# Getting Results with HiQ™

June 2000 Edition
Part Number 321886C-01

**Worldwide Technical Support and Product Information**

www.ni.com

**National Instruments Corporate Headquarters**

11500 North Mopac Expressway  Austin, Texas 78759-3504  USA  Tel: 512 794 0100

**Worldwide Offices**

Australia 03 9879 5166, Austria 0662 45 79 90 0, Belgium 02 757 00 20, Brazil 011 284 5011,
Canada (Calgary) 403 274 9391, Canada (Ontario) 905 785 0085, Canada (Québec) 514 694 8521,
China 0755 3904939, Denmark 45 76 26 00, Finland 09 725 725 11, France 01 48 14 24 24,
Germany 089 741 31 30, Greece 30 1 42 96 427, Hong Kong 2645 3186, India 91805275406,
Israel 03 6120092, Italy 02 413091, Japan 03 5472 2970, Korea 02 596 7456, Mexico (D.F.) 5 280 7625,
Mexico (Monterrey) 8 357 7695, Netherlands 0348 433466, New Zealand 09 914 0488, Norway 32 27 73 00,
Poland 0 22 528 94 06, Portugal 351 1 726 9011, Singapore 2265886, Spain 91 640 0085,
Sweden 08 587 895 00, Switzerland 056 200 51 51, Taiwan 02 2528 7227, United Kingdom 01635 523545

For further support information, see the *Technical Support Resources* appendix. To comment on the documentation, send e-mail to `techpubs@ni.com`

© Copyright 1998, 2000 National Instruments Corporation. All rights reserved.

# Important Information

## Warranty

The media on which you receive National Instruments software are warranted not to fail to execute programming instructions, due to defects in materials and workmanship, for a period of 90 days from date of shipment, as evidenced by receipts or other documentation. National Instruments will, at its option, repair or replace software media that do not execute programming instructions if National Instruments receives notice of such defects during the warranty period. National Instruments does not warrant that the operation of the software shall be uninterrupted or error free.

A Return Material Authorization (RMA) number must be obtained from the factory and clearly marked on the outside of the package before any equipment will be accepted for warranty work. National Instruments will pay the shipping costs of returning to the owner parts which are covered by warranty.

National Instruments believes that the information in this document is accurate. The document has been carefully reviewed for technical accuracy. In the event that technical or typographical errors exist, National Instruments reserves the right to make changes to subsequent editions of this document without prior notice to holders of this edition. The reader should consult National Instruments if errors are suspected. In no event shall National Instruments be liable for any damages arising out of or related to this document or the information contained in it.

EXCEPT AS SPECIFIED HEREIN, NATIONAL INSTRUMENTS MAKES NO WARRANTIES, EXPRESS OR IMPLIED, AND SPECIFICALLY DISCLAIMS ANY WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. CUSTOMER'S RIGHT TO RECOVER DAMAGES CAUSED BY FAULT OR NEGLIGENCE ON THE PART OF NATIONAL INSTRUMENTS SHALL BE LIMITED TO THE AMOUNT THERETOFORE PAID BY THE CUSTOMER. NATIONAL INSTRUMENTS WILL NOT BE LIABLE FOR DAMAGES RESULTING FROM LOSS OF DATA, PROFITS, USE OF PRODUCTS, OR INCIDENTAL OR CONSEQUENTIAL DAMAGES, EVEN IF ADVISED OF THE POSSIBILITY THEREOF. This limitation of the liability of National Instruments will apply regardless of the form of action, whether in contract or tort, including negligence. Any action against National Instruments must be brought within one year after the cause of action accrues. National Instruments shall not be liable for any delay in performance due to causes beyond its reasonable control. The warranty provided herein does not cover damages, defects, malfunctions, or service failures caused by owner's failure to follow the National Instruments installation, operation, or maintenance instructions; owner's modification of the product; owner's abuse, misuse, or negligent acts; and power failure or surges, fire, flood, accident, actions of third parties, or other events outside reasonable control.

## Copyright

Under the copyright laws, this publication may not be reproduced or transmitted in any form, electronic or mechanical, including photocopying, recording, storing in an information retrieval system, or translating, in whole or in part, without the prior written consent of National Instruments Corporation.

## Trademarks

ActiveMath™, ComponentWorks™, CVI™, DataSocket™, HiQ™, HiQ-Script™, LabVIEW™, Measurement Studio™, National Instruments™, and ni.com™ are trademarks of National Instruments Corporation.

Product and company names mentioned herein are trademarks or trade names of their respective companies.

## WARNING REGARDING USE OF NATIONAL INSTRUMENTS PRODUCTS

(1) NATIONAL INSTRUMENTS PRODUCTS ARE NOT DESIGNED WITH COMPONENTS AND TESTING FOR A LEVEL OF RELIABILITY SUITABLE FOR USE IN OR IN CONNECTION WITH SURGICAL IMPLANTS OR AS CRITICAL COMPONENTS IN ANY LIFE SUPPORT SYSTEMS WHOSE FAILURE TO PERFORM CAN REASONABLY BE EXPECTED TO CAUSE SIGNIFICANT INJURY TO A HUMAN.

(2) IN ANY APPLICATION, INCLUDING THE ABOVE, RELIABILITY OF OPERATION OF THE SOFTWARE PRODUCTS CAN BE IMPAIRED BY ADVERSE FACTORS, INCLUDING BUT NOT LIMITED TO FLUCTUATIONS IN ELECTRICAL POWER SUPPLY, COMPUTER HARDWARE MALFUNCTIONS, COMPUTER OPERATING SYSTEM SOFTWARE FITNESS, FITNESS OF COMPILERS AND DEVELOPMENT SOFTWARE USED TO DEVELOP AN APPLICATION, INSTALLATION ERRORS, SOFTWARE AND HARDWARE COMPATIBILITY PROBLEMS, MALFUNCTIONS OR FAILURES OF ELECTRONIC MONITORING OR CONTROL DEVICES, TRANSIENT FAILURES OF ELECTRONIC SYSTEMS (HARDWARE AND/OR SOFTWARE), UNANTICIPATED USES OR MISUSES, OR ERRORS ON THE PART OF THE USER OR APPLICATIONS DESIGNER (ADVERSE FACTORS SUCH AS THESE ARE HEREAFTER COLLECTIVELY TERMED "SYSTEM FAILURES"). ANY APPLICATION WHERE A SYSTEM FAILURE WOULD CREATE A RISK OF HARM TO PROPERTY OR PERSONS (INCLUDING THE RISK OF BODILY INJURY AND DEATH) SHOULD NOT BE RELIANT SOLELY UPON ONE FORM OF ELECTRONIC SYSTEM DUE TO THE RISK OF SYSTEM FAILURE. TO AVOID DAMAGE, INJURY, OR DEATH, THE USER OR APPLICATION DESIGNER MUST TAKE REASONABLY PRUDENT STEPS TO PROTECT AGAINST SYSTEM FAILURES, INCLUDING BUT NOT LIMITED TO BACK-UP OR SHUT DOWN MECHANISMS. BECAUSE EACH END-USER SYSTEM IS CUSTOMIZED AND DIFFERS FROM NATIONAL INSTRUMENTS' TESTING PLATFORMS AND BECAUSE A USER OR APPLICATION DESIGNER MAY USE NATIONAL INSTRUMENTS PRODUCTS IN COMBINATION WITH OTHER PRODUCTS IN A MANNER NOT EVALUATED OR CONTEMPLATED BY NATIONAL INSTRUMENTS, THE USER OR APPLICATION DESIGNER IS ULTIMATELY RESPONSIBLE FOR VERIFYING AND VALIDATING THE SUITABILITY OF NATIONAL INSTRUMENTS PRODUCTS WHENEVER NATIONAL INSTRUMENTS PRODUCTS ARE INCORPORATED IN A SYSTEM OR APPLICATION, INCLUDING, WITHOUT LIMITATION, THE APPROPRIATE DESIGN, PROCESS AND SAFETY LEVEL OF SUCH SYSTEM OR APPLICATION.

