• Getting Started

This help file contains overview and guided tour material and constitutes an essential introduction to Igor Pro. The main sections are:

We strongly recommend that you read at least the first two sections.

The material in this help file is duplicated in Volume I of the Igor Pro PDF manual which is accessible through the Help menu.

• Introduction to Igor Pro

Igor is an integrated program for visualizing, analyzing, transforming and presenting experimental data.

Igor's features include:

- Publication-quality graphics
- High-speed data display
- Ability to handle large data sets
- Curve-fitting, Fourier transforms, smoothing, statistics and other data analysis
- Waveform arithmetic
- Image display and processing
- Combination graphical and command-line user interface
- Automation and data processing via a built-in programming environment
- Extensibility through modules written in the C and C++ languages

Some people use Igor simply to produce high-quality, finely-tuned scientific graphics. Others use Igor as an all-purpose workhorse to acquire, analyze and present experimental data using its built-in programming environment. We have tried to write the Igor program and this manual to fulfill the needs of the entire range of Igor users.

Igor Objects

The basic objects that all Igor users work with are:

- Waves
- Graphs
- Tables
- Page layouts

A collection of objects is called an "experiment" and is stored in an experiment file. When you open an experiment, Igor recreates the objects that comprise it.

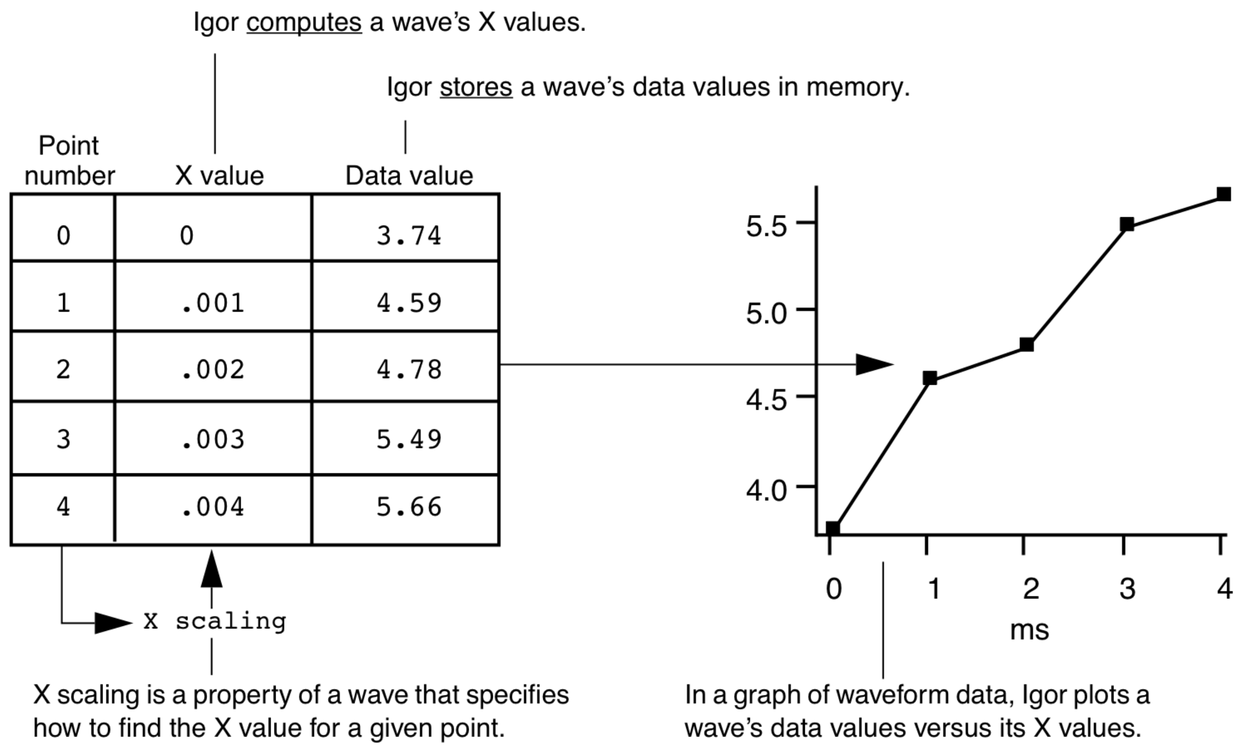
Waves - The Key Igor Concept

We use the term "wave" to describe the Igor object that contains an array of numbers. Wave is short for "waveform". The wave is the most important Igor concept.

Igor was originally designed to deal with waveform data. A waveform typically consists of hundreds to thousands of values measured at evenly spaced intervals of time. Such data is usually acquired from a digital oscilloscope, scientific instrument or analog-to-digital converter card.

The distinguishing trait of a waveform is the uniform spacing of its values along an axis of time or other quantity. An Igor wave has an important property called "X scaling" that you set to specify the spacing of your data. Igor stores the Y component for each point of a wave in memory but it computes the X component based on the wave's X scaling.

In the following illustration, the wave consists of five data points numbered 0 through 4. The user has set the wave's X scaling such that its X values start at 0 and increment by 0.001 seconds per point. The graph displays the wave's stored data values versus its computed X values.



Igor computes a wave's X values.
Igor stores a wave's data values in memory.
X scaling
X scaling is a property of a wave that specifies how to find the X value for a given point.
In a graph of waveform data, Igor plots a wave's data values versus its X values.
5.5
5.0
4.5
4.0
0
1
2
3
4
ms

Point number	X value	Data value
0	0	3.74
1	.001	4.59
2	.002	4.78
3	.003	5.49
4	.004	5.66

Waves can have from one to four dimensions and can contain either numeric or text data.

Igor is also capable of dealing with data that does not fit the waveform metaphor. We call this XY data. Igor can treat two waves as an XY pair. In an XY pair, one wave supplies an X component and another wave supplies a Y component for each point in the pair.

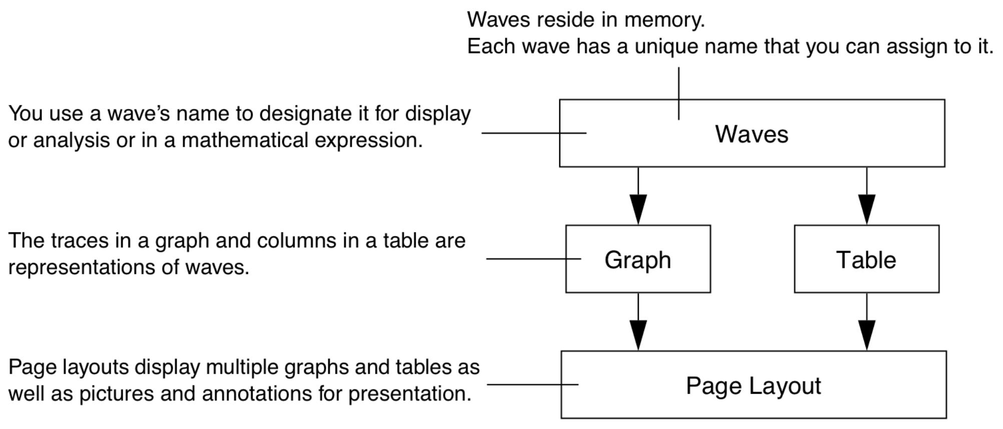
A few analysis operations, such as Fourier transforms, inherently work only on waveform data. They take a wave's X scaling into account.

Other operations work equally well on waveform or XY data. Igor can graph either type of data and its powerful curve fitting works on either type.

Most users create waves by loading data from a file. You can also create waves by typing in a table, evaluating a mathematical expression, acquiring from a data acquisition device, and accessing a database.

How Objects Relate

This illustration shows the relationships among Igor's basic objects. Waves are displayed in graphs and tables. Graphs and tables are displayed in page layouts. Although you can display a wave in a graph or table, a wave does not need to be displayed to exist.


Waves reside in memory.
Each wave has a unique name that you can assign to it.
You use a wave's name to designate it for display or analysis or in a mathematical expression.
Waves
The traces in a graph and columns in a table are representations of waves.
Graph
Table
Page layouts display multiple graphs and tables as well as pictures and annotations for presentation.
Page Layout

Each object has a name so that it can be referenced in an Igor command. You can explicitly set an object's name or accept a default name created by Igor.

Graphs are used to visualize waves and to generate high-quality printouts for presentation. The traces in a graph are representations of waves. If you modify a wave, Igor automatically updates graphs. Igor labels the axes of a graph intelligently. Tick marks never run into one another and are always "nice" values no matter how you zoom in or pan around.

In addition to traces representing waveform or XY data, a graph can display an image or a contour plot generated from 2D data.

Tables are used to enter, inspect or modify wave data. A table in Igor is not the same as a spreadsheet in other graphing programs. A column in a table is a *representation* of the contents of a wave. The wave continues to exist even if you remove it from the table or close the table entirely.

Page layouts permit you to arrange multiple graphs and tables as well as pictures and annotations for presentation. If you modify a graph or table, either directly or indirectly by changing the contents of a wave, Igor automatically updates its representation in a layout.

Both graphs and layouts include drawing tools for adding lines, arrows, boxes, polygons and pictures to your presentations.

More Objects

Here are some additional objects that you may encounter:

- Numeric and string variables
- Data folders
- Notebooks
- Control panels
- 3D plots
- Procedures

A numeric variable stores a single number and a string variable stores a text string. Numeric and string variables are used for storing bits of data for Igor procedures (described below).

A data folder can contain waves, numeric variables, string variables and other data folders. Data folders provide a way to keep a set of related data, such as all of the waves from a particular run of an experiment, together and separate from like-named data from other sets.

A notebook is like a text-editor or word-processor document. You can use a notebook to keep a log of results or to produce a report. Notebooks are also handy for viewing Igor technical notes or other text documentation.

A control panel is a window containing buttons, checkboxes and other controls and readouts. A control panel is created by an Igor user to provide a user interface for a set of procedures.

A 3D plot displays three-dimensional data as a surface, a scatter plot, or a path in space.

A procedure is a programmed routine that performs a task by calling Igor's built-in operations and functions and other procedures. Procedures range from very simple to very complex and powerful. You can run procedures written by WaveMetrics or by other Igor users. If you are a programmer or want to learn programming, you can learn to write your own Igor procedures to automate your work.

Igor's Toolbox

Igor's toolbox includes a wide range of built-in routines. You can extend it with user-defined procedures written in Igor itself and separately-compiled Igor extensions (plug-ins) that you obtain from WaveMetrics, from a colleague, from a third-party, or write yourself.

Built-In Routines

Each of Igor's built-in routines is categorized as a function or as an operation.

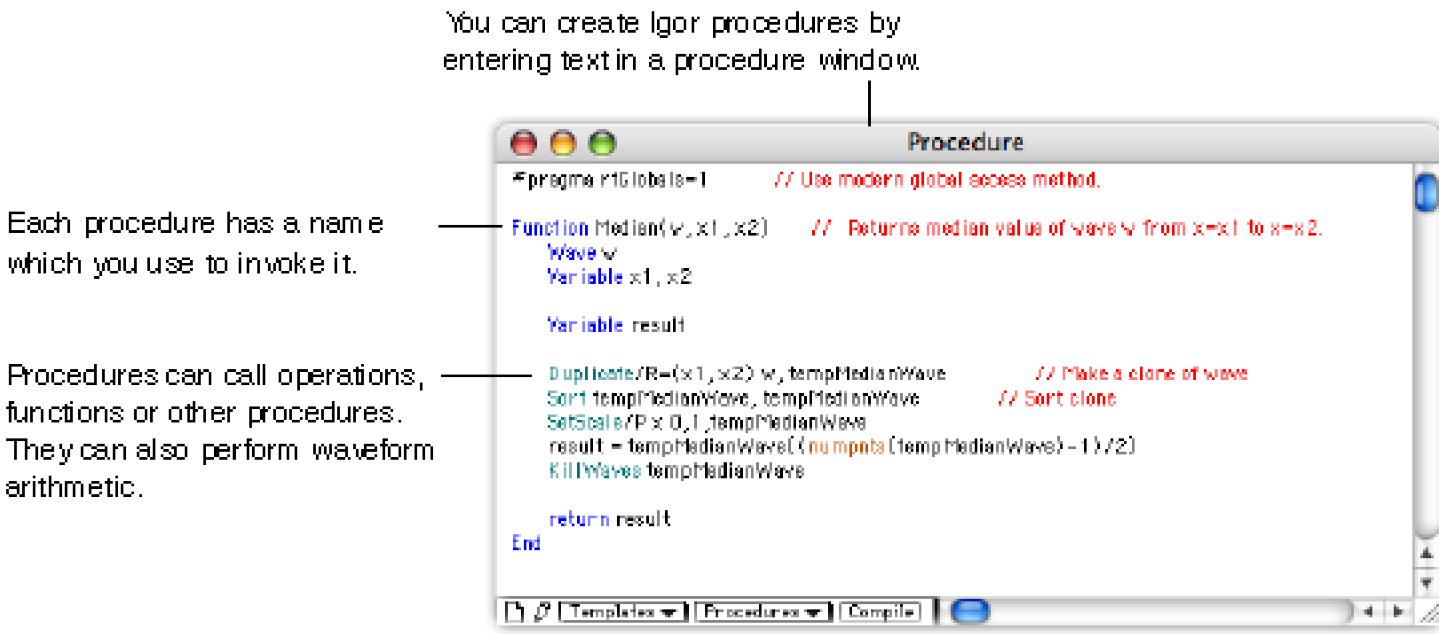
A built-in function is an Igor routine, such as sin, exp or ln, that directly returns a result. A built-in operation is a routine, such as Display, FFT or Integrate, that acts on an object and may create new objects but does not directly return a result.

The best way to get a sense of the scope of Igor's built-in routines is to open the reference volume of the PDF manual (choose Help->Manual) and scan the sections "Built-in Operations by Category" and "Built-in Functions by Category".

For getting reference information on a particular routine it is usually most convenient to choose Help->Command Help and use the Igor Help Browser.

User-Defined Procedures

A user-defined procedure is a routine written in Igor's built-in programming language by entering text in a procedure window. It can call upon built-in or external operations and functions as well as other user-defined procedures to manipulate Igor objects. Sets of procedures are stored in procedure files.


You can create Igor procedures by entering text in a procedure window.
Each procedure has a name which you use to invoke it.
Procedures can call operations, functions or other procedures. They can also perform waveform arithmetic.
Procedure
Templates
Procedures
Compile

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

Function Median(w,x1,x2)	//  Returns median value of wave w from x=x1 to x=x2.
	Wave w
	Variable x1, x2

	Variable result

	Duplicate/R=(x1,x2) w, tempMedianWave		// Make a clone of wave
	Sort tempMedianWave, tempMedianWave		// Sort clone
	SetScale/P x 0,1,tempMedianWave
	result = tempMedianWave((numpnts(tempMedianWave)-1)/2)
	KillWaves tempMedianWave

	return result
End
```

Igor Extensions

An "extension" is a "plug-in" - a piece of external C or C++ code that adds functionality to Igor. We use the terms "external operation" or "XOP" and "external function" or "XFUNC" for operations and functions added by extensions. An extension can add menus, dialogs and windows to Igor as well as operations and functions.

To create an extension, you must be a programmer and you need the optional Igor XOP Toolkit. See Creating Igor Extensions.

Although creating an extension is a job for a C or C++ programmer, anyone can use an extension. The Igor installer automatically installs commonly used extensions in "Igor Pro Folder/Igor Extensions". These extensions are available for immediate use. An example is the Excel file loader accessible through Data->Load Waves.

Less commonly used extensions are installed in "Igor Pro Folder/More Extensions". Available extensions are described under External Operations Index. To activate an extension, see Activating Extensions.

Igor's User Interface

Igor uses a combination of the familiar graphical user interface and a command line interface. This approach gives Igor both ease-of-use and programmability.

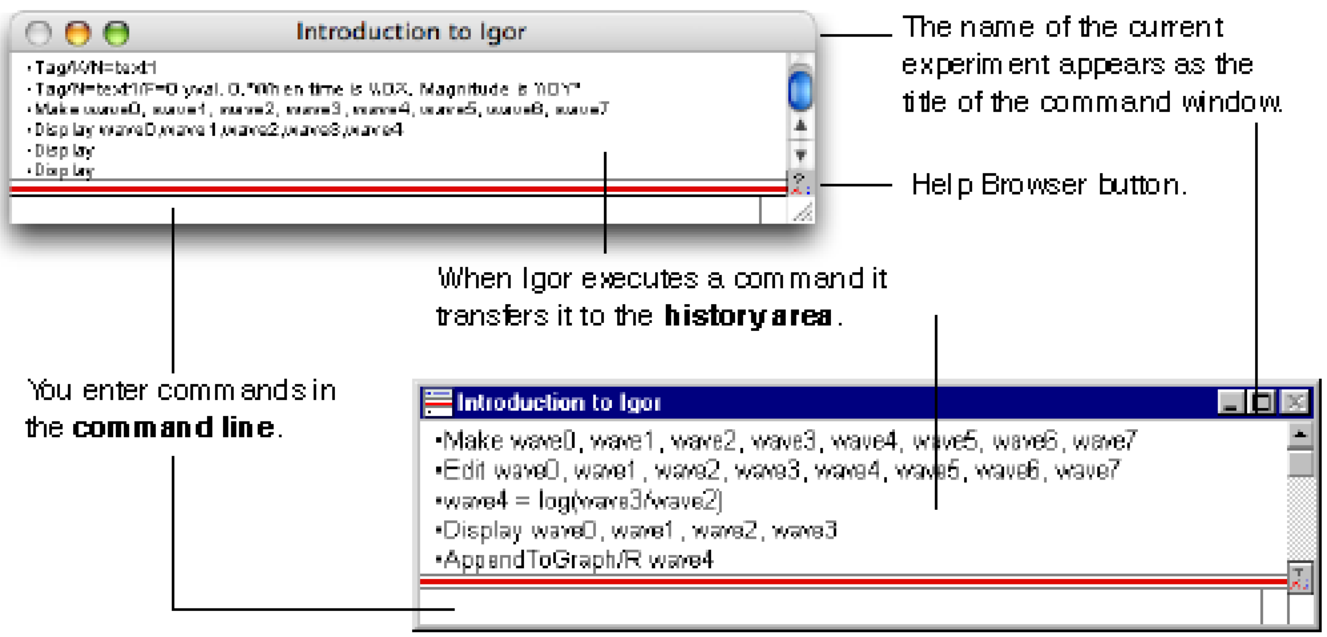
The job of the user interface is to allow you to apply Igor's operations and functions to objects that you create. You can do this in three ways:

- Via menus and dialogs
- By typing Igor commands directly into the command line
- By writing Igor procedures

The Command Window

The command window is Igor's control center. It appears at the bottom of the screen.

At the bottom of the command window is the command line. Above the red divider is the history area where executed commands are stored for review. Igor also uses the history area to report results of analyses like curve-fitting or waveform statistics.



Introduction to Igor
The name of the current experiment appears as the title of the command window.
Help Browser button.
When Igor executes a command it transfers it to the **history area**.
You enter commands in the **command line**.
Introduction to Igor

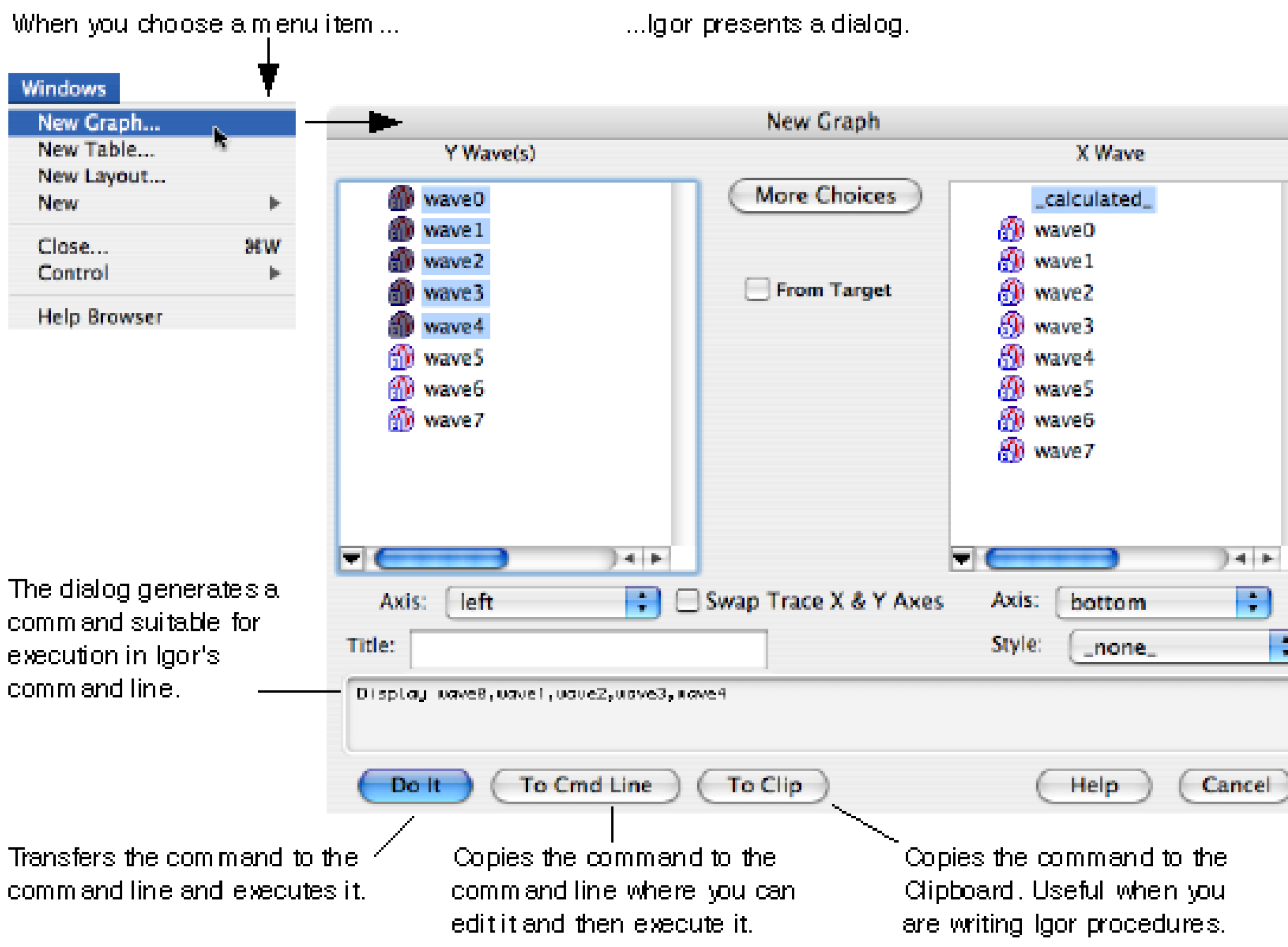
```
•Tag/K/N=text1
•Tag/N=text1/F=0 yval, 0,"When time is \\OX, Magnitude is \\OY"
•Make wave0, wave1, wave2, wave3, wave4, wave5, wave6, wave7
•Display wave0,wave1,wave2,wave3,wave4
•Display
•Display
```

```
•Make wave0, wave1, wave2, wave3, wave4, wave5, wave6, wave7
•Edit wave0, wave1, wave2, wave3, wave4, wave5, wave6, wave7
•wave4 = log(wave3/wave2)
•Display wave0, wave1, wave2, wave3
•AppendToGraph/R wave4
```

Menus, Dialogs and Commands

Menus and dialogs provide easy access to the most commonly-used Igor operations.

When you choose a menu item associated with an Igor operation, Igor presents a dialog. As you use the dialog, Igor generates a command and displays it in the **command box** near the bottom of the dialog. When you click the Do It button, Igor transfers the command to the command line where it is executed.


When you choose a menu item ...
...Igor presents a dialog.
Windows
New Graph...
New Table...
New Layout...
New
Close... ⌘W
Control
Help Browser
New Graph
Y Wave(s)
X Wave
wave0
wave1
wave2
wave3
wave4
wave5
wave6
wave7
calculated
More Choices
From Target
Axis: left
Swap Trace X & Y Axes
Axis: bottom
Title:
Style: _none_
Do It
To Cmd Line
To Clip
Help
Cancel
The dialog generates a command suitable for execution in Igor's command line.
Transfers the command to the command line and executes it.
Copies the command to the command line where you can edit it and then execute it.
Copies the command to the Clipboard. Useful when you are writing Igor procedures.

