Experimental Physical Chemistry

CH 346

Laboratory Information
Course Syllabus
and
Data Analysis Tutorial

John W. Shriver and Michael George
Department of Chemistry
The University of Alabama in Huntsville
Fall 2006

Preface

This document covers the necessary basic information needed to do well in CH 346. This is the second semester of a two semester series for chemistry majors and others requiring experimental physical chemistry laboratory experience. The course is largely structured around the standard experimental physical chemistry text by Garland, Nibler and Shoemaker (***Experiments in Physical Chemistry***, 6th Ed., McGraw-Hill, Boston, 2003). Detailed information on instrumentation, experimental methods, data analysis, notebooks, and reports can be found there. Additional information is supplied here, and supplemental information and resources may be provided as needed.

You will be expected to have read all of the relevant background material in these sources prior to coming to the laboratory.

I hope that you find this document useful. Please feel free to notify me of any errors so that they may be removed and the document updated as soon as possible. Any suggestions on ways to improve either this information or the course would be appreciated.

John Shriver
29 November 2006

Contents

1. Preface …….. 2

2. Laboratory Introduction

2.1 Meeting place and times…….. 5

2.2 Instructor……. 5

2.3 Course goals……...…….. 5

2.4 Pre-requisites…… 5

2.5 Textbooks…… 5

2.6 Software requirements……...…….. 6

2.7 Assignments and grading……. 6

2.8 Attendance and missed laboratories…...…….. 6

2.9 Classroom conduct ……..…… 7

2.10 Academic honesty ……..…….. 7

2.11 Turnitin.com…… 7

2.12 Disabilities…… 8

2.13 Complaints…… 8

2.14 Acknowledgements……..……. 8

3. Course Syllabus

3.1 Intro - Notebooks, lab reports, and error analysis…….…….… 9

3.2 Joule-Thomson Effect……..…… 9

3.3 Transient kinetic methods……..…… 9

3.4 Steady-state enzyme kinetics……. 9

3.5 Exchange kinetics and NMR…….…… 10

3.6 UV-Visible spectroscopy……...……..……... 10

4. Data Analysis

4.1 Data analysis…… 12

4.2 Data fitting…… 12

4.3 Software…… 13

4.4 Igor Pro 5 Tutorial…… 13

4.5 Data…… 14

4.6 Data entry…… 14

4.7 Objects and experiments…… 15

4.8 Saving data…… 15

4.9 Command window…… 15

4.10 Plotting data…… 15

4.11 Simulations…… 16

4.12 Simulating experimental noise…… 18

4.13 Differentiation and integration…… 18

4.14 Least squares fitting of data…… 18

4.15 User defined functions…… 20

4.16 User defined menus…… 23

Appendices

A. Laboratory notebook guidelines…… 26

B. Laboratory reports guidelines …… 27

2. Laboratory Introduction

Meeting place and times

Classroom and times: WH 327, Tuesday 7:05 – 10:00 PM

Instructor

Instructor: Michael A. George
Office: 103 MSB
Phone: 236-824-6916
Email: georgem@uah.edu
Office hours: by appointment

Course goals

To convey the joys of experimental physical chemistry and the satisfaction obtained from doing quality work. You should obtain an appreciation and understanding of experimental methods and equipment used in chemical thermodynamics, kinetics, and spectroscopy, including data collection methods, instrumentation, data reduction, error analysis, and report writing.

Pre-requisite

You should have taken, or are currently taking, CH 341 or 347. You should also have taken CH 345 or CHE 295 You should also have taken courses in differential and integral calculus.

Textbooks

"Experiments in Physical Chemistry, 7th Edition" by Carl Garland, Joseph Nibler, and David Shoemaker, McGraw-Hill, Boston 2003. **(Referred to below as GNS.) The majority of this course is built around this text. It is highly recommended that you obtain a copy.**

Recommended:
"Data Reduction and Error Analysis for the Physical Sciences, 3rd Edition" by Philip Bevington, McGraw Hill, 2002.

"An Introduction to Error Analysis: The Study of Uncertainties in Physical Measurement, 2nd Edition" by John Taylor, University Science Books, 1996.

"Mathematics for Physical Chemistry, 3rd Edition" by Robert Mortimer, Elsevier, Academic Press, Boston, 2005.

"How to Write and Publish a Scientific Paper, 6th Edition" by Robert Day and Barbara Gastel, ISI Press, 2006.

Software requirements

You will be expected to be able to use **Igor Pro 5** (Wavemetrics) for doing simulations, plotting data, and performing linear and nonlinear least squares fitting of data. An academic distribution license has been obtained from Wavemetrics which allows a fully functional version of the software to be distributed to students in this class at no charge with the understanding that the software will be used for educational purposes only. This is a considerable savings over the nonrmal price (academic: $400, standard: $600). If you desire to use this software in your research or other work, you should purchase another copy with the appropriate license from http://www.wavemetrics.com/.

Assignments and grading

The course is built around five sets of experiments which are devoted to various aspects of chemical data collection and analysis. Laboratory reports will be required for all experiments. Reports are due no later than one week after completion of each module. Guidelines for preparation of reports will be covered in the first lab meeting (see GNS, pp. 10-25). Students will work in the laboratory in pairs, but each student will be required to keep their own laboratory notebook, which will be graded periodically in class by the instructor. Although students will work in pairs, laboratory reports should be written independently. Occasionally a take-home exam will be given that will require independent reading and research to complete. Although resources may be shared for these take home exams, the writing of the exam should be done by each student independently.

The relative weighting of each contribution to your grade is as follows:

5 laboratory reports	75 points
Take-home exams	20 points
Laboratory notebook	5 points

Grades will be assigned using the following scale:
100-90 A; 80-89 B; 70-79 C; 60-69 D; below 60 F.

