slope and slope uncertainty in the array slopes and dslopes by

referring to the corresponding array members. 10pt

30. Plot the new fitted line and print (on the graph, not in the console) the new results as in item 22, over the graph in same graph 4 created in item 23. Make sure the texts are aligned vertically and not printed on top of the previous ones. You might want to let all the texts start at the same x position, and change the y position using such as "0.2+index*0.1 10pt (Graph 4)

31. End the for loop (2pt)

32. Label the new graph axes accordingly. Also title the graph as 'Test of the Stefan-Boltzmann law in different fitting ranges' **(5pt)** (subtotal 77 pt item 21 to 32) (Graph 4)

33. Create a new figure and define the y-axis range to be 3.0 to 5.0. Make a graph of slopes (with uncertainties) VS tmin, using the 8 fitting results from the loop. Label axes accordingly. (Extra 5pt (Graph 5)

34. Also create a new graph plotting reduced chi-squared VS tmin graph, using the 8 fitting results from the loop. Label axes accordingly. You might need to modify/add variables in item 24 to item 29. (Extra 5pt) (Graph 6)

Total number of points: 224 (plus 15 extra) be re-scaled to 100.