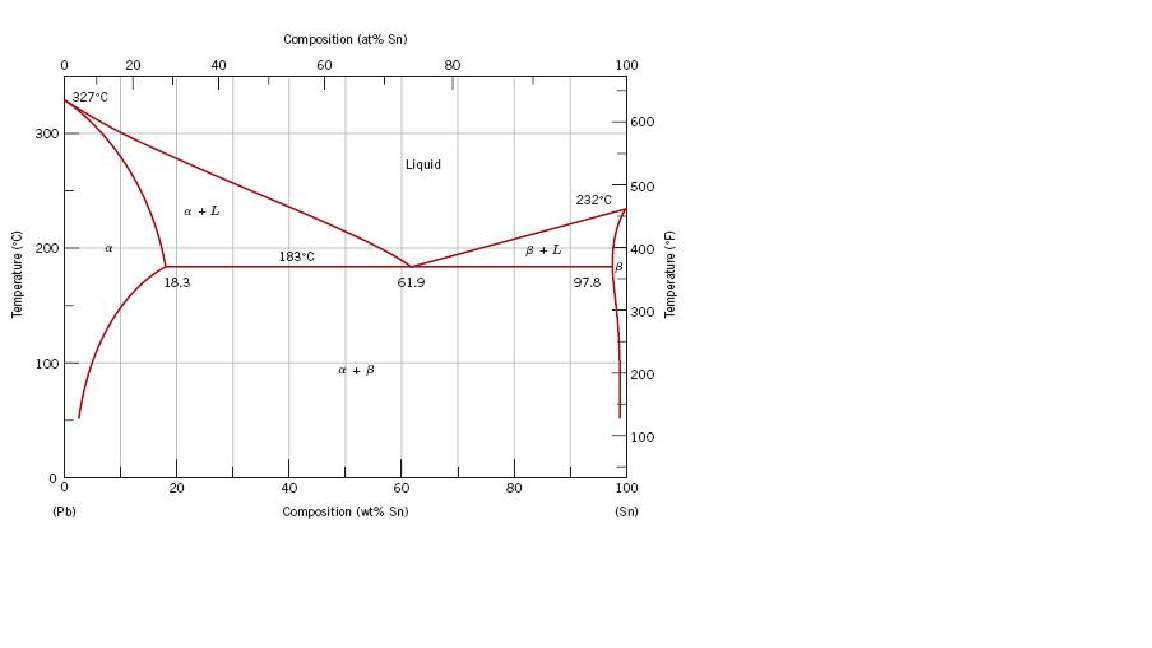
**MSE 360-Fall 2017**

**Matlab Data Analysis**

**Week 1 Laboratory Worksheet, Due by the Week 2 lab (Sept 12-14)**

**The purpose of this lab and assignment is to get hands-on experience in using the MATLAB programing language to understand and analyze experimental results. This assignment is worth 100 points total. All plots and the answers to all questions should be included in a write-up to be turned in at the beginning of next week’s lab. A formal lab report is not required, but the document should be professionally formatted.**

**1.** Figure 1 shows the phase diagram for a binary Pb-Sn alloy, with the composition by weight % (wt%) Sn shown along the x-axis and the temperature, in °C (left) and °F (right), shown along the y-axis.



**Figure 1.** Phase diagram of a binary Pb-Sn alloy

This lab will involve using MATLAB to perform data analysis on (simulated) measurements of the liquidus temperature, *TL*, of a Pb-Sn alloy with the wt% of Sn varying between 20 and 40 %.

**2.** To open MATLAB, first click on the CAEN software icon on your desktop. Use the search function to search the list of available software to find ‘MATLAB’. The page that lists the available software may automatically launch when you log in to the CAEN computer. Select the 2017a MATLAB Student Instructional version of MATLAB and click ‘Run’. A new window will launch that is titled ‘Cloudpaging player’. When MATLAB has been completely loaded in the Cloudpaging player window, select ‘MATLAB’ and click ‘Launch’. Contact an instructor if you have trouble opening MATLAB.