# Conventions

The following conventions appear in this manual:

| | |
|---|---|
| » | The » symbol leads you through nested menu items and dialog box options to a final action. The sequence **Help»HiQ Help Topics** directs you to pull down the **Help** menu and select the **HiQ Help Topics** item. This symbol also represents the MATLAB prompt. |
| | This icon denotes a tip, which alerts you to advisory information. |
| | This icon denotes a note, which alerts you to important information. |
| **bold** | Bold text denotes items that you must select or click on in the software, such as menu items and dialog box options. |
| *italic* | Italic text denotes emphasis, a cross reference, or an introduction to a key concept. |
| `monospace` | Text in this font denotes text or characters that you should literally enter from the keyboard, sections of code, programming examples, and syntax examples. This font is also used for the proper names of disk drives, paths, directories, parameters, functions, variables, filenames and extensions, and for statements and comments taken from programs. |
| *`monospace italic`* | Italic text in this font denotes text that is a placeholder for a word or value that you must supply. |

# Contents

## Chapter 1
## Introduction to HiQ


System Requirements ..... 1-1
Installing HiQ ..... 1-1
Launching HiQ ..... 1-2
Getting Started ..... 1-2
The HiQ Notebook ..... 1-3
Understanding Objects ..... 1-5
    Deleting Objects and Object Views ..... 1-6
        Deleting the View of an Object from a Page ..... 1-6
        Deleting an Object from a Notebook ..... 1-6
Experimenting with Example Notebooks ..... 1-6
    Working with Multiple Windows ..... 1-6
    Interacting with a Notebook and Running Scripts ..... 1-7
Accessing Online Help ..... 1-7


## Chapter 2
## Getting Results with a HiQ Notebook


Notebook Examples ..... 2-1
    Data Fitting Notebook ..... 2-1
        Opening a Notebook ..... 2-2
        Running a Script ..... 2-3
        Viewing an Object List ..... 2-4
    Expression Evaluator Problem Solver Notebook ..... 2-4
        Entering Data and Running Scripts ..... 2-6


## Chapter 3
## Getting Results with the HiQ Command Window


Navigating the Command Window ..... 3-1
Getting Immediate Results with the Command Window ..... 3-4
    Solving a Linear System of Equations ..... 3-4
    Solving a System of Ordinary Differential Equations ..... 3-5
Getting Help from the Command Window ..... 3-9
Getting Notebook Information from the Command Window ..... 3-10
Modifying a Notebook from the Command Window ..... 3-12
Command Window Commands ..... 3-14
MATLAB Mode Commands ..... 3-15

## Chapter 4
## Getting Your Data into HiQ

Climate Notebook: Importing Data ..... 4-1
Seismic Notebook: Importing Data ..... 4-4

## Chapter 5
## Publishing and Subscribing to Live Data

Climate Notebook: Subscribing to Live Temperature Measurements over the Internet ..... 5-2
Seismic Notebook: Publishing Data from HiQ ..... 5-4

## Chapter 6
## Visualizing Data with 2D and 3D Graphs

Climate Notebook: Visualizing Rainfall and Temperature Data in a 2D Graph ..... 6-1
Creating a Graph ..... 6-1
Plotting a HiQ Data Object ..... 6-2
Modifying the Graph and Plot ..... 6-4
Working with Multiple Plots in a Graph ..... 6-7
Seismic Notebook: Visualizing Seismic Data in a 3D Graph ..... 6-11
Visualizing Live Data in 3D ..... 6-11
Modifying a 3D Graph and Plot ..... 6-13
Rotating, Zooming, and Panning a 3D Graph ..... 6-16

## Chapter 7
## Analyzing Data with HiQ-Script

Climate Notebook: Analyzing Rainfall Data ..... 7-1
Setting Default View Properties ..... 7-3
Looking for Trends ..... 7-4
Maintaining the Notebook ..... 7-5
Seismic Notebook: Computing the Energy in the Signal ..... 7-6
Automating the Analysis ..... 7-9

## Chapter 8
## Taking Advantage of ActiveX

Climate Notebook: Adding ActiveX Objects ..... 8-2
Adding a ComponentWorks ActiveX Control ..... 8-2
Programmatically Accessing the ActiveX Control ..... 8-4
Adding a Microsoft Word Document ..... 8-6

Seismic Notebook: Adding a ComponentWorks Control....8-8
Adding an ActiveX Control....8-9
Programmatically Accessing the ActiveX Control....8-10

## Chapter 9
## Converting a Notebook to a Problem Solver

Climate Notebook: Converting to a Problem Solver....9-1
Seismic Notebook: Converting to a Problem Solver....9-5

## Chapter 10
## Sharing Notebooks with Others

About the HiQ Reader....10-1
Climate Notebook: Previewing the Notebook in the HiQ Reader....10-1
Distributing ActiveX Controls in Your Notebooks....10-2
Automatically Executing Functions....10-3
Distributing Help for a Notebook....10-3
Climate Notebook: Maintaining the HTML Help File....10-4
Seismic Notebook: Viewing Compiled HTML Help....10-5

## Chapter 11
## Getting Results as a MATLAB User

About HiQ....11-1
HiQ Command Window....11-2
Accessing the HiQ Command Window....11-2
Accessing and Using MATLAB 5 Compatibility Mode....11-3
Sharing Data and Functionality....11-5
Accessing and Using HiQ Mode....11-7
Frequently Asked Questions....11-8
Comparing HiQ and MATLAB Commands....11-10
Automated M-file to HiQ-Script Translator....11-13
Using the Translator....11-13
Working with the Results....11-15
HiQ-Script for MATLAB Users....11-16
HiQ Online Help for MATLAB Users....11-21

## Appendix A
## Technical Support Resources

**Glossary**

**Index**

# 1

# Introduction to HiQ

This chapter lists system requirements and installation instructions, introduces the HiQ environment, and describes ways in which you can apply HiQ analysis and visualization to your projects.

## System Requirements

HiQ requires the following minimum system configuration:

- Windows 2000/NT/9*x* (NT 4.0 or later)
- 486 CPU with coprocessor
- 8 MB RAM with Windows 95, 16 MB RAM with Windows 2000/NT/98
- 256-color, 800 by 600 VGA display
- 20–60 MB free disk space (You need 20 MB for HiQ Reader, 40 MB for HiQ, and 60 MB for HiQ Professional.)

The following specifications are the recommended system configuration for HiQ:

- Windows NT 4.0
- Pentium 90 or higher
- 16 MB RAM with Windows 95, 32 MB RAM with Windows 2000/NT/98
- Accelerated True Color, 1024 by 768 display
- 20–60 MB free disk space (You need 20 MB for HiQ Reader, 40 MB for HiQ, and 60 MB for HiQ Professional.)

## Installing HiQ

1. Insert the HiQ CD in the CD-ROM drive of your computer. If the startup screen does not appear, use Windows Explorer to run the `SETUP.EXE` program from your CD.
2. Follow the *Setup* instructions you see on your screen.

By default, the HiQ installation program creates a new folder, `C:\Program Files\National Instruments\HiQ`, that contains the following items:

- `Program` folder—`HiQ.exe`, HiQ help files, and related files.
- `Examples` folder—Example Notebooks demonstrating many of the analysis and visualization capabilities of HiQ, organized by category.
- `Readme.txt` file—Late-breaking information about HiQ, known limitations, and corrections.
- `Manuals` folder—PDF versions of the HiQ manuals.

# Launching HiQ

Use one of the following methods to launch HiQ:

- Use the **Start** Button.
  1. Select **Programs»National Instruments»HiQ**.
  2. Select the HiQ icon.
- Use Windows Explorer.
  1. Locate the HiQ executable, `HiQ.exe`, in Windows Explorer.
  2. Double click `HiQ.exe`.

# Getting Started

This manual contains basic information you need to plan projects, understand the HiQ environment, navigate efficiently in the HiQ environment, and implement basic analysis and visualization features.

This manual was designed to teach you the fundamental features of HiQ through interactive discussions and examples. If you are new to HiQ, try reading this manual at your computer so you can test the discussion concepts and examples. If you are already a HiQ user and need information about completing a specific task, turn directly to the section discussing that task and complete the examples. Use the following questions and answers to assess where you should begin in this manual.

**Are you new to HiQ and need to learn the interface elements and how to get results quickly?**

Chapters 1–3 introduce HiQ, the primary interface element (the HiQ Notebook), and the Command Window. Try reading Chapters 2 and 3 at your computer so you can explore and experiment with the Notebooks and Command Window as you learn about their features and capabilities.

**Do you understand the HiQ interface and Notebook metaphor but need information about constructing Notebooks and enhancing them?**

In Chapters 4–9, you create Notebooks from scratch and expand those Notebooks to include data visualization and analysis, ActiveX capabilities, and interactive components. Each chapter contains two examples—a Climate Notebook and a Seismic Notebook—that build on the results of the previous chapter. At the end of Chapter 9, you will have created a complete Problem Solver Notebook for both examples.

**Do you want to share your Notebook with others?**

Chapter 10, *Sharing Notebooks with Others*, discusses important issues about distributing your Notebooks with the HiQ Reader. Only the HiQ Professional version can create Notebooks that others can view in the HiQ Reader.

**Do you need information to complete a specific task?**

Use the Contents to locate the task, and complete the examples in that section to learn about it. If you are working in HiQ, select **Help»HiQ Help Topics** and search for the task about which you want more information.