```
Display wave0,wave1,wave2,wave3,wave4
```

As you get to know Igor, you will find that some commands are easier to invoke from a dialog and others are easier to enter directly in the command line.

There are some menus and dialogs that bypass the command line. Examples are the Save Experiment and Open Experiment items in the File menu.

Using Igor for Heavy-Duty Jobs

If you generate a lot of raw data you may find it worthwhile to learn how to put Igor to heavy-duty use. It is possible to automate some or all of the steps involved in loading, processing and presenting your data. To do this, you must learn how to write Igor procedures.

Igor includes a built-in programming environment that lets you manipulate waves, graphs, tables and all other Igor objects. You can invoke built-in operations and functions from your own procedures to build higher-level operations customized for your work.

Learning to write Igor procedures is easier than learning a lower-level language such as FORTRAN or C. The Igor programming environment is interactive so you can write and test small routines and then put them together piece-by-piece. You can deal with very high level objects such as waves and graphs but you also have fine control over your data. Nonetheless, it is still programming. To master it requires an effort on your part.

The Igor programming environment is described under Programming Overview.

You can also learn about Igor programming by examining the WaveMetrics Procedures and example experiments that were installed on your hard disk.

Igor Documentation

Igor includes an extensive online help system and a comprehensive PDF manual.

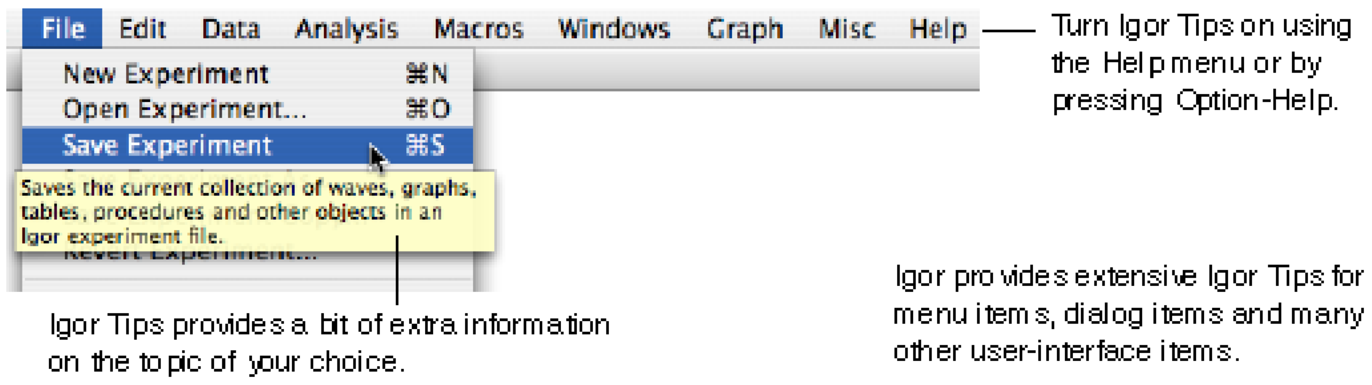
The online help provides guided tours, context-sensitive tips, general usage information for all aspects of Igor, and reference information for Igor operations, functions and keywords.

The PDF manual contains mostly the same information except for the context-sensitive tips.

The PDF manual, being in book format, is better organized for linear reading while the online help is usually preferred for reference information.

Igor Tips (Macintosh only)

Igor Tips present brief explanations of menus, dialogs and other user interface items. You turn Igor Tips on and off using the Help menu. When on, tips appear as you move your mouse over icons, menu items and dialog items.


File
Edit
Data
Analysis
Macros
Windows
Graph
Misc
Help
New Experiment ⌘N
Open Experiment... ⌘O
Save Experiment ⌘S
Saves the current collection of waves, graphs, tables, procedures and other objects in an Igor experiment file.
Turn Igor Tips on using the Help menu or by pressing Option-Help.
Igor provides extensive Igor Tips for menu items, dialog items and many other user-interface items.
Igor Tips provides a bit of extra information on the topic of your choice.

Status Line Help, Tool Tips and Context-Sensitive Help (Windows only)

Igor's Windows help system provides three ways to get immediate help for an icon, a menu item, or a dialog item.

Status line help automatically shows brief descriptions of menu items and icons in the status line area at the bottom of the main Igor Pro window.

Tool tips provide very brief descriptions of icons when you point the cursor at the item.

Context-sensitive help displays a pop-up window containing information about the menu item, icon or dialog item of interest. Context-sensitive help is accessed as follows:

Menus and Icons:	Press Shift+F1 to get the question-mark cursor and click the item of interest.
Dialogs:	Click the question-mark button in the upper-right corner of dialog to get the question-mark cursor , then click a dialog item.

The Igor Help System

The Help menu provides access to Igor's help system, primarily through the Igor Help Browser.

- Use the Help menu or press Help (*Macintosh*) or F1 (*Windows*) to display the Igor Help Browser.

- Use the Igor Help Browser Help Topics tab to browse help topics.
- Use the Igor Help Browser Shortcuts tab to get a list of handy shortcuts and techniques.
- Use the Igor Help Browser Command Help tab to get reference information on Igor operations and functions. You can also right-click operation and function names in Igor windows to access the reference help.
- Use the Igor Help Browser Search tab to search Igor help, procedure and example files.

Most of the information displayed by the help browser comes from help files that are automatically loaded at launch time. The Windows->Help Windows submenu provides direct access to these help files.

The Igor Manual

The Igor PDF manual resides in "Igor Pro Folder/Manual". You can access it by choosing Help->Manual.

The manual consists of five volumes and an index.

Volume I contains the Getting Started material, including the Guided Tour of Igor Pro.

Volumes II and III contain general background and usage information for all aspects of Igor other than programming.

Volume IV contains information for people learning to do Igor programming.

Volume V contains reference information for Igor operations, functions and keywords.

Hard copy of the manual can be purchased from <http://www.lulu.com/wavemetrics>.

Learning Igor

To harness the power of Igor, you need to understand its fundamental concepts. Some of these concepts are familiar to you. However, Igor also incorporates a few concepts that will be new to you and may seem strange at first. The most important of these are waves and experiments.

In addition to this introduction, the primary resources for learning Igor are:

- The Guided Tour of Igor Pro

  The guided tour shows you how to perform basic Igor tasks step-by-step and reinforces your understanding of basic Igor concepts along the way. It provides an essential orientation to Igor and is highly-recommended for all Igor users.
- The Igor Pro PDF manual and online help files

  You can access the PDF manual through Igor's Help menu or by opening it directly from the Manual folder of the Igor Pro Folder where Igor is installed.

  You can access the help files through the Igor Help Browser (choose Help->Igor Help Browser) or directly through the Windows->Help submenu.
- The "Examples" experiments

  The example experiments illustrate a wide range of Igor features. They are stored in the Examples folder in the Igor Pro Folder. You can access them using the File->Example Experiments submenu or directly from the Examples folder of the Igor Pro Folder where Igor is installed.

You will best learn Igor through a combination of doing the guided tour, reading select parts of the manual (see suggestions following the guided tour), and working with your own data.

Getting Hands-On Experience

This introduction has presented an overview of Igor, its main constituent parts, and its basic concepts. The next step is to get some hands-on experience and reinforce what you have learned by doing the Guided Tour of Igor Pro.

• Guided Tour of Igor Pro

In this tour we take a look at the main functions of Igor by stepping through some typical operations. Our goal is to orient you so that you can comfortably explore the program on your own. You will benefit most from this tour if you actually do the instructed operations on your computer as you read this tour. Screen shots are provided to keep you synchronized with the tour.

Terminology

If you have read Introduction to Igor Pro, you already know these terms:

Experiment:	The entire collection of data, graphs and other windows, program text and whatnot that make up the current Igor environment or workspace.
Wave:	Short for waveform, this is basically a named column of data with optional extra information.
Name:	Because Igor contains a built-in programming and data transformation language, each object must have a unique name.
Command:	This is a line of text that instructs Igor to do some task. Igor is command-driven so that it can be easily programmed.

About the Tour

This tour consists of three sections: "Guided Tour 1 - General Tour", "Guided Tour 2 - Data Analysis" and "Guided Tour 3 - Histograms and Curve Fitting".

The General Tour is a rambling exploration intended to introduce you to the way things work in Igor and give you a general orientation. This tour takes 2 to 4 hours but does not have to be performed in one sitting.

The second and third tours guide you through Igor's data analysis facilities including simple curve fitting.

When you've completed the first tour you may want to explore freely on your own before starting the second tour.

Note: **You can execute the example commands in this help file by selecting the line or lines and pressing Control-Enter (*Macintosh*) or Ctrl+Enter (*Windows*).**

• Guided Tour 1 - General Tour

In this exercise, we will generate data in three ways (typing, loading, and synthesizing) and we will generate graph, table, and page layout windows. We will jazz up a graph and a page layout with a little drawing and some text annotation. At the end we will explore some of the more advanced features of Igor Pro.

Launching Igor Pro

The Igor Pro application is typically installed in:

