

Chapter
II-8

Data Folders

Overview	116
Data Folder Syntax.....	117
Data Folder Operations and Functions.....	118
Data Folders Reference Functions.....	119
Data Folders and Commands.....	119
Data Folders and User-Defined Functions.....	119
Data Folders and Window Macros	119
Data Folders and Assignment Statements	120
Data Folders and Controls.....	120
Data Folders and Traces.....	121
Using Data Folders	121
Hiding Waves, Strings, and Variables.....	121
Separating Similar Data	121
Using Data Folders Example.....	122
Problems with Data Folders	124
Data Browser	124
Current Data Folder	125
Display Checkboxes	126
Info Checkbox.....	126
Plot Checkbox.....	126
New Folder Button	126
Browse Expt. Button.....	127
Save Copy Button	128
Delete Button.....	128
The Preferences Button	128
The Execute Cmd Button.....	129
Using the Data Browser Find Dialog.....	129
Programming the Browser	130
Browser Pop-Up Menu	130
Other Browser Operations.....	130
Data Browser Shortcuts	131

Overview

Using data folders, you can store your data within an experiment in a hierarchical manner. Hierarchical storage is useful when you have multiple sets of similar data. By storing each set in its own data folder, you can organize your data in a meaningful way and also avoid name conflicts.

Data folders contain four kinds of **data objects**:

- Waves
- Numeric variables
- String variables
- Other data folders

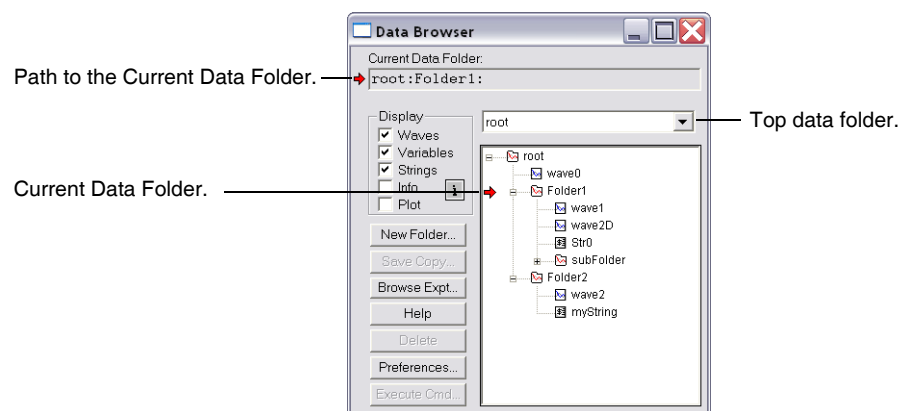
Igor's data folders are very similar to a computer's hierarchical disk file system except they reside wholly in memory and not on disk. This similarity can help you understand the concept of data folders but you should take care not to confuse them with the computer's folders and files.

Data folders are particularly useful when you conduct several runs of an experiment. You can store the data for each run in a separate data folder. The data folders can have names like "run1", "run2", etc., but the names of the waves and variables in each data folder can be the same as in the others. In other words, the information about which run the data objects belong to is encoded in the data folder name, allowing the data objects themselves to have the same names for all runs. You can write procedures that use the same wave and variable names regardless of which run they are working on.

Data folders are very handy for programmers who need to create temporary waves during a procedure. You can create a temporary data folder with a name designed not to conflict with any other Igor object names and then create waves without having to worry about conflict. When done, you can kill the data folder and everything it contains with a single command rather than having to kill waves, variables, and strings. You can also use a data folder to store persistent waves and global variables not intended to be seen by the end user. As a programmer, you can use nested data folders as a sort of data structure.

All operations in Igor Pro are data-folder aware. On the command line you can specify waves from several different data folders within one command.

You can use the Data Browser window (Data menu) to see the data folder hierarchy and to set the current data folder:



You can use the Data Browser not only to see the hierarchy and set the current data folder but also to:

- Create new data folders.
- Move, duplicate, rename and delete objects.
- Browse other Igor experiment files and load data from them into memory.
- Save a copy of data in the current experiment to an experiment file or folder on disk.
- See and edit the contents of variables, strings or waves in the information pane by selecting an object

- See a simple plot of 1D or 2D waves by selecting one wave at a time in the main list while the Plot pane is visible.
- See a simple plot of a wave while browsing other Igor experiments.
- See variable, string and wave contents by double-clicking their icons.
- See a simple histogram or wave statistics for one wave at a time.

Before using data folders, be sure to read **Using Data Folders** on page II-121, and **Problems with Data Folders** on page II-124.

Programmers should read **Programming with Data Folders** on page IV-148.

A similar browser is used for wave selection in dialogs. For details see **Dialog Wave Browser** on page II-175.

Data Folder Syntax

Data folders are named objects like other Igor objects such as waves and variables. Data folder names follow the same rules as wave names. See **Liberal Object Names** on page III-412.

Like the Macintosh file system, Igor Pro's data folders use the colon character (:) to separate components of a path to an object. This is analogous to Unix which uses / and Windows which uses \. (**Reminder:** Igor's data folders exist wholly in memory while an experiment is open. It is not a disk file system!)

A data folder named "root" always exists and contains all other data folders.

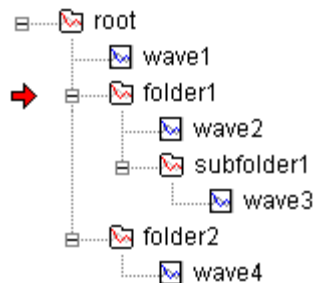
A given object can be specified in a command using:

- A full path
- A partial path
- Just the object name

The object name alone can only be used when the current data folder contains the object.

A full path starts with "root" and does not depend on the current data folder. A partial path starts with ":" and is relative to the current data folder.

Assume the data folder structure shown below, where the arrow indicates that folder1 is the current data folder.