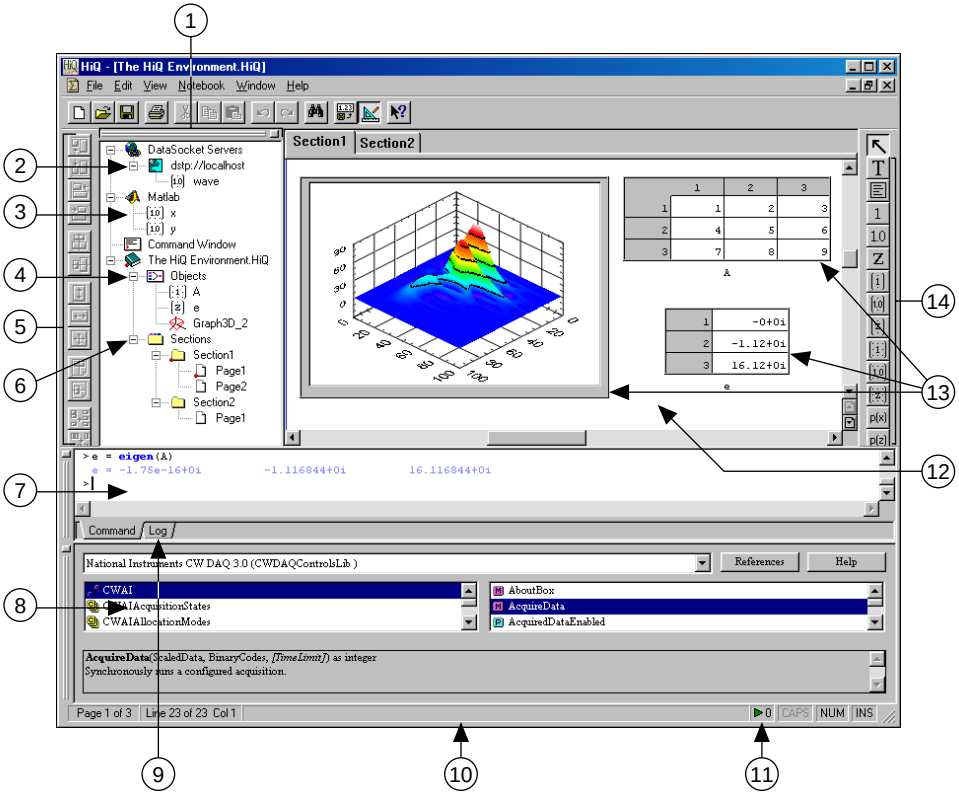
**Are you a MATLAB user looking for a way to leverage your work?**

Chapter 11, *Getting Results as a MATLAB User*, offers suggestions that can help you leverage the work you have already done in MATLAB. Try reading this chapter at your computer so you can test the suggestions and examples.

# The HiQ Notebook

Imagine preparing a management study for a water reservoir. To report the results of your study you might use text to describe the problem, tables of data to support your claim, graphs to analyze the data, analysis to formulate projections or solutions, and conclusions to suggest ways to better manage the reservoir in the future. HiQ handles all these reporting, analysis, and visualization tasks. Furthermore, you can include an interactive script and graph so your audience can observe the effect of changing variables on inflow and outflow water levels. With HiQ, you can create interactive tools for managing the reservoir in the future.

HiQ uses a notebook as the basic interface metaphor. The HiQ *Notebook* has features that you normally associate with a real engineering or scientific Notebook, such as pages, sections, and tabs. Figure 1-1 labels all of the interface elements in the HiQ environment.




| | | | |
|---|---|---|---|
| 1 HiQ Explorer | 5 Alignment Tools | 9 Log Window | 12 Notebook Page |
| 2 DataSocket Servers Browser | 6 Notebook Organization | 10 Status Bar | 13 Objects |
| 3 MATLAB Browser | 7 Command Window | 11 Script Status | 14 Object Tools |
| 4 Object List | 8 ActiveX Object Browser | | |

**Figure 1-1.** HiQ Environment

On a single page you can arrange information in a free-form manner much like in a typical drawing program. Each piece of information that you place on a Notebook page is known as an *object*. As with a real notebook, you can put many types of objects on a page: text, numerics, graphs, and more. HiQ offers these objects plus some that you do not find in a regular

notebook. You also can place any ActiveX control or ActiveX object on a Notebook page, significantly increasing your flexibility within the HiQ environment.

As your Notebook grows, you can add more pages. You also can group your work in sections to better organize a project.

# Understanding Objects

All the items on a Notebook page are called objects. Notice that any object you click on becomes active and displays a border, as shown in Figure 1-2. You can interact with any activated object directly on the Notebook page.

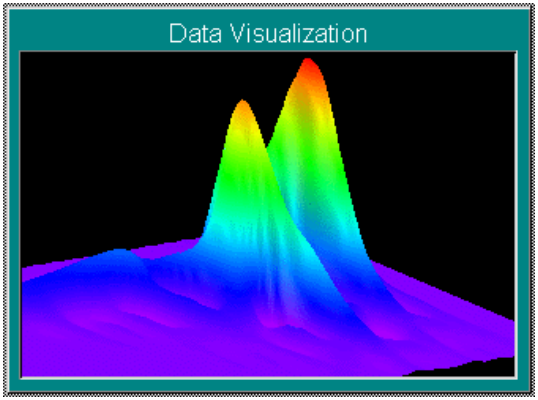



**Figure 1-2.** Object in Active Mode

Objects have the following characteristics:

- Contained within a Notebook.
- Native (if found in the Objects toolbar) or ActiveX.
- Associated with a Notebook, but not necessarily visible on the Notebook page until you place a view there.
- Typed numeric, graphic, HiQ-Script, text, or ActiveX. Each of these object type categories has a distinct purpose in HiQ.

## Deleting Objects and Object Views

Objects in your HiQ Notebooks can exist without being visible on the Notebook page. For example, you might want to display the plot of a data set without displaying the numeric object that is the source of the data.

**Tip** Remember the distinction between objects and object views when deleting. You can delete an object entirely from the Notebook or delete only a view of that object from the Notebook page.

### Deleting the View of an Object from a Page

Deleting the view of an object from the Notebook page is not the same as deleting the object. When you delete a view, the object remains in the Notebook Object List. You can still access the object through a script or in the Object List.

To delete a view, right click the object on the Notebook page and use the **View»Delete** command. Alternatively, you can select the object view and press the <Delete> key. The selected view is removed from the Notebook page, but the object remains in the Notebook.

### Deleting an Object from a Notebook

To entirely delete an object from a Notebook, right click the object and use the **Object»Delete** command. Alternatively, you can select the object view and press <Shift-Delete>. The object and all of its views are removed completely from the Notebook.

# Experimenting with Example Notebooks

Looking at and experimenting with some example Notebooks gives you an idea of what you can accomplish with HiQ. In this manual, you learn how to use and create Notebooks to visualize and analyze data. You can browse the `Examples` folder, which contains a variety of Notebooks illustrating the analysis and visualization capabilities of HiQ.

## Working with Multiple Windows

A HiQ Notebook uses the standard Multiple Document Interface (MDI): a main window exists within which multiple HiQ Notebooks may reside. From the **Windows** menu, you can maximize, overlay, or tile the Notebook windows you open.

## Interacting with a Notebook and Running Scripts

A well-designed Notebook should be self-documenting, describing how to interact with the Notebook. Interaction might be as simple as entering a start time and a stop time for simulation of a system of differential equations. Or a Notebook might instruct you to select a data file to import in order to calculate and graph the amount of rainfall in a particular area.

Most Notebooks include scripts that you can run to solve the type of mathematical or visualization problem the Notebook was designed to handle.

**Note** To execute a script, right click on the script object and select **Run**.

# Accessing Online Help

HiQ offers tooltips, context-sensitive help, and online help that you can access in the following ways.

- Access the HiQ online help by selecting **Help»HiQ Help Topics**. There you find information not available in this manual, such as complete function reference and answers to *How Do I...?* questions.
- Move the mouse cursor over any button in HiQ to see a tooltip describing the function of the button.
- Watch the status bar at the bottom of a HiQ Notebook as you move the mouse pointer over the menu items to learn more about each item.
- Click the **Help** button (**?**) in other areas of the HiQ user interface, such as dialog boxes, to access context-sensitive help for the next item on which you click. Alternatively, use the <F1> key to access context-sensitive help for a selected item or for an area where the cursor is located.

# 2

# Getting Results with a HiQ Notebook

This chapter introduces the fundamental features of the HiQ Notebook interface and shows you how to interact with existing Notebooks.

## Notebook Examples

This section presents two different HiQ Notebooks—a data fitting example and an expression evaluator Problem Solver—to demonstrate the use of the HiQ Notebook. The data fitting example, Population Fit, illustrates how a Notebook performs a single data analysis, and the Expression Evaluator Problem Solver demonstrates how a Notebook can solve a class of problems. A Problem Solver Notebook accepts varied input from users without requiring them to alter the contents of the script.

### Data Fitting Notebook

In this example, you work with a Notebook that performs data fitting analysis. The purpose of this example is to present the fundamental interface elements in HiQ, general parts of a Notebook, and common objects.

This Notebook demonstrates how HiQ analyzes a set of data, in this case population data, and graphically displays it. More specifically, the Population Fit Notebook graphically presents the change in U.S. population over time.

This Notebook contains a script and a graph. The script object contains HiQ-Script code that performs the data fit. The graph object displays the data fit after the script executes.

## Opening a Notebook

1. To open a Notebook, use the **File»Open** command.
2. Open the `Examples` folder within the `HiQ` folder.

   The `Examples` folder contains Notebooks grouped by category. Later you can explore the different examples to get a feel for HiQ capabilities.
3. From the `Examples` folder, open the `Data Fitting` folder and select `Population Fit`.
4. Click **Open**. The Population Fit Notebook appears, as shown in Figure 2-1.