Attendance and missed laboratories

You are responsible for attending all laboratories, and you are expected to start all laboratories at the designated times. Any delays or absences must be approved by the instructor. Only excused absences may be made up. No separate, make-up laboratories will be scheduled. Missed laboratories which have been excused by the instructor will have to be made up during regularly scheduled class time by arrangement with the instructor. These must be done as soon as possible. No experiments may be performed without supervision by the instructor or a designated teaching assistant. Failure to make up an experiment will result in a zero on the lab report associated with that experiment.

Classroom conduct requirements

The following three requirements are essential in order to conduct a safe and productive teaching laboratory:

1. All students in the laboratory must behave in a safe manner as described by the instructor, and Departmental and University policies. Safety procedures are posted at the entrance to the laboratory and at various places throughout the lab. **Students are expected to have thoroughly read all of the material describing an experiment before coming to the laboratory and they should be aware of all safety concerns.**

2. All students must respect the equipment which they use. This means understanding how to use it safely and correctly before starting to use it. This also means that every precaution should be taken to prevent damage to laboratory equipment. Quality equipment used to make accurate and precise measurements is very expensive. **You are required to immediately inform the instructor of any equipment malfunction or damage.**

3. All students are expected to maintain a professional attitude in the laboratory. All students must treat others in the laboratory with civility and respect. They must conduct themselves during class in a way that does not interfere with the opportunity of others to learn. This includes keeping unnecessary noise and discussions to a minimum.

Failure to comply with these requirements may result in points being deducted from a student's final grade, up to a maximum of 20 points. Serious misconduct will result in the student being asked to leave the laboratory, and may be referred to the appropriate University office with a recommendation that the student be removed from the course.

Academic honesty

Your written assignments and examinations must be your own work. **Scientific and academic misconduct (e.g. fabrication of data, copying and plagiarism) is a serious offence and will not be tolerated.** To insure that you are aware of what is considered misconduct, you should review carefully the definitions and examples provided in Article III, Code of Student Conduct, Student Handbook, p. 93. If you have any questions related to this point, please contact the instructor as soon as possible.

Consent to the use of Turnitin.com

UAH is committed to the fundamental values of preserving academic honesty as defined in the Student Handbook (7.III.A, Code of Student Conduct). The instructor reserves the right to utilize electronic means to help prevent plagiarism. Students agree that by taking this course all assignments are subject to submission for textual similarity review to Turnitin.com. **All laboratory reports must be submitted in hard copy and electronic forms.** Assignments submitted to Turnitin.com will be included as source documents in Turnitin.com's restricted access database solely for the purpose of detecting plagiarism in such documents. The terms that

apply to the University's use of the Turnitin.com service, as well as additional information about the company, are described at www.uah.edu/library/turnitin.

Disabilities

If you have a disability that may require some modification of seating, testing, or other class procedures, please see me after class or during my office hours to discuss appropriate modifications. You should also contact Student Development Services in UC 113 (telephone 824-6203) for further assistance.

Complaints

If you have difficulties or complaints related to this course, your first action usually should be to discuss them with the instructor. If such a discussion would be uncomfortable for you or fails to resolve your difficulties, you should contact Professor William Setzer, Chair of the Department of Chemistry. Professor Setzer's office is MSB 203C. His telephone number is 2416. If you still are unsatisfied, you should discuss the matter with the Dr. Daniel Rochowiak, Associate Dean of the College of Science. Dean Rochowiak's office and telephone number are MSB C207 and 824-6605.

Acknowlegement

This course material is developed in part using material supplied by previous instructors of this course, including Dr. James Baird, Dr. Y.W. Kim, and Dr. Jeffrey Weimer.

3. Syllabus and Course Outline

This course is composed of five modules or experiments. The five experiments will be preformed simultaneously by rotating teams of two students each. Each team will rotate through the set of five experiments, with at least two weeks devoted to each of the labs.

There will be occasional take home exams which will require independent study, web and literature searches for completion. These will be devoted to various topics important in experimental physical chemistry, such as temperature measurement, pressure measurement, voltage and current measurement, equipment design, computer interfacing and data logging, vacuum methods, high pressure methods, balances, pH meters, signal averaging, and timing.

Lab 1 Intro - Notebooks, lab reports, and error analysis

Introduction and basic methods (GNS,pp.1-66)

1. Lab preparation
2. Safety
3. Recording experimental data
4. Literature search
5. Lab notebook – **required format and procedures - Appendix A**
6. Lab reports – **Journal of Physical Chemistry A/B format - Appendix B**
7. Significant figures, errors, and noise - Accuracy vs. precision
8. Analytical and Numerical methods
9. Graphs
10. Errors and discordant data
11. Statistical treatment of data
12. Calculations and propagation of error

Lab 2 Joule-Thomson effect

Joule-Thomson Effect (GNS, Expt. 2)

1. Gas Expansion
2. Temperature measurement
3. Inversion temperature

Lab 3 Transient kinetic methods

Method of Initial Rates: Iodine Clock (GNS, Expt. 20)

1. Rate laws
2. Initial rates
3. Flow methods

Lab 4 Steady state enzyme kinetics

Inversion of sucrose (GNS, Expt. 22)

1. Michaelis-Menten kinetics
2. Activation energy
3. Temperature and pH dependence
4. Lineweaver-Burk and Eadie-Hofstee plots

Lab 5 Exchange kinetics and NMR

NMR Study of a Reversible Hydrolysis Reaction (GNS, Expt. 21)

1. Forward and reverse rates
2. Effect of temperature
3. Relaxation times
4. NMR linewidth analysis

Lab 6 UV-Visible spectroscopy and fluorescence

Absorption Spectrum of a Dye (GNS, Expt. 34)

1. Absorbance
2. Conjugation and delocalized electrons
3. Fluorescence

4. Data Analysis

Data analysis includes:

1. An assessment of data quality (error analysis)
2. A derivation of information from the data (data fitting)
3. A quantitative measure of the quality of the information gained (error analysis)

The first requires an analysis of the noise or error in the data; the other two require a mathematical model which is used to fit the data to obtain values for adjustable parameters in the model. A good introduction to error analysis can be found in Chapter 2 of the GNS text. This is the focus of the first module in this course.