Each of the following commands creates a graph of one of the waves in this hierarchy:

```

Display wave2
Display :subfolder1:wave3
Display root:folder1:subfolder1:wave3
Display ::folder2:wave4
  
```

The last example illustrates the rule that you can use multiple colons to walk back up the hierarchy: from folder1 (the current data folder), up one level, then down to folder2 and wave4. Here is another valid (but silly) example:

```

Display root:folder1:subfolder1:::folder2:wave4
  
```

Chapter II-8 — Data Folders

Occasionally you will need to specify a data folder itself rather than an object in a data folder. In that case, just leave off the object name. The path specification should therefore have a trailing colon. However, Igor will generally understand what you mean if you forget the trailing colon.

If you need to specify the current data folder, you can use just a single colon. For example:

```
KillDataFolder :
```

kills the current data folder (and all its contents) and then sets the current data folder to the parent of the current. Nonprogrammers might prefer to use the Data Browser to delete data folders.

Recall that the \$ operator converts a string expression into a single name. Since data folders are named, the following is valid:

```
String df1 = "folder1", df2="subfolder1"  
Display root:$(df1):$(df2):wave3
```

This is a silly example but the technique would be useful if df1 and df2 were parameters to a procedure.

Note that parentheses must be used in this type of statement. That is a result of the precedence of \$ relative to .

When used at the beginning of a path, the \$ operator works in a special way and can (and must) be used on the entire path:

```
String path1 = "root:folder1:subfolder1:wave3"  
Display $path1
```

When liberal names are used within a path, they must be in single quotes. For example:

```
Display root:folder1:'subfolder 1':'wave 3'  
String path1 = "root:folder1:'subfolder 1':'wave 3'"  
Display $path1
```

However, when a simple name is passed in a string, single quotes must not be used:

```
Make 'wave 1'  
String name  
name = "'wave 1'"           // Wrong.  
name = "wave 1"           // Correct.  
Display $name
```

Data Folder Operations and Functions

Most people will use the Data Browser (Data menu) to create, view and manipulate data folders. The following operations will be mainly used by programmers, who should read **Programming with Data Folders** on page IV-148.

```
NewDataFolder path  
SetDataFolder path  
KillDataFolder path  
DuplicateDataFolder srcPath, destPath  
MoveDataFolder srcPath, destPath  
MoveString srcPath, destPath  
MoveVariable srcPath, destPath  
MoveWave wave, destPath [newname]  
RenameDataFolder path, newName  
Dir
```

The following are functions that are used with data folders.

```
GetDataFolder (mode)  
CountObjects (pathStr, type)  
GetIndexedObjName (pathStr, type, index)  
GetWavesDataFolder (wave, mode)
```

`DataFolderExists` (*pathStr*)
`DataFolderDir` (*mode*)

Data Folders Reference Functions

As of Igor Pro 6.1, function programmers can utilize data folder references in place of paths. Data folder references are lightweight objects that refer directly to a data folder whereas a path, consisting of a sequence of names, has to be looked up in order to find the actual target folder.

Here are functions that work with data folder references:

`GetDataFolderDFR` ()
`GetIndexedObjNameDFR` (*dfr*, *type*, *index*)
`GetWavesDataFolderDFR` (*wave*)
`CountObjectsDFR` (*dfr*, *type*)
`DataFolderRefStatus` (*dfr*)
`NewFreeDataFolder` ()

For information on programming with data folder references, see **Data Folder References** on page IV-63.

Data Folders and Commands

Igor normally evaluates commands in the context of the current data folder. This means that, unless qualified with a path to a particular data folder, object names refer to objects in the *current* data folder. For example:

```
Macro MyMacro()
    Make wave1
    Variable/G myGlobalVariable
EndMacro
```

creates `wave1` and `myGlobalVariable` in the current data folder. Likewise executing:

```
WaveStats wave1
```

creates `WaveStats` output variables (`V_avg`, etc.) in the current data folder.

Data Folders and User-Defined Functions

You must exercise some care when accessing global variables from “data-folder ignorant” user-defined functions. See **Accessing Global Variables and Waves** on page IV-52 for details.

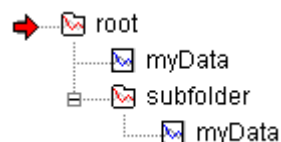
Data Folders and Window Macros

Window macros are evaluated in the context of the *root* data folder. Window macros begin with the `Window` keyword, as in the example below. Macros that begin with the “`Macro`” or `Proc` keywords evaluate their commands in the context of the *current* data folder.

Evaluating window macros this way ensures that a window is recreated correctly regardless of the current data folder, and provides some compatibility with window macros created with prior versions of Igor Pro which didn’t have data folders.

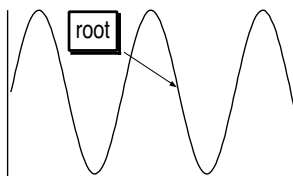
This means that object names within window macros or functions that don’t explicitly contain a data folder path refer to objects in the root data folder. This is important when the current data folder is not the root data folder.

For example, given identically named waves organized as follows:



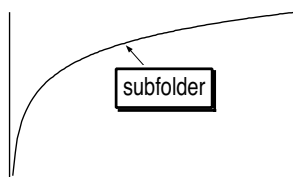
Chapter II-8 — Data Folders

A window recreation macro for a graph of root:myData (whose tag shows the wave's data folder) will resemble the following:



```
Window Graph0() : Graph
  PauseUpdate; Silent 1 // building window...
  Display myData // note: no data folder specified
  Tag/N=text0/X=21.15/Y=35.00 myData, 50
  AppendText/N=text0 "\\{\\"%s\",GetWavesDataFolder(TagWaveRef(),0)}"
EndMacro
```

Observe that myData is referred to in the Display command without its data folder (root:myData would be the fully qualified name of the wave object). If you change the current data folder to the subfolder and run the window macro, the resulting graph will be identical because the myData wave in the root data folder would be graphed.



```
Window Graph1() : Graph
  PauseUpdate; Silent 1 // building window...
  String fldrSav= GetDataFolder(1)
  SetDataFolder root:subfolder: // note: data folder is specified
  Display myData
  SetDataFolder fldrSav
  Tag/N=text0/X=21.15/Y=35.00 myData, 50
  AppendText/N=text0 "\\{\\"%s\",GetWavesDataFolder(TagWaveRef(),0)}"
EndMacro
```

Data Folders and Assignment Statements

Wave and variable assignment statements are evaluated in the context of the data folder containing the wave or variable on the left-hand side of the statement:

```
root:subfolder:wave0 = wave1 + var1
```

is a shorter way of writing the equivalent:

```
root:subfolder:wave0 = root:subfolder:wave1 + root:subfolder:var1
```

This rule also applies to dependency formulae which use := instead of = as the assignment operator.