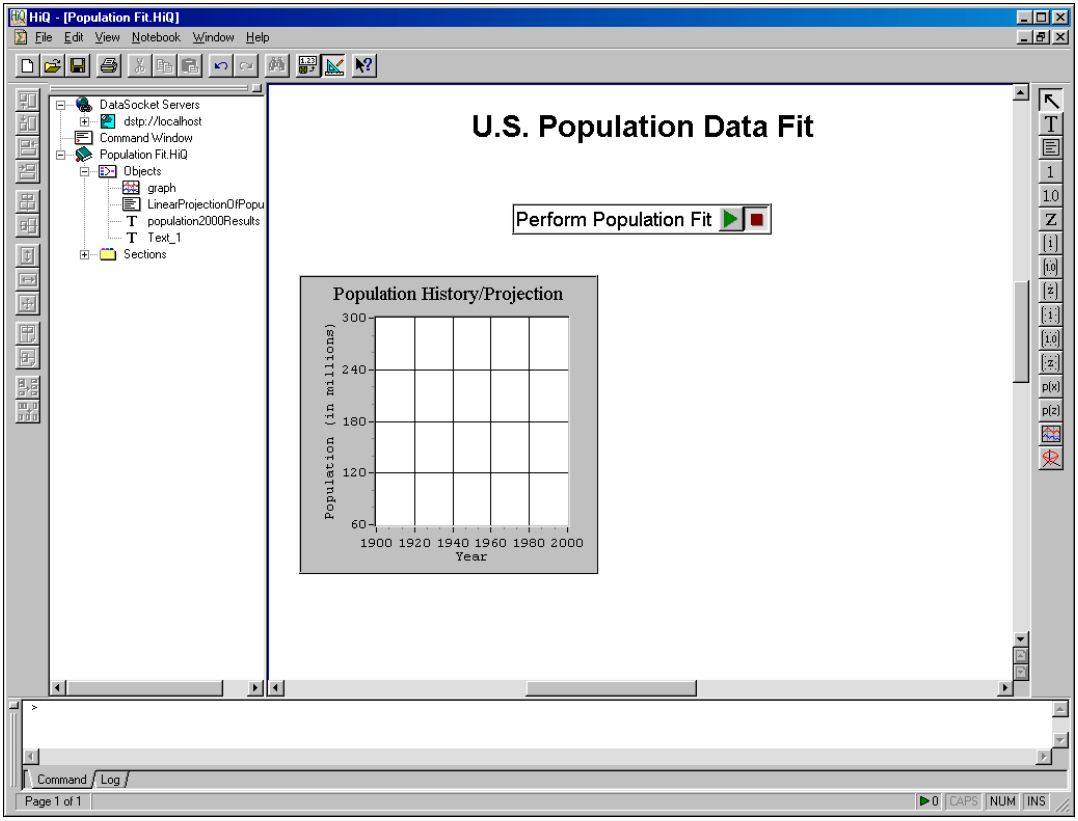



**Figure 2-1.** Population Fit Notebook

## Running a Script

To run the script, press the **Perform Population Fit** run button. The output of the script appears as a plot within the graph and represents the increase of U.S. population over time, as shown in Figure 2-2.

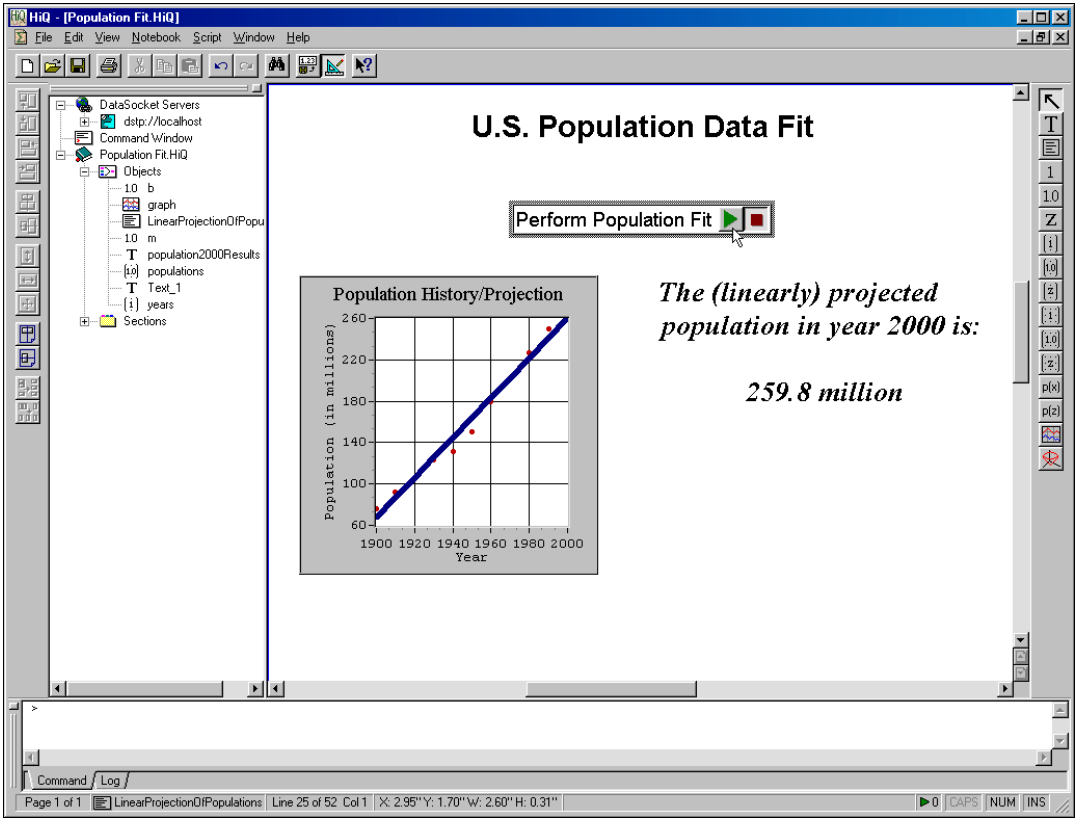



**Figure 2-2.** Running a Script in the Population Fit Notebook

Four objects are visible on the Notebook page: the graph, two text boxes, and the script. However, these are not the only objects in this Notebook, as you will learn in the next section.

Right click on the run button, select **View Source**, and scroll through the script to get a feel for how HiQ performs analysis. In this example, a single function `fit` performs the data fitting analysis. You can access most analysis capability in HiQ through calls to single functions like `fit`.

## Viewing an Object List

Many, but not all, objects in a Notebook appear on the Notebook page. To view all objects in the Notebook, use the Object list in the HiQ Explorer, as shown in Figure 2-3.

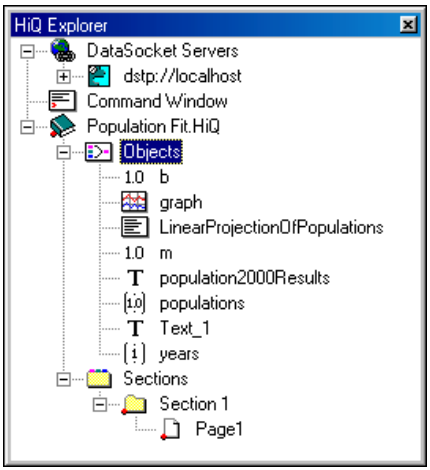



**Figure 2-3.** Expanding the Object List

The *Object List* contains all objects in the active Notebook. With the exception of the graph and HiQ-Script objects, all the objects in the list resulted from the execution of the script. As you can see, objects do not have to be visible to exist within a Notebook.

**Note** Any object created by a script exists in the Notebook and appears in the Object List after that script runs. The only exception is objects with local scope.

When HiQ-Script objects are generated, they appear only in the Object List. If you want to see the object on the Notebook page, select it in the Object List and drag-and-drop it on the page.

# Expression Evaluator Problem Solver Notebook

Problem solvers are HiQ Notebooks designed to interact with users to analyze or solve a wide range of problems belonging to a certain class. The previous example Notebook, Population Fit, does not qualify as a problem solver because it solves a single problem. Population Fit does not give you the option to fit populations from different countries or different time ranges. To analyze a different set of data, you would need to modify the script. A problem solver is a HiQ Notebook that allows the user to enter a

different set of data or analysis problem without having to modify script. The Expression Evaluator Notebook illustrates the problem solver features you can use in HiQ. Perform the following steps to learn more about the Expression Evaluator Notebook.

1. To open a Notebook, use the **File»Open** command.
2. Open the `Examples\Problem Solvers` folder within the `HiQ` folder.
3. Select `Expression Evaluator.HiQ` and click **Open**. The Expression Evaluator Notebook appears, as shown in Figure 2-4.