An introduction to plotting, simulating, and fitting data is provided here. This will be the focus of the second module in this course. Additional information on data fitting can be found in Chapters 2 and 22 of the GNS text. In particular, Chapter 22 provides an excellent introduction to the least squares method. The tutorial below provides a practical introduction to using a computer program to fit data to models. You will be required to be able to use the program Igor Pro to do both linear and nonlinear least squares fitting of data.

Data fitting

The fitting of data to a model is often called least squares fitting, curve fitting, regression, optimization, or data reduction. All of these essentially refer to the same thing.

Fitting data to a straight line (linear least squares or regression) is very easy and is commonly the only method used in introductory courses. Nonlinear least squares however is not difficult and is the preferred method for fitting data that is intrinsically defined by a nonlinear function (e.g. a binding curve, an exponential decay, a Gaussian distribution). In this course we will stress the use of nonlinear least squares when appropriate.

Considerable effort has been devoted to developing ways to transform nonlinear data into a linear form so that linear regression can be applied. Examples include the Lineweaver-Burke and Scatchard plots. These transformations were developed in the middle of the last century so that data could be fit with graph paper and a straight edge. Today coordinate transformations to linearize data are unnecessary given the availability and speed of modern computers and the development of robust nonlinear regression routines (linearizations can be useful to obtain initial parameters and for presentations). More importantly, linearization results in distortions of the noise distributions in the data so that the basic assumption of Gaussian distributed noise that is required for the application of least squares is violated (this is true even if the data is weighted according to the linearization). Nonlinear least squares is now just as easy as linear least squares on most computers. By fitting the raw data directly, the Gaussian distributed noise is retained and the most accurate definition of the fitted parameters is obtained. In addition, the parameters (and their errors) are obtained directly from the fit, i.e. they do not have to be calculated from linear fit parameters by reversing the transformation and applying the appropriate error propagation formulas to define their precision.

Fitting intrinsically linear data to a linear function (linear least squares) such as y = mx + b uses straightforward equations to calculate the fitting parameters m (slope) and b (y-intercept). In this case an analytical solution is defined. In contrast, fitting to nonlinear functions requires an iterative, numerical approach and there is no closed-form analytical solution. Linear least squares is easily performed on a hand calculator and is also very accessible in Excel. Nonlinear least squares or regression requires a computer with sufficient power to perform multiple iterations to optimize an initial set of guesses in a reasonable period of time. Although there is no analytical solution to the nonlinear fitting problem, if a sufficient number of iterations are performed, the fitted parameters can be defined to any level of precision. There is a certain amount of user intervention required to perform nonlinear least squares (e.g. supplying reasonable initial guesses, determining if the search has stopped in a local minimum, and specification when convergence has been achieved). There are also a number of different algorithms used to perform the numerical calculations, e.g. grid search, Marquardt-Levenberg, and Simplex routines. An excellent source for these and other methods is the book *Numerical Recipes in C: The Art of Scientific Computing*, by W.H. Press, S.A. Teukolsky, W.T.Vetterling, and B.P. Flannery, Cambridge University Press, New York, 1992.

Software

There are many good software packages available for data analysis. Most students taking this course are familiar with Microsoft Excel. The Solver routine that can be installed with Excel permits nonlinear least squares optimization, but the program is very limited and will not be utilized in this course. Plotting packages such as KaleidaGraph (http://www.synergy.com/) also permit data fitting, but the emphasis is on creating excellent, publication quality graphics, and the data fitting options can be cumbersome. Mathematica (http://www.wolfram.com/), Origin (http://www.originlab.com/), and Igor Pro (http://www.wavemetrics.com/) are three of the best data analysis products currently available. Wavemetrics provides a very generous educational distribution license, and we will therefore focus on Igor Pro (version 5). A copy of Igor Pro 5 can be obtained from the instructor for your personal use in this course. You are free to install it on your personal computers as long as it is used only by you for this course.

Igor Pro 5 Tutorial

Igor Pro can be used at various levels for data analysis and presentation. This introduction is intended to provide the basics that will get you started. More detailed information can be found using the online help that is available in the Igor menu after starting up the program. In addition, the complete Igor manual is available as a pdf file after installation. You should look over the first few chapters of that manual to obtain a guided tour of Igor.

Please note that the Igor manual is very large (>2000 pages). Do not let this intimidate you. Learn to use the software in stages. Use the introduction provided here as well as the guided tour provided with the software. After that you can use the manual to obtain more information on specific topics as needed.

Data

Data is typically used in Igor in the form of x,y pairs ("XY data" in the Igor terminology), although the x and y data values are stored separately. For historical reasons, a data file is referred to as a "wave" in Igor. Igor was originally designed to work with waveforms with hundreds of y-values as a function of an x variable (e.g. time), with each value equally spaced along the x-axis. It would have been better if these were referred to as vectors or arrays, but the initial name has been retained, and we will have to live with it. You will have to get used to the fact that your data files are called waves even if they are not oscillating functions.

"XY data" in Igor does not require equal spacing, and is more appropriate for the type of data that we will typically encounter in physical chemistry and biophysics. Real data will normally be composed of a series of points which are *not* equally spaced along the x-axis. In contrast, for simulations we will often set up a table of equally spaced x-values and calculate the corresponding y values using the appropriate function.