```
/Applications/Igor Pro Folder (Macintosh)
C:\Program Files\WaveMetrics\Igor Pro Folder (Windows)
```

1. **Double-click the Igor Pro application file on your hard disk.**

   On Windows you can also start Igor using the Start menu.

   If Igor was already running, choose the File->New Experiment menu item.

2. **Use the Misc menu to turn preferences off.**

   Turning preferences off ensures that the tour works the same for everyone.

Entering Data

1. **If a table window is showing, click on it to bring it to the front.**

   When Igor starts up, it creates a new blank table unless this feature is turned off in the Miscellaneous Settings dialog. If the table is not showing, perform the following two steps:

   1a. **Choose the Windows->New Table menu item.**

   The New Table dialog appears.

   1b. **Click the Do It button.**

   A new blank table is created.

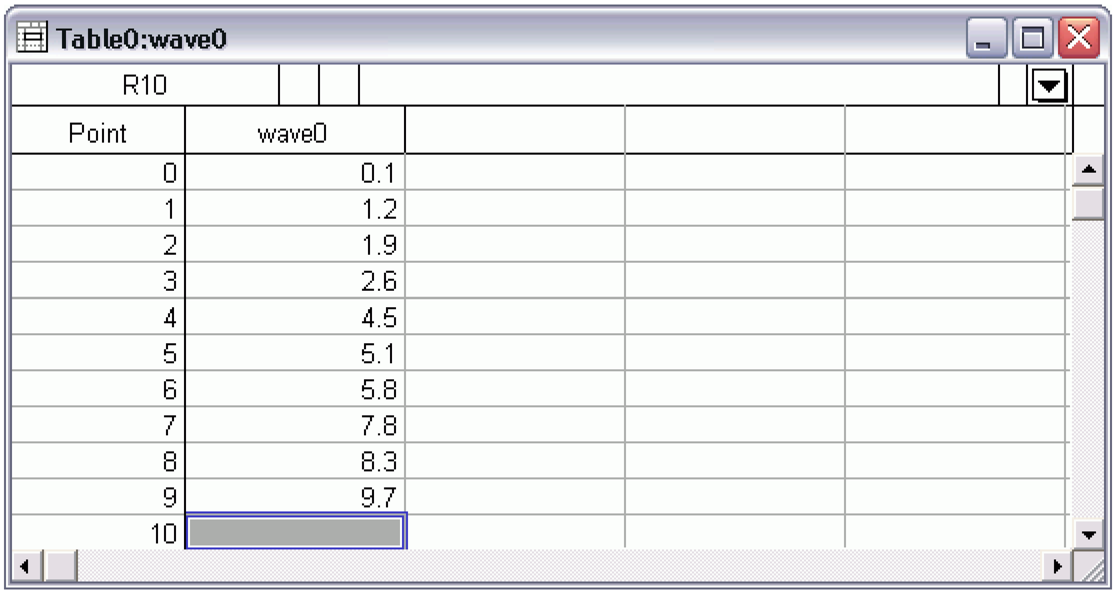
2. **Type "0.1" and then press Return or Enter on your keyboard.**

   This creates a wave named “wave0” with 0.1 for the first point. Entering a value in the first row (point 0) of the first blank column automatically creates a new wave.

3. **Type the following numbers, pressing Return or Enter after each one:**

   1.2
   1.9
   2.6
   4.5
   5.1
   5.8
   7.8
   8.3
   9.7

   Your table should look like this:



Point	wave0
0	0.1
1	1.2
2	1.9
3	2.6
4	4.5
5	5.1
6	5.8
7	7.8
8	8.3
9	9.7
10	

4. **Click in the first cell of the first blank column**
5. **Enter the following numbers in the same way:**

   -0.12
   -0.08
   1.3
   1
   0.54
   0.47
   0.44
   0.2
   0.24

0.13

6. **Choose Data->Rename.**
7 **Click "wave0" in the list and then click the arrow icon.**
8. **Replace "wave0" with "time".**

   Notice that you can't use the name "time" because it is the name of a built-in string function. We apologize for usurping such a common name.
9. **Change the name to "timeval".**
10. **Select "wave1" from the list, click the arrow icon, and type "yval".**
11. **Click Do It.**

    Notice the column headers in the table have changed to reflect the name changes.

Making a Graph

1. **Choose the Windows->New Graph menu item.**

   The New Graph dialog will appear. This dialog comes in a simple form that most people will use and a more complex form that allows you to create complex multiaxis graphs in one step.
2. **If you see a button labeled Fewer Choices, click it.**

   The button is initially labeled More Choices because the simpler form of the dialog is the default.
3. **From the Y Wave(s) list, pick "yval".**
4. **From the X Wave list, pick "timeval".**
5. **Click Do It.**

   A simple graph is created.

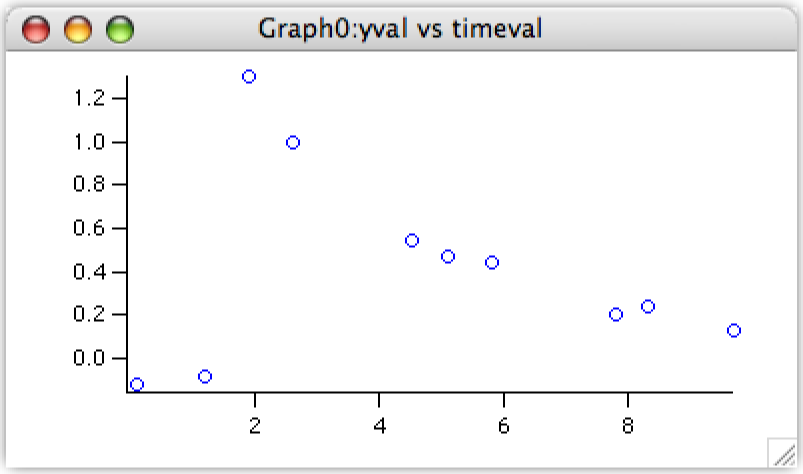
Touching up a Graph

1. **Position the cursor directly over the trace in the graph and double-click.**

   The Modify Trace Appearance dialog appears. You could also have selected the corresponding menu item from the Graph menu.

   Note: The Graph menu appears only when a graph is the target window. The target window is the window that menus and dialogs act on by default.
2. **Choose Markers from the Mode pop-up menu.**
3. **Select the open circle from the pop-up menu of markers.**
4. **Set the marker color to blue.**
5. **Click Do It.**

   Your graph should now look like this:



6. **Position the cursor over the bottom axis line.**

   The cursor changes to this shape: ↕. This indicates the cursor is over the axis and also that you can offset the axis (and the corresponding plot area edge) to a new position.

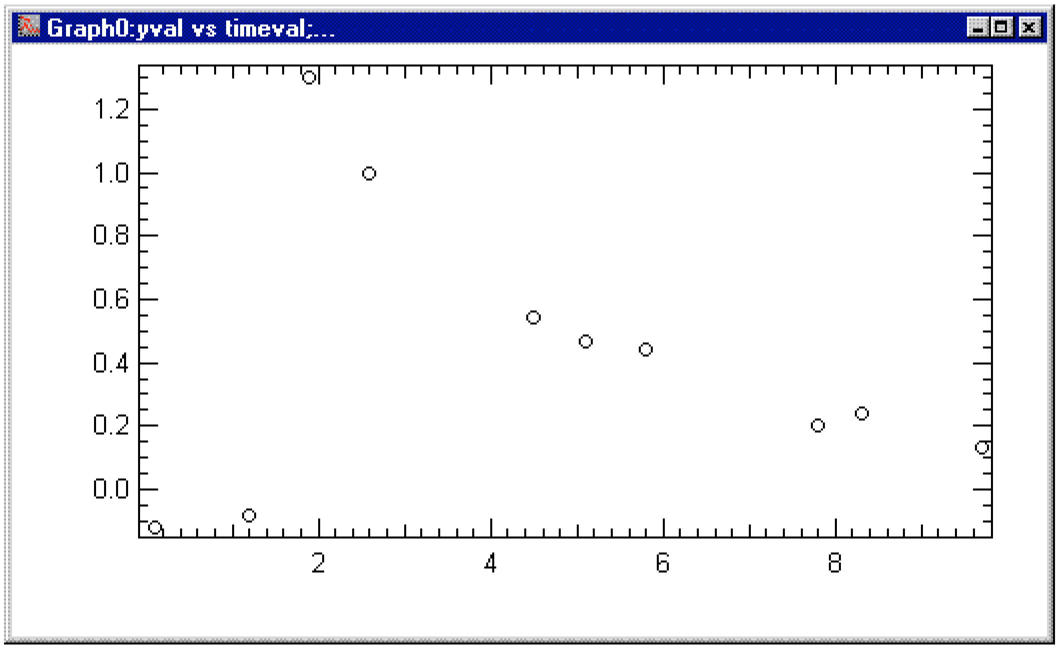
7. **Double click directly on the axis.**

   The Modify Axis dialog appears. If another dialog appears, click cancel and try again, making sure the ↕ cursor is showing.

   Note the Live Update checkbox in the top/right corner of the Modify Axis dialog. When it is checked, changes that you make in the dialog are immediately reflected in the graph. When it is unchecked, the changes appear only when you click Do It. The Modify Axis dialog is the only one with a Live Update checkbox.

8. **If it is not already showing, click the Axis tab.**
9. **Choose On from the Mirror Axis pop-up.**
10. **Click the Auto/Man Ticks tab.**
11. **Click the Minor Ticks checkbox so it is checked.**
12. **Click the Ticks and Grids tab.**
13. **Choose Inside from the Location pop-up.**
14. **Choose the left axis from the Axis pop-up menu in the top-left corner of the dialog and then repeat steps 8 through 13.**
15. **Click Do It.**

    Your graph should now look like this:



16. **Again double click the bottom axis.**

    The Modify Axis dialog appears again.

17. **Click the Axis tab.**
18. **Uncheck the Standoff box.**
19. **Choose the left axis from the Axis pop-up menu and repeat step 18.**
20. **Click Do It.**

    Notice that some of the markers now overlap the axes. The axis standoff setting pushes out the axis so that markers and traces do not overlap the axis. You can use Igor's preferences to ensure this and other settings default to your liking, as explained below.

21. **Double-click one of the tick mark labels (e.g., "6") on the bottom axis.**

    The Modify Axis dialog reappears, this time with the Axis Range tab showing. If another dialog or tab appears, cancel and try again, making sure to double click one of the tick mark labels on the bottom axis.

22. **Choose "Round to nice values" from the pop-up menu that initially reads "Use data limits".**
23. **Choose the left axis from the Axis pop-up menu and repeat step 22.**
24. **Click Do It.**

    Notice that the limits of the axes now fall on "nice" values.

Adding A Legend

1. **Choose the Graph->Add Annotation menu item.**

   The Add Annotation dialog appears.

2. **Click the Text tab if it is not already selected.**
3. **Choose Legend from the pop-up menu in the upper-left hand corner.**

   Igor inserts text to create a legend in the Annotation text entry area. The Preview area shows what the annotation will look like. Note that the text "\s(yval)" generates the symbol for the yval wave. This is an "escape sequence," which creates special effects such as this.

4. **In the Annotation area, change the second instance of "yval" to "Magnitude".**
5. **Click the Frame tab and choose Box from the Annotation Frame pop-up menu.**
6. **Choose Shadow from the Border pop-up menu.**
7. **Click the Position tab and choose Right Top from the Anchor pop-up menu.**

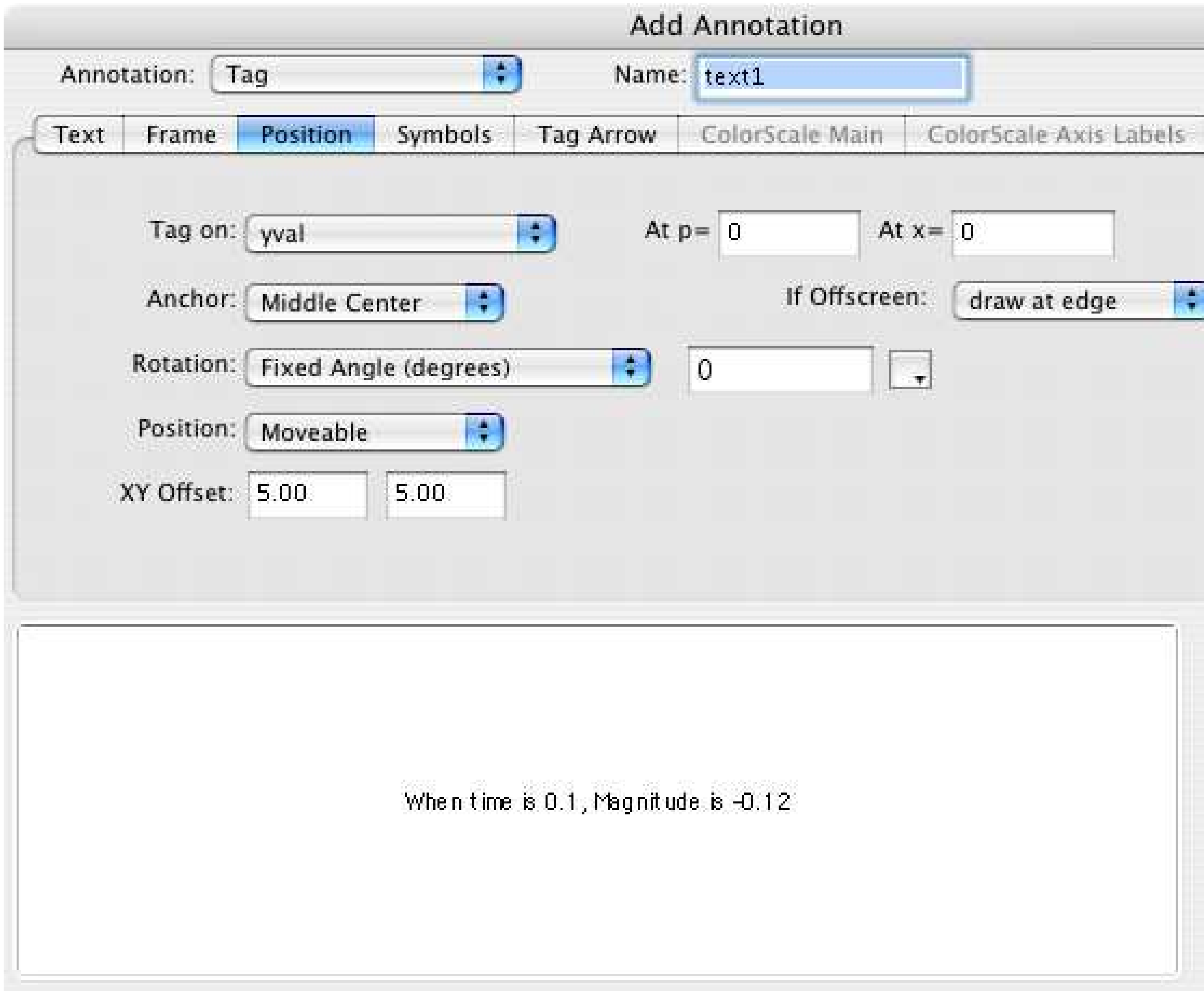
   Specifying an Anchor point helps Igor keep the annotation in the best location as you make the graph bigger or smaller.
8. **Click Do It.**

Adding A Tag

1. **Choose the Graph->Add Annotation menu item.**
2. **Choose Tag from the pop-up menu in the upper-left hand corner.**
3. **In the Annotation area, type "When time is ".**
4. **Choose Attach point X value from the Dynamic pop-up menu in the Insert area of the dialog.**

   Igor inserts the "\0X" escape code into the Annotation text entry area.
5. **Type ", Magnitude is ".**
6. **Choose Attach point Y value from the Dynamic pop-up menu.**
7. **Switch to the Frame tab and choose None from the Annotation Frame pop-up menu.**
8. **Switch to the Tag Arrow tab and choose Arrow from the Connect Tag to Wave With pop-up menu.**
9. **Click the Position tab and choose "Middle center" from the Anchor pop-up menu.**

   The dialog should now look like this:



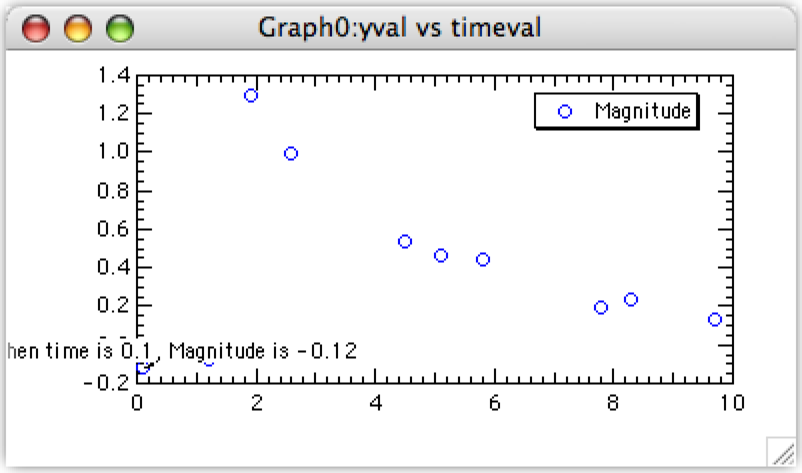
Annotation: Tag
Name: text1

Tag on: yval
At p= 0
At x= 0
Anchor: Middle Center
If Offscreen: draw at edge
Rotation: Fixed Angle (degrees) 0
Position: Moveable
XY Offset: 5.00 5.00

When time is 0.1, Magnitude is -0.12

10. Click Do It.

Your graph should now look like this:



The tag is attached to the first point. An arrow is drawn from the center of the tag to the data point but you can't see it because it is hidden by the tag itself.

11. **Position the cursor over the text of the tag.**

    The cursor changes to a hand. This indicates you can reposition the tag relative to the data point it is attached to.

12. **Click and drag the tag up and to the right about 1 cm.**

    You can now see the arrow.

13. **With the cursor over the text of the tag, press Option (*Macintosh*) or Alt (*Windows*).**

    The cursor changes to this shape: ▭. (You may need to nudge the cursor slightly to make it change.)

14. **With the key still down, click and drag the box cursor to a different data point.**

    The tag jumps to the new data point and the text is updated to show the new X and Y values. Option-drag (*Macintosh*) or Alt-drag (*Windows*) the tag to different data points to see their X and Y values..

    Notice that the tip of the arrow touches the marker. This doesn't look good, so let's change it:

15. **Double-click on the tag.**

    The Modify Annotation dialog appears.

16. **Click the Tag Arrow tab and change the Line/Arrow Standoff from "Auto" to "10".**

17. **Click the Change button.**

    The tip of the arrow now stops 10 points from the marker.

Using Preferences

If you have already set preferences to your liking and do not want to disturb them, you can skip this section.

1. **Use the Misc menu to turn preferences on.**
2. **Click the graph window if it is not already active.**
3. **Choose the Graph->Capture Graph Prefs menu item.**

   The Capture Graph Preferences dialog appears.

4. **Click the checkboxes for XY plot axes and for XY plot wave styles.**
5. **Click Capture Preferences.**
6. **Choose Windows->New Graph.**

7. **Choose "yval" as the Y wave and "timeval" as the X wave.**
8. **Click Do It.**

   The new graph is created with a style similar to the model graph.
9. **Press Option (*Macintosh*) or Alt (*Windows*) while clicking the close button of the new graph.**

   The new graph is killed without presenting a dialog.
10. **Choose Graph->Capture Graph Prefs.**
11. **Click the checkboxes for XY plot axes and for XY plot wave styles.**

12 **Click Revert to Defaults.**

13. **Use the Misc menu to turn preferences off.**

    We turn preferences off during the guided tour to ensures that the tour works the same for everyone. This is not something you would do during normal work.

<u>Making a Page Layout</u>

1. **Choose the Windows->New Layout menu item.**

   The New Page Layout dialog appears. The names of all tables and graphs are shown in the list.
2. **From the Objects to Lay Out list, pick Graph0.**
3. **Command-click (Macintosh) or Ctrl-click (Windows) on Table0.**
4. **Click Do It.**

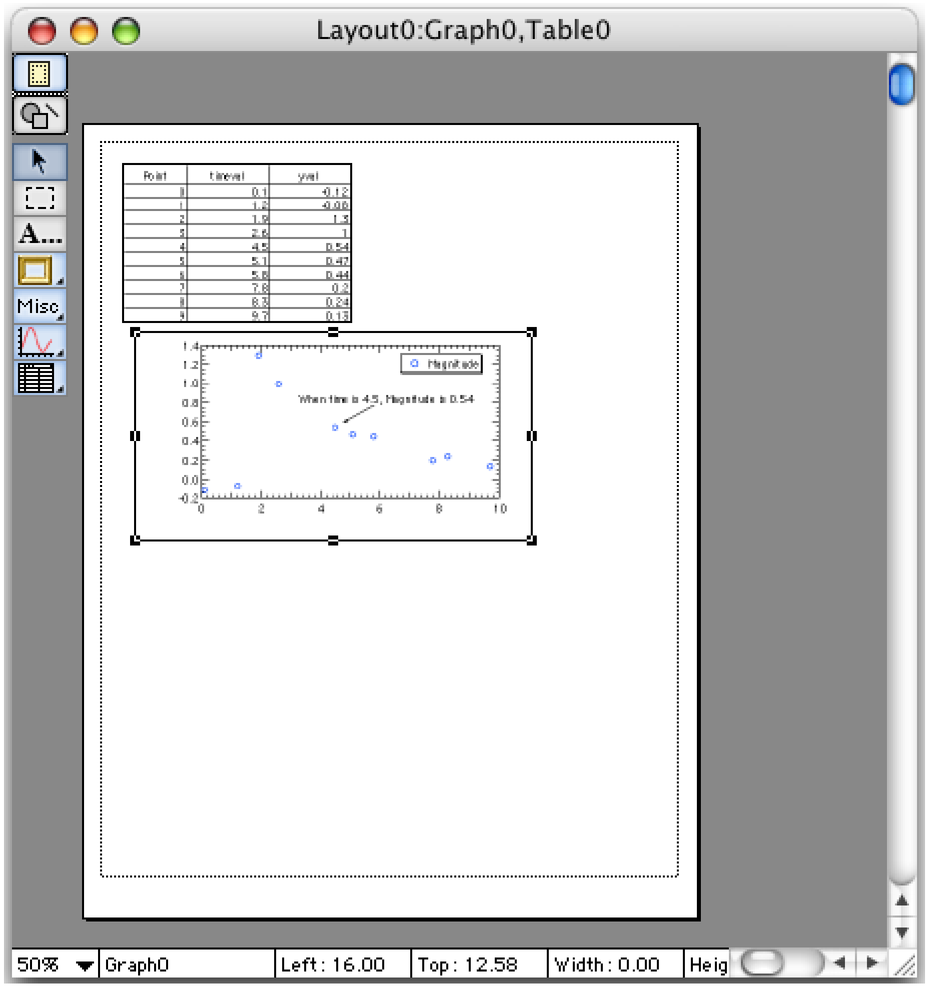
   A page layout window appears with a Table0 object on top of a Graph0 object.

   The layout initially shows objects at 50% but you may prefer to work at 100%. You can use the pop-up menu in the lower left corner of the window to change magnification.
5. **Click the Table0 object in the layout window.**

   The table object becomes selected, resize handles are drawn around the edge and the cursor becomes a hand when over the table.
6. **Click in the middle of the table and drag it so you can see the right edge of the table.**
7. **Position the cursor over the small black square (handle) in the middle of the right side of the table.**

   The cursor changes to a two headed arrow indicating you can drag in the direction of the arrows.
8. **Drag the edge of the table to the left until it is close to the edge of the third column of data.**

   You need only get close -- Igor snaps to the nearest grid line.
9. **In a similar fashion, adjust the bottom of the table to show all the data but without any blank lines.**
10. **Drag the graph and resize it to match the picture:**



Point	timeval	yval
0	0.1	-0.12
1	1.2	-0.06
2	1.9	1.3
3	2.6	1
4	4.5	0.54
5	5.1	0.47
6	5.8	0.44
7	7.8	0.2
8	8.3	0.24
9	9.7	0.13

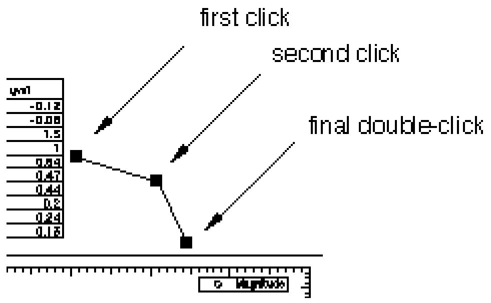
11. **Click on this icon in the tool palette:**

    This activates the drawing tools.

12. **Click on this icon in the drawing tool palette:**

    This is the polygon tool.

13. **Click once just to the right of the table, click again about 2 cm right and 1 cm down and finally double-click a bit to the right of the last click and just above the graph.**



    The double click exits the "draw polygon" mode and enters "edit polygon mode". If you wish to touch up the defining vertices of the polygon, do so now by dragging the "handles" (the square boxes at the vertices).

14. **Click on the Arrow tool in the palette.**

    This exits polygon edit mode.

15. **Click on the polygon to select it.**

16. **Click on the draw environment pop-up icon, , and choose At End in the Line Arrow submenu.**

17. **Click on this icon in the tool palette:**

    This is the operate icon. The drawing tools are replaced by the normal tools.

    We are finished with the page layout for now.

18. **Choose Windows->Send To Back.**

Saving Your Work

1. **Identify or create a folder on your hard disk for saving your Igor files.**

   For example, you might create a folder for your Igor files in your user folder.

   Don't save your Igor files in the Igor Pro folder as this complicates updating Igor and making backups.

2. **Choose File->Save Experiment.**

   The save file dialog appears.

3. **Make sure that Packed Experiment File is selected as the file format.**

4. **Type "Tour #1 a.pxp" in the name box.**

5. **Navigate to the folder where you want to keep your tour files.**

6. **Click Save.**

   The "Tour #1a.pxp" file contains all of your work in the current experiment, including waves that you created, graphs, tables and page layout windows.

   If you want to take a break, you can quit from Igor now.

Loading Data

Before loading data we will use a Notebook window to look at the data file.

0. **If you are returning from a break, launch Igor and open your "Tour #1 a.pxp" experiment file. Then turn off preferences using the Misc menu.**

   Opening the "Tour #1 a.pxp" experiment file restores the Igor workspace to the state it was in when you saved the file. You can open the experiment file by using the Open Experiment item in the File menu or by double-clicking the experiment file.

1. **Choose the File->Open File->Notebook menu item.**

2. **Navigate to the folder "Igor Pro Folder:Learning Aids:Sample Data" folder and open "Tutorial Data #1.txt."**

   A Notebook window showing the contents of the file appears. If desired, we could edit the data and then save it. For now we just observe that the file appears to be tab-delimited (tabs separate the columns) and contains names for the columns. Note that the name of the first column will conflict with the data we just entered and the other names have spaces in them.

3. **Click the close button or press Command-W (*Macintosh*) or Ctrl+W (*Windows*).**

   A dialog appears asking what you want to do with the window.

4. **Click the Kill button.**

   The term Kill means to "completely remove from the experiment". The file will be unharmed.

   Now we will actually load the data.

5. **Choose Data->Load Waves->Load Delimited Text.**

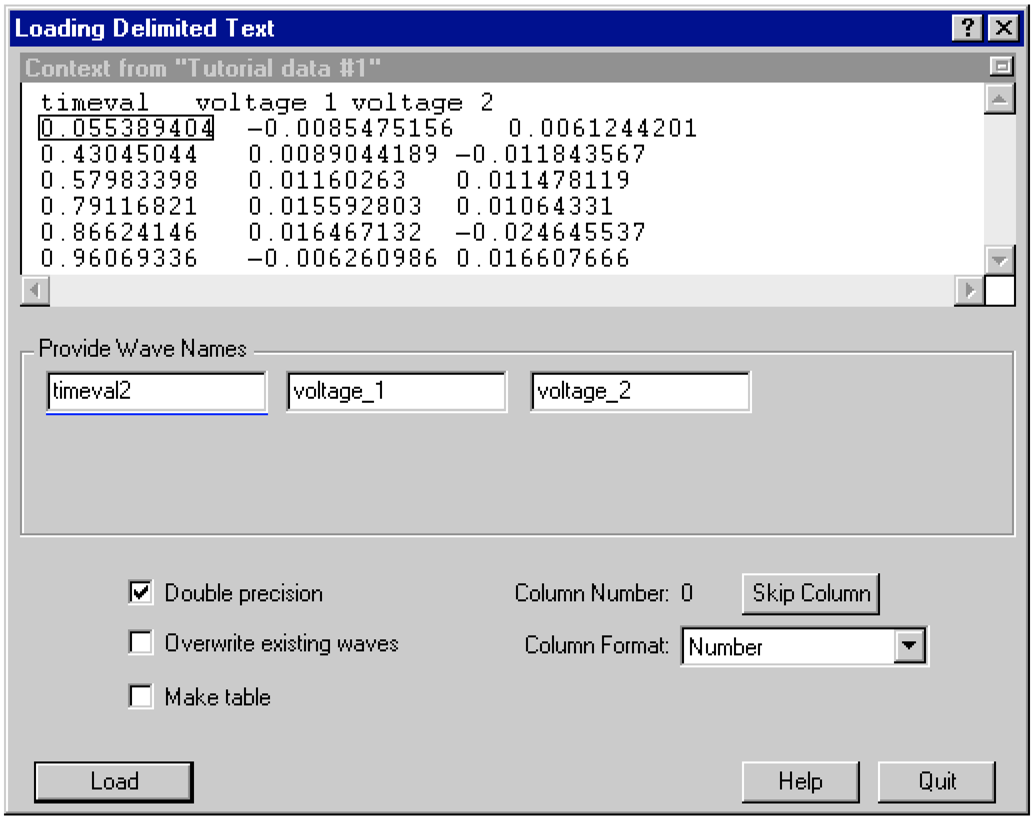
   An Open File dialog appears.

6. **Again choose "Tutorial Data #1.txt" and click Open.**

The Loading Delimited Text dialog appears. The name "timeval" is highlighted and an error message is shown. Observe that the names of the other two columns have been fixed up by replacing the spaces with underscore characters.

7. Change "timeval" to "timeval2".

The dialog should now look like this:



Loading Delimited Text

Context from "Tutorial data #1"