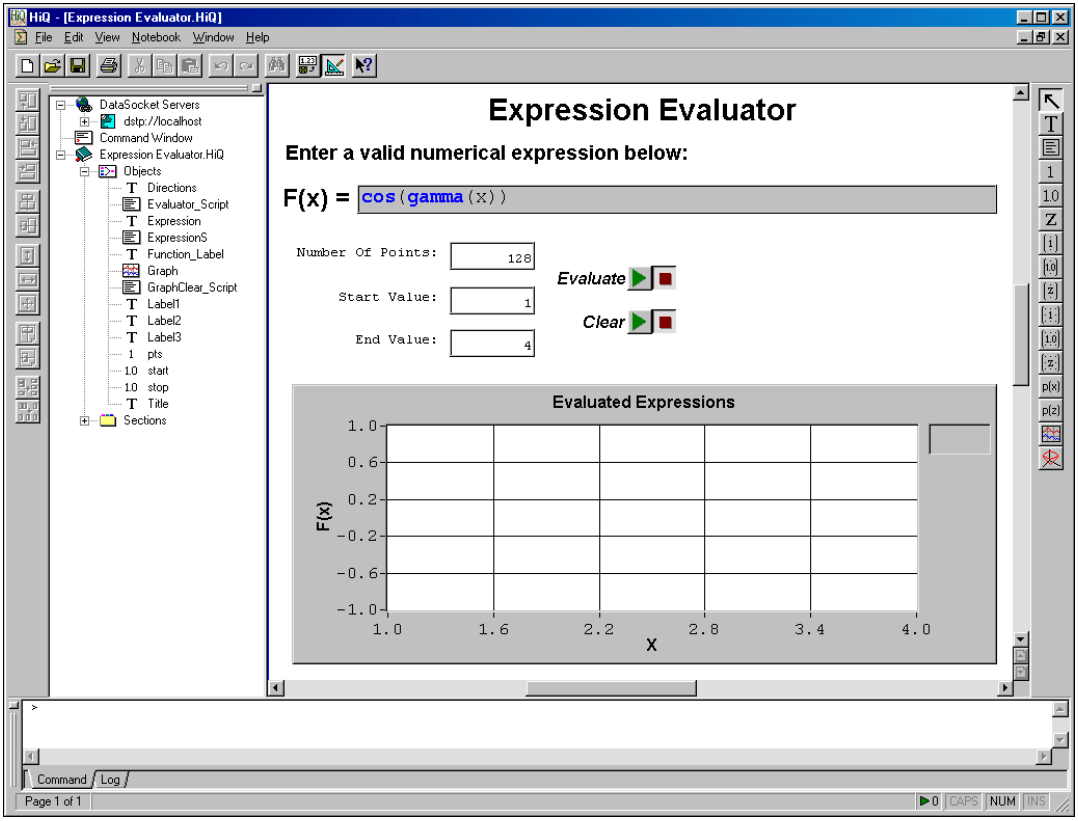



**Figure 2-4.** Expression Evaluator Notebook

## Entering Data and Running Scripts

This Notebook can evaluate any mathematical expression and display its behavior graphically. Unlike the Population Fit Notebook, objects in the Expression Evaluator Notebook expect user input: a mathematical expression, the number of points to plot, and the interval of the domain. In this example, all objects have default values so you can just run the Notebook.

1. Press the **Evaluate** button to evaluate the expression. The Notebook evaluates the expression over 128 points, starting at 1 and ending at 4. The graph displays the result.
2. Enter the mathematical expression `sin(x)*cos(x)` into the text object labeled **F(x) =**.
3. Press the **Evaluate** button to evaluate the expression.

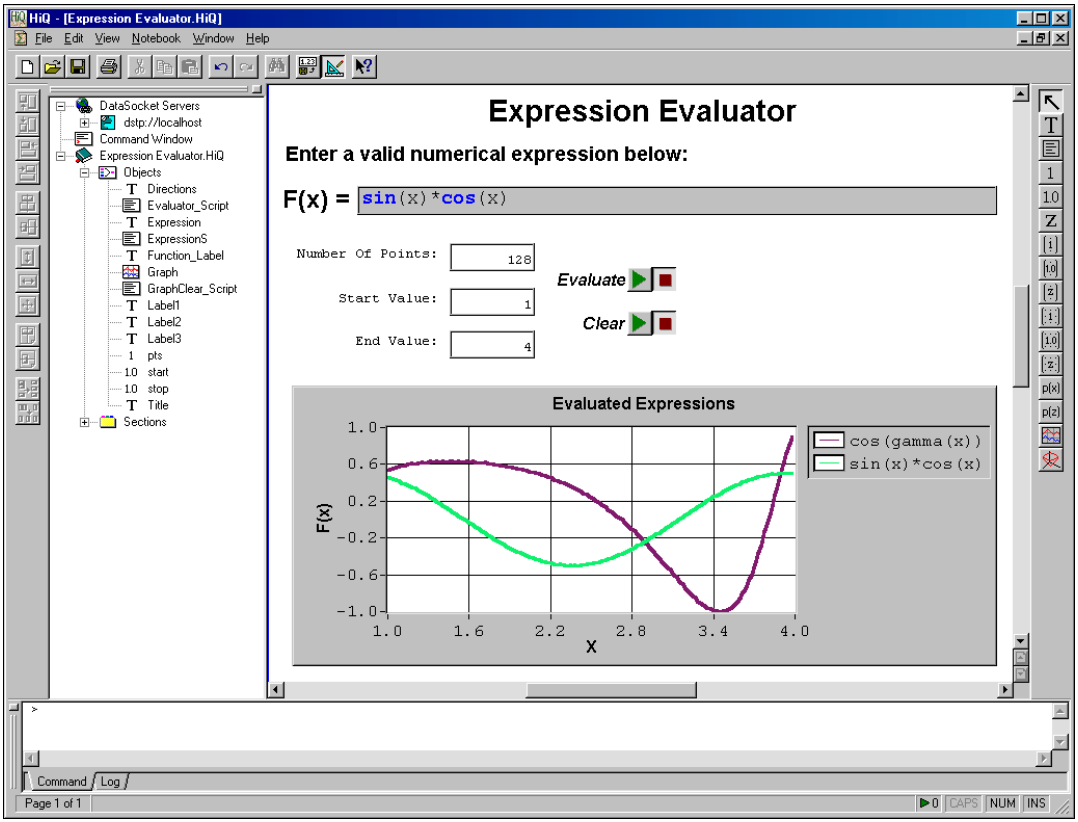



**Figure 2-5.** Evaluating Expressions with the Expression Evaluator Notebook

The expression you entered is fed into the numerical algorithm in the script. The plot that appears when you run the script is a graphical representation of the mathematical expression `sin(x)*cos(x)` that you entered. To observe the role of the numeric input of this Notebook, enter various values for the domain interval and number of plotted points and rerun the script.

Enter other mathematical expressions and run this Notebook to see how they display. Keep in mind that you must use valid operators and valid HiQ built-in functions. In this example, your expression must be in terms of *x*. Refer to the HiQ online help for complete descriptions of operators and built-in functions.

# 3

# Getting Results with the HiQ Command Window

This chapter presents the Command Window. It explains how to navigate the Command Window, get help from the Command Window, create and modify notebooks from the Command Window, and get immediate results using the Command Window.

## Navigating the Command Window

The Command Window provides immediate access to all analysis and visualization functions in HiQ-Script. If you are looking for quick answers or you are performing exploratory analysis, use the Command Window to display your results immediately.

To experiment with the Command Window, open a new Notebook. By default, the Command Window is attached to the bottom of the HiQ window. You can adjust the position of the Command Window by clicking on the double gray border on the left and dragging it. The window detaches from the bottom frame and becomes a floating window, as shown in Figure 3-1.

**Tip** To return the Command Window to its default position, click in the Command Window title bar and drag the window toward the bottom of the HiQ window. As you drag the window, notice the gray frame. When the window is in a position where it can be docked in the HiQ window, the frame becomes thinner. When you see the thin frame, release the mouse button. This procedure works for the toolbars and other detachable windows, such as the ActiveX Object Browser and the Explorer.

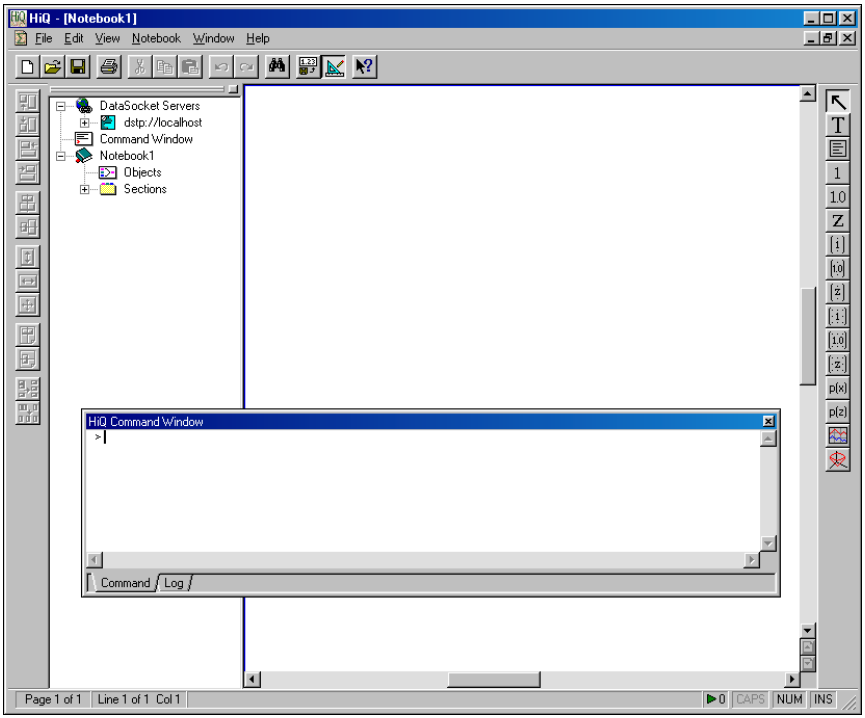



**Figure 3-1.** Floating Command Window

The Command Window prompt, >, indicates that the Command Window is ready to receive a command. You can create different objects at the prompt. For example, you can create a matrix object named A by typing the following command at the prompt:

```
> A = {1,2,3;4,5,6;7,8,9}
```

Notice that the Command Window echoes the result:

```
A=
123
456
789
```

**Tip** If you do not want the Command Window to display the results, include a semicolon after the command.

You can create a polynomial object named p by typing the following command at the prompt:

```
> p = {poly:"x^2+2x+1"}
```

The value for the polynomial object p is echoed by the Command Window:

```
p = x^2 + 2x + 1
```

If an object is too large to view in the Command Window, HiQ displays the object in an individual floating window. For example, create a graph by typing the following command at the prompt:

```
> MyGraph = createGraph(A)
```

**Tip** HiQ highlights all built-in functions, such as `createGraph`, in blue to indicate that you have correctly entered the function name.