**3.** Load the dataset ‘Sn30Pb70.mat’ into your MATLAB workspace**.** The data set *Sn30Pb70* simulates measurements of *TL* for a 30/70 Sb-Pb alloy. To open this data set this you will need to download the file ‘Sn30Pb70.mat’ from the MSE course website found on Canvas (<https://canvas.umich.edu>). Save this file to your current folder in MATLAB. By default this is the …/Documents/MATLAB directory. Once this is done, you can use the *load* function to load the file into the MATLAB workspace. If you need help using the load function, type ‘help load’ at the command line in the MATLAB command window. If you have loaded the file correctly, you should see *Sn30Pb70 1x25* double in your workspace window. By default the workspace window can be found in the upper right corner of the MATLAB window

The most common problem student will have is that they don’t save the .mat file in the correct location. Make sure that Sn30Pb70.mat can be seen in the Current Folder (which is on the left hand side of the MATLAB interface).

**4 [15 points].** Use the *histogram* function to plot the distribution of this data.For more instructions on how to use the *histogram* function, use ‘help histogram’. Include this plot in your write-up.

The histogram function defaults to 4 bins. I would recommend having the students start at 10 bins. You can chance the number of bins by calling histogram(*var\_name,* X) where *var\_name* contains the values being binned, X is the number of bins to used.

*Q1*) Explain what values are plotted on the x and y axes of the histogram. How many bins does the *histogram* function use by default? Should the number of bins change as the number of data points increases/decreases? Justify your answer.

*Q2*) What is the difference between a histogram and a bar chart?

**5. [10 points]** Use the *histogram* function to plot the probability distribution of the data.Instructions on how to do this can be found by using ‘help histogram’. You will need to use the ‘Normalization’ and ‘Probability’ keywords when you call the *histogram* function. Include this plot in your write-up.

The syntax for this is histogram(*var\_name*, X, ‘Normalization’, ‘Probability’)

*Q3*) Verify that the sum of the bins of the probability distribution histogram is equal to 1. Explain why this sum is equal to 1.

There is no function to do this in MATLAB (that I’m aware of). They should do this by hand. Depending on rounding, etc… the sum may not be exactly 1, but it should be very close.

**6. [20 points]** The Gaussian function describes a peak centered about  and takes the form

 Eq. (1)

where *A* controls the height of the peak, and *σ* controls the width of the peak. Using *A* = 1, *σ* = 1, and = 260, plot a Gaussian function in MATLAB. Use a range of *x* values from 255 to 265, with *dx* = 0.1. Contact an instructor if you are unsure of how to obtain the values of *f(x)*.

General syntax of this should be something along the lines of:

A=1;

sigma = 1;

x\_bar = 260;

x = 255:0.1:265

OR

x = linspace(255,265,101)

y = A\*exp(-(0.5\*(x-x\_bar).^2)/(sigma\*sigma));

Note the period in the .^2 operation. This period tells MALTAB that the ^2 operation should be applied to each element of (*x-x\_bar*). Failure to include the period will result in the error ‘Inputs must be scalar and a square matrix’.

Next, integrate the area under the curve of the Gaussian function. The MATLAB function *trapz* can be used to perform this calculation. **Be careful** - depending on how you use the *trapz* function, you may need to account for the value of *dx* in your calculation of the integral. After you have done this, divide each point of your original Gaussian function by the area you just calculated. As a result of this operation, the area under the curve of the new Gaussian function will be equal to 1 (verify this!). If done correctly, you will now have a probability distribution function (PDF) similar to the probability distribution histogram you have already made. Include the plot of the PDF in your write-up.

General syntax should be something along the lines of:

area = trapz(y)\*0.1; %% need to account for the value of *dx*

OR

area = trapz(x,y); %% Matlab uses the values in the vector *x* to determine the value of *dx*

y2 = y/area.

h1=plot(x,y);

set(gca,’Fontsize’, N) %% sets the font size, try N=40;

set(h1,’LineWidth’,N) %% sets the line width of the plot, try N =4;

*Q4*) Compare the PDF described by the Gaussian function against the probability distribution histogram of the *Sb30Pb70* data set. How are they similar? How are they different?