Data files are saved in Igor as two waves, one containing the x values, and one containing the y values.

Data entry

Data can be typed into the table that is opened after starting Igor. This is typically called TableO:waveO in the title bar at the top of the table window. You can also load data from text files using the Data->Load Waves->Load General Text menu option. Basic operations on the data can be performed using the options under Data (e.g. naming the columns of the data table). Data will be typed into the second column; the first column simply enumerates the data points.

Note: Be careful. When you make or load a wave, it is retained as part of Igor until it is "killed". You can see which waves are active by looking at the list of waves in the "Data Browser" window. If this is not visible, it can be opened by selecting it under the Window menu item.

Data will often exist as text files on a computer. You can enter data in this manner, or it may have been collected using another program or instrument (such as a spectrophotometer, data logger, hand held PC, etc.) and saved as a text file.

Data can be typed into a text editor (e.g. Word) in tab delimited columns. To demonstrate, open a word processing program and type the following x,y pairs using the Tab key to separate the first and second elements in each row:

1 2
2 4
3 6
4 16
5 24

Save this file as a "text" file (specified under the file options upon selection of the save command).

In Igor, under the Data menu item, select Load Waves -> Load Delimited Text. Navigate to the file you want to load, select it, and in the window that follows, you should see the text entries and two suggested names for the two waves which will be created, each containing one of the columns (e.g. wave 1 and wave2). Other options can be selected, but you can ignore these for now. Hit the Load button to load the data as waves. In the "Data Browser" window you should now see these waves listed (if you do not see the "Data Browser", open it under the Window menu option).

The tables showing the contents of the waves are not yet displayed. To see these, simply double click on the wave entries in the "Data Browser" list.

Note: The "Current Data Folder" is the data folder which is currently active (and will be used for saving data) is indicated in the Data Browser window with a red arrow.

Objects and Experiments

As you use Igor to perform a data analysis, you will generate a number of so-called "objects" including tables, procedures, and graphs or plots. These may be saved as a group called an "experiment" and recalled as a group later for further work.

Saving data

Save data, and the associated collection of objects (e.g. plots), by choosing Save Experiment from the File menu and make sure you choose Packed Experiment File in file format.

Command Window

You can type Igor commands into the Command window at the bottom of the screen. Commands are typically performed using menu items, and you will see these appear in the Command window as well. This can be used to view a history of previous actions. You can use the arrow keys to scroll up in the history, and hitting return on a selected line will place that line in the active area at the bottom of the command window. This line can be edited or modified before hitting a return to repeat the command.

Command syntax is described in Chapter IV-1 of the Igor manual and will not be duplicated here. You should print out pages IV-1 through IV-14 now and become familiar with this material.

Plotting data

Data plotting (or graphing) is done using the New Graph option under Windows. The data to be plotted is specified by selecting the appropriate x and y waves from the New Graph window. Be careful, the x wave is on the right, and the y wave is on the left (which seems backwards to me).

You can modify plotting options by clicking on the plot area that you want to change (e.g. axes, plot, axis labels). This will open the Modify Axis window. The axis to be modified is specified at

the top of the window with a pull down menu. Try the various options by selecting the Axis, Auto/Man Ticks, Ticks&Grids, Axis Label, Label Options, and Axis Range buttons. Most of this is self-explanatory. Use the mirror option (select Axis options) to place axes around the plot, rather than on just the left and bottom.

A legend can be added to the graph by clicking the menu item Graph-> Add annotation. There is an interesting option for labeling data points in the tutorial.

Simulations

You must be able to simulate data in order to fit it.

The first step to fitting data is obtaining a function which will reasonably simulate the data. This is the mathematical model. The function will contain adjustable parameters which will allow the function to be adjusted to fit the data. For example, a single exponential decay would be described by the function $x = a \bullet e^{-kt}$, where "a" and "k" are the parameters which define the dependence of x on time t: "a" specifies the initial value or amplitude, and k specifies how fast the function decays (the rate constant).

Simulating (synthesizing) data can be done using the Make Waves option under the Data menu item. Specify the names for the waves you want to create in the boxes of the Make Waves window that opens following execution of this menu command (e.g. timevalues, and yvalues). Set the number of elements (size) by defining the number of rows. Set dimensions to 1 (one value for each element in the wave). Try this now by selecting the Make Wave option under Data and make two waves called timevalues and yvalues, with a size of 10 elements in each. Note what happens if you attempt to label a wave with "x", "y", "sin", "exp", or "time". The orange and red highlight indicates that this name cannot be used: reserved words cannot be used as wave names in Igor.

Once you create a wave, its elements are filled with 0's. You have created an empty array of defined size with a name. You have simply declared that you will be using arrays of this dimension, nothing more.

The contents of a wave can be filled with numbers based on a value for an index. For each wave dimension, you can set the size and step size for the index for that wave by using Change Wave Scaling under Data. Unfortunately, for all one dimensional waves, this variable is called x. Thus, you might have a wave called xvalues, and another called yvalues, but the index for each is still called x. The index does not correspond to the point number (unless you choose to define it that way). Each wave has its own x value. The characteristics of x are set by using Change Wave Scaling under Data. As an example, lets adjust the wave timevalues that was created above. Go to the Change Wave Scaling window and select the "timevalues" wave and set SetScale Mode to "Start and Right" and define the Start to be 0 and the Right to be 2. x now runs from 0 to 2 in 10 steps (the increment size must therefore be 0.2). If you look at the contents of the timevalues wave, you see that it is still filled with zeros. We have not used the index yet. Type in the command window xwave=x and you will now see the xwave values changed. In this case we

have used the x index to specify the timevaues. The values for each element are now specified along the x-axis. These values can now be used to calculate corresponding y values.