HiQ creates a graph named `MyGraph` of the matrix A and displays the graph in its own window.

Notice that each HiQ-Script command assigns a new object name, such as `p`, `A`, or `MyGraph`. All objects in HiQ have a name; however, you do not have to name every object you create in the Command Window. For example, you can quickly determine the norm of a matrix by typing in the following command at the prompt:

```
> norm(A)
```

The Command Window echoes the result:

```
ans = 16.848103
```

Because you did not specify the object name, HiQ chose a default name. In this case, the default object name is `ans`.

**Note** You can change the default name in the Command Window property pages. Right click in the Command Window and edit the **Name** option in the **Default Object Assignment** area.

Because the Command Window allows multiline commands, you can issue commands over several lines to help you visually create a matrix or organize a script, as in Figure 3-2.

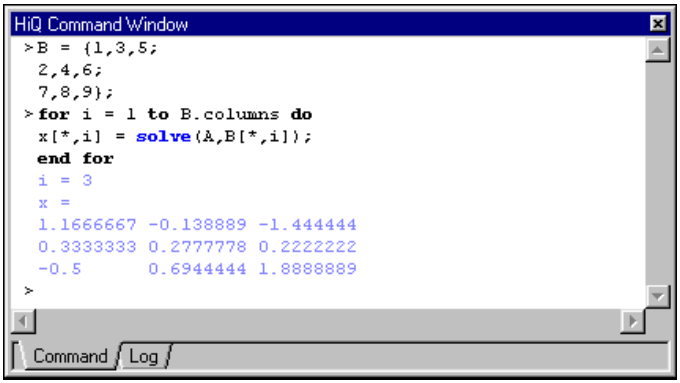



**Figure 3-2.** Multiline Commands in the Command Window

You can recall and edit previous commands by using the arrow keys. The Command Window accepts all special Command Window commands and all HiQ-Script built-in functions.

# Getting Immediate Results with the Command Window

The Command Window offers access to all of the analysis and visualization capabilities of HiQ-Script. The following two examples show you how to use the Command Window to quickly compute answers. The first example computes the solution of a linear system of equations. The second example simulates a system of ordinary differential equations.

## Solving a Linear System of Equations

Linear algebra is a widely used area of analysis for describing linear dynamic systems, computing the best fit to a set of data, optimizing equations, and many other areas. One of the most common operations in linear algebra is solving a system of linear equations.

Consider a set of three linear equations with three unknowns:

$$1\mathbf{x}_1 + 2\mathbf{x}_2 + 3\mathbf{x}_3 = 6$$
$$4\mathbf{x}_1 + 5\mathbf{x}_2 + 6\mathbf{x}_3 = 2$$
$$7\mathbf{x}_1 + 8\mathbf{x}_2 + 9\mathbf{x}_3 = 7$$

These equations can be represented in linear algebra syntax:

$$\mathbf{Ax} = \mathbf{b}$$

where **b** is a vector containing the values on the righthand side of the equations, **A** is a matrix containing the values of the coefficients on the lefthand side of the equations, and **x** is a vector of unknown quantities.

To create the matrix **A** and the vector **b**, type the following at the prompt:

```
> A = {1,2,3;4,5,6;7,8,9};
> b = {v:6,2,7};
```

One obvious way to solve this linear algebra equation for the vector **x** is to multiply the vector **b** by the inverse of the matrix **A**:

```
> x = inv(A) * b
```

This method works only if the matrix **A** has an inverse. In this case, **A** is singular, and, therefore, it is not invertible. HiQ displays the following message to convey this information: `The matrix is not invertible because it is numerically singular`.

Multiplying the vector **b** on the left by the inverse of **A**, as above, is semantically equivalent to dividing the vector **b** on the left by the matrix **A**. HiQ uses the backslash operator (\) to indicate left division and chooses an appropriate method for solving the equation:

```
> x = A\b
```

In this case, the left divide operator is equivalent to calling the built-in function `solve`:

```
> x = solve(A, b)
```

Both of the solutions share the same result:

```
x=-2.0277780.05555562.1388889
```

## Solving a System of Ordinary Differential Equations

Scientists and engineers use differential equations in many areas to describe the dynamics of a physical, chemical, or electrical system. HiQ contains a set of methods for numerically solving first-order ordinary differential equations for both initial values and boundary values.

For example, consider a second-order, initial-value system described by the following equations:

$$\frac{dx_1}{dt} = -5x_1 + 5$$

$$\frac{dx_2}{dt} = -10x_2 + 10$$

These equations describe how two states, $x_1$ and $x_2$, change in time. This example solves for the states $x_1$ and $x_2$ as time $t$ moves from 0 to 2. The initial value for the states $x_1$ and $x_2$ at time = 0 are 4 and 9, respectively.

`ODEIVP` is the HiQ built-in function for solving ordinary differential equations (initial value problems). To get started, type `ODEIVP` at the Command Window prompt and press <F1> to display the online help for this function. Study the first function listed under the *Usage* heading:

```
[tOut, X] = ODEIVP(fct,x0,start,stop,stepSize,
IVPAlg,absTolr,relTolr,outStep,callback)
```

**Note** A HiQ function call consists of a list of input parameters enclosed by parentheses to the right of the function name and a list of output objects enclosed by square brackets to the left of the equal sign.

The first five parameters for the `ODEIVP` function are required and consist of a user function that describes the differential equations, a vector representing the initial values of the states, a starting value, a stopping value, and a step size for the solution. The remaining parameters are optional.

**Tip** As a convention, all optional parameters appear in an italic font.

To create the user function for the differential equations, click on the parameter `fct` in the online help parameter table to view a detailed window with a template for the user function `fct`, as shown in Figure 3-3. Copy this function (by selecting the text and **Options»Copy**).

In the Command Window, press the <Esc> key to clear the current command. Paste it at the Command Window prompt (by selecting **Edit»Paste**).

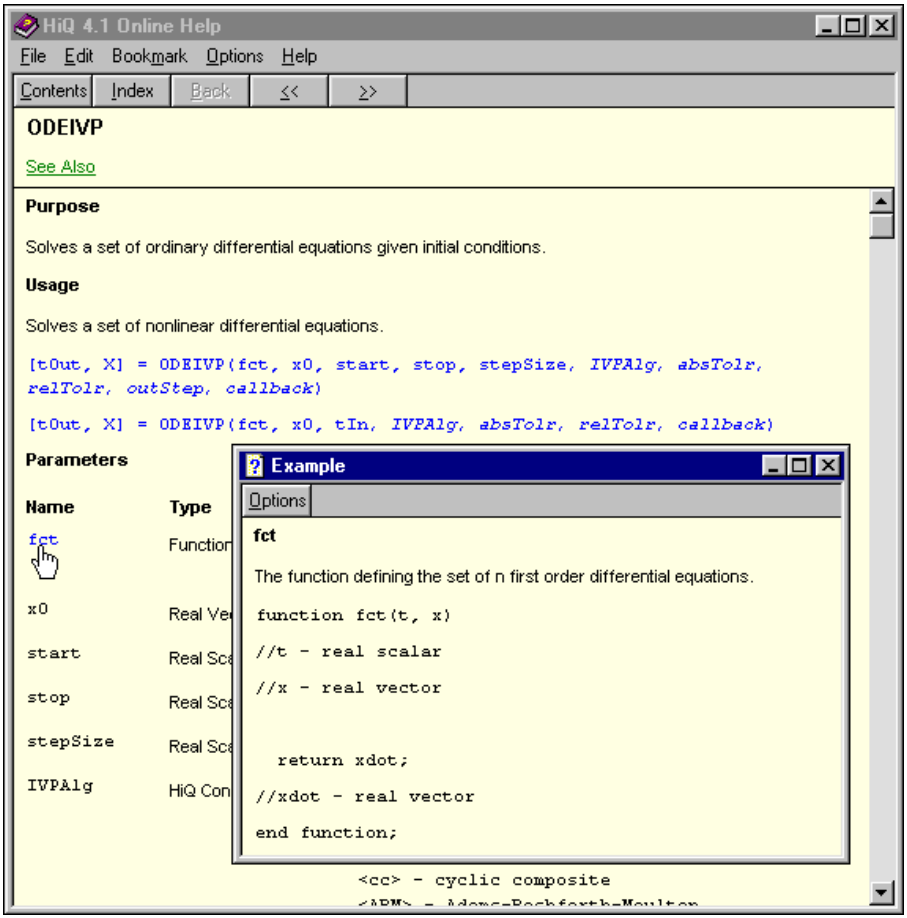



**Figure 3-3.** Accessing the fct Function Template

Complete the function as follows. Insert the cursor in the last line and press <Enter>.

```
> function fct(t, x)
    //t – real scalar
    //x – real vector
    xdot[1] = –5*x[1] + 5;
    xdot[2] = –10*x[2] + 10;
    return xdot;
    //xdot – real vector
end function;
```

If you have already pressed <Enter> and need to edit what you typed, recall the function with the up-arrow key and then edit it. After you make your edits, insert the cursor at the end of the function and press <Enter>. HiQ creates a user function named `fct`.