Data Folders and Controls

ValDisplay controls evaluate their value expression in the context of the root data folder.

SetVariable controls remember the data folder in which the controlled global variable exists, and continue to function properly when the current data folder is different than the controlled variable.

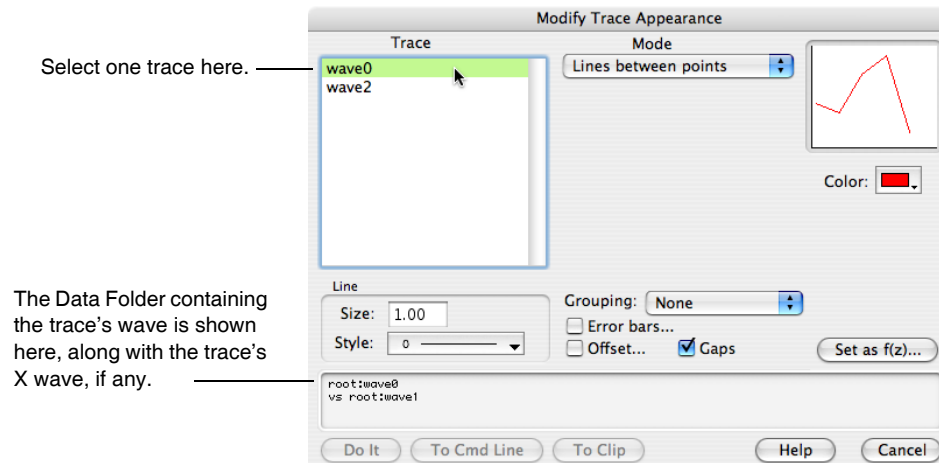
Note: The system variables (K0 through K19) belong to no particular data folder (they are available from any data folder), and there is only *one* copy of these variables. If you create a SetVariable controlling K0 while the current data folder is "aFolder", and another SetVariable controlling K0 while the current data folder is "bFolder", *they are actually controlling the same K0.*

See Chapter III-14, **Controls and Control Panels**, for details about controls.

Data Folders and Traces

You cannot tell by looking at a trace in a graph which data folder it resides in. You could save and examine the graph window recreation macro. The easiest way to find out what data folder a trace's wave resides in is to use the trace info help. On Macintosh, press Command-Option-Control and click on the trace in the graph window. On Windows, press Shift+F1 to summon context-sensitive help and then click on the trace to get trace info.

Another method is to use the Modify Trace Appearance dialog. When you press and hold down the mouse button on a trace in the dialog's Trace list, Igor displays data folder (and X wave) information where the commands are usually shown:



Using Data Folders

You can use data folders for many purposes, just like you use the folders on your hard disk for organizing files in many different ways. Here are two common uses of data folders.

Hiding Waves, Strings, and Variables

Sophisticated Igor procedures may need a large number of global variables, strings and waves that aren't intended to be directly accessed by the user. The programmer who creates these procedures should keep all such items within data folders they create with unique names designed not to conflict with other data folder names.

Users of these procedures should leave the current data folder set to the data folder where their raw data and final results are kept, so that the procedure's globals and waves won't clutter up the dialog lists.

Programmers creating procedures should read **Programming with Data Folders** on page IV-148.

Separating Similar Data

One situation that arises during repeated testing is needing to keep the data from each test "run" separate from the others. Often the data from each run is very similar to the other runs, and may even have the same name. Without data folders you would need to choose new names after the first run.

By making one data folder for each test run, you can put all of the related data for one run into each folder. The data can use identical names, because other identically named data is in different data folders.

Using data folders also keeps the data from various runs from being accidentally combined, since only the data in the current data folder shows up in the various dialogs or can be used in a command without a data folder name.

The Wavemetrics-supplied "Multi-peak Fitting" example experiment's procedures work this way: they create data folders to hold separate peak curve fit runs and global state information.

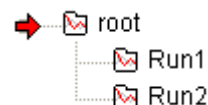
Using Data Folders Example

This example will use data folders to:

- load data from two test runs into separate data folders
- create graphs showing each test run by itself
- create a graph comparing the two test runs

First we'll use the Data Browser to create a data folder for each test run.

Open the Data Browser (in the Data menu), and set the Current Data Folder to root. Click the root data folder, and click the New Folder button. Enter "Run1" for the new data folder's name. Click New Folder again and enter "Run2". The Data Browser window should resemble the one shown here.

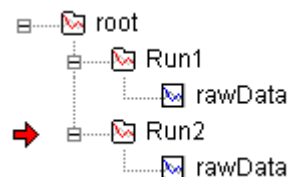


Now let's load sample data into each data folder, starting with Run1.

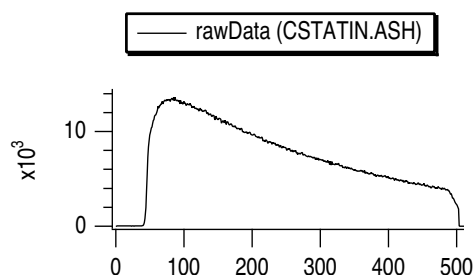
Set the Current Data Folder to Run1, then choose Load Delimited Text from the Data menu's Load Data submenu. Select the CSTATIN.ASH file from the Sample Data subfolder of the Learning Aids folder, and click Open. In the resulting Load Waves dialog, name the loaded wave "rawData". We will pretend this data is the result of Run 1. Type "Display rawData" on the command line to graph the data.