To demonstrate, let's simulate an exponential decay. Let's simulate a decay curve with 100 points over a time period of 5 seconds, an initial amplitude of 2.5, a decay constant of 2 sec^{-1}. Use Make Wave to create two waves, which we will call xtime and fluor. Use ChangeWaveScale and set x to run from 0 to 5 with 100 points for both. In the Command window type

xtime =x.

Double click on the wave xtime in the Data Browser to confirm that xtime contains 100 values running from 0 to 2. In the Command window, type

Variable A = 2.5, k = 2

and hit return (note you must use the command Variable to set numeric variables and constants). Type

fluor=A * exp(-k * xtime).

This is an unusually powerful command. Note that fluor is a wave (or array), A and k are real numbers, and xtime is a wave. Each element of the fluor wave is calculated using the corresponding element in the xtime wave. That is, every y value is calculated using the appropriate x value. This one line specifies every value of the fluor wave according to the values specified by A and k, and the xwave value. Obviously, the xwave and ywave must be of the same size. Make sure that you completely understand this.

Now double click the fluor wave in the Date Brower window. You should see a table of 100 values starting at 2.5 and decreasing to near zero. Open the xtime wave table and make sure that you understand the correspondence between each x and y value and its associated x index.

Go to New Graph under window and set up a plot with fluro for the y axis and xtime for the x-axis. You should see a plot of a single exponential decay displayed in a new window. Resize the window to make it clearly visible.

You can change the value of A and/or k (use the Variable command in the Command line as above), and repeat the calculation of yvalues to check the influence of the parameters on the curve. Try it. For those interested, Igor will allow you to place the values of parameters on a slidebar to allow continuous variation of numbers and plots.

Simulating Experimental Noise

You can add noise to simulated data using a Gaussian random number generator. There is a function built into Igor called gnoise() which returns a random number with a Gaussian distribution centered at 0 and the standard deviation specified in parentheses. For example

gnoise(0.05) returns a random number from a Gaussian distribution centered at zero with a standard deviation of 0.05. Type

fluor=A*exp(-k*x) + gnoise(0.05)

This adds onto the smooth theoretical curve a random noise level that is similar to what might be observed experimentally. The table of fluor values and the plot change instantaneously. If you wanted to keep the original smooth curve, you could have set up a new wave that would hold the simulated noisy data, and then overlay simulated data onto the theoretical curve for comparison.

Doing Math in Igor

The contents of waves can be added and inserted into another wave by typing

wave2 = wave0 + wave1

or

avgwave=(wave0 + wave1)/2

(wave2 and avgwve must be created before these commands will work.)

Under Analysis there is the option Wave Stats. This will give you the average, standard deviation, and other statistical parameters associated with a wave. Each is assigned a name which can be used in the command line for further calculations. This can only be used for one wave at a time, since the output parameters such as V_avg are updated each time you run Wave Stats. If you want to retain these for further use you could assign the values that you want to a new variable, e.g.

WaveStats wave0
Variable wave0std=V_sdev

There is a sum() command which allows you to sum the contents of a wave. Check the Manual for details (page V-627). To view the results of the sum command use the Print command, e.g. type Print sum(wave0).
You could also set up a variable called sumwave0 and type

Variable sumwave0=sum(wave0)

or

Variable Npts=40
Variable avgwave0=sum(wave0)/Npts
Print avgwave0

Differentiation and Integration

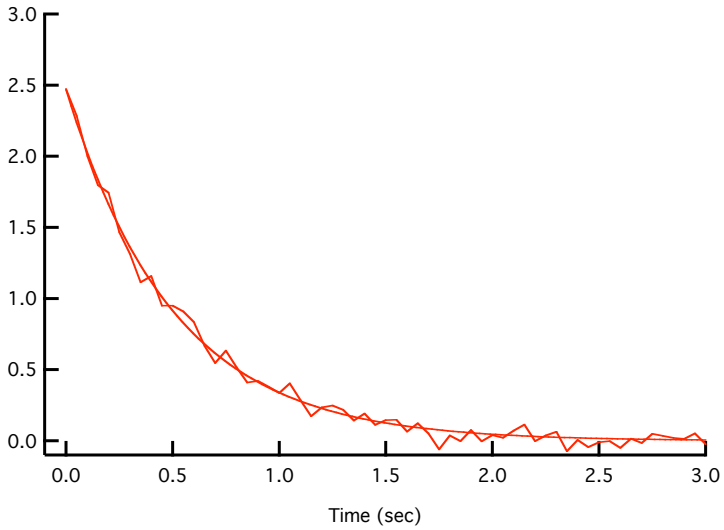
Note that under the "Analysis" menu option there are various options including the ability to integrate and differentiate x,y data. The trapezoidal option under integration works best with x,y data. For differentiation you will probably want to use Central Differences and use Approximate for the end points. Try it.