To create the vector of initial values of the states of the two differential equations, type the following command:

```
> x0 = {v: 4, 9};
```

To solve the differential equations starting at 0, ending at 2, and with an output step size of 0.2, type the following command:

```
> [t, X] = ODEIVP(fct, x0, 0, 2, .2);
```

HiQ solves the differential equation and returns the independent solution values in the vector `t` and the solutions to the two differential equations in the first and second columns of the matrix `X`.

Finally, visualize the results using the `createGraph` and `addPlot` functions:

```
> g = createGraph(t, X[*,1])
> addPlot(g, t, X[*,2])
```

The first line creates a new graph named `g` with a new plot. The second line adds a new plot to the graph `g`. HiQ creates a floating view of the graph, as shown in Figure 3-4.

When you save this Notebook, all of the objects you have created are saved with it. You can load the Notebook later and change the parameters in the differential equations, alter the starting and stopping values of the solution, or place the objects on the Notebook page to create a document. You do not have to recreate results because you save them as you save the Notebook.

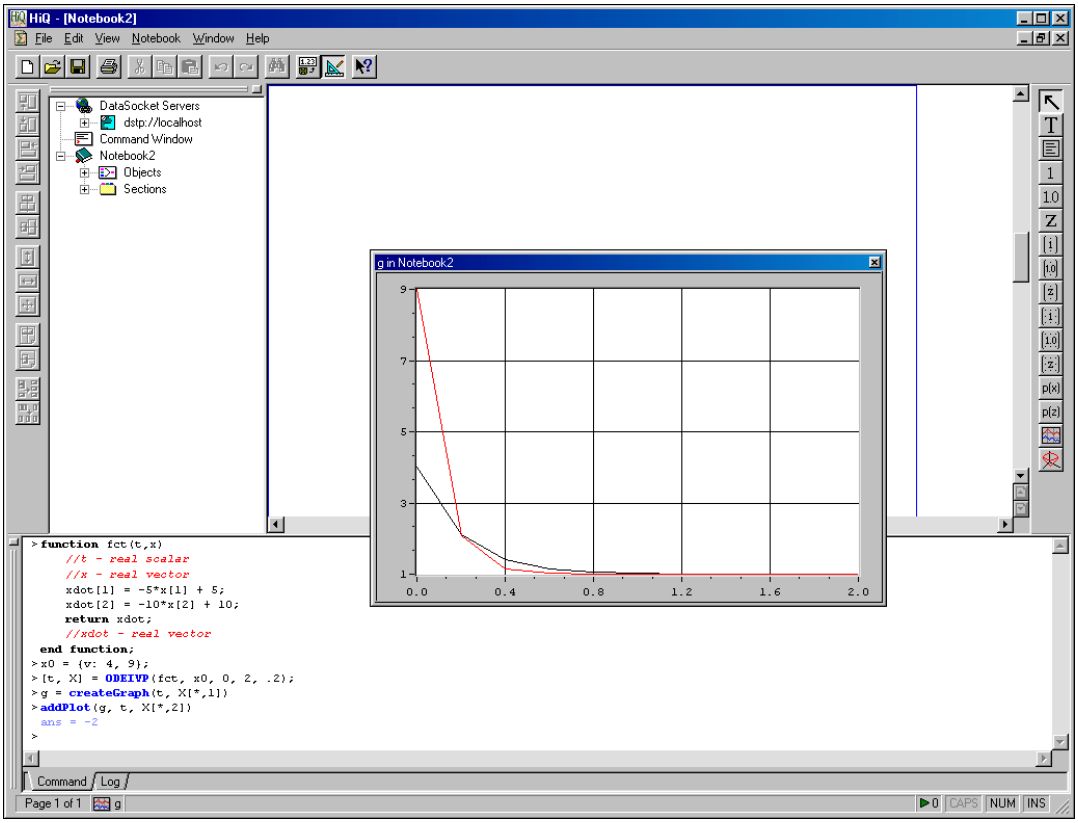



**Figure 3-4.** Creating an Object View

# Getting Help from the Command Window

When working in the Command Window, you can access the online help using any of the following methods.

- Type `help` at the command prompt or press the <F1> key when you are in the Command Window to display help for the Command Window.
- Type a function name at the prompt, place your cursor in the function name, and press <F1>. The online help opens the topic information for that function.
- Type `help` *keyword* at the command prompt, where *keyword* is the function or help file entry for which you are looking. The online help opens the topic information related to the keyword.

# Getting Notebook Information from the Command Window

A Notebook is an organizational tool as well as a documentation tool. HiQ saves all objects created in a Notebook with the Notebook so you can access them in a later session to continue your analysis.

This section describes how you can use the Command Window with an existing Notebook to quickly find answers to questions about Notebook data. For this discussion, open and refer to the `Command Window.HiQ` example in the `Examples\Getting Results` folder. This Notebook contains a matrix of data and a 3D graph representing seismic data collected at a seismographic station, as shown in Figure 3-5.

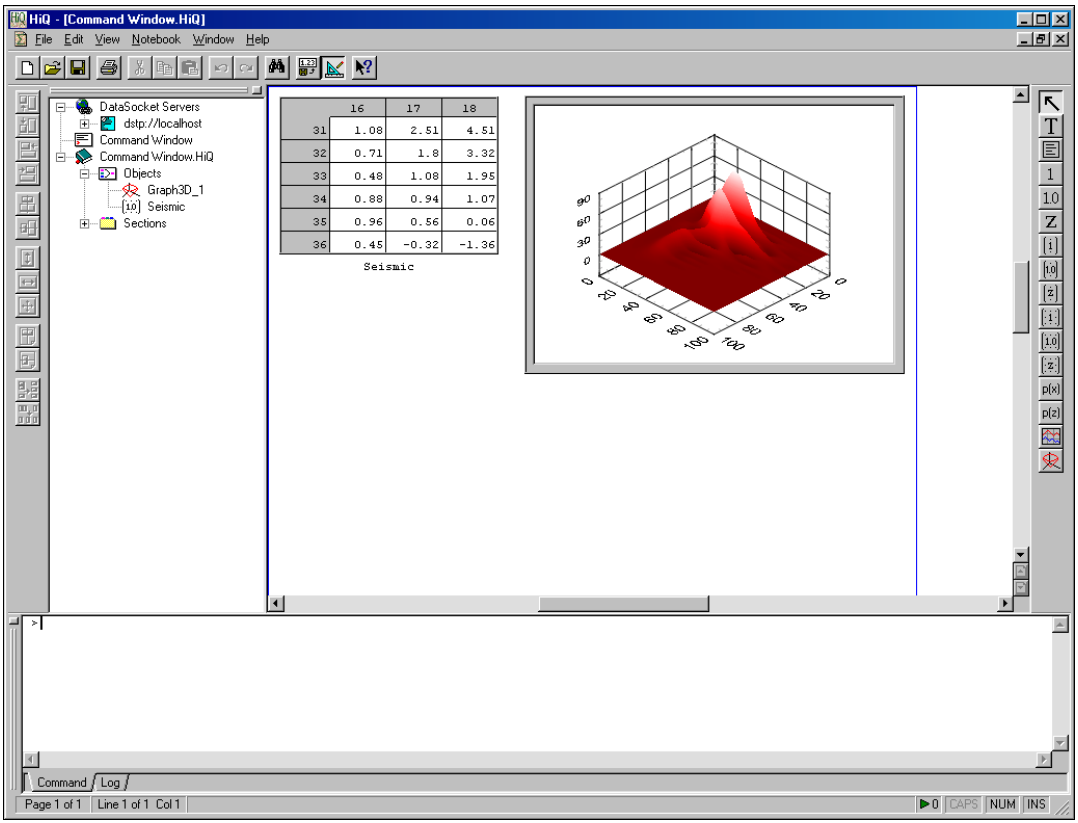



**Figure 3-5.** Command Window Notebook

In this example, you create a 2D graph of a signal and compute its average.

Suppose you want to know what happened thirty seconds into the seismic experiment. Specifically, you want to graph the data and calculate the average. First, you need to extract the desired data from the Seismic matrix. Type the following command at the command prompt to create a new vector of data from row 30 of the Seismic matrix:

```
> slice = Seismic[30,*]
```

This line of HiQ-Script creates a new vector object named `slice` using row 30 and all of the columns of the Seismic matrix. This object is visible in its own window floating above the Notebook. You can scroll down to view all elements of the vector.