```
timeval    voltage 1 voltage 2
0.055389404   -0.0085475156    0.0061244201
0.43045044    0.0089044189 -0.011843567
0.57983398    0.01160263    0.011478119
0.79116821    0.015592803   0.01064331
0.86624146    0.016467132   -0.024645537
0.96069336    -0.006260986  0.016607666
```

Provide Wave Names: timeval2 | voltage_1 | voltage_2

[✓] Double precision

[ ] Overwrite existing waves

[ ] Make table

Column Number: 0

Column Format: Number

8. Click the Make Table box to select it and then click Load.

The data is loaded and a new table is created to show the data.

9. Click the close button of the new table window.

A dialog is presented asking if you want to create a recreation macro.

10. Click the No Save button.

The data we just loaded is still available in Igor. A table is just a way of viewing data and is not necessary for the data to exist.

The Load Delimited Text menu item that you used is a shortcut that uses default settings for loading delimited text. Later, when you load your own data files, choose Data->Load Waves->Load Waves so you can see all of the options.

Appending to a Graph

0. If necessary, click on Graph0 to bring it to the front.

The Graph menu is available only when the target window is a graph.

1. Choose the Graph->Append Traces to Graph menu item.

The Append Traces dialog appears. It is very similar to the New Graph dialog that you used to create the graph.

2. From the Y Wave(s) list, pick voltage_1 and voltage_2.

3. From the XWave list, pick timeval2.

4. **Click Do It.**

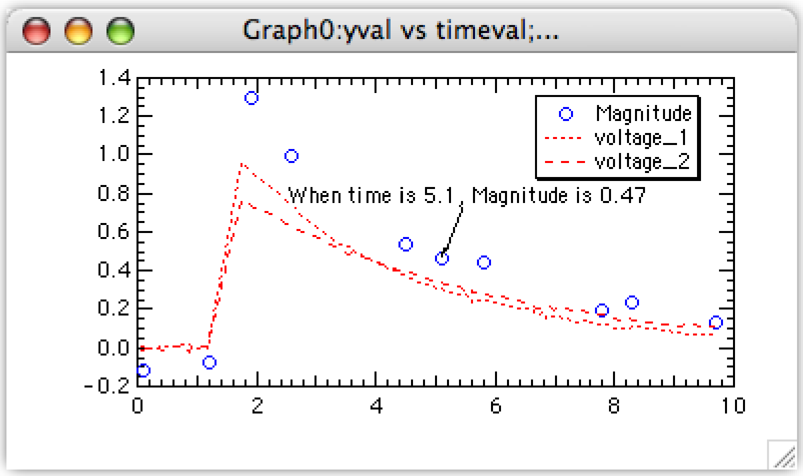
   Two additional traces are appended to the graph. Notice that they are also appended to the Legend.

5. **Position the cursor over one of the traces in the graph and double-click.**

   The Modify Trace Appearance dialog appears with the trace you clicked on already selected.

6. **If necessary, select voltage_1 from the list of traces.**
7. **Choose dashed line #2 from the Line Style pop-up menu.**
8. **Select voltage_2 from the list of traces.**
9. **Choose dashed line #3 from the Line Style pop-up menu.**
10. **Click Do It.**

    Your graph should now look like this:



Offsetting a Trace

1. **Position the cursor directly over the voltage_2 trace.**

   The voltage_2 trace has the longer dash pattern.

2. **Press and hold the mouse button for about 1 second.**

   An XY readout appears in the lower-left corner of the graph and the trace will now move with the mouse.

3. **With the mouse button still down, press Shift and drag the trace up about 1 cm and release.**

   The Shift key constrains movement to vertical or horizontal directions.

   You have added an offset to the trace. If desired, you could add a Tag to the trace indicating that it has been offset and by how much.

Unoffsetting a Trace

1. **Choose the Edit->Undo Trace Drag menu item.**

   You can undo many of the interactive operations on Igor windows if you do so before performing the next interactive operation.

2. **Choose Edit->Redo Trace Drag.**

   The following steps show how to remove an offset after it is no longer undoable.

3. **Double-click the voltage_2 trace.**

   The Modify Trace Appearance dialog will appear with voltage_2 selected. (If voltage_2 is not selected, click on it.) The Offset checkbox will be checked.

4. **Click on the Offset checkbox.**

   This turns offset off for the selected trace.

5. **Click on the Offset checkbox again.**

   The Trace Offset dialog appears showing the offset value you introduced by dragging.

6. **Click the Cancel button or press Escape.**

   The Offset checkbox should still be unchecked.

7. **Click Do It.**

   The voltage_2 trace is returned to its original position.

Drawing in a Graph

1. **If necessary click on Graph0 to bring it to the front.**

2. **Choose the Graph->Show Tools menu item or press Command-T (*Macintosh*) or Ctrl+T (*Windows*).**

   A toolbar is added to the graph. The second icon from the top ( ) is selected indicating that the graph is in drawing mode as opposed to normal (or "operate") mode.

3. **Click the top icon ( ) to go into normal mode.**

   Normal mode is for interacting with graph objects such as traces, axes and annotations. Drawing mode is for drawing lines, rectangles, polygons and so on.

4. **Click the second icon to return to drawing mode.**

5. **Press Option (*Macintosh*) or Alt (*Windows*) and hold down the mouse button while the cursor is in the draw environment icon (tree and grass).**

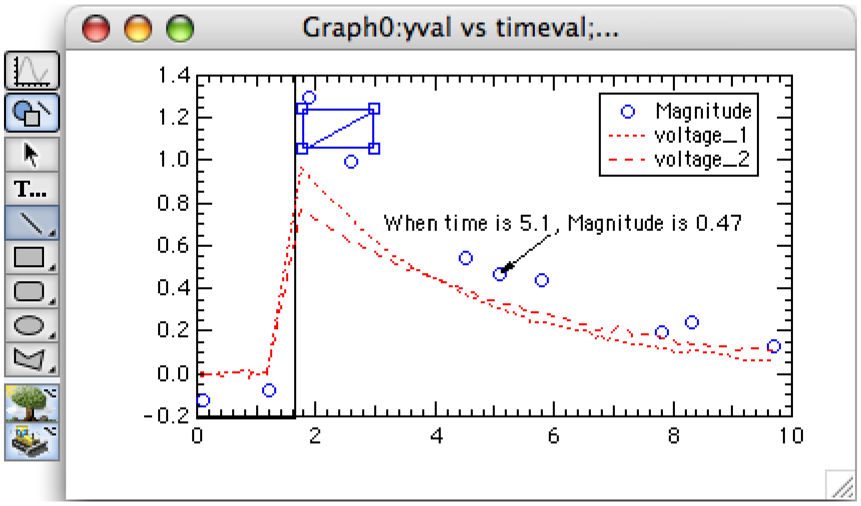
   A pop-up menu showing the available drawing layers and their relationship to the graph elements appears (the items in the menu are listed in back-to-front order).

6. **Choose UserBack from the menu.**

   We will be drawing behind the axes, traces and all other graph elements.

7. **Click the rectangle tool and drag out a rectangle starting at the upper left corner of the plot area (y= 1.4, x=0 on the axes) and ending at bottom of the plot area and about 1.5 cm in width (y= -0.2, x= 1.6).**

8. **Click on the line tool and draw a line as shown, starting at the left and ending at the right:**



9. **Click in the draw environment icon and choose At Start from the Line Arrow item.**
10. **Click the Text tool icon T... .**
11. **Click just to the right of the line you have just drawn.**

    The Create Text dialog appears.
12. **Type "Precharge".**
13. **From the Anchor pop-up menu, choose Left center.**
14. **Click Do It.**
15. **Click the graph's zoom button (*Macintosh*) or maximize button (*Windows*).**

    Notice how the rectangle and line expand with the graph. Their coordinates are measured relative to the plot area (rectangle enclosed by the axes).
16. **Click the graph's zoom button (*Macintosh*) or restore button (*Windows*).**
17. **Click the Arrow tool and then double click on the rectangle.**

    The Modify Rectangle dialog appears showing the properties of the rectangle.
18. **Enter "0" in the Thickness box in the Line Properties section.**

    This turns off the frame of the rectangle.
19. **Choose Light Gray from the Fill Mode pop-up menu.**
20. **Choose black from the Fore Color pop-up menu under the Fill Mode pop-up menu.**
21. **Click Do It.**

    Observe that the rectangle forms a gray area behind the traces and axes.
22. **Again, double click on the rectangle.**

    The Modify Rectangle dialog appears.
23. **From the X Coordinate pop-up menu, choose Axis Bottom.**

    The X coordinates of the rectangle will be measured in terms of the bottom axis - as if they were data values.
24. **Press Tab until the X0 box is selected and type "0".**
25. **Tab to the Y0 box and type "0".**
26. **Tab to the X1 box and type "1.6".**
27. **Tab to Y1 and type "1".**

    The X coordinates of the rectangle are now measured in terms of the bottom axis and the left side will be at zero while the right side will be at 1.6.

    The Y coordinates are still measured relative to the plot area. Since we entered zero and one for the Y coordinates, the rectangle will span the entire height of the plot area.
28. **Click Do It.**

    Notice the rectangle is nicely aligned with the axis and the plot area.
29. **Click in the operate icon, , to exit the drawing mode.**
30. **Press Option (*Macintosh*) or Alt (*Windows*), click in the middle of the plot area and drag about 2 cm to the right.**

    The axes are rescaled. Notice that the rectangle moved to align itself with the bottom axis.
31. **Choose Edit->Undo Scale Change.**

Making a Window Recreation Macro

1. **Click the graph's close button.**

   Igor presents a dialog which asks if you want to save a display recreation macro. The graph's name is "Graph0" so Igor suggests "Graph0" as the macro name.

2. **Click Save.**

   Igor generates a display recreation macro in the currently hidden procedure window. A display recreation macro contains the commands necessary to recreate a graph, table or page layout. You can invoke this macro to recreate the graph you just closed.

3. **Choose the Windows->Procedure Windows->Procedure Window menu item.**

   The procedure window is always present but is usually hidden to keep it out of the way. The window now contains the recreation macro for Graph0. You may need to scroll up to see the start of the macro. Because of the way it is declared, `Window Graph0() : Graph`, this macro will be available from the Graph Macros submenu of the Windows main menu.

4. **Click the procedure window's close button.**

   This hides the procedure window. Most other windows will put up a dialog asking if you want to kill or hide the window, but the built-in procedure window and the help windows simply hide themselves.

Recreating the Graph

1. **Choose the Windows->Graph Macros->Graph0 menu item.**

   Igor executes the Graph0 macro which recreates a graph of the same name.

2. **Repeat step #1.**

   The Graph0 macro is executed again but this time Igor gave the new graph a slightly different name, Graph0_1, because a graph named Graph0 already existed.

3. **While pressing Option (*Macintosh*) or Alt (*Windows*) click the close button of Graph0_1.**

   The window is killed without presenting a dialog.

Saving Your Work

1. **Choose the File->Save Experiment As menu item.**
2. **Navigate back to the folder where you saved the first time.**
3. **Change the name to "Tour #1 b.pxp" and click Save.**

   If you want to take a break, you can quit from Igor now.

Using Igor Documentation

Now we will take a quick look at how find information about Igor.

In addition to guided tours such as this one, Igor includes context-sensitive help, general usage information and reference information. The main guided tours as well as the general and reference information are available in both the online help files and in the Igor Pro PDF manual.

We'll start with context-sensitive help.

1. **On Macintosh only, turn Igor Tips on by choosing Help->Show Igor Tips.**

   On Macintosh Igor tips for icons, menu items and dialog items appear in yellow textboxes.

   On Windows tips for icons and menu items appear in the status line at the bottom of the Igor Pro frame window.

2. **Click the Data menu item and move the cursor over the items in the menu.**

   Notice the tips in yellow textboxes on Macintosh and in the status line on Windows.

   You can also get textbox tips on Windows by pressing Shift-F1 and then clicking an icon or menu item, but the status line is usually more convenient.

3. **Choose Data->Load Waves->Load Waves.**

Igor displays the Load Waves dialog. This dialog provides an interface to the LoadWave operation which is how you load data into Igor from text data files.

4. **On Macintosh only, move the cursor over the Load Columns Into Matrix checkbox.**

   An Igor tip appears in a yellow textbox. You can get a tip for most dialog items this way.

5. **On Windows only, click the question-mark icon in the top-right corner of the dialog window and then click the Load Columns Into Matrix checkbox.**

   Context-sensitive help appears in a yellow textbox. You can get a tip for most dialog items this way.

6. **Click the Cancel button to quit the dialog.**

Now let's see how to get reference help for a particular operation.

7. **Choose Help->Command Help.**

   The Igor Help Browser appears with the Command Help tab displayed.

   The information displayed in this tab comes from the Igor Reference help file - one of many help files that Igor automatically opens at launch. Open help files are directly accessible through Windows->Help Windows but we will use the Igor Help Browser right now.

8. **Make sure all three checkboxes in the Command Help tab of the help browser are checked and that all three pop-up menus are set to All.**

   These checkboxes and pop-up menus control which operations, functions and keywords appear in the list.

9. **Click any item in the list and then type "Loa".**

   Igor displays help for the LoadData operation. We want the LoadWave operation.

10. **Press the down-arrow key a few times until LoadWave is selected in the list.**

    Igor displays help for the LoadWave operation in the Help windoid.

    Another way to get reference help is to Ctrl-click (*Macintosh*) or right-click (*Windows*) the name of an operation or function and choose the "Help For" menu item. This works in the command window and in procedure, notebook and help windows.

    While we're in the Igor Help Browser, let's see what the other tabs are for.

11. **Click each of the Help Browser tabs and note their contents.**

    You can explore these tabs in more detail later.

    Next we will take a quick trip to the Igor Pro PDF manual. If you are doing this guided tour using the PDF manual, you may want to just read the following steps rather than do them to avoid losing your place.

12. **Click the Manual tab and then click the Open Online Manual button.**

    Igor opens the PDF manual in your PDF viewer - typically Adobe Reader or Apple's Preview program.

    If you use Adobe Reader for viewing PDF files, you should have a Bookmarks pane on the left side of the PDF manual window. If not, choose View->Navigation Panel->Bookmarks in Reader.

    If you use Apple's Preview for PDF files, you should have a drawer on one side of the main page. If not, choose View->Drawer in Preview.

    Note in the Reader Bookmarks pane or the Preview drawer that the PDF manual is organized into five volumes plus an index.

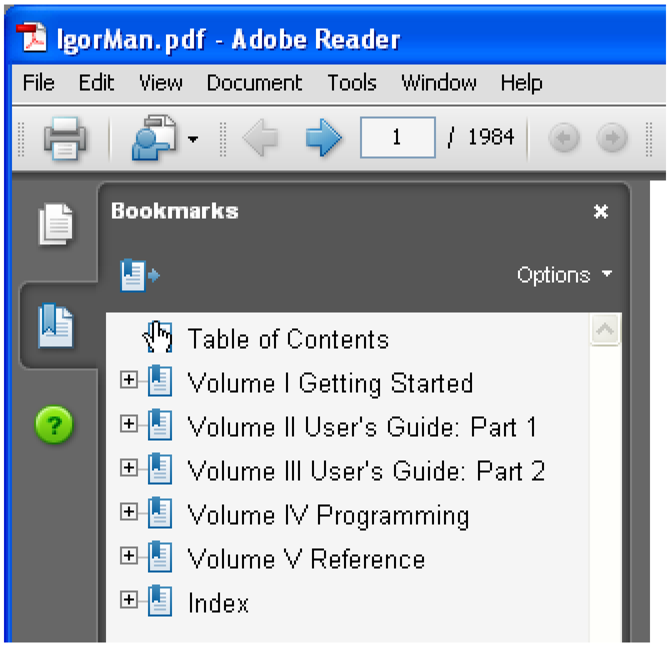

IgorMan.pdf - Adobe Reader
File Edit View Document Tools Window Help
1 / 1984
Bookmarks
Options

Table of Contents
Volume I Getting Started
Volume II User's Guide: Part 1
Volume III User's Guide: Part 2
Volume IV Programming
Volume V Reference
Index

13. **Use the Bookmarks pane to get a sense of what's in the manual.**

    Expand the volume bookmarks to see the chapter names.

    You may have noticed that the Igor PDF manual is rather large - about 2,000 pages at last count. You'll be happy to know that we don't expect you to read it cover-to-cover. Instead, chapters as the need arises.

    The information in the manual is also in the online help files. The manual, being in book format, is better organized for linear reading while the online help is usually preferred for accessing reference information.

    In case you ever want to open it directly, you can find the PDF manual in "Igor Pro Folder/Manual".

That should give you an idea of where to look for information about Igor. Now let's get back to our hands-on exploration of Igor.

Graphically Editing Data

0. **If you quit Igor after the last save, open your "Tour #1 b.pxp" experiment and turn off preferences.**
1. **Adjust the positions of the graph and table so you can see both.**

   Make sure you can see the columns of data when the graph is the front window.
2. **If necessary, click on the Graph0 window to bring it to the front.**
3. **Click on the drawing mode icon, , to activate the drawing tools.**
4. **While pressing Option (*Macintosh*) or Alt (*Windows*) on the keyboard, move the cursor over the polygon icon () and click and hold the mouse button.**

   A pop-up menu appears.
5. **Choose the Edit->Edit Wave menu item.**
6. **Click on one of the open circles of the yval trace.**

   (yval is labeled "Magnitude" in the legend.)

   The trace is redrawn using lines and squares to show the location of the data points.
7. **Click on the second square from the left and drag it 1 cm up and to the right.**

   Notice point 1 of yval and timeval changes in the table.

8. **Press Command-Z (*Macintosh*) or Ctrl+Z (*Windows*) or choose Edit->Undo.**
9. **Click midway between the first and second point and drag up 1 cm.**

   Notice an new data point 1 of yval and timeval appear in the table.
10. **Press Option (*Macintosh*) or Alt (*Windows*) and click the new data point with the tip of the lightning bolt.**

    The new data point is zapped.

    You could also have pressed Command-Z (*Macintosh*) or Ctrl+Z (*Windows*) to undo the insertion.
11. **Press Command (*Macintosh*) or Ctrl (*Windows*), click the line segment between the second and third point and drag a few cm to the right.**

    The line segment is moved and two points of yval and timeval are changed in the table.
12. **Press Command-Z (*Macintosh*) or Ctrl+Z (*Windows*) or choose Edit->Undo.**
13. **Click in the operate icon, , to exit draw mode.**
14. **Choose File->Revert Experiment and answer Yes to the resulting dialog.**

    This returns the experiment to the state it was in before we started editing the data.

Making a Category Plot (Optional)

Category plots show continuous numeric data plotted against non-numeric text categories.

1. **Choose the Windows->New Table menu item.**
2. **Click the Do It button.**

   A new blank table is created. We could have used the existing table but its best to keep unrelated data separate.
3. **Type "Monday" and then press Return or Enter.**

   A wave named "textWave0" was created with the text Monday as the value of the first point. Entering a non-numeric value in the first row of the first blank column automatically creates a new text wave.
4. **Type the following lines, pressing Enter after each one:**

   Tuesday

   Wednesday

   Thursday
5. **Click in the first cell of the next column and enter the following values:**

   10

   25

   3

   16
6. **Click in the first cell of the next column and enter the following values:**

   0

   12

   30

   17
7. **Choose Windows->New->Category Plot.**

   A dialog similar to the New Graph dialog appears. This dialog shows only text waves in the right-hand list.
8. **Click the From Target checkbox to select it.**

   This limits the list of waves to those in the target window. The target window is the table we just made.

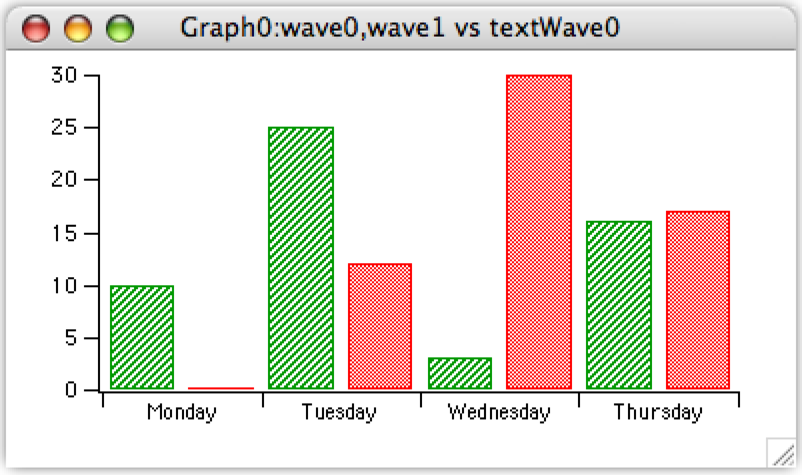
9. **In the Y Wave(s) list, select both items and select textWave0 in the X Wave list.**
10. **Click Do It.**

    A category plot is created.
11. **Double-click on one of the bars.**

    The Modify Trace Appearance dialog appears.
12. **Using the "+Fill Type" pop-up menu, change the fills of each trace to any desired pattern.**

    You might also want to change the colors.
13. **Click Do It.**

    Your graph should now look like this:



Category Plot Options (Optional)

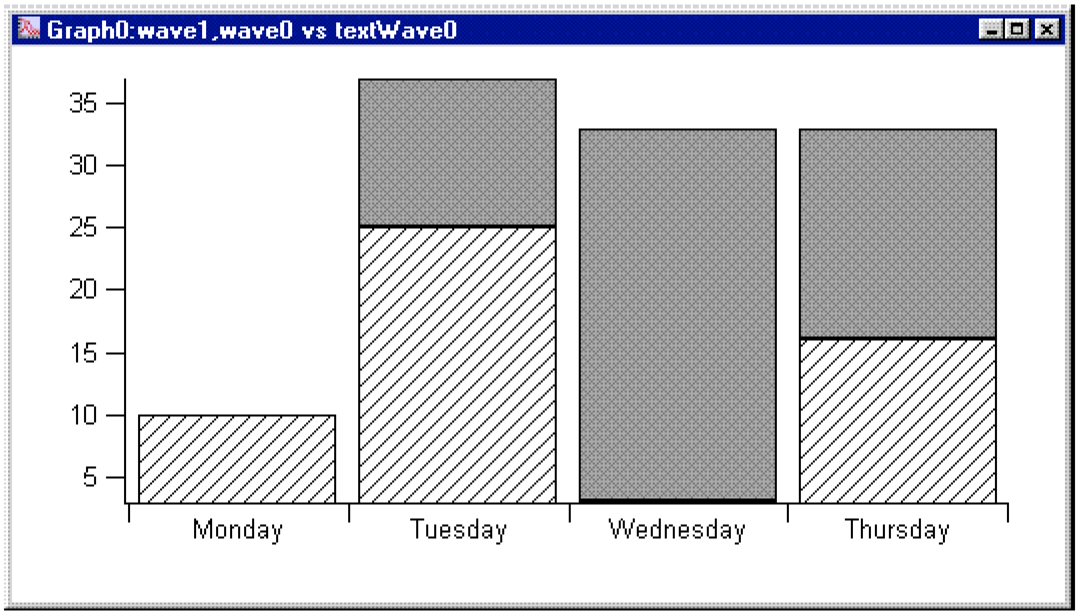
This section explores various category-plot options. If you are not particularly interested in category plots, you can stop now, or at any point in the following steps, by closing the graph and table and skipping forward to the next section.

1. **Double-click on one of the bars and, if necessary, select the top-most trace in the list.**
2. **From the Grouping pop-up menu, choose Stack on Next.**
3. **Click Do It.**

   The left bar in each group is now stacked on top of the right bar.
4. **Choose the Graph->Reorder Traces menu item.**
5. **Reverse the order of the items in the list by dragging the top item down. Click Do It.**

   The bars are no longer stacked and the bars that used to be on the left are now on the right. The reason the bars are not stacked is that the trace that we set to Stack on Next mode is now last and there is no next trace.
6. **Again using the Modify Trace Appearance dialog, set the top-most trace to Stack on next. Click Do It.**

   The category plot graph should now look like this:



7. **Enter the following values in the next blank column in the table:**

   7

   10

   15

   9

   This creates a new wave named wave2.

8. **Click on the graph to bring it to the front .**

9. **Choose Graph->Append to Graph->Category Plot.**

10. **From the Y list, select wave2 and click Do It.**

    The new trace is appended after the previous two. Because the second trace was in Stack on Next mode, the new trace is on the bottom of each set of three stacked bars.

11. **Using the Modify Trace Appearance dialog, change the grouping mode of the middle trace to none.**

    Now the new bars are to the right of a group of two stacked bars. You can create any combination of stacked and side-by-side bars.

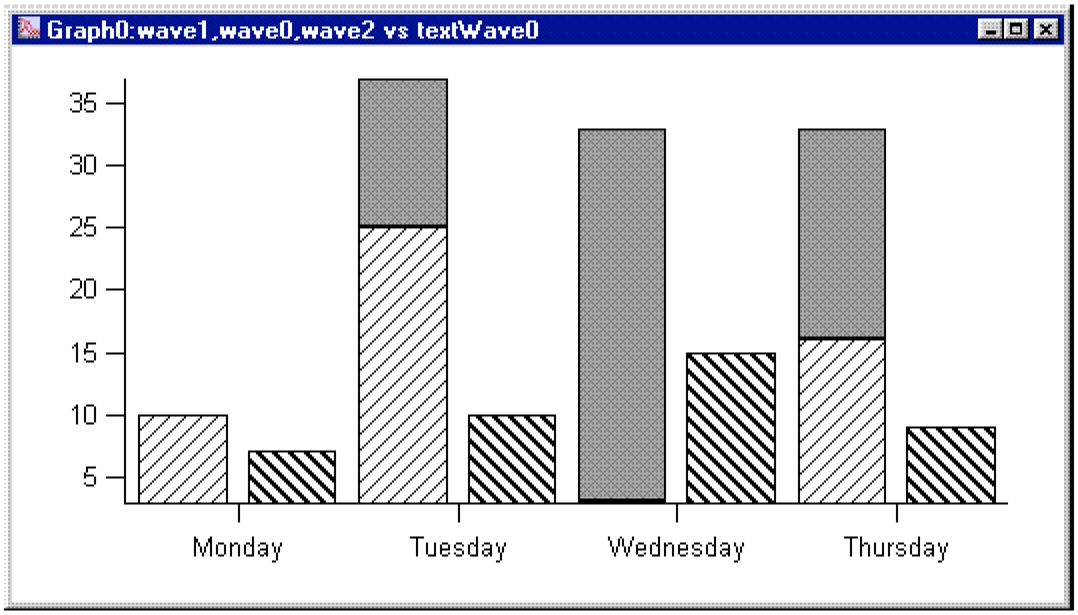
12. **Double-click on the bottom axis.**

    The Modify Axis dialog appears with the bottom axis selected.

13. **Click the Auto/Man Ticks tab.**

14. **Select the Tick In Center checkbox and then click Do It.**

    Notice the new positions of the tick marks.



15. **Again double-click on the bottom axis.**
16. **Click the Axis tab.**
17. **Change the value of Bar gap to zero and then click Do It.**

    Notice that the bars within a group are now touching.
18. **Use the Modify Axis dialog to set the Category Gap to 50%.**

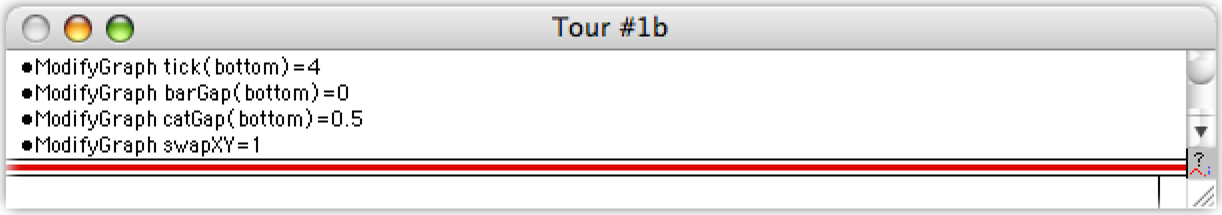
    The widths of the bars shrink to 50% of the category width.
19. **Choose Graph->Modify Graph.**
20. **Select the Swap X & Y Axes checkbox and then click Do It.**

    This is how you create a horizontal bar plot.
21. **Close both the graph and table windows without saving recreation macros.**

The Command Window

Parts of this tour make use of Igor's command line to enter mathematical formulae. Let's get some practice now.

Your command window should look something like this:



```
•ModifyGraph tick(bottom)=4
•ModifyGraph barGap(bottom)=0
•ModifyGraph catGap(bottom)=0.5
•ModifyGraph swapXY=1
```

The command line is the space below the separator while the space above the separator is called the history area.

1. **Click in the command line, type the following line and press Return or Enter.**

   ```
   Print 2+2
   ```

   The Print command as well as the result are placed in the history area.
2. **Press Up Arrow.**

The line containing the print command is selected, skipping over the result printout line.

3. **Press Return or Enter.**

   The selected line in the history is copied to the command line.

4. **Edit the command line so it matches the following and press Return or Enter.**

   ```
   Print "The result is ",2+2
   ```

   The Print command takes a list of numeric or string expressions, evaluates them and prints the results into the history.

5. **Choose the Help->Igor Help Browser menu item.**

   The Igor Help Browser appears.

   You can also display the help browser by pressing Help (*Macintosh*) or F1 (*Windows*), or by clicking the question-mark icon near the right edge of the command window.

6. **Click the Command Help tab in the Igor Help Browser.**

7. **Deselect the Functions and Programming checkboxes and select the Operations checkbox.**

   A list of operations appears.

8. **From the pop-up menu next to the Operations checkbox, choose About Waves.**

9. **Select PlaySound from the list.**

   Tip: Click in the list to activate it and then type "p" to jump to PlaySound.

10. **Click the Help windoid, scroll down to the Examples section, and select the first four lines of example text (starting with "Make", ending with "PlaySound sineSound).**

11. **Choose the Edit->Copy menu to copy the selection.**

12. **Close the Igor Help Browser.**

13. **Choose Edit->Paste.**

    All four lines are pasted into the command line area. You can view the lines using the miniature scroll arrows that appear at the right-hand edge of the command line.

14. **Press Return or Enter to execute the command.**

    The four lines are executed and a short tone plays. (*Windows* : You may see an error message if your computer is not set up for sound.)

15. **Click once on the last line in the history area (PlaySound sineSound).**

    The entire line (less the bullet) is selected just as if you pressed the arrow key.

16. **Press Return or Enter once to transfer the command to the command line and a second time to execute it.**

    The tone plays again as the line executes.

    We are finished with the "sineSound" wave that was created in this exercise so let's kill the wave to prevent it from cluttering up our wave lists.

17. **Choose Data->Kill Waves.**

    The Kill Waves dialog appears.

18. **Choose "sineSound" and click Do It.**

    The sineSound wave is removed from memory.

19. **Again click once on the history line "PlaySound sineSound".**

20. **Press Return or Enter twice to re-execute the command.**

    An error dialog is presented because the sineSound wave no longer exists.

21. **Click OK to close the error dialog.**

22. **Choose Edit->Clear Command Buffer or press Command-K (*Macintosh*) or Ctrl+K (*Windows*).**

When a command generates an error, it is left in the command line so you can edit and re-execute it. In this case we just wanted to clear the command line.

Browsing Waves

1. **Choose the Data->Browse Waves menu item.**

   The Browse Waves dialog appears. You can view the properties of the waves that are in memory and available for use in the current experiment and can also examine waves that are stored in individual binary files on disk.

2. **Click on "timeval" in the list.**

   The dialog shows the properties of the timeval wave.

3. **Click in the Wave Note area of the dialog.**

   A wave note is text you can associate with a wave. You can both view and edit the text of the note. The other fields in this dialog are read-only.

4. **Type the following:**

   This wave was created by typing data into a table.

5. **Click on the other waves in the list while observing their properties.**

6. **Click the done button to exit the dialog.**

Using the Data Browser

The Data Browser provides another way to browse waves. You can also browse numeric and string variables.

1. **Choose the Data->Data Browser menu item.**

   The Data Browser appears.

2. **Make sure all of the checkboxes in the top-left corner of the Data Browser are checked.**

3. **Click on the timeval wave icon to select it.**

   Note that the wave is displayed in the plot pane at the bottom of the Data Browser and the wave's properties are displayed just above in the info pane.

4. **Control-click (Macintosh) or right-click (Windows) on the timeval wave icon.**

   A contextual menu appears with a number of actions that you can perform on the selection.

5. **Press Escape to dismiss the contextual menu.**

   You can explore that and other Data Browser features later on your own.

6. **Click the Data Browser's close box to close it.**

Synthesizing Data

In this section we will make waves and fill them with data using arithmetic expressions.

1. **Choose the Data->Make Waves menu item.**

   The Make Waves dialog appears.

2. **Type "spiralY", Tab and then "spiralX" in the second box.**

3. **Change Rows to 1000.**

4. **Click Do It.**

   Two 1000 point waves have been created. They are now part of the experiment but are not visible because we haven't put them in a table or graph.

5. **Choose Data->Change Wave Scaling.**

6. **If a button labeled More Options is showing, click it.**
7. **In the Wave(s) list, click spiralY and then Command-click (*Macintosh*) or Ctrl-click (*Windows*) spiralX.**
8. **Choose Start and Right for the SetScale Mode pop-up menu.**
9. **Enter "0" for Start and "50" for Right.**
10. **Click Do It.**

    This executes a SetScale command specifying the X scaling of the spiralX and spiralY waves. X scaling is a property of a wave that maps a point number to an X value. In this case we are mapping point numbers 0 through 999 to X values 0 through 50.
11. **If necessary, click in the command window to bring it to the front.**
12. **Type the following on the command line and then press Return or Enter:**

    ```
    spiralY= x*sin(x)
    ```

    This is a waveform assignment statement. It assigns a value to each point of the destination wave (spiralY). The value stored for a given point is the value of the righthand expression at that point. The meaning of *x* in a waveform assignment statement is determined by the X scaling of the destination wave. In this case, *x* takes on values from 0 to 50 as Igor evaluates the righthand expression for points 0 through 999.
13. **Execute this in the command line:**

    ```
    spiralX= x*cos(x)
    ```

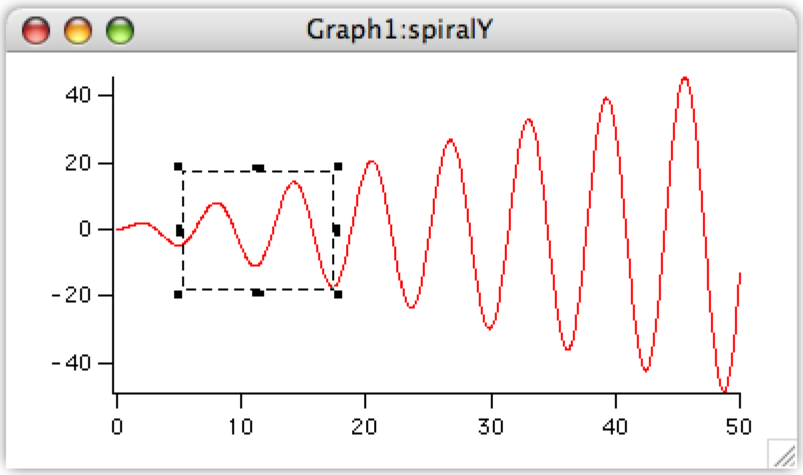
    Now both spiralX and spiralY have their data values set.

Zooming and Panning

1. **Choose the Windows->New Graph menu item.**
2. **If necessary, uncheck the From Target checkbox.**
3. **From the Y Wave(s) list, pick "spiralY".**
4. **From the X Wave list, pick "_calculated_".**
5. **Click Do It.**

    Note that the X axis goes from 0 to 50. This is because the SetScale command we executed earlier set the X scaling property of spiralY which tells Igor how to compute an X value from a point number. Choosing _calculated_ from the X Wave list graphs the spiralY data values versus these calculated X values.
6. **Position the cursor in the interior of the graph.**

    The cursor changes to a cross-hair shape.
7. **Click and drag down and to the right to create a marquee as shown:**



You can resize the marquee with the black squares (handles). You can move the marquee by dragging the dashed edge of the marquee.

8. **Position the cursor inside the marquee.**

   The mouse pointer changes to this shape: ▬ , indicating that a pop-up menu is available.

9 **Click and choose Expand from the pop-up menu.**

   The axes are rescaled so that the area enclosed by the marquee fills the graph.

10. **Choose Edit->Undo Scale Change or press Command-Z (*Macintosh*) or Ctrl+Z (*Windows*).**

11. **Choose Edit->Redo Scale Change or press Command-Z (*Macintosh*) or Ctrl+Z (*Windows*).**

12. **Press Option (*Macintosh*) or Alt (*Windows*) and position the cursor in the middle of the graph.**

    The cursor changes to a hand shape. You may need to move the cursor slightly before it changes shape.

13. **With the hand cursor showing, click and drag about 2 cm to the left.**

14. **While pressing Option (*Macintosh*) or Alt (*Windows*), click the middle of the graph and gently fling it to the right.**

    The graph continues to pan until you click again to stop it.

15. **Choose Graph->Autoscale Axes or press Command-A (*Macintosh*) or Ctrl+A (*Windows*).**

    Continue experimenting with zooming and panning as desired.

16. **Press Command-Option-W (*Macintosh*) or Ctrl+Alt+W (*Windows*).**

    The graph is killed. Pressing Option (*Macintosh*) or Alt (*Windows*) avoided the normal dialog asking whether to save the graph.

Making a Graph with Multiple Axes

1. **Choose the Windows->New Graph menu item.**

2. **If you see a button labeled More Choices, click it.**

   We will use the more complex form of the dialog to create a multiple-axis graph in one step.

3. **From the Y Wave(s) list, pick "spiralY".**

4. **From the X Wave list, pick "spiralX".**

5. **Click Add.**

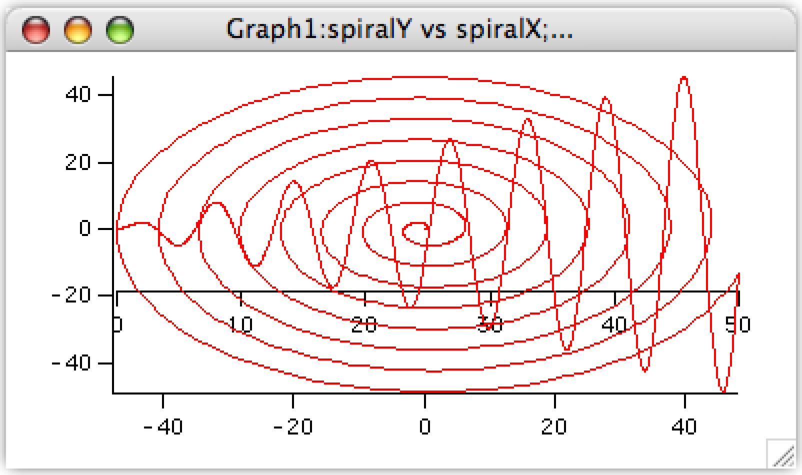
The selections are inserted into the lower list in the center of the dialog.

6. **From the Y Wave(s) list, again pick "spiralY".**
7. **From the X Wave list, pick "_calculated_".**
8. **Choose New from the Axis pop-up menu under the X Wave(s) list.**
9. **Enter "B2" in the name box.**
10. **Click OK.**

    Note the command box at the bottom of the dialog. It contains two commands: a Display command corresponding to the initial selections that you added to the lower list and an AppendToGraph command corresponding to the current selections in the Y Wave(s) and X Wave lists.

11. **Click Do It.**

    The following graph is created.



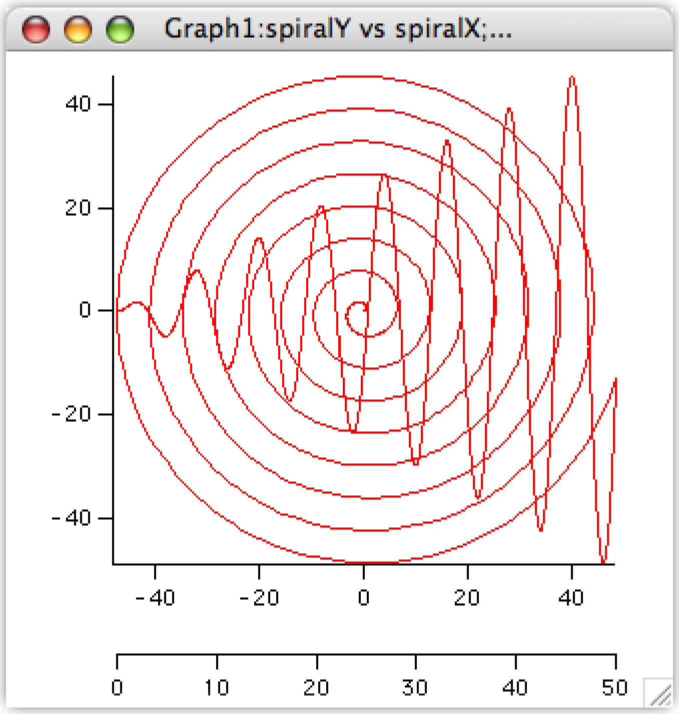
The interior axis is called a "free" axis because it can be moved relative to the plot rectangle. We will be moving it outside of the plot area but first we must make room by adjusting the plot area margins.

12. **Press Option (*Macintosh*) or Alt (*Windows*) and position the cursor over the bottom axis until the cursor changes to this shape: ╪.**

    This cursor indicates you are over an edge of the plot area rectangle and that you can drag that edge to adjust the margin.

13. **Click and drag the margin up about 2 cm. Release Option or Alt.**
14. **Drag the interior axis down into the margin space you just created.**
15. **Resize the graph so the spiral is nearly circular.**

    Your graph should now look like this:



Saving Your Work

1. **Choose the File->Save Experiment As menu item.**
2. **Type "Tour #1 c.pxp" in the name box and click Save.**

   If you want to take a break, you can quit from Igor now.

Using Cursors

0. **If you are returning from a break, open your "Tour #1 c.pxp" experiment and turn off preferences.**
1. **Click in the graph and choose the Graph->Show Info menu item.**

   A cursor info panel appears below the graph.
2. **Turn on** Igor Tips **(*Macintosh*) or use** context-sensitive help **(*Windows*) to examine the info panel.**

   On Macintosh, choose Help->Show Igor Tips and let the cursor hover over an item in the info panel. When you have seen all the help, turn Igor Tips off.

   On Windows, press Shift+F1 to get the cursor and click an item in the info panel. You can see similar help information in the status bar at the bottom of the Igor Pro frame window as you let the cursor hover over an item in the info panel
3. **Control-click (*Macintosh*) or right-click (*Windows*) in the name area for graph cursor A (the round one).**
4. **Choose "spiralY" from the pop-up menu.**

   The A cursor is placed on point zero of spiralY.
5. **Repeat for cursor B but choose "spiralY#1" from the pop-up menu.**

   The wave spiralY is graphed twice. The #1 suffix is used to distinguish the second instance from the first. It is #1 rather than #2 because in Igor, indices start from zero.
6. **Position the mouse pointer over the center of the slide control bar .**
7. **Click and gently drag the slide bar to the right.**

   Both cursors move to increasing point numbers. They stop when one or both get to the end.

8. **Practice moving the slide bar to the left and right.**

   Notice that the cursors move with increasing speed as the bar is displaced farther from the center.

9. **Click once on the dock for cursor A (the round black circle).**

   The circle turns white.

10. **Move the slide bar to the left and right.**

    Notice that only cursor B moves.

11. **Click on cursor B in the graph and drag it to another position on either trace.**

    You can also drag cursors from their docks to the graph.

12. **Click on cursor A in the graph and drag it completely outside the graph.**

    The cursor is removed from the graph and returns to its dock.

13. **Choose Graph->Hide Info.**

14. **Click on the command window, type the following and press Return or Enter.**

    ```
    Print vcsr(B)
    ```

    The Y value at cursor B is printed into the history area. There are many functions available for obtaining information about cursors.

15. **Click in the graph and then drag cursor B off of the graph.**

Removing a Trace and Axis

1. **Choose the Graph->Remove from Graph menu item.**

   The Remove From Graph dialog appears with spiralY listed twice. When we created the graph we used spiralY twice, first versus spiralX to create the spiral and second versus calculated X values to show the sine wave.

2. **Click on the second instance of spiralY (spiralY#1) and click Do It.**

   The sine wave and the bottom-most (free) axis are removed. An axis is removed when its last trace is removed.

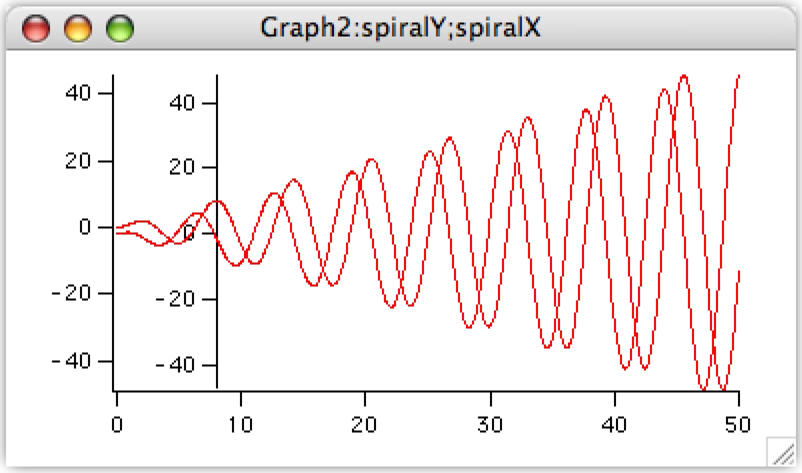
3. **Drag the horizontal axis off the bottom of the window.**

   This returns the margin setting to auto. We had set it to a fixed position when we option-dragged (*Macintosh*) or Alt-dragged (*Windows*) the margin in a previous step.

Creating a Graph with Stacked Axes

1. **Choose the Windows->New Graph menu item.**
2. **If you see a button labeled More Choices, click it.**
3. **From the Y Wave(s) list, pick "spiralY".**
4. **From the X Wave list, pick "_calculated_".**
5. **Click Add.**
6. **From the Y Wave(s) list, pick "spiralX".**
7. **From the X Wave list, pick "_calculated_".**
8. **Choose New from the Axis pop-up menu under the Y Wave(s) list.**
9. **Enter "L2" in the name box.**
10. **Click OK.**
11. **Click Do It.**

    The following graph is created.



In the next step we will stack the interior axis on top of the left axis.

12. **Double-click on the left axis.**

The Modify Axis dialog appears. If any other dialog appears, cancel and try again making sure the cursor is over the axis.

13. **Click the Axis tab.**

The Left axis should already be selected in the pop-up menu in the upper left corner.

14. **Set the Left axis to draw between 0 and 45% of normal.**
15. **Choose L2 from the Axis pop-up menu.**
16. **Set the L2 axis to draw between 55 and 100% of normal.**
17. **In the Free Axis Position box, pop up the menu reading Distance from Margin and select Fraction of Plot Area.**
18. **Verify that the box labeled "% of Plot Area" is set to zero.**

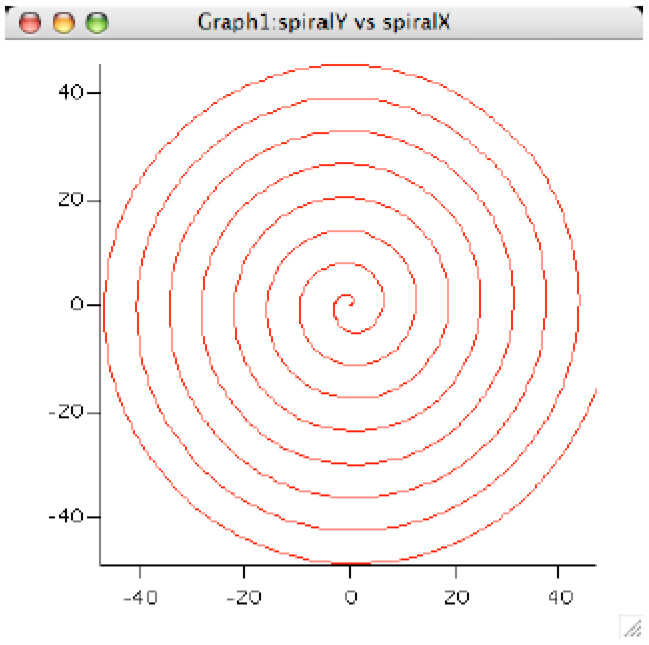
Steps 17 and 18 move the L2 axis so it is in line with the Left axis.

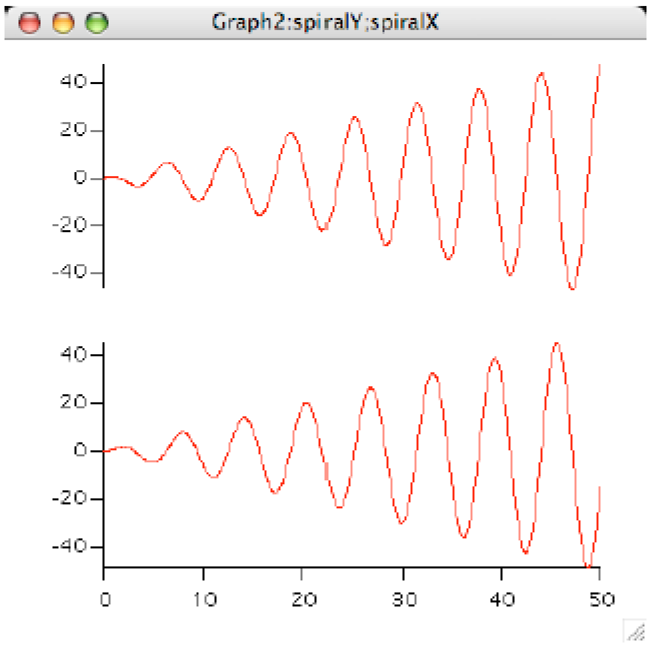
Why don't we make this the default? Good question - positioning as percent of plot area was added in Igor Pro 6; the default maintains backward compatibility.

19. **Choose Bottom from the Axis pop-up menu.**

20 **Click the Axis Standoff checkbox to turn standoff off.**

21. **Click Do It.**
22. **Resize and reposition the top two graph windows so they are side-by-side and roughly square.**
23. **The graphs should look like this:**





Appending to a Layout

1. **Choose the Windows->Layouts->Layout0 menu item.**
2. **Adjust the layout window size and scrolling so you can see the blank area below the graph that is already in the layout.**
3. **Click in the graph icon, , and choose "Graph1".**
   Graph1 is added to the layout.
4. **Again, click in the graph icon and choose "Graph2".**
   Graph2 is added to the layout.
5. **Click the marquee icon .**
6. **Drag out a marquee that fills the printable space under the original graph.**
7. **Choose Layout->Arrange Objects.**
   The Arrange Objects dialog appears.
8. **Select both Graph1 and Graph2. Leave the Use Marquee checkbox checked.**
9. **Click Do It.**
   The two graphs are tiled inside the area defined by the marquee.
10. **Click in the page area outside the marquee to dismiss it.**
11. **Choose Windows->Control->Send Behind or press Command-E (*Macintosh*) or Ctrl+E (*Windows*).**

Saving Your Work

1. **Choose the File->Save Experiment As menu item.**
2. **Type "Tour #1 d.pxp" in the name box and click Save.**
   If you want to take a break, you can quit from Igor now.

Creating Controls

This section illustrates adding controls to an Igor graph -- the type of thing a programmer might want to do. If you are not interested in programming, you can skip to the End of General Tour.

0. **If you are returning from a break, open your "Tour #1 d.pxp" experiment and turn off preferences.**

1. **Click on the graph with the spiral (Graph1) to bring it to the front.**

2. **Choose the Graph->Show Tools menu item or press Command-T (*Macintosh*) or Ctrl+T (*Windows*).**

   A toolbar is displayed to the left of the graph. The second icon is selected indicating that the graph is in the drawing as opposed to the normal mode.

   The selector tool (arrow) is active. It is used to create, select, move and resize controls.

3. **Choose Graph->Add Controls->Control Bar.**

   The Control Bar dialog appears.

4. **Enter a height of 30 pixels and click Do It.**

   This reserves a space at the top of the graph for controls.

5. **Click on the command window, type the following and press Return or Enter.**

   ```
   Variable ymult=1, xmult=1
   ```

   This creates two numeric variables and sets both to 1.0.

6. **Click in the graph and then choose Graph->Add Controls->Add Set Variable.**

   The SetVariable Control dialog appears.

   A SetVariable control provides a way to display and change the value of a variable.

7. **Choose "ymult" from the Variable pop-up menu.**

8. **Enter 80 in the Width edit box.**

   This setting is back near the top of the scrolling list.

9 **Set the High Limit, Low Limit, and Increment values to 10, 0.1, and 0.1 respectively.**

   You may need to scroll down to find these settings.

10. **Click Do It.**

    A SetVariable control attached to the variable ymult appears in the upper left of the control bar.

11. **Double-click the ymult control.**

    The SetVariable Control dialog appears.

12. **Click the Duplicate button (it's at the bottom center of the dialog).**

13. **Choose xmult as the value.**

14. **Click Do It.**

    A second SetVariable control appears in the control bar. This one is attached to the xMult variable.

15. **Choose Graph->Add Controls->Add Button.**

    The Button Control dialog appears.

16. **Enter "Update" in the Title box.**

17. **Click the New button adjacent to Procedure.**

    The Control Procedure dialog appears.

18. **Make sure the checkbox labelled "Prefer structure-based procedures" is not selected.**

19 **Edit the procedure text so it looks like this:**