Set the Current Data Folder to Run2, and repeat the wave loading steps, selecting the CSTATIN.ASV file instead. In the resulting Load Waves dialog, name the loaded wave "rawData". We will pretend this data is the result of Run 2. Repeat the "Display rawData" command to make a graph of this data.

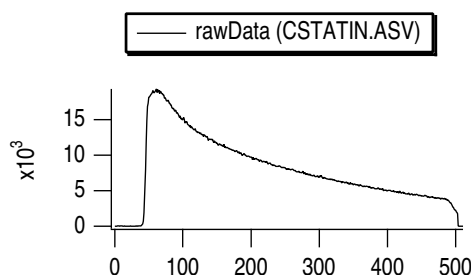
Notice that we used the same name for the loaded data. No conflict exists because the other rawData wave is in another data folder. At this point, the Data Browser should look something like this example (we've deselected Display Variables and Display Strings).



The graphs of our loaded waves look like this:

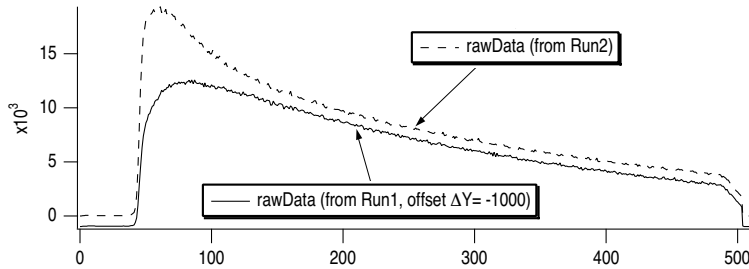


Graph of data in data folder "Run1"



Graph of data in data folder "Run2"

You can easily make a graph displaying both rawData waves to compare them better. Using the New Graph dialog, make sure Show Data Folders is selected in the Wave Browsers (see **Dialog Wave Browser** on page II-175). You can then select both waves to display in a graph. Alternatively, you can execute two commands on the Command Line: first execute `Display rawData`, change the current data folder to Run1, and then execute `AppendToGraph rawData` (or use the Append To Graph dialog)



You can change the current data folder to anything you want and the graphs will continue to display the same data; graphs remember which data folder the waves belong to, and so do graph recreation macros. This is often what you want, but not always.

Suppose you have many test runs in your experiment, each safely tucked away in its own data folder, and you want to “visit” each test run by looking at the data using a single graph which displays data from the test run’s data folder only. When you visit another test run, you want the graph to display data from that other data folder only.

Additionally, suppose you want the graph characteristics to be the same (the same axis labels, annotations, line styles and colors, etc.). You could:

- Create a graph for the first test run
- Kill the window, and save the graph window macro.
- Edit the recreation macro to reference data in another data folder.
- Run the edited recreation macro.

The recreated graph will have the same appearance, but use the data from the other data folder. The editing usually involves changing a command like:

```
SetDataFolder root:Run1:
```

to:

```
SetDataFolder root:Run2:
```

If the graph displays waves from more than one data folder, you may need to edit commands like:

```
Display rawData,::Run1:rawData
```

as well.

However, there is another way that doesn’t require you to edit recreation macros: use the **ReplaceWave** operation to replace waves (traces) in the graph with waves from the other folder.

- Switch to the other data folder.
- Select the desired graph
- Type in the command line:

```
ReplaceWave allinCDF
```

This replaces all the waves in the graph with identically named waves from the Current Data Folder, if they exist. There is no dialog for this command; see the **ReplaceWave** operation on page V-504 for more details.

Though we have only one wave, we can try it out:

- Set the Current Data Folder to Run1.
- Select the graph showing data from Run2 only (CSTATIN.ASV).
- Type in the command line:

```
ReplaceWave allinCDF
```

The graph will be updated to show the **rawData** wave from Run1.

You could create a Button control in the graph (see **Button** on page III-361) that executes a macro containing the ReplaceWave allinCDF command. Then you would use the Data Browser to change the Current Data Folder, and click the button to update the graph with waves from that data folder. You could also execute the same macro directly from the Data Browser in response to the user dragging the current folder indicator. To do so, use the command:

```
ModifyBrowser command3="ReplaceWave allinCDF"
```

For another Data Folder example, see the Data Folder Tutorial in “Igor Pro Folder:Learning Aids:Tutorials”.

Problems with Data Folders

If you are a nonprogrammer and do not use procedures written by others then you can probably use data folders without problems. Just be aware that you need to set the current data folder (using the Data Browser) to the data folder of interest and Igor will behave as if the other data folders do not exist.

If you are a programmer and have written your own procedures, you can use data folders *after* you have made your procedures data-folder aware. However, rewriting legacy code to be data folder aware can be a big job and you should make sure the benefits will outweigh the costs before undertaking such a project.

Nonprogrammers who use procedures written by others should avoid data folders until the procedures are updated.

Igor procedures written for versions of Igor prior to Igor Pro 3.1 may not work properly if the current data folder is not root and yet will not be able to access data in other data folders if the current data folder is set to root. If you rely on procedures (and even some XOPs and XFUNCS) that are not data-folder aware, you should do some testing to verify that they work properly before committing to data folder use.

Procedures that rely on global variables and waves are likely to fail when the current data folder is not root. Unfortunately this is a very common occurrence. The reason for this is that non data-folder-aware procedures refer to waves and variables with simple object names (no data folder paths). Igor will look in the current data folder for objects that are actually in the root data folder. If the current data folder is not root, Igor will not find the named objects and will generate an error.

Also with the introduction of data folders, the name of a wave is no longer sufficient to uniquely identify it because the name does not tell you in which data folder the wave can be found (and waves with the same name can exist in different data folders). For a discussion of how to deal with this problem, see **Wave Reference Functions** on page IV-168.