**7. [10 points]** Use the ‘*randn*’ function in MATLAB to simulate a data set with a sample size, *N*, of 200, with a mean value of 260 and standard deviation of 1.See ‘help *randn*’ for an example of how this can be done using the *randn* function. Use the *histogram* function to plot the probability distribution of this data.Include this plot in your write-up.

Syntax is:

vals = mean\_val + stand\_dev\*randn(N,1);

Note that if randn(N) is used you will get an NxN matrix of random numbers, which is not what we want.

*Q5*) Compare the PDF described by the Gaussian function against the probability distribution of this new data set. Does the new probability distribution histogram look more or less similar to PDF described by the Gaussian function? Explain your answer.

**8. [10 points]** Use the *randn* function in MATLAB to simulate a data set with *N* = 500, 1000, 5000, and 10000.Use the *histogram* function to plot the probability distributions of these data sets.Include these plots in your write-up.

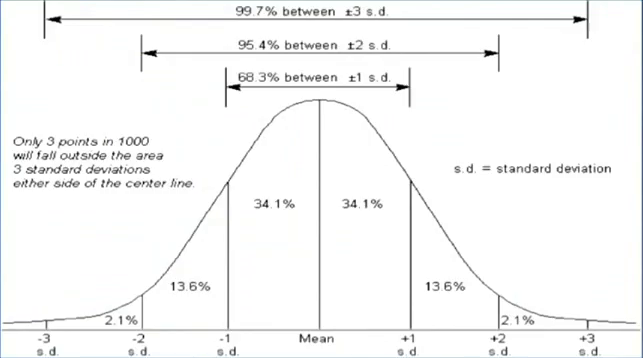
*Q6*) For a PDF, the term in Eq. 1 corresponds to the mean value of the data. For each histogram, estimate the value of  How does change as N increases? If the location ofdoes not appear to change, you may need to increase the number of bins you are using in your histogram.

This is meant to be a qualitative exercise, but you can use the *mean*() function to check the mean value of each data set, if needed.

*Q7*) The Gaussian PDF, also commonly referred to as the *normal distribution*, is a *limiting distribution function* for samples of multiple measurements with small variations. Why? Justify your answer.

**9.** **[5 points]** The width of the probability distribution function, controlled by the *σ* term in Eq. 1, is related to a statistical quantity known as the *standard deviation.* The calculation of the standard deviation will be discussed in a later lesson, but qualitatively it describes how far from the mean a set of values will tend to be distributed. As such, the standard deviation of the normal distribution can be used to make predictions on the range of likely values for a measured quantity.

There is a useful approximation that is called the ‘68-95-99.7’ rule. This refers to the fact that when making multiple measurements of a normally distributed quantity, approximately 68.3% of the measured values fall within one standard deviation of the mean, 95.4% of the measured values fall within two standard deviations of the mean, and 99.7% of the measured values fall within three standard deviations of the mean. This is illustrated in Figure 2.



**Figure 2.** Illustration of the ’68-95-99.7’ rule for a probability distribution function.

Figure 3 shows the plot of two probability distribution functions:

Gaussians

**Figure 3.** Plot of two different probability distribution functions.

*Q8*) According to Figure 3, which PDF would have a smaller standard deviation? Is there a parameter and/or calculated quantity that is equal for these two distributions; why or why not? What characteristic(s) of the PDFs did you use to come to this conclusion?

**10. [5 points]** Calculate and report the mean and standard deviation for the Sn30Pb70 data set. These values can be determined using the *mean* and *std* functions in MATLAB.

Mean = 260.2281

Std = 1.0343

*Q9*) If the you were run this experiment again, assuming no other outside factors change your results, estimate the probability that you would measure *TL* to be in the range of 260 +/- 1.5 °C. Explain your reasoning behind the estimation.