Least squares fitting of data

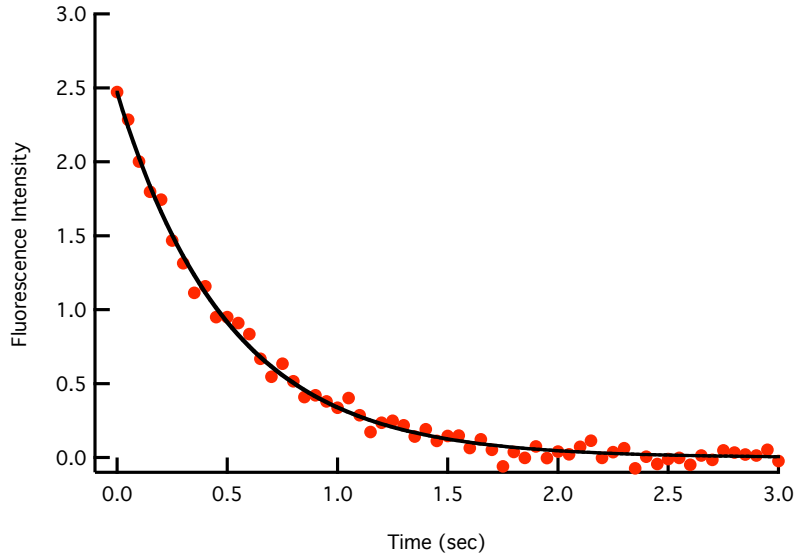
A brief tutorial is provided here to get you started doing linear and nonlinear regression with Igor. See Chapter III-8 in the Igor manual for a complete survey of Curve Fitting with Igor.

Under the "Analysis" menu item, chose "Curve Fitting" and select the "Function and Data" tab option. Various possible built-in functions are available under the Function pull-down menu (e.g. line, polynomial, exp, gauss). As we will see below, if the appropriate function is not listed, you can create it. Selection of "line" results in a linear least squares fit. As you can see, it is no more difficult to do a nonlinear fit than a linear fit. For demonstration purposes, select the "exp" function under Function. The exponential function with the fitting parameters is shown in the window in the lower part of the Curve Fitting window. There are 3 fitting parameters: yo (a vertical offset), A, invTau (what we called k above).

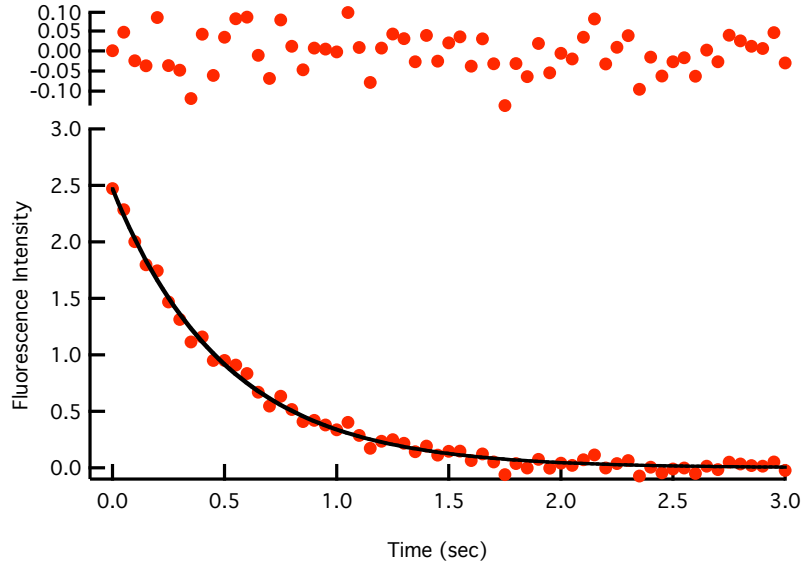
Under Y Data select the wave used for the y-axis (e.g fluor in the example above), and under X data select the x-axis wave (e.g. xtime). Under Data Options tab select "none" for both Weighting and Data Mask, under Coefficients you could enter initial guesses for the parameters, which may be necessary to get things to behave well (not necessary here), and under Output Options choose "auto" for Destination and "none" for Residual. Press "Do It", and the fitted curve should be overlayed on the data in the plot window, and the final parameters (coefficients, listed as K0, K1, and K2, since this is a 3 parameter fit) are listed in a separate window with the function shown to facilitate correlation of K0, K1, and K2 to the elements in the equation (K0 is the offset here, K1 is the amplitude, and K2 is the rate constant). Press OK and the coefficients (fitted parameters) are printed in the Command window along with the errors (± one standard deviation). The graph can be saved by going to Save Graphics under File and selecting a high resolution format. The resulting graphics image can be pasted into any document.



The line and points styles can be changed by clicking on the plot and setting the various options to more appropriate values. For example, the data can be shown with individual points, and the fitted curve shown with a bold smooth curve:



You can look at the scatter of the data around the fitted curve by plotting the residuals (the difference between the data and the calculated curve at each point). In the Curve Fitting Window, select the Output Options and select "auto trace" under Residual. Do It and the following should result:



Note that the distribution of the residuals is the same across the entire fitted region, indicating that the fit is excellent and the distribution of the residuals is totally random. The width of the residuals reflects the Gaussian noise in the original "data" which we synthesized. The standard deviation of the residuals is about 0.05 (the value used in the gnoise function above).

User Defined Fitting Functions

In addition to using the fitting functions included under the pull down menu in the Curve Fitting window, you can define your own function by selecting the "New Fit Function" button in that window. This opens up a new window where you can specify a name for the function (be careful to use a unique name), as well as names for the fitting coefficients, and the independent variable (usually x). The actual function is typed into the Fit Expression window below this in the form f(x)= …..