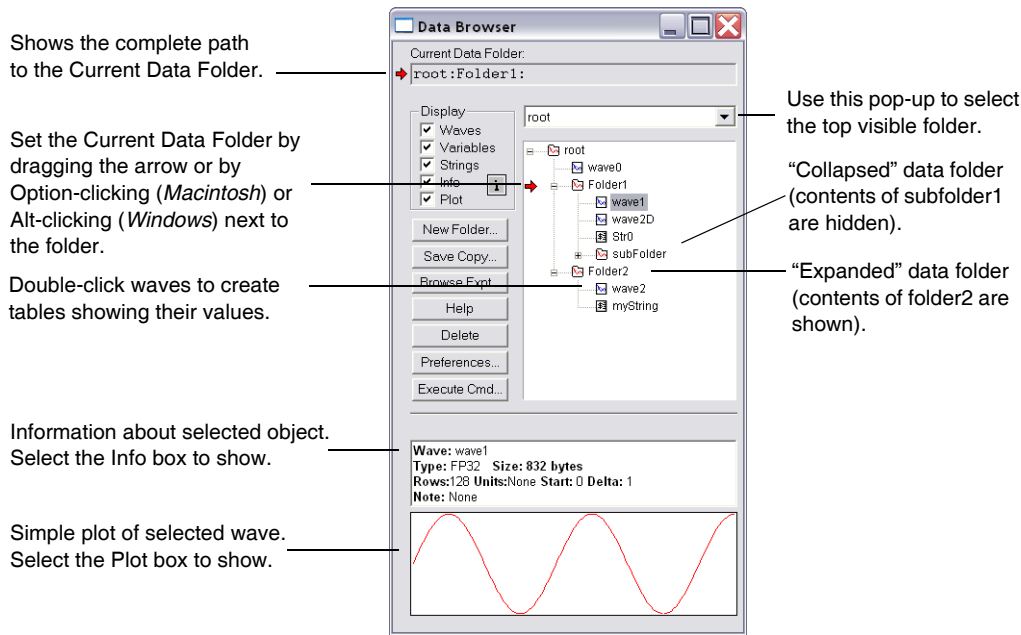
Note that the window recreation macros generated by Igor itself when you click the close button of a graph or table window are data-folder aware and will work properly regardless of the current data folder setting.

Data Browser

The Data Browser is an extension that lets you navigate through the different levels of data folders, examine values of variables, strings and waves, load data objects from other Igor experiments, and save a copy of data from the current experiment to an experiment file or folder on disk.

To open the browser choose Data Browser from the Data menu.

The user interface of the browser is similar to that of the computer desktop. The basic Igor data objects (variables, strings, waves and data folders) are represented by unique icons and arranged in the main list based on their hierarchy in the current experiment. The browser also sports several buttons that provide you with additional functionality:

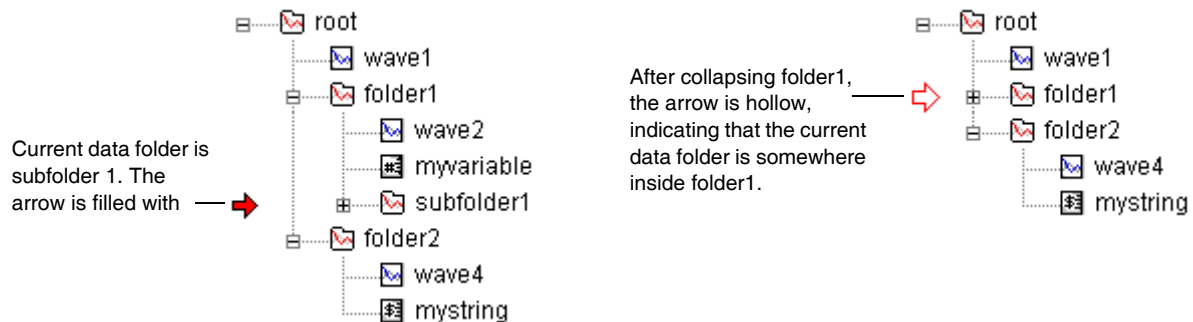


The main list occupies most of the browser when it is first invoked. At the top of the data tree is the root data folder which by default appears expanded. By double-clicking a data folder icon you can change the display so that the tree is displayed with your selection as the top data folder instead of root. You can use the pop-up menu above the main list to replace the current top data folder with another folder in the hierarchy. Following the top folder are all the data objects that it contains. Objects are grouped by type and by default they are listed in the order that they were created.

Current Data Folder

The “current data folder” is the folder that Igor uses by default for storing newly-created variables, strings, waves and other data folders. There are two indicators for the current data folder. First, above the main list there is a text box that contains the full path to the current data folder. Second, the main list has a painted red arrow to the left of the icon for the current data folder.

When the current data folder is contained inside a collapsed data folder, an unpainted (empty) arrow indicator points to the icon of the data folder containing the current data folder.



To set the current data folder, drag the current-folder indicator (red arrow) until it points to the desired data folder. You can also set the current data folder directly by clicking next to the desired data folder while pressing Option (Macintosh) or Alt (Windows).

This “skeleton” arrow indicates you can’t set the current data folder here (it is pointing at a wave, not a data folder).



Display Checkboxes

The Display checkboxes group lets you determine which object types are shown in the main list. Data folders are always shown.

Info Checkbox

Click in the Info Checkbox to display the Information pane of the Data Browser. The Information pane is situated below the main list. When you select a data object in the main list, its properties or contents appear in the information pane. For example, when you select a variable, its name is displayed in bold face and its value is displayed below the name. You can edit the numerical value by selecting it and typing in a new numerical value. If you modify the value of the variable, Accept and Cancel buttons will appear above the Information pane. You must either accept the change or cancel it before doing anything else with Igor.

When you select a string in the main list, the contents of the string (up to 32000 characters) will be displayed in the Information pane. Longer strings will be clipped. You can then select and edit any part of the string.

If you select a wave in the main list, the Information pane displays the wave type, size, dimensions, units, start, delta and note. Each one of these fields is displayed as a bold face name followed by plane text value. You can select and modify each one of the plane text fields by typing the new values. The only exception here is the wave type field, where you need to Control-click (*Macintosh*) or right-click (*Windows*) to select a new wave type from a pop-up menu. Note that when you change the wave type or any one of its dimensions, you might irreversibly change your data.