```
Function ButtonProc(ctrlName) : ButtonControl
	String ctrlName

	Wave spiralY, spiralX
	NVAR ymult, xmult
```

```
	spiralY= x*sin(ymult*x)
	spiralX= x*cos(xmult*x)
End
```

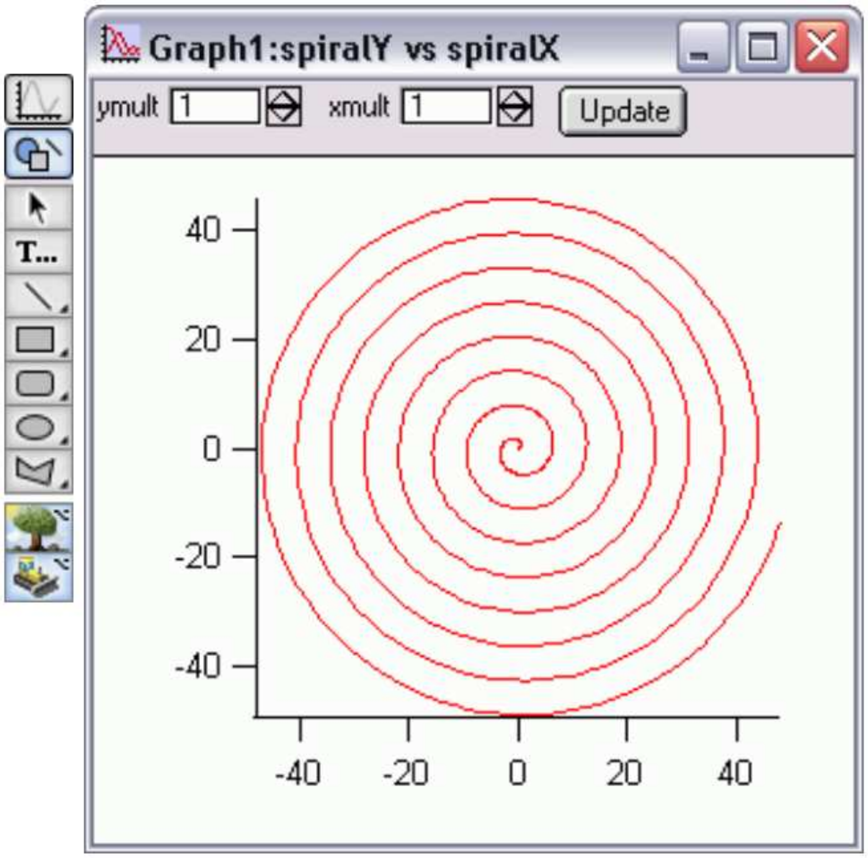
20. **Click the Save Procedure Now button.**

    The Control Procedure dialog disappears and the text you were editing is inserted into the (currently hidden) built-in procedure window. (See also Procedure Windows).

21. **Click Do It.**

    A Button control is added to the control bar.

    The three controls are now functional but are not esthetically arranged.



22. **Use the Arrow tool to rearrange the three controls into a more pleasing arrangement. Expand the button so it doesn't crowd the text by dragging its handles.**
23. **Click the top icon in the tool palette to enter "operate mode".**
24. **Choose Graph->Hide Tools or press Command-T (*Macintosh*) or Ctrl+T (*Windows*).**
25. **Click the up arrow in the ymult control.**

    The value changes to 1.1.

26. **Click the Update button.**

    The ButtonProc procedure that you created executes. The spiralY and spiralX waves are recalculated according to the expressions you entered in the procedure and the graphs are updated.

27. **Experiment with different ymult and xmult settings as desired.**
28. **Set both values back to 1 and click the Update button.**

You can use Tab to select a value and then simply type "1" followed by Return or Enter.

Creating a Dependency

A dependency is a rule that relates the value of an Igor wave or variable to the values of other waves or variables. By setting up a dependency you can cause Igor to automatically update a wave when another wave or variable changes.

1. **Click on the command window to bring it to the front.**
2. **Execute the following commands in the command line:**

   ```
   spiralY := x*sin(ymult*x)
   spiralX := x*cos(xmult*x)
   ```

   This is exactly what you entered before except here ":=" is used in place of "=". The := operator creates a dependency formula. In the first expression, the wave spiralY is made dependent on the variable ymult. If a new value is stored in ymult then the values in spiralY are automatically recalculated from the expression.
3. **Click on the graph with the spiral (Graph1) to bring it to the front.**
4. **Adjust the ymult and xmult controls but do not click the Update button.**

   When you change the value of ymult or xmult using the SetVariable control, Igor automatically executes the dependency formula. The spiralY or spiralX waves are recalculated and both graphs are updated.
5. **On the command line, execute this:**

   ```
   ymult := 3*xmult
   ```

   Note that the ymult SetVariable control as well as the graphs are updated.
6. **Adjust the xmult value.**

   Again notice that ymult as well as the graphs are updated.
7. **Choose the Misc->Object Status menu item.**

   The Object Status dialog appears. You can use this dialog to examine Igor objects that might otherwise have no visual representation such as string and numeric variables.
8. **Click the "The Current Object" pop-up menu and choose spiralY from the Dependent Objects item (*Macintosh*) or drop-down list (*Windows*).**

   The list on the right indicates that spiralY depends on the variable ymult.
9. **Double-click on the ymult entry in the right hand list.**

   ymult becomes the current object. The list on the right now indicates that ymult depends on xmult.
10. **Click the Delete Formula button.**

    Now ymult no longer depends on xmult.
11. **Click Done.**
12. **Adjust the xmult setting.**

    The ymult value is no longer automatically recalculated but the spiralY and spiralX waves still are.
13. **Click the Update button.**
14. **Adjust the xmult and ymult settings.**

    The spiralY and spiralX waves are no longer automatically recalculated. This is because the ButtonProc function called by the Update button does a normal assignment using "=" rather than ":=" and that action removes the dependency formulae.

    Note: In real work, you should avoid the kind of multilevel dependencies that we created here because they are too confusing.

    In fact, it is best to avoid dependencies altogether as they are hard to keep track of and debug. If a button action procedure or menu item procedure can do the job then use the procedure rather than the dependency.

Saving Your Work

1. **Choose the File->Save Experiment As menu item.**
2. **Type "Tour #1 e.pxp" in the name box and click Save.**

End of General Tour

This is the end of this tour.

If you want to take a break, you can quit from Igor Pro now.

• Guided Tour 2 - Data Analysis

In this tour we will concentrate on the data analysis features of Igor Pro. We will generate synthetic data and then manipulate it using sorting and curve fitting.

Launching Igor Pro

1. **Double-click the Igor Pro application file on your hard disk.**

   If Igor was already running, choose New Experiment from the File menu.
2. **Use the Misc menu to turn preferences off.**

Creating Synthetic Data

We need something to analyze, so we generate some random X values and create some Y data using a math function.

NOTE: You can execute the example commands given in this help file by selecting the line or lines and pressing Control-Enter (*Macintosh*) or Ctrl+Enter (*Windows*).

1. **Type the following in the command line and then press Return or Enter:**
   ```
   SetRandomSeed 0.1
   ```
   This initializes the random number generator so you will get the same results as this guided tour.
2. **Type the following in the command line and then press Return or Enter:**
   ```
   Make/N=100 fakeX= enoise(5)+5,fakeY
   ```
   This generates two 100 point waves and fills fakeX with evenly distributed random values ranging from 0 to 10.
3. **Execute this in the same way:**
   ```
   fakeY= exp(-(fakeX-4)^2)+gnoise(0.1)
   ```
   This generates a Gaussian peak centered at 4.
4. **Choose the Windows->New Graph menu item.**
5. **From the Y Wave(s) list, pick "fakeY".**
6. **From the XWave list, pick "fakeX".**
7. **Click Do It.**

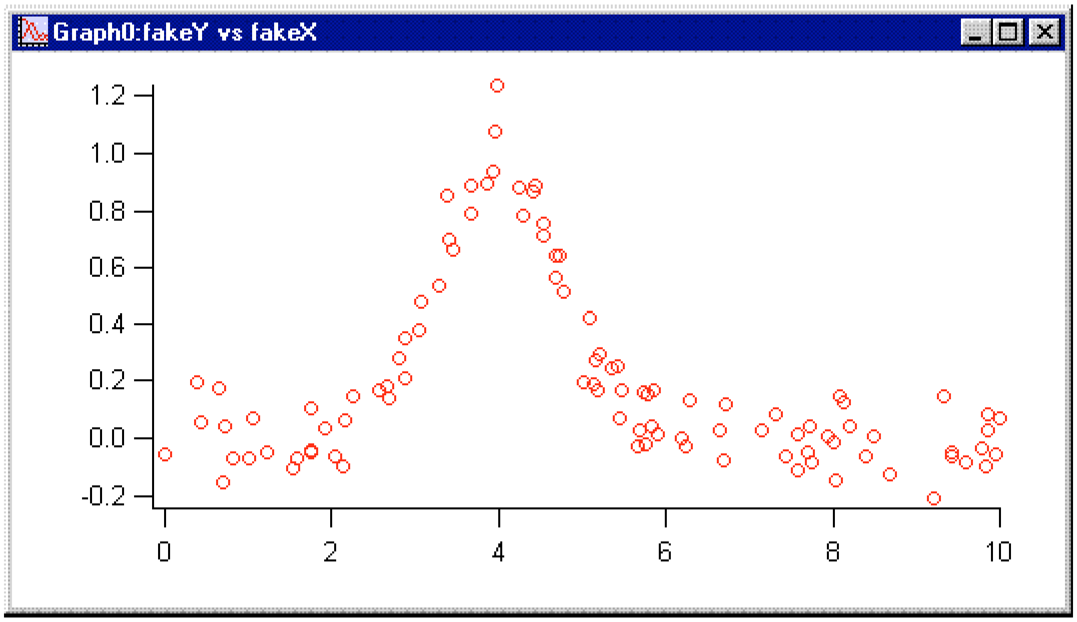
   The graph is a rat's nest of lines because the X values are not sorted.
8. **Double-click in the center of the graph.**

   The Modify Trace Appearance dialog appears.
9. **From the Mode pop-up choose Markers.**

10. From the pop-up menu of markers choose the open circle.

11. Click Do It.

Now the graph makes sense.



Quick Curve Fit to a Gaussian

Our synthetic data was generated using a Gaussian function so let's try to extract the original parameters by fitting to a Gaussian of the form:

```
y = y0 + A*exp(-((x-x0)/width)^2)
```

Here y0, A, x0 and width are the parameters of the fit.

1. Choose the Analysis->Quick Fit->gauss menu item.

Igor generated and executed a CurveFit command which you can see if you scroll up a bit in the history area of the command window. The CurveFit command performed the fit, appended a fit result trace to the graph, and reported results in the history area.

At the bottom of the reported results we see the values found for the fit parameters. The amplitude parameter (A) should be 1.0 and the position parameter (x0) should be 4.0. We got 0.99222 +/- 0.0299 for the amplitude and 3.9997 +/- 0.023 for the position.

Let's add this information to the graph.

2. Choose Analysis->Quick Fit->Textbox Preferences.

The Curve Fit Textbox Preferences dialog appears.

You can add a textbox containing curve fit results to your graph. The Curve Fit Textbox Preference dialog has a checkbox for each component of information that can be included in the textbox.

3. Click the Display Curve Fit Info Textbox to select it and then click OK.

You have specified that you want an info textbox. This will affect future Quick Fit operations.

4. Choose Analysis->Quick Fit->gauss again.

This time, Igor displays a textbox with the curve fit results. Once the textbox is made, it is just a textbox and you can double-click it and change it. But if you redo the fit, your changes will be lost unless you rename the textbox.

That textbox is nice, but it's too big. Let's get rid of it.

5. Choose Analysis->Quick Fit->Textbox Preferences again. Click the Display Curve

Fit Info Textbox to deselect it. Click OK.

6. **Choose Analysis->Quick Fit->gauss again.**

   The textbox is removed from the graph.

   You could just double-click the textbox and click Delete in the Modify Annotation dialog. The next time you do a Quick Fit you would still get the textbox unless you turn the textbox feature off.

More Curve Fitting to a Gaussian

The Quick Fit menu provides easy access to curve fitting using the built-in fit functions, with a limited set of options, to fit data displayed in a graph. You may want more options. For that you use the Curve Fitting dialog.

1. **Choose the Analysis->Curve Fitting menu item.**

   The curve fitting dialog appears.

2. **Click the Function and Data tab.**
3. **From the Function pop-up menu, choose gauss.**
4. **From the Y data pop-up menu, choose fakeY.**
5. **From the X data pop-up menu, choose fakeX.**
6. **Click the Data Options tab.**

   The Weighting and Data Mask pop-up menus should read "_none_".

7. **Click the Output Options tab.**

   The Destination pop-up menu should read "_auto_", and Residual should read "_none_".

8. **Click Do It.**

   During the fit a Curve Fit progress window appears. After a few passes the fit is finished and Igor waits for you to click OK in the progress window.

9. **Click OK.**

   The curve fit results are printed in the history. They are the same as in the previous section.

Sorting

In the next section we will do a curve fit to a subrange of the data. For this to work, the data must be sorted by X values.

1. **Double-click one of the open circle markers in the graph.**

   The Modify Trace Appearance dialog appears with fakeY selected. If fakeY is not selected, click it.

2. **From the Mode pop-up choose Lines between points and click Do It.**

   The fakeY trace reverts to a rat's next of lines.

3. **Choose the Analysis->Sort menu item.**

   The Sorting dialog appears.

4. **If necessary choose Sort from the Operation pop-up menu.**
5. **Select "fakeX" in the "Key Wave" list and both "fakeX" and "fakeY" in the "Waves to Sort" list.**

   This will sort both fakeX and fakeY using fakeX as the sort key.

6. **Click Do It.**

   The rat's nest is untangled. Since we were using the lines between points mode just to show the results of the sort, we now switch back to open circles but in a new way.

7. **Control-click (*Macintosh*) or right-click (*Windows*) on the fakeY trace.**

   A pop-up menu appears with the name of the trace at the top. If it is not "Browse fakeY" try again.

8. **Choose Markers from the Mode item.**

Fitting to a Subrange

Here we will again fit our data to a Gaussian but using a subset of the data. We will then extrapolate the fit outside of the initial range.

1. **Choose the Graph->Show Info menu item.**

   A cursor info panel is appended to the bottom of the graph.

   Two cursors are "docked" in the info panel, Cursor A and Cursor B.

2. **Place cursor A (the round one) on the fakeY trace.**

   One way to place the cursor is to drag it to the trace. Another way is to control-click (*Macintosh*) or right-click (*Windows*) on the name area which is just to the right of the cursor icon in the cursor info panel.

   Note that the cursor A icon in the dock is now black. This indicates that cursor A is selected, meaning that it will move if you use the arrow keys on the keyboard or the slider in the cursor info panel.

3. **Move cursor A to point #14.**

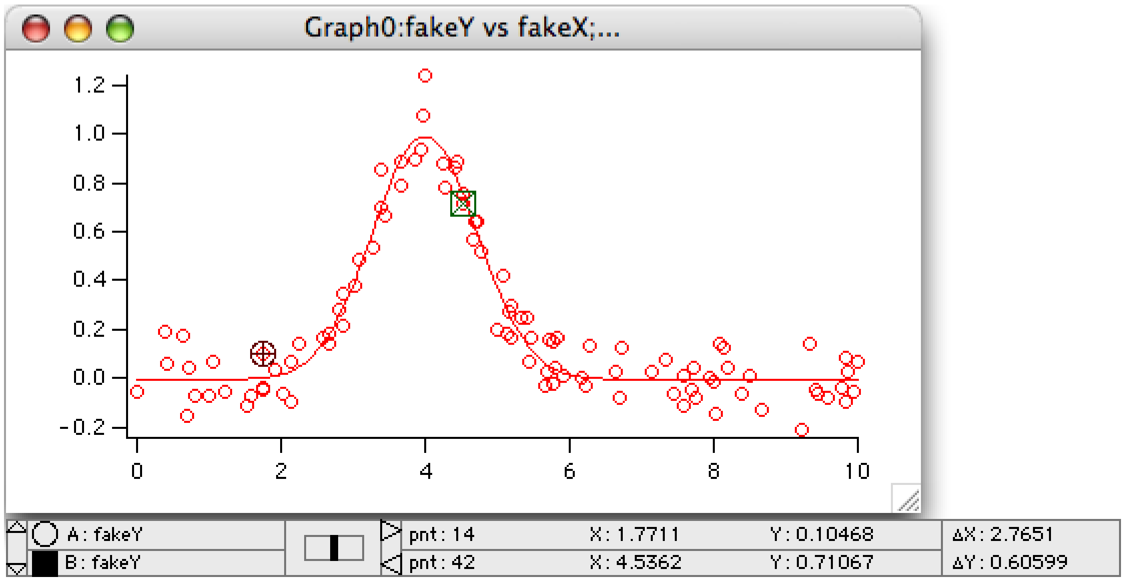
   To move the cursor one point at a time, use the arrow keys on the keyboard or click on either side of the slider in the cursor info panel.

4. **Click the dock for cursor A in the cursor info panel to deselect it.**

   This is so you can adjust cursor B without affecting the position of cursor A.

5. **Place cursor B (the square one) on the fakeY trace and move it to point #42.**

   Your graph should look like this:

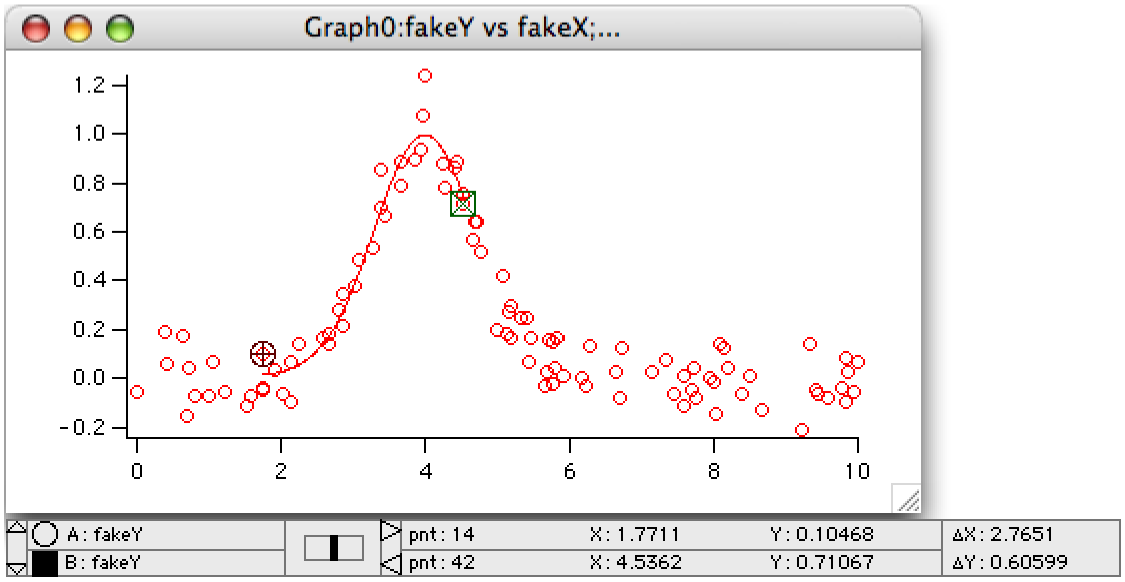


```
pnt: 14   X: 1.7711   Y: 0.10468   ΔX: 2.7651
pnt: 42   X: 4.5362   Y: 0.71067   ΔY: 0.60599
```

6. **In the Analysis->Quick Fit menu make sure the Fit Between Cursors item is checked. If it is not, select it to check it.**

7. **Choose Analysis->Quick Fit->gauss.**

   Note that the fit curve is evaluated only over the subrange identified by the cursors.



pnt: 14 X: 1.7711 Y: 0.10468 ΔX: 2.7651
pnt: 42 X: 4.5362 Y: 0.71067 ΔY: 0.60599

We would like the fit trace to extend over the entire X range, while fitting only to the data between the cursors. This is one of the options available only in the Curve Fitting dialog.

8. **Choose Analysis->Curve Fitting and then click the Function and Data tab.**

   The curve fitting dialog appears and the settings should be as you left them. Check that the function type is gauss, the Y data is fakeY, the X data is fakeX.

9. **Click the Data Options tab.**

10. **Click the Cursors button in the Range area.**

    This puts the text "pcsr(A)" and "pcsr(B)" in the range entry boxes.

    pcsr is a function that returns the wave point number at the cursor position.

11. **Select the Output Options tab and click the X Range Full Width of Graph checkbox to select it.**

12. **Click Do It.**

    The curve fit starts, does a few passes and waits for you to click OK.

13. **Click OK.**

    The fit has been done using only the data between the cursors, but the fit trace extends over the entire X range.

    In the next section, we need the short version of the fit curve, so we will simply do the fit again:

14. **Choose Analysis->Quick Fit->gauss.**

Extrapolating a Fit After the Fit is Done

When you used the Quick Fit menu, and when you chose "_auto_" from the Destination pop-up menu in the curve fit dialog, Igor created a wave named fit_fakeY to show the fit results. This is called the "fit destination wave." It is just an ordinary wave whose X scaling is set to the extent of the X values used in the fit.

In the preceding sections you learned how to make the curve fit operation extrapolate the fit curve beyond the subrange. Here we show you how to do this manually to illustrate some important wave concepts.

To extrapolate, we simply change the X scaling of fit_fakeY and re-execute the fit destination wave assignment statement which the CurveFit operation put in the history area.

1. **Choose the Data->Change Wave Scaling menu item.**
2. **If you see a button labeled More Options, click it.**

3. **From the SetScale Mode pop-up menu, choose Start and End.**
4. **Double-click on "fit_fakeY" in the list.**

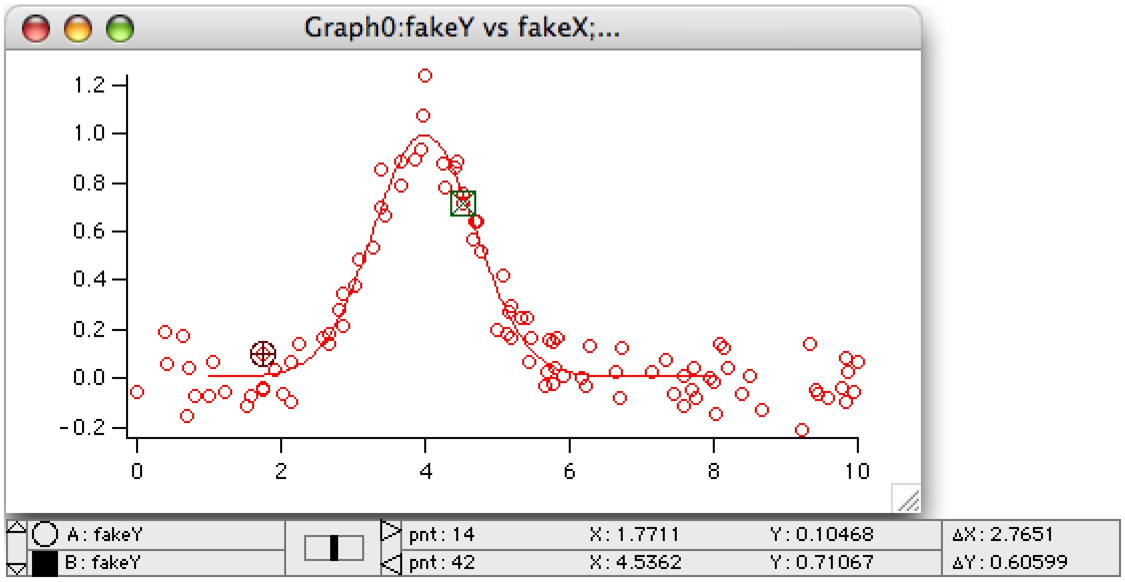
   This reads in the current X scaling values of fit_fakeY. The starting X value will be about 1.77 and the ending X will be about 4.53.
5. **Press Tab until the Start box is selected and type "1.0".**
6. **Tab to the End box and type "8.0".**
7. **Click Do It.**

   The fit_fakeY trace is stretched out and now runs between 1 and 8.

   Now we need to calculate new Y values for fit_fakeY using its new X values.
8. **In the history, find the line that starts "fit_fakeY=" and click it.**

   The entire line is selected. (The line in question is near the top of the curve fit report printed in the history.)
9. **Press Return or Enter once to copy the selection from the history to the command line and a second time to execute it.**

   The fit_fakeY wave now contains valid data between 1 and 8.



A: fakeY pnt: 14 X: 1.7711 Y: 0.10468 ΔX: 2.7651
B: fakeY pnt: 42 X: 4.5362 Y: 0.71067 ΔY: 0.60599

Appending a Fit

The fit trace added automatically when Igor does a curve fit uses a wave named by adding "fit_" to the start of the Y data wave's name. If you do another fit to the same Y data, that fit curve will be overwritten. If you want to show the results of several fits to the same data, you will have to somehow protect the fit destinaton wave from being overwritten. This is done by simply renaming it.

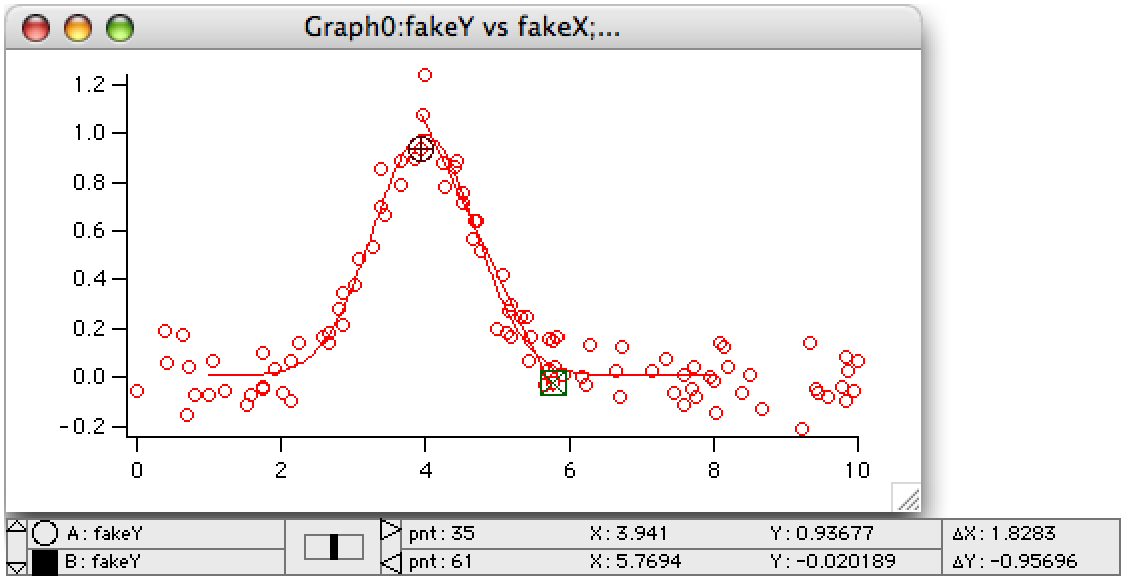
1. **Choose the Data->Rename menu item.**
2. **Double-click the wave named fit_fakeY to move it into the list on the right.**
3. **Edit the name in the New Name box to change the name to "gaussFit_fakeY" and click Do It.**
4. **Position the A and B cursors to point numbers 35 and 61 respectively.**

   Tip: Click in the dock for a given cursor to enable/disable its being moved by the slide control and arrow keys. Click on either side of the central slide or use the arrow keys to move the cursor one point number at a time.
5. **Choose Analysis->Quick Fit->line.**

   Because there are two traces on the graph, Quick Fit doesn't know which one to fit and puts up the Which Trace to Fit dialog.

6. **Select fakeY from the menu and click OK.**

   The curve fit is performed without displaying the fit progress window because the line fit is not iterative.



This concludes Guided Tour 2.

- ## Guided Tour 3 - Histograms and Curve Fitting

In this tour we will explore the Histogram operation and will perform a curve fit using weighting. The optional last portion creates a residuals plot and shows you how to create a useful procedure from commands in the history.

Launching Igor Pro

1. **Double-click the Igor Pro application file on your hard disk.**

   If Igor was already running, choose New Experiment from the File menu.

2. **Use the Misc menu to turn preferences off.**

Creating Synthetic Data

We need something to analyze, so let's generate some random values.

1. **Type the following in the command line and then press Return or Enter:**

   ```
   SetRandomSeed 0.1
   ```

   This initializes the random number generator so you will get the same results as this guided tour.

2. **Type the following in the command line and then press Return or Enter:**

   ```
   Make/N=10000 fakeY= enoise(1)
   ```

   This generates a 10,000 point wave filled with evenly distributed random values ranging from -1 to 1.

Histogram of White Noise

Here we will generate a histogram of the evenly distributed "white" noise.

1. **Choose the Analysis->Histogram menu item.**

   The Histogram dialog appears.

2. **Select fakeY from the Source Wave list.**
3. **Verify that Auto is selected in the Output Wave menu.**
4. **Select the Auto-set bin range radio button.**
5. **Set the Number of Bins box to 100.**

   Note in the command box at the bottom of the dialog there are two commands:

   ```
   Make/N=100/O fakeY_Hist;DelayUpdate
   Histogram/B=1 fakeY,fakeY_Hist
   ```

   The first command makes a wave to receive the results, the second performs the analysis. The Histogram operation in the "Auto-set bin range" mode takes the number of bins from the output wave.
6. **Click the Do It button.**

   The histogram operation is performed.

   Now we need to display the results.
7. **Choose Windows->New Graph.**
8. **Select fakeY_Hist from the Y Wave(s) list and _calculated_ from the X list.**
9. **Click the Do It button.**

   A graph is created showing the histogram results. We need to touch it up a bit.
10. **Double-click the trace in the graph.**

    The Modify Trace Appearance dialog appears.
11. **Choose "Sticks to zero" from the Mode pop-up menu and click the Do It button.**

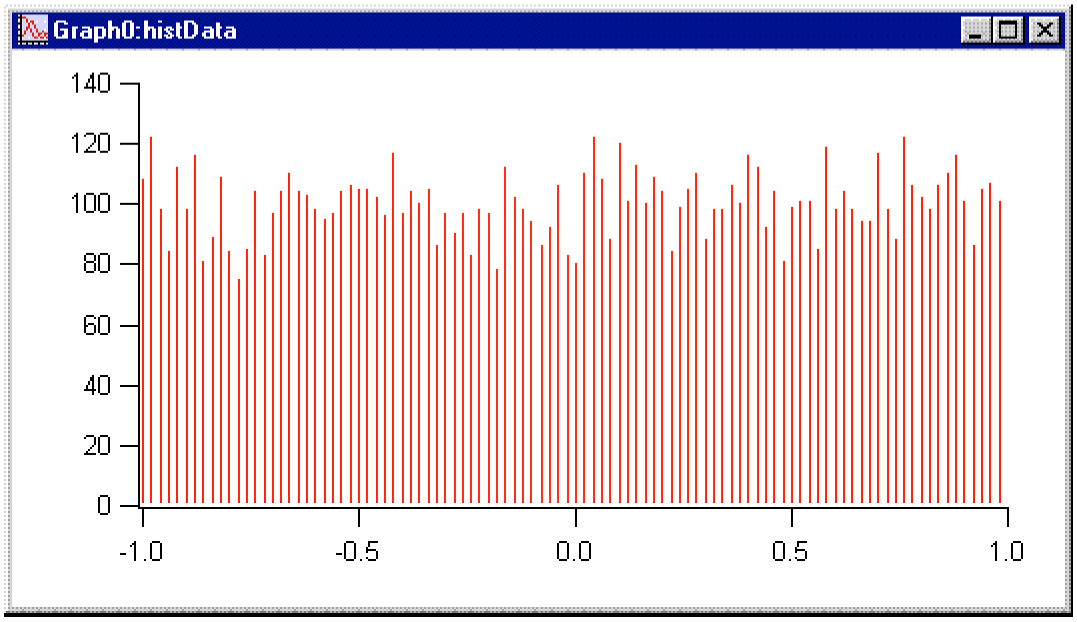
    The graph is redrawn using the new display mode.
12. **Double-click one of the tick mark labels (e.g., "100") of the left axis. of the left axis.**

    The Modify Axis dialog appears, showing the Axis Range tab.

    "Left" is selected in the Axis pop-up menu in the top/left corner of the dialog indicating that changes made in the dialog will affect the left axis.
13. **From the two pop-up menus in the Autoscale Settings area, choose "Round to nice values" and "Autoscale from zero".**
14. **Choose Bottom from the Axis pop-up menu.**
15. **From the two pop-up menus in the Autoscale Settings area, choose "Round to nice values" and "Symmetric about zero".**

16 **Click the Do It button.**

   Your graph should now look like this:



Histogram of Gaussian Noise

Now we'll do another histogram, this time with Gaussian noise.

1. **Type the following in the command line and then press Return or Enter:**

   ```
   fakeY = gnoise(1)
   ```

2. **Choose the Analysis->Histogram menu item.**

   The dialog should still be correctly set up from the last time.

3. **Click the radio button labeled "Auto-set bins: 3.49*Sdev*N^-1/3".**

   The information text at the bottom of the Destination Bins box shows you that he histogram will have 48 bins.

   This is a method by Sturges for selecting a "good" number of bins for a histogram. See the Histogram reference information for a reference.

4. **Select the Bin-Centered X Values checkbox.**

   By default, the Histogram operation sets the X scaling of the output wave such that the X values are at the left edge of each bin, and the right edge is given by the next X value. This makes a nice bar plot.

   In the next section you will do a curve fit to the histogram. For curve fitting you need X values that represent the center of each bin.

5. **Select the Create Square Root(N) Wave checkbox.**

   Counting data, such as a histogram, usually has Poisson-distributed values. The estimated mean of the Poisson distribution is simply the number of counts (N) and the estimated standard deviation is the square root of N.

   The curve fit will be biased if this is not taken into account. You will use this extra wave for weighting when you do the curve fit.

6. **Click the Do It button.**

   Note that the histogram output as shown in Graph0 has a Gaussian shape, as you would expect since the histogram input was noise with a Gaussian distribution.

7. **Choose Data->Data Browser.**

   The Data Browser shows you the waves and variables in your experiment. You should see three waves now: fakeY, fakeY_Hist, and W_SqrtN. FakeY_Hist contains the output of the Histogram operation and W_SqrtN is the wave created by the Histogram operation to receive the square root of N data.

8. **Click in Graph0 and then double-click the trace to bring up the Modify Trace Appearance dialog.**
9. **Select Markers from the Mode menu, then select the open circle marker.**
10. **Click the Error bars checkbox.**

    The Error Bars dialog appears.
11. **Select "+/- wave" from the Y Error Bars menu.**
12. **Pop up the Y+ menu and select W_SqrtN.**

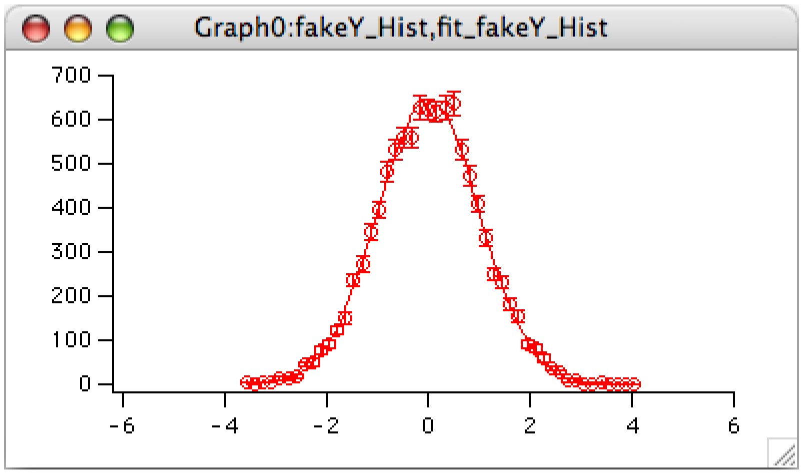
    Note that W_SqrtN is also selected in the Y- menu. You could now select another wave from the Y- menu if you needed asymmetric error bars.
13. **Click OK, then Do It.**

Curve Fit of Histogram Data

The previous section produces all the pieces required to fit a Gaussian to the histogram data, with proper weighting to account for the variance of Poisson-distributed data.

1. **Click in the graph to make sure it is the target window.**
2. **In the Analysis->Quick Fit menu make sure the Weight from Error Bar Wave item is checked. If it is not, select it to check it.**
3. **Choose Analysis->Quick Fit->gauss.**

    With all the changes you've made, by now the graph looks like this:



As shown in the history area, the fit results are:

```
Coefficient values ± one standard deviation
  y0    =-0.35284 ± 0.513
  A     =644.85 ± 7.99
  x0    =-0.0014111 ± 0.00997
  width =1.406 ± 0.0118
```

The original data was made with a standard deviation of 1. Why is the width 1.406? The way Igor defines its gauss fit function, width is sigma*$2^{1/2}$.

4. **Enter this command in the command line:**

```
Print 1.406/sqrt(2)
```

The result, 0.994192, is pretty close to 1.0.

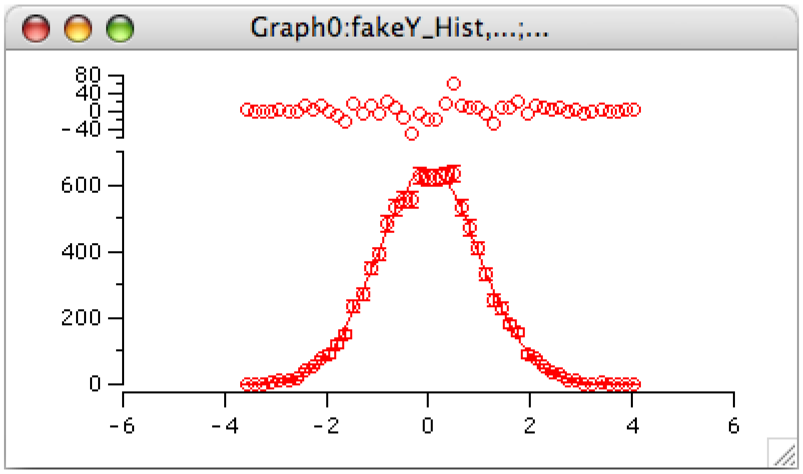
It is often useful to plot the residuals from a fit to check for various kinds of problems. For that you need to use the Curve Fit dialog.

5. **Choose Analysis->Curve Fitting.**
6. **Click the Function and Data tab and choose gauss from the Function menu.**

7. **Choose fakeY_Hist (not fakeY) from the Y data menu.**
8. **Leave the X data pop-up menu set to "_calculated_".**
9. **Click the Data Options tab. If there is text in the Start or End Range boxes, click the Clear button in the Range section.**
10. **Choose W_SqrtN from the Weighting pop-up menu.**
11. **Just under the Weighting pop-up menu there are two radio buttons. Click the top one which is labeled "Standard Deviation".**
12. **Click the Output Options tab and choose "_auto_" from the Destination pop-up menu.**
13. **Set the Residual pop-up menu to "_autotrace_".**

    Residuals will be calculated automatically and added to the curve fit in our graph.
14. **Click Do It.**

    The curve fit starts, does a few passes, and waits for you to click OK.



There is one small issue not addressed above. One of the bins contains zero; the square root of zero is, of course, zero. So the weighting wave contains a zero, which causes the curve fit to ignore that data point. It's not clear what is the best approach to fixing that problem. Some people replace the zero with a one. These commands replace any zeroes in the weighting wave and re-do the fit:

```
W_SqrtN = W_SqrtN[p] == 0 ? 1 : W_SqrtN[p]
CurveFit/NTHR=0 gauss  fakeY_Hist /W=W_SqrtN /I=1 /D /R
```

This doesn't change the result very much, since there was just one zero in the histogram:

```
Coefficient values ± one standard deviation
y0    =-0.40357 ± 0.464
A     =644.76 ± 7.98
x0    =-0.0014186 ± 0.00996
width =1.4065 ± 0.0115
```

<u>Curve Fit Residuals</u> (Optional)

This section and the next one are primarily of interest to people who want to use Igor programming to automate tasks.

In the next section, as an illustration of how the history area can be used as a source of commands to generate procedures, we will create a procedure that appends residuals to a graph. The preceding section illustrated that Igor is able to automatically display residuals from a curve fit, so the procedure that we write in the next section is not needed. Still, it demonstrates the process of creating a procedure. In preparation for writing the procedure, in this section we append the residuals manually.

If the curve fit to a Gaussian function went well and if the gnoise function truly produces noise with a Gaussian distribution, then a plot of the difference between the histogram data and the fitted function should not reveal any curvature.

0. **To remove the automatically generated residual from the Gaussian fit in the previous section, Control-click (*Macintosh*) or right-click (*Windows*) directly on the residual trace at the top of the graph and select Remove Res_fakeY_Hist from the pop-up menu.**
1. **Choose the Data->Duplicate Waves menu item.**
2. **Choose fakeY_Hist from the Template pop-up menu.**
3. **In the first Names box, enter "histResids".**
4. **Click Do It.**

   You now have a wave suitable for containing residuals.
5. **In the history area of the command window, find the line that reads:**

   ```
   fit_fakeY_Hist = W_coef[0]+ W_coef[1]*exp(-((x-W_coef[2])/W_coef[3])^2)
   ```

   W_coef is a wave created by the CurveFit operation to contain the fit parameters. W_coef[0] is the y0 parameter, W_coef[1] is the A parameter, W_coef[2] is the x0 parameter and W_coef[3] is the width parameter.

   This line shows conceptually what the CurveFit operation did to set the data values of the fit destination wave.
6. **Click once on the line to select it and then press Return or Enter.**

   The line is transferred to the command line.
7. **Edit the line to match the following:**

   ```
   histResids = fakeY_Hist-(W_coef[0]+ W_coef[1]*exp(-((x-W_coef[2])/W_coef[3])^2))
   ```

   In other words, change "fit_fakeY_Hist" to "histResids", click after the equals and type "fakeY_Hist - (" and then add a ")" to the end of the line.

   The expression inside the parentheses that you added represents the model value using the parameters determined by the fit. This command computes residuals by subtracting the model values from the data values on which the fit was performed.

   Note: If the fit had used an X-wave rather than calculated X values then it would have been necessary to substitute the name of the X-wave for the "x" in the expression.
8. **Press Return or Enter.**

   This wave assignment statement calculates the difference between the measured data (the output of the Histogram operation) and the theoretical Gaussian (as determined by the CurveFit operation).

   Now we will append the residuals to the graph stacked above the current contents.
9. **Choose Graph->Append Traces to Graph.**
10. **Select histResids from the Y wave(s) list and _calculated_ from the X wave list.**
11. **Choose New from the Axis pop-up menu under the Y Wave(s) list.**
12. **Enter "Lresid" in the Name box and click OK.**
13. **Click Do It.**

    The new trace and axis is added.

    Now we need to arrange the axes. We will do this by partioning the available space between the Left and Lresid axes.
14. **Double-click the far-left axis.**

    The Modify Axis dialog appears. If any other dialog appears, cancel and try again making sure the cursor is over the axis.

    If you have enough screen space you will be able to see the graph change as you change settings in the dialog. Make sure that the Live Update checkbox in the top/right corner of the dialog is selected.

15. Click the Axis tab.

The Left axis should already be selected in the pop-up menu in the top-left corner of the dialog.

16. Set the Left axis to draw between 0 and 70% of normal.

17. Choose Lresid from the Axis pop-up menu.

18. Set the Lresid axis to draw between 80 and 100% of normal.

19. Choose Fraction of Plot Area in the "Free axis position" menu.

The Lresid axis is a "free" axis. This moves it horizontally so it is in line with the Left axis.

20. Choose Bottom from the Axis pop-up menu.

21. Click the Axis Standoff checkbox to turn standoff off.

Just a couple more touch-ups and we will be done. The ticking of the Lresid axis can be improved. The residual data should be in dots mode.

22. Choose Lresid from the Axis pop-up menu again.

23. Click the Auto/Man Ticks tab.

24. Change the Approximately value to 2.

25. Click the Axis Range tab.

26. In the Autoscale Settings area, choose "Symmetric about zero" from the menu currently reading "Zero isn't special".

27 Click the Do It button.

28. Double-click the histResids trace.

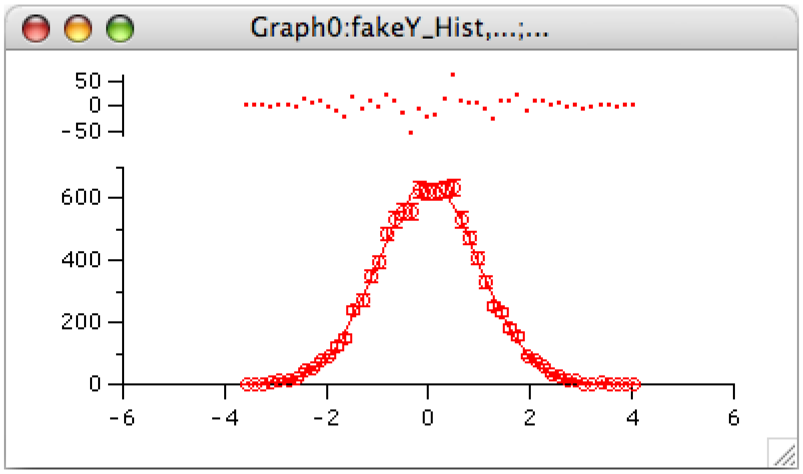
The Modify Waves Appearance dialog appears with histResids already selected in the list.

29. Choose Dots from the Mode pop-up menu.

30. Set the line size to 2.00.

31. Click Do It.

Your graph should now look like this:



Writing a Procedure (Optional)

In this section we will collect commands that were created as we appended the residuals to the graph. We will then use them to create a procedure that will append a plot of residuals to a graph.

1. Click the zoom button (*Macintosh*) or the maximize button (*Windows*) of the

command window to enlarge it to fill the screen.

2. **Find the fifth line from the bottom that reads:**

   ```
   •AppendToGraph/L=Lresid histResids
   ```

3. **Select this line and all the lines below it and press Command-C (*Macintosh*) or Ctrl+C (*Windows*) to copy them to the clipboard.**

4. **Click the zoom button (*Macintosh*) or the restore button (*Windows*) of the command window to return it to its normal size.**

5. **Choose the Windows->New->Procedure menu item.**

6. **Type "Append Residuals" (without the quotes) in the Document Name box and click New.**

   A new procedure window appears. We could have used the always-present standard procedure window, but we will save the procedure window as a stand-alone file.

7. **Add a blank line to the window and then type "Function AppendResiduals()" and press Return or Enter.**

8. **Press Command-V (Macintosh) or Ctrl+V (Windows) to paste the commands from the history into the new window.**

9. **Type "End" and press Return or Enter.**

10. **Select the five lines that you pasted into the procedure window and then choose Edit->Adjust Indentation.**

    This removes the bullet characters from the history and prepends tabs to apply the normal indentation for procedures.

    If you are running on an Asian-language system, you will have asterisks at the start of each line and you must remove them manually.

    Your procedure should now look like this:

    ```
    Function AppendResiduals()
        AppendToGraph/L=Lresid histResids
        ModifyGraph nticks(Lresid)=2,standoff(bottom)=0, axisEnab(left)={0,0.7};DelayUpdat
        ModifyGraph axisEnab(Lresid)={0.8,1}, freePos(Lresid)=0;DelayUpdate
        SetAxis/A/E=2 Lresid
        ModifyGraph mode(histResids)=2,lsize(histResids)=2
    End
    ```

11. **Delete the ";DelayUpdate" at the end of the two ModifyGraph commands.**

    DelayUpdate has no effect in a function.

    We now have a nearly functional procedure but with a major limitation -- it only works if the residuals wave is named "histResids". In the following steps, we will change the function so that it can be used with any wave and also with an XY pair, rather than just with equally-spaced waveform data.

12. **Convert the first two lines to match the following:**

    ```
    Function AppendResiduals(ywave,xwave)
        String ywave,xwave

        if (CmpStr("_calculated_",xwave) == 0)
            AppendToGraph/L=Lresid $ywave
        else
            AppendToGraph/L=Lresid $ywave vs $xwave
        endif
    ```

13. **In the last ModifyGraph command in the function, change both "histResids" to "$ywave".**

    The "$" character is a signal to Igor to take the string expression that follows and convert it into the name of an Igor object.

    Here is the completed procedure:

    ```
    Function AppendResiduals(ywave,xwave)
    ```

```
	String ywave,xwave
	if (CmpStr("_calculated_",xwave) == 0)
		AppendToGraph/L=Lresid $ywave
	else
		AppendToGraph/L=Lresid $ywave vs $xwave
	endif
	ModifyGraph nticks(Lresid)=2,standoff(bottom)=0, axisEnab(left)={0,0.7}
	ModifyGraph axisEnab(Lresid)={0.8,1},freePos(Lresid)=0
	SetAxis/A/E=2 Lresid
	ModifyGraph mode($ywave)=2,lsize($ywave)=2
End
```

Let's try it out.

14. **Click the Compile button at the bottom of the procedure window to compile the function.**

    If you get an error, edit the function text to match the listing above.

15. **Click the close button in the Append Residuals procedure window. A dialog will ask if you want to kill or hide the window. Click Hide.**

    If you press Shift while clicking the close button, the window will be hidden without a dialog. (Use the Help->Shortcuts menu to learn about this and other shortcuts.)

16. **Choose Windows->New Graph.**

17. **Choose fakeY_Hist from the Y Wave(s) list and _calculated_ from the X Wave list and click Do It.**

    A graph without residuals is created.

18. **In the command line, execute the following command:**

    ```
    AppendResiduals("histResids", "_calculated_")
    ```

    The AppendResiduals function runs and displays the residuals in the graph, above the original histogram data.

    Next, we will add a function that displays a dialog so we don't have to type wave names into the command line.

19. **Use the Windows->Procedure Windows menu to show the Append Residuals procedure window.**

20. **Enter the following function, below the end of the AppendResiduals function:**

```
Function AppendResidualsDialog()
	String ywave,xwave

	Prompt ywave,"Residuals Data",popup WaveList("*",";","")
	Prompt xwave,"X Data",popup "_calculated_;"+WaveList("*",";","")
	DoPrompt "Append Residuals", ywave, xwave
	if (V_flag != 0)
		return -1; // User canceled.
	endif
	AppendResiduals(ywave,xwave)
End
```

This function will display a dialog to get parameters from the user and will then call the AppendResiduals function.

Let's try it out.

21. **Click the Compile button at the bottom of the procedure window to make Igor compile our function.**

    If you get an error, edit the function text to match the listing above.

22. **Shift-click the close button to hide the procedure window. Then activate the graph.**

23. **Control-click (*Macintosh*) or right-click (*Windows*) on the residual trace at the top of the graph and select Remove histResids from the pop-up menu.**

    The axis displaying histData will stay short because the residuals were not appended to the graph automatically.

24. **In the command line, execute the following command:**

```
AppendResidualsDialog()
```

The AppendResidualsDialog function displays a dialog to let you choose parameters.

25. **Choose histResids from the Residuals Data pop-up menu.**
26. **Leave the X Wave pop-up set to "_calculated_".**
27. **Click Continue.**

The graph should once again contain the residuals plotted on a new axis above the main data.

Next we will add a menu item to the Macros menu.

28. **Use the Windows->Procedure Windows menu to open the Append Residuals procedure window.**
29. **Enter the following above the AppendResiduals function:**

```
Menu "Macros"
     "Append Residuals...", AppendResidualsDialog()
End
```

30. **Click the Compile button.**

Igor compiles the function and adds the menu item to the Macros menu.

31. **Press Command-E (*Macintosh*) or Ctrl+E (*Windows*) to send the procedure window to the back, and then activate the graph.**
32. **Control-click (*Macintosh*) or right-click (*Windows*) on the residual trace at the top of the graph and select "Remove histResids" from the pop-up menu.**
33. **Click the Macros menu and choose the "Append Residuals" item.**

The procedure displays a dialog to let you choose parameters.

34. **Choose histResids from the Residuals Data pop-up menu.**
35. **Leave the X Wave pop-up menu set to "_calculated_".**
36. **Click Continue.**

The graph should once again contain the residuals plotted on a new axis above the main data.

Saving a Procedure File (Optional)

Demo NOTE: If you are using the demo version of Igor Pro beyond the 30-day trial period, you cannot save a procedure file.

Now that we have a working procedure, let's save it so it can be used in the future. We will save the file in the "Igor Pro User Files" folder - a folder created by Igor for you to store your Igor files.

1. **Choose Help->Show Igor Pro User Files.**

Igor opens the "Igor Pro User Files" folder on the desktop.

By default, this folder has the Igor Pro major version number in its name, for example, "Igor Pro 6 User Files", but it is generically called the "Igor Pro User Files" folder.

Note where in the file system hierarchy this folder is located as you will need to know this in a subsequent step. The default locations are:

```
Macintosh: /Users/<user>/Documents/WaveMetrics/Igor Pro 6 User Files
Windows:   <My Documents>\WaveMetrics\Igor Pro 6 User Files
```

We will save the procedure file in the "User Procedures" subfolder of the Igor Pro User Files folder. You could save the file anywhere on your hard disk, but saving in the User Procedures subfolder makes it easier to access the file as we will see in the next section.

2. **Back in Igor, activate the Append Residuals procedure window again.**
3. **Choose the File->Save Procedure As menu item.**
4. **Enter the file name "Append Residuals.ipf".**
5. **Navigate to your User Procedures folder inside your Igor Pro User Files folder and click Save.**

   The Append Residuals procedure file is now saved in a stand-alone file.

6. **Click the close button on the procedure window.**

   Igor will ask if you want to kill or hide the file. Click Kill. This removes the file from the current experiment, but it still exists on disk and you can open it as needed.

   There are several ways to open the procedure file to use it in the future. One is to double-click it. Another is to choose the File->Open File->Procedure menu item. A third is to put a #include statement in the built-in procedure window, which is how we will open it in the next section.

Including a Procedure File (Optional)

The preferred way to open a procedure window that you intend to use from many different experiments is to use a #include statement. This section demonstrates how to do that.

Demo NOTE: If you are using the demo version of Igor Pro beyond the 30-day trial period, you did not create the Append Residuals.ipf file in the preceding section so you can't do this section. See The Include Statement for details about including procedure files.

1. **In Igor, use the Windows->Procedure Windows menu to open the built-in procedure window.**
2. **At the top of the built-in procedure window, notice the line that says:**

   ```
   #pragma rtGlobals = 1
   ```

   This is technical stuff that you can ignore.

3. **Under the rtGlobals line, leave a blank line and then enter:**

   ```
   #include "Append Residuals"
   ```

4. **Click the Compile button at the bottom of the built-in procedure window.**

   Igor compiles the procedure window. When it sees the #include statement, it looks for the Append Residuals.ipf procedure file in the User Procedures folder and opens it. You don't see it because it was opened hidden.

5. **Use the Windows->Procedure Windows menu to verify that the Append Residuals procedure file is in fact open.**

   To remove the procedure file from the experiment, you would remove the #include statement from the built-in procedure window.

   #include is powerful because it allows procedure files to include other procedure files in a chain. Each procedure file automatically opens any other procedure files it needs.

   User Procedures is special because Igor searches it to satisfy #include statements.

   Another special folder is Igor Procedures. Any procedure file in Igor Procedures is automatically opened by Igor at launch time and left open till Igor quits. This is the place to put procedure files that you want to be open all of the time.

This concludes Guided Tour 3.

• For Further Exploration

We developed the guided tours in this chapter to provide an overview of the basics of using Igor Pro and to give you some experience using features that you will likely need for your day-to-day work. Beyond these fundamentals, Igor includes a wide variety of features to facilitate much more advanced graphing and analysis of your data.

As you become more familiar with using Igor, you will want to further explore some of the additional learning and informational aids that we have included with Igor Pro.

- The Igor Pro manual is installed on your hard disk in PDF format. You can access it through the Igor Help Browser or open it directly from the Manual folder in the Igor Pro Folder.

  The material in the manual is the same as the material in the online help files but is organized in book format and is therefore better suited for linear reading. Unlike the help files, the PDF manual includes an index. You may want to print selected chapters. You can purchase hard copy of the Igor Pro manual from <http://www.lulu.com/wavemetrics>.

  The most important chapters at this point in your Igor learning curve are Chapter II-3, Experiments, Files and Folders and Chapter II-5, Waves. If you want to learn Igor programming, read Chapter IV-1, Working with Commands, Chapter IV-2, Programming Overview, and Chapter IV-3, User-Defined Functions.
- The Igor Help Browser provides online help, including reference material for all built-in operations and functions, an extensive list of shortcuts, and the ability to search Igor help files, procedure files, examples and technical notes for key phrases.
- The Examples folder contains a wide variety of sample experiments illustrating many of Igor's advanced graphing and programming facilities. You can access these most easily through the File->Example Experiments submenus.
- The Learning Aids folder contains additional guided tours and tutorials including a tutorial on image processing. You can access these most easily through the File->Example Experiments->Tutorials submenus.
- A tutorial on 3D visualization can be found in the Visualization help window, accessible through the Windows->Help Windows menu.
- The More Help Files folder contains several supplementary help files. Use the File->Open File->Help Files menu item to open them.
- The WaveMetrics Procedures folder contains a number of utility procedures that you may find useful for writing your own procedures and for your more advanced graphing requirements. For an overview of the WaveMetrics procedures and easy loading of the procedure files, choose Windows->Help Windows->WM Procedures Index.
- The More Extensions folder contains a number of External Operations (XOPs), which add functionality not built into the Igor Pro application. Read the included help files to find out more about the individual XOPs and how to install them, or consult the External Operations Index in the XOP Index help file, which has brief description of each XOP.
- The Technical Notes folder contains miscellaneous additional information and services. Tech Note #000 contains an index to all of the other notes.
- The Igor Mailing List is an Internet mailing list where Igor users share ideas and help each other. See The Igor Mailing List or select the Help->Support menu and then select Igor Mailing List from the Support Options list for information about the mailing list.
- IgorExchange is a user-to-user support web page and a repository for user-created Igor Pro projects. Choose Help->IgorExchange to visit it.