Note that this window is actually a convenient way to write an Igor Procedure for your function. Once it is saved, you can go to the Procedure window and view what you have written. When you save the experiment, this function is also saved. If you would like this fitting function to be present for other experiments, then you need to save the Function in the Igor Procedures directory. Open a Procedure window, and cut and paste the Function into the Procedure window and save in a directory outside of Igor called "My Fitting Function". Create an alias to this folder, and insert the alias into Igor Procedures (Igor Procedures are run at startup, and so the contents of My Fitting Function will be activated and available after startup). Note that the name of the Fit function will only appear in the Curve Fitting window Function list if the function is specified as a FitFunc by using the :FitFunc specifier in the Function line. An example of a simple function is shown here:

```
#pragma rtGlobals=1          // Use modern global access method.

Function TestFunc(w,x) : FitFunc
        Wave w
        Variable x

        //CurveFitDialog/ These comments were created by the Curve Fitting dialog. Altering them will
        //CurveFitDialog/ make the function less convenient to work with in the Curve Fitting dialog.
        //CurveFitDialog/ Equation:
        //CurveFitDialog/ f(x) = C1 + C2 * x + C3 * Sin(x)
        //CurveFitDialog/ End of Equation
        //CurveFitDialog/ Independent Variables 1
        //CurveFitDialog/ x
        //CurveFitDialog/ Coefficients 3
        //CurveFitDialog/ w[0] = C1
        //CurveFitDialog/ w[1] = C2
        //CurveFitDialog/ w[2] = C3

        return w[0] + w[1] * x + w[2] * Sin(x)
End
```

Note the comment lines (starting with the double slash "//") containing the command CurveFitDialog contain information on the number of coefficients and the parameters. Surprisingly, this is used by the Curve Fitting function and is required. Comment lines are usually ignored by computer programs, but in this case these are actually read.

You can define your own fitting functions without using the "New Fit Function" window by simply writing a Procedure and including the :FitFunc specifier. A more complicated example is shown below for a DSC fitting function. Note that this function is a multiline function and cannot be written simply as a f(x) function. In this case the New Fit Function would not work. The equation in the comments appears in the Fit Function window when selected, but the actual function is more complicated. The value returned by the function appears at the end of the Procedure (i.e. ctot).

```
#pragma rtGlobals=1             // Use modern global access method.

Function DSCdimerfit(w,x) : FitFunc
        Wave w
        Variable x


        //CurveFitDialog/ These comments were created by the Curve Fitting dialog.
                Altering them will
        //CurveFitDialog/ make the function less convenient to work with in the Curve
                Fitting dialog.
        //CurveFitDialog/ Equation:
        //CurveFitDialog/ f(x) = (w[1] - (w[0] - x) * w[2])* dalpha
        //CurveFitDialog/ End of Equation
        //CurveFitDialog/ Independent Variables 1
        //CurveFitDialog/ x
        //CurveFitDialog/ Coefficients 8
        //CurveFitDialog/ w[0] = Tm
        //CurveFitDialog/ w[1] = DH
        //CurveFitDialog/ w[2] = DCp
        //CurveFitDialog/ w[3] = A1
        //CurveFitDialog/ w[4] = B1
        //CurveFitDialog/ w[5] = C1
        //CurveFitDialog/ w[6] = D1
        //CurveFitDialog/ w[7] = beta

        Variable Hterm, Gterm, K, alpha
        Variable Hterm1, Gterm1, Kplus, alpha1
        Variable Hterm2, Gterm2, Kminus, alpha2
        Variable Q, Qplus, Qminus
```

```
        Variable dalpha, cexcess, cav, ctot
        Variable R = 1.99

    Variable Po = 0.001

        Hterm = w[1]  -  (w[0] - x) * w[2]
        Gterm=w[1]*((w[0] - x)/w[0]) - w[2]*(w[0]-x)+x*w[2]*ln(w[0]/x)
        K = exp(-Gterm/(R * x))
                // calculating the progress from the partition function is stable
                Q = 1 + 2/(sqrt(1 + 8* Po / K) - 1)
                alpha = 1 - 1/Q

        //  Calculate derivative of alpha by perturbing temperature up and down 0.05 degree
        Hterm1 = w[1]  -  (w[0] - (x+0.05)) * w[2]
        Gterm1=w[1]*((w[0] - (x+0.05))/w[0]) - w[2]*(w[0]-
                        (x+0.05))+(x+0.05)*w[2]*ln(w[0]/(x+0.05))
        Kplus = exp(-Gterm1/(R * (x+0.05)))
                Qplus = 1 + 2/(sqrt(1 + 8* Po / Kplus) - 1)
                alpha1 = 1 - 1/Qplus

        Hterm2 = w[1]  -  (w[0] - (x-0.05)) * w[2]
        Gterm2=w[1]*((w[0] - (x-0.05))/w[0]) - w[2]*(w[0]-(x-0.05))+(x-
                        0.05)*w[2]*ln(w[0]/(x-0.05))
        Kminus = exp(-Gterm2/(R * (x-0.05)))
                Qminus = 1 + 2/(sqrt(1 + 8* Po / Kminus) - 1)
                alpha2 = 1 - 1/Qminus

        dalpha = (alpha1 - alpha2)/0.1
        // Calculate excess Cp
        cexcess = (w[1] - (w[0] - x) * w[2])* dalpha*w[7]
        cav = (1 - alpha) * (w[3] + w[4] * x) + alpha * (w[5] + w[6] * x)
        ctot = cexcess + cav

        return ctot

End
```

User Defined Menu Items and Activating User Procedures

You can customize your menus so that commonly used functions and macros which you have written are readily available in a pull-down menu. Somewhere in your directory system, create a folder called "My Igor Procedures". Place your procedures in this folder, so that they are separate

and outside the main Igor Program directories. To make these Procedures available to Igor, create an alias to this folder and place the alias in a folder titled "User Procedures" in the Igor Directory. All procedures in the User Procedures directory are available to be loaded with an "#include" statement at the beginning of a program.

Procedures in the Igor Procedures folder differ from the User Procedures in that they are not only loaded but also run upon startup.