Another option offered by the Information pane is to display WaveStats for any selected wave. The **WaveStats** operation on page V-703 provides several statistical properties of a wave. To show WaveStats, click the sigma icon next to the Info checkbox. Note that WaveStats calculations are performed in the background and should not affect your interaction with or the performance of Igor. When you click the sigma icon, it changes to an *i* icon which you can click to return to normal mode.

Plot Checkbox

Click the Plot Checkbox to display the Plot pane of the Data Browser. The plot pane is situated below the main list and the optional Information pane. It displays a small graph or image of a wave selected in the main list above it.

Simple 1D real waves are drawn in red on a white background. Complex 1D waves are drawn as two traces with the real part drawn in red and the imaginary in blue. 2D waves are drawn as an image that by default is scaled to the size of the Plot pane and uses the Rainbow color table. To display the image using the aspect ratio implied by the number of samples in each direction or to change the color table, Control-click in the Plot pane (*Macintosh*) or right-click (*Windows*) and make the appropriate choice in the pop-up menu.

When you select 3D or 4D wave in the main list, the Plot pane displays animated images of one slice at a time. The slices represent layers relative to the data cube (3D) or the selected chunk (4D). You can stop the animation at any time by selecting from a pop-up menu in the Plot pane. You can also select the Plot pane by clicking in it and then toggling the animation by pressing Enter. When the animation is stopped you can use the cursor keys to navigate through layers and chunks.

New Folder Button

The New Folder button is used to create a new data folder inside the current data folder. The browser provides a simple dialog for specifying the name of the new data folder and tests that the name provided is valid. When entering liberal object names, you should not use single quotes around the name.

Browse Expt. Button

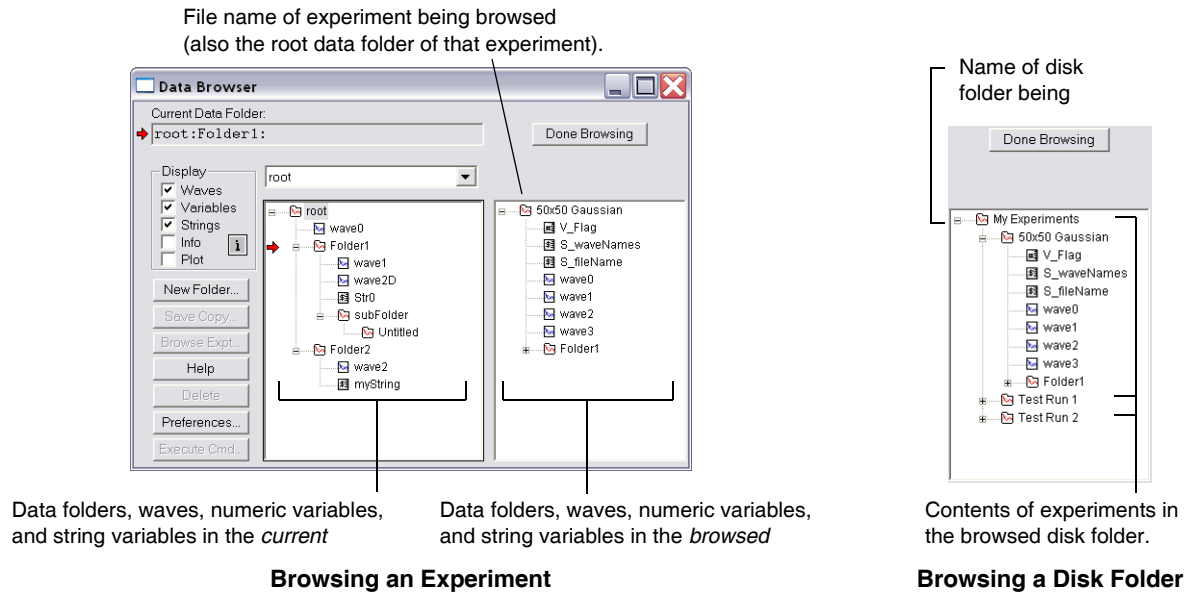
The Browse Expt. button loads data objects from Igor packed or unpacked experiments into the current experiment (in memory). When you click Browse Expt., the browser presents the standard Open dialog. You can choose to browse a packed Igor experiment file or to browse a folder on your hard disk and any subfolders.

To browse a disk folder, select the folder and click the Folder button.

When you browse a disk folder, the browser shows you all packed Igor experiment files or unpacked Igor data files in the selected folder as well as in any subfolders.

To browse a packed experiment file, select the file in the dialog and click the Choose button.

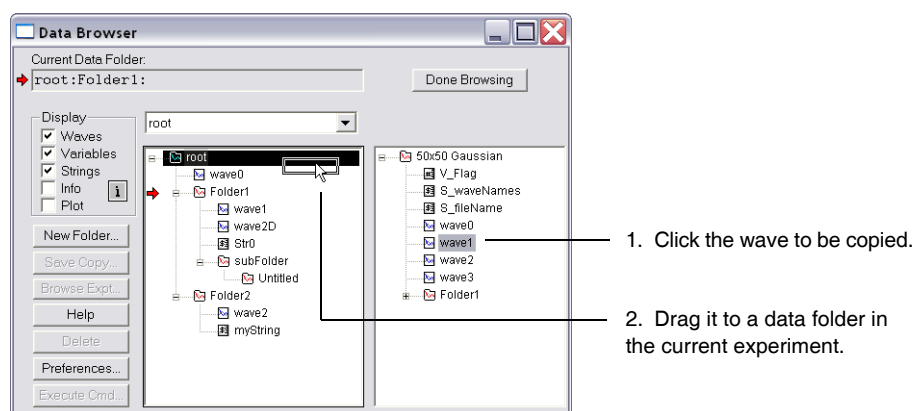
At this stage, the browser will display, on the right-hand side, a new list containing icons representing the data in the file or folder that you selected for browsing.



Each experiment in the new list is represented by a data folder which may contain any number of data folders and data objects.