**11. [10 points]** The measure of uncertainty in experimental observations is often reported in terms of a range of values around the measured mean. One common method is to use a 95% confidence interval, meaning that you are 95% confident that the true mean is within the range of values you report. Applying this method, if you report a *TL* of 262 +/- 3 °C, you are claiming to be 95% confident that the true value of *TL* falls within the range of 259 to 265 °C. **It is very important to explicitly state the underlying statistics behind your reported uncertainty value to avoid any confusion in how it should be interpreted since there are different methods for generating confidence intervals and uncertainties.**

When plotting your results, this uncertainty is often expressed using *error bars.* It should be noted that **the name ‘error bar’ does not refer to mistakes** made during measurement, but rather to the range of uncertainty in the possible values of the quantity being measured. Often error bars are drawn such that the width of the error bar spans the 95% confidence interval for the measurement.

Table 1 shows a simulated sample of liquidus temperatures for different compositions of a Sn-Pb alloy. The column on left gives the wt % Sn of the Sn-Pb alloy, the column in the middle gives the mean value of the simulated liquidus temperatures, and the column on the right gives the uncertainty of the data within two standard deviations from the mean.

**Table 1.** Simulated liquidus temperatures for 20-40 wt% Sn in a Sn-Pb alloy.

Plot liquidus temperature vs. wt% Sn using the MATLAB *plot* command. Make sure each data point is discrete (don’t connect the points with a line). **Using the Sn-Pb phase diagram**, find a linear approximation for what *TL* “should be” between 20 and 40 wt% Sn. This linear approximation will serve as the theoretical (i.e. predicted value) for the liquidus temperature of the Sn-Pb alloy. Add this linear approximation for *TL* to the plot of melting temperature vs. wt% Sn (this should be plotted as a solid line).

Linear approximation should be something close to *y* = -2*x* + 320.

Plotting multiple data sets on the same MATLAB plot can be done a couple different ways. Read ‘help plot’ and ‘help hold’ for more information. For this exercise, using the *hold* function will likely prove to be more useful. Include this plot in your write-up.

General syntax should be something along the lines of:

x = 20:0.1:40;

y = -2\*x+320;

WT = [20,25,30,35, 40];

T = [282, 273, 262, 253, 243];

hold on %% ALL plots will be place on top of each other. If you need to start over just close the figure window.

h1 = plot(x,y,’r’) %% ‘r’ sets the color red to differentiate the line from the marker colors

h2 = plot(WT,T,’o’) %% ‘o’ set each point to be plotted as a separate ‘o’ shaped marker

|  |  |  |
| --- | --- | --- |
| **wt% Sn** | ***TL*** | **Uncertainty** |
| 20 | 282 | +/- 1.67 |
| 25 | 273 | +/- 1.83 |
| 30 | 262 | +/- 2.03 |
| 35 | 253 | +/- 1.63 |
| 40 | 243 | +/- 1.76 |

*Q10*) Looking at this plot what can you say about the accuracy and precision of your data? Does it appear to be in good agreement with the theoretical fit? Can you make any quantitative assertions about your results as currently plotted? Why or why not?

**12. [15 points]** Add error bars to your plot of the data. Error bars can be added to a plot using the *errorbar*function (use the *errorbar* function in place of the *plot* function).

Errorbar syntax is:

errorbar(x,y, uncertainty)

OR

errorbar(x,y,UL,LL) where UL = upper limit and LL = lower limit of uncertainty.

Each error bar needs to be plotted separately:

hold on

plot\*(x,y,’r’)

errorbar(20,282,1.67,’o’)

errorbar(25,273,1.83,’o’)

etc …

*Q11*) Can you make any additional quantitative statements about your results after you have added the error bars? Given that these temperatures were measured by a thermocouple, can you infer anything about the accuracy of the thermocouple based on the plot? Estimate a lower bound for the systematic error in its measurements.

*Q12*) What are some of the possible sources of experimental uncertainty in the data? Assuming the reported uncertainties are trustworthy, what is the upper bound for the value for the precision of the thermocouple that was used to measure ?