To add menus items to the Igor menu which can be used to initiate a commonly used procedure, create an Igor procedure similar to the following:

```
#pragma rtGlobals=1             // Use modern global access method.
#include "StraightLine"
#include "CalcLorentzian"
#include "BindingSimulation"
#include "ProteinStabilityCurveSim"
#include "DSCSimProc"

Menu  "Analysis"
    "---"
        Submenu "CH347Stuff"
        "Straight Line Draw", StraightLine()
        "Lorentzian Draw", CalcLorentzian()
        "BindingSimulation", BindingSim()
        "Protein Stability Curve Simulation", ProteinStabilityCurveSim()
        "DSC Simulation", DSCsim()
        End
End

Menu "Macros"
"Display Straight Line Control Panel", StartStraightLineFunc()
End
```

This creates a Menu item called "CH347 Stuff" under Analysis with entries to start procedures. There is also another menu item added under Macros. The actual procedures are activated with the initial procedures that include "Execute/P "INSERTINCLUDE …." statements so that selection of the item in the menu list causes the procedure to be loaded and then to run. This is a convenient way to not have to load a large number of procedures before they are needed. Save this procedure in your My Igor Menus directory, create an alias, and place the alias in "Igor Procedures". This will cause the menu items to be installed up startup.

Appendix A

Laboratory Notebook Guidelines

A laboratory notebook should provide a complete description of everything done by a researcher in the laboratory. In a research environment (both academic and corporate), it is essentially a legal document that clearly indicates what a single investigator has done. It should be complete enough to permit anyone with proper training to repeat the work. The following is required in this course and should serve as a guide to how to maintain a laboratory notebook after you graduate:

1. The notebook should be a bound notebook. Loose leaf binders and spiral notebooks are not permitted. The pages should be permanently bound and page numbers should be printed on each page by the manufacturer. Pages should never be removed from a notebook. If a page is not to be used, a line should be drawn across the page and an explanation should be written at the bottom explaining the problem.

2. All entries in the lab notebook should be made in ink. Mistakes should not be erased. Instead a single line should be drawn through the error and the correction placed adjacent to the error. An explanation should be provided on the same page so that there is no ambiguity about why the change was made.

3. The first few pages of the notebook should be used to maintain a table of contents. Start the first experiment entry on page 5 or so. Fill in the table of contents as you do the experiments.

4. The notebook should be complete, neat, and well organized. Details may be perfectly clear to you at the moment, but a few months later you may have trouble recalling what was done. If in doubt, write more. Write down everything you see. If an instrument is drifting or behaving in an unusual way, make a note of it.

5. Each entry page for an experiment should have the days date at the top right corner. Write on one side of a page. As additional pages are used for an experiment, always place the date at the top right corner. Always start a new page for each experiment and each day. Do not crowd too much information onto a page.

6. Provide a brief summary of the purpose of an experiment on the first page of each day's work. If you are doing two or more different experiments on the same day, do not interleave entries for the two experiments on the same page. Instead, keep the two experiments on separate pages. This may require alternating pages if both extend beyond a single page. Make sure there is no ambiguity. Use sketches to describe experimental setups.

7. All information relevant to the experiment should be recorded in the notebook. Loose paper and notes (e.g. from a weighing) should not be used. Take your notebook with you to the balance. Any information that you think should be on a scrap piece of paper or a paper towel should be entered directly in the lab notebook. All calculations should be clearly written. The notebook is a detailed record of everything you have done. For example, if you have to make up

a new solution in the middle of an experiment, make sure this is perfectly clear and there is no ambiguity in the data where this change was made.

8. All raw data should be entered directly in the notebook when practical. Plots of the data should be entered into the notebook as the data is being collected. This can help in detecting problems with experimental design. It is normally not a good idea to do an experiment "blind". Look at the data as it is being collected. At the same time, you must be careful that such information does not bias your observations.

9. Computer data should be printed out if possible and attached with tape or staples on the appropriate page of the notebook. Clearly this is not possible with large data files containing hundreds if not many thousands of data points. The computer file names and the storage location (e.g. hard drive, CD, or DVD name) must be clearly indicated instead. A printout of spectra and scans should be attached whenever possible for clarity.

10. The laboratory instructor should initial and date the last page of each days work before you leave the laboratory.

Appendix B

Laboratory Report Guidelines

Laboratory reports in CH 345 and 346 should follow the guidelines for manuscripts submitted to the Journal of Physical Chemistry (A or B). These can be found at that journals web site (http://pubs.acs.org/journals/jpcafh/index.html) under information for authors. The Guidelines for Authors can be downloaded in pdf format. The guidelines for the A and B series are identical. Follow the guidelines for feature articles. Be sure to organize the sections as described in the guidelines on page 3A.

Prepare the reports using a word processing program, and submit them both in hard copy form and electronically by email to the instructor. Do not use binders for the hard copy, but staple your report in the upper left corner.

Additional guidelines for preparation of graphics

1. Plots should be prepared using Igor Pro. Follow the guidelines in GNS (pp. 34-35) for computer-generated plots. Do not use manual plots.

2. Use a consistent plot style throughout. Use papers in a recent issue of the Journal of Physical of Chemistry to get an idea of an acceptable style.

3. As you can see from inspection of the Journal of Physical Chemistry, the use of color in plots is now standard. Keep it professional. Use color to aid in distinguishing multiple curves (e.g. the ray data and a fit). A single curve should be plotted in black. Backgrounds and axes are never in color.

4. Labels, axis legends, and symbols should be large enough to be clearly read.

5. Units should be displayed in the axis legends.

6. Figures should not be embedded in the report text, but should be on separate pages at the end of the report. Figure captions or legends should not appear below the figure, but should appear together on a separate page preceding the figures.

Further information:

"The ACS Style Guide: Effective Communication of Scientific Information, 3rd Edition", American Chemical Society Publication, 2006.

"How to Write and Publish a Scientific Paper, 6th Edition" by Robert Day and Barbara Gastel, Greenwood Press, 2006.