Note: Although data folders exist wholly in memory while an experiment is active, *unpacked* experiments create a disk folder hierarchy that mirrors the data folder hierarchy. *Packed* experiments do not create a disk folder hierarchy at all. The Data Browser displays a saved experiment's data folders by examining either the packed experiment's file contents or the unpacked experiment's disk hierarchy.

You may select one or more data objects and drag them to the main list. When the cursor appears on top of a valid drop target (a data folder in the current experiment), the target is highlighted. When you release the mouse button on a valid drop target, the browser loads the corresponding data objects into the specified data folder. There is no change to the experiment from which the data is loaded.



Copying a wave from a browsed experiment into the current experiment

Clicking in the Done Browsin' button removes the additional list and resets the browser window to its size prior to the load operation.

Save Copy Button

The Save Copy button copies data objects from the current experiment to an Igor packed experiment file or to an unpacked folder on disk. Most users will not need to do this because the data will be saved when the current experiment is saved.

Before clicking Save Copy, select the data that you want to save. When you click Save Copy the browser presents a dialog in which you specify the name and location of the packed Igor experiment file which will contain a copy of the saved data.

If you press Option (*Macintosh*) or Alt (*Windows*) while clicking Save Copy, the browser presents a dialog in which you specify a folder on disk in which the data is to be saved in unpacked format. The unpacked format is intended for advanced users with specialized applications.

By default, objects are written to the output without regard to the state of the Waves, Variables and Strings checkboxes in the Display section of the Data Browser. However, there is a preference that you use change this behavior. If you enable the appropriate checkbox in the Data Browser preferences dialog, then Save Copy writes a particular type of object only if the corresponding Display checkbox is selected.

The Data Browser does not provide a method for adding or deleting data to or from a packed experiment file on disk. It can only overwrite an existing file. To add or delete, you need to open the experiment (Open Experiment in the File menu), make additions and deletions and then save the experiment. Advanced users can add data to an unpacked folder using the **SaveData** operation on page V-515.

Delete Button

The Delete button is enabled whenever data objects are selected in the main list. The browser provides a warning message listing the number of items that will be deleted. Note that clicking this button when the root folder is selected deletes all data objects in the current experiment.

Note that if you try to delete a wave that's displayed in a graph or table, it will not be deleted and you will not get an error message.

To skip the warnings, press Option (*Macintosh*) or Alt (*Windows*) when clicking the Delete button.

Warning: If you mistakenly delete something, you cannot undo it except by reverting the entire experiment to its last saved state.

The Preferences Button

The Preferences button sets the following:

- The font and font size used in the browser's directory window.

- Whether the browser will remember its window size and position when you relaunch Igor.
- The order of objects in data folders. You can choose to sort them by creation date, by name or by name and type. If you choose Creation date, objects are ordered according to the time they were created, but they are grouped according to type with waves appearing first followed by variables and strings. If you choose to order objects by name only, objects are arranged in alphabetical order within each data folder. Data folders always appear based on the tree structure and the order in which they were created.
- Whether the Save Copy button affects only the currently visible objects.

The Execute Cmd Button

The Execute Cmd button provides you with a shortcut for executing a command on selected objects in the Data Browser window. When you click in the button you get a dialog where you can specify the command, the execution mode and a secondary command for an overflow. If you set the commands once, you can skip the dialog by pressing Option when you click in the button.

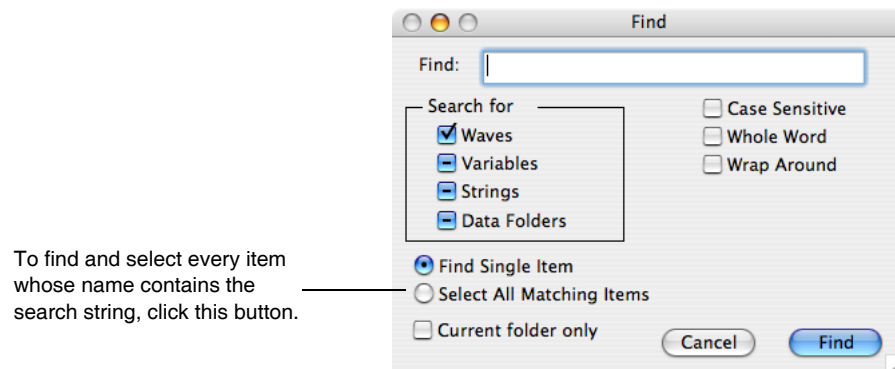
The format of commands is exactly the same as any Igor commands except that you use %s where the selection is to be inserted, e.g.,

```
Display %s
Print "%s"
```

When “Execute once for all selected items” is chosen, the Data Browser enters the full path for each selection in place of %s. This may cause the command to exceed the maximum of 400 characters. Before that limit is reached, the Data Browser executes the first command on the selection followed by executing the overflow command on the remaining objects in the selection. For example, if you want to display many waves use “Display %s” for your first command and “AppendToGraph %s” as the overflow command. The command syntax should not include printf, sprintf or sscanf because of conflict between the formatting string and the %s used here.

Using the Data Browser Find Dialog

If you choose the Find item in the Edit menu, the Data Browser displays a Find dialog. This dialog finds waves, variables and data folders that might be buried in subdata folders. It also provides a convenient way to select a number of objects at one time, based on a search string. Any object with a name containing your search string will be found and selected. You can then use the Execute Cmd button to operate on the selection.



The Data Browser’s Find dialog specifies the objects that you would like to find or select. You may use the “*” wildcard to specify object names of the form “abc*ef” where * represents zero or more arbitrary characters. Note that “abc*”, “*abc” and “abc” are completely equivalent.

Choosing Find Same in the Edit menu or pressing Command-G (*Macintosh*) or Ctrl+G (*Windows*) performs a search in the forward direction for an item matching the same search string as was used in the previous Find. When the search reaches the end of the data objects list, it will wrap around only if the wrap around box is selected in the find dialog.

Chapter II-8 — Data Folders

Choosing Find Selection in the Edit menu or pressing Command-H (*Macintosh*) or Ctrl+H (*Windows*) searches for an item matching the first item that is currently selected in the main list. All other search settings are those specified in the Find dialog.

Programming the Browser

The Data Browser can be controlled from the command line or from Igor procedures. Detailed reference information about the CreateBrowser, ModifyBrowser, and GetBrowserSelection commands can be found in the Command Help tab of the Igor Help Browser.

Advanced Igor programmers can use the browser as an input device via the GetBrowserSelection function or by modifying stored command strings. For an example, see the Data Folder Tutorial in “Igor Pro Folder:Learning Aids:Tutorials”.

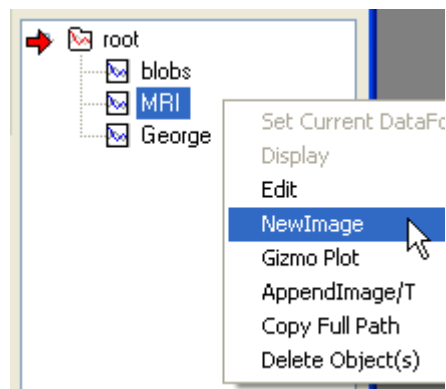
You can use the Data Browser as a modal dialog permitting a user to select one or more waves from multiple data folders. For details, see the Data Browser help file.

Browser Pop-Up Menu

You can apply various Igor operations to objects by selecting the objects in the Data Browser and choosing the operation from a pop-up menu you obtain by Control-clicking (*Macintosh*) or right-clicking (*Windows*).

Using the Display and New Image pop-up items, you can create a new graph or image plot of the selected wave. You can also select multiple waves, in the same or different data folders, to display together in the same graph.

The Copy Full Path item copies the complete data folder paths of the selected objects to the clipboard.



Other Browser Operations

You can rename data objects by clicking the name of the object and editing the name.

The browser also supports icon dragging as means of moving or copying data objects from one data folder to another. You can select multiple data objects by Shift-clicking (*Macintosh*) or Ctrl-clicking (*Windows*) on them.

You can move data objects from one data folder to another by dragging them.

You can copy data objects from one data folder to another by holding down Option (*Macintosh*) or Alt (*Windows*) while dragging.

You can duplicate data objects within a folder by choosing Duplicate from the Edit menu or by pressing Command-D (*Macintosh*) or Ctrl+D (*Windows*).

Note: Objects remain selected even when they are hidden inside collapsed data folders. If you select a wave, collapse its data folder, Shift-select another wave, and drag it to another data folder, both waves will be moved there.

However, when a selected object is hidden by deselecting the relevant Display checkbox, no action (e.g., delete or duplicate) is taken upon it except if you use Save Copy and your preference setting is to save nonvisible objects.

Data Browser Shortcuts

Action	Shortcut
To set the current data folder	Drag the red arrow until it points to the desired data folder, or Option-click (<i>Macintosh</i>) or Alt-click (<i>Windows</i>) next to the desired data folder.
To display a graph or an image of a wave	Control-click (<i>Macintosh</i>) or right-click (<i>Windows</i>) and select an option from the pop-up menu.
To view the contents of a “collapsed” data folder	Click the triangle (<i>Macintosh</i>) or the plus button (<i>Windows</i>) next to the data folder.
To “collapse” a data folder	Click the triangle (<i>Macintosh</i>) or the minus button (<i>Windows</i>) next to the data folder.
To move the selection up or down by one object	Press Up Arrow or Down Arrow.
To move an object from one data folder to another	Drag the object onto the destination data folder.
To move several objects from one data folder to another	Select the objects by Shift-clicking them (<i>Macintosh</i>) or Ctrl-clicking them (<i>Windows</i>). Drag the selected objects onto the desired data folder.
To copy an object from one data folder to another	Drag the object while holding down Option (<i>Macintosh</i>) or Alt (<i>Windows</i>).
To duplicate an object	Select the object and press Command-D (<i>Macintosh</i>) or Ctrl+D (<i>Windows</i>).
To rename an object	Click the object’s name and type a new name. To finish, press Return, Enter, Tab, or click outside the name.
To view a wave’s values in a table	Double-click the wave’s icon.
To print the value of a variable or string in the history area	Double-click the variable or string icon.
To delete an object without the confirmation dialog	Press Option (<i>Macintosh</i>) or Alt (<i>Windows</i>) while clicking the Delete button.
To find objects in the browser list	Choose Find from the Edit menu, or press Command-F (<i>Macintosh</i>) or Ctrl+F (<i>Windows</i>).
To find the same thing again	Choose Find Same from the Edit menu, or press Command-G (<i>Macintosh</i>) or Ctrl+G (<i>Windows</i>).
To find names containing selected text	Choose Find Selection from the Edit menu, or press Command-H (<i>Macintosh</i>) or Ctrl+H (<i>Windows</i>).
To execute a command on a set of waves	Select the icon for each wave that the command is to act on and click the Execute Cmd button.
To execute a command on a set of waves without going through the Execute Cmd dialog	Select the icon for each wave that the command is to act on, press Option (<i>Macintosh</i>) or Alt (<i>Windows</i>), and click the Execute Cmd button. This reexecutes the command entered previously in the Execute Cmd dialog.
To see WaveStats for a selected wave	Control-click (<i>Macintosh</i>) or right-click (<i>Windows</i>) in the information pane and select the WaveStats mode from the pop-up menu.
To change colormap, activate animation in plot pane	Control-click (<i>Macintosh</i>) or right-click (<i>Windows</i>) in the plot pane and select the appropriate option from the pop-up menu.

Chapter II-8 — Data Folders

Action	Shortcut
To stop the animation in the plot pane	Click in the plot pane and then press Return or Enter.
To navigate between displayed layers or chunks	Stop the animation in the plot pane and use the cursor keys, Page Down, Page Up, Home, and End.
To close the information pane	Double click the horizontal separator bar.
To save a copy of selected data in unpacked format	Press Option (<i>Macintosh</i>) or Alt (<i>Windows</i>) while clicking the Save Copy button.