Although a tabular view of this data is useful, a graph can convey this information faster and more effectively. Type the following command at the prompt to create a graph of `slice`:

```
> myGraph = createGraph(slice)
```

With this command, you have created a new graph object named `myGraph` that displays a plot of the object `slice`. If you need to make this graph larger, resize the window.

To compute the average, or arithmetic mean, of the elements in `slice`, type the following command at the prompt:

```
> ave = mean(slice)
```

The new scalar object `ave` represents the average of all elements in the vector `slice`. Notice that you can see the `ave` object in the Command Window immediately after you type the command. To view the `ave` object in its own window, type the following command at the prompt:

```
> view ave
```

To dismiss the floating window view, use the close box in the title bar of the view. The `view` command is one of several Command Window commands you can use to manipulate objects from the Command Window. In the next section, you use the Command Window to modify this Notebook.

# Modifying a Notebook from the Command Window

The Command Window is a convenient place to create or modify a Notebook as you generate results. For example, you can place notebook views of the objects created in the previous section by using the `place` command. Place the new objects on the page by typing the following commands:

```
> place slice
> place myGraph
> place ave
```

These commands place three new views on the Notebook: one view for `slice`, one view for `myGraph`, and one view for `ave`.

Use the following procedure to move objects on the Notebook page:

1. Right click on the object you want to move and drag the object to a new position.
2. When you have the object positioned where you want it, release the right mouse button.

Use the following procedure to resize an object:

1. Click on the object to activate it.
2. Position the mouse on the object corner to activate the resize cursor, as in Figure 3-6.

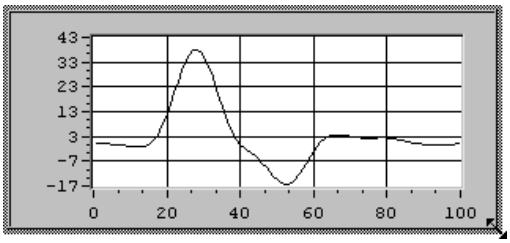



**Figure 3-6.** Resizing an Object

3. Click and drag the object border to the appropriate size.

Figure 3-7 shows the modified Notebook, which now analyzes and documents the results of a seismic experiment. To print this Notebook for a report or hardcopy documentation, use the **File»Print** command.

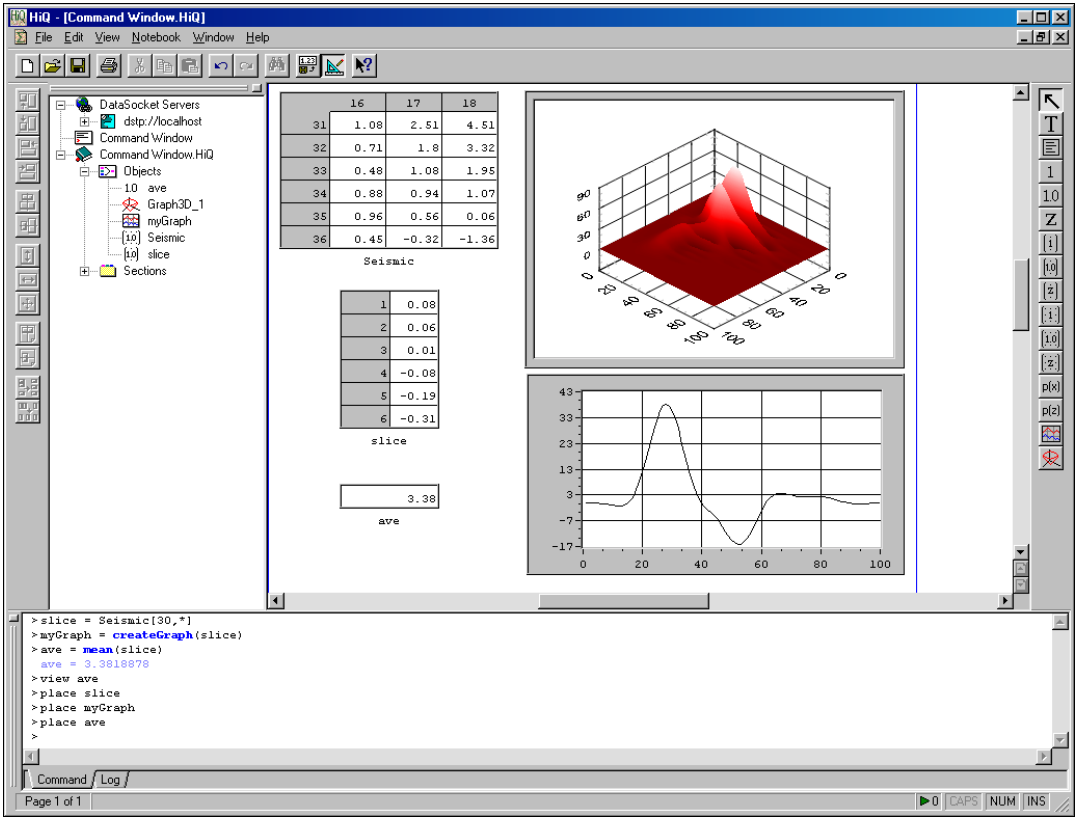



**Figure 3-7.** Modifying a Notebook from the Command Window

# Command Window Commands

The Command Window evaluates any valid HiQ-Script statement or expression and special Command Window commands. Table 3-1 lists the Command Window commands. You can type an object name at the command line to display its value.

**Note** If a Command Window command and an object share the same name and you type the name at the command line, the object is displayed, and the command does not execute. To execute the command in this situation, prefix the command with the pound sign (#). For example, type `#clear`, instead of `clear`.

**Table 3-1.** Command Window Commands

| Command | Explanation |
|---|---|
| `clear` | Clears the contents of the Command Window. |
| `clearHistory` | Clears the contents of the history. |
| `quit` | Quits HiQ, which is the same as selecting **File»Exit**. |
| `help` *`keyword`* | Displays the online help topic for the word you specify. |
| `delete` *`object_name`* | Removes *`object_name`* from the Object List. |
| `openNotebook` *`name`* | Opens the HiQ Notebook *`name`*. |
| `whatChanged` | Lists all objects that changed as a result of executing the previous command. |
| `objects` | Lists the names of all objects in the Object List. |
| `place` *`object_name`* | Places a view of *`object_name`* on the active page of the current notebook. |
| `terse` | Enters terse mode. Results are not echoed on the Command Window. |
| `verbose` | Exits terse mode. Results are echoed on the Command Window. |
| `detach` | Enters detached mode. Results are not placed or saved in any Notebook. |
| `attach` | Attaches the Command Window to the currently active Notebook. Subsequent objects created using the Command Window are placed in the Object List. |

**Table 3-1.** Command Window Commands (Continued)

| Command | Explanation |
|---|---|
| `whatis` *`object_name`* | Displays type information about *`object_name`*. |
| `cd` *`path`* | Changes the current directory to *`path`*. *`path`* can be a relative pathname or an absolute path name. *`path`* also can specify a network computer and share name. If you do not specify *`path`*, the current directory is displayed. |
| `pwd` | Displays the current directory. |
| `dir` | Lists the names of the files in the current directory. |
| `ls` | Lists the names of the files in the current directory. |
| `run` *`file_name`* | Runs the HiQ-Script contained in *`file_name`*. You can omit the `.hqs` extension. |
| `view` *`object_name`* | Places a view of *`object_name`* in an individual window. |
| `browse` *`object_name`* | Displays the ActiveX information for *`object_name`* in the ActiveX Object Browser. *`object_name`* must be an ActiveX object. |
| `matlab` | Enters MATLAB mode. |

# MATLAB Mode Commands

If you have MATLAB 5.0 or greater installed on you computer, you can enter MATLAB mode by typing `matlab` in the Command Window. The Command Window enters MATLAB mode and displays the » prompt. While you are in MATLAB mode, you can invoke any MATLAB file command and call any MATLAB file you have. Results are displayed in the Command Window and MATLAB variables are displayed in the HiQ Explorer in the familiar MATLAB manner. Refer to Chapter 11, *Getting Results as a MATLAB User*, for information about sharing information between HiQ and MATLAB.

To return to HiQ mode, type `hiq`.

# 4

# Getting Your Data into HiQ

This chapter explains how you can get different types of data into HiQ. In this chapter, you build two Notebooks—the Climate Notebook and the Seismic Notebook—and use the Import Wizard to get your data into HiQ. You enhance these Notebooks over the next several chapters.

Many Notebooks you create with HiQ involve analyzing and visualizing data. Usually this data exists in a file on your hard drive. When you need to import text or binary data from a file, you can take advantage of the HiQ Import Wizard.

Although you can use the default import settings in most cases, the Import Wizard allows you to define the data format in your file. When you change any of the import setting options in the Import Wizard, the data Preview window automatically updates to indicate the format of the imported data.

You also might choose to create a custom import option for special kinds of data. Refer to the documentation on the `import` built-in function in the online help.

If you need to export data from HiQ, use the Export Wizard, which works similarly to the Import Wizard. To access the Export Wizard, right click on the object you want to export and select Export.

## Climate Notebook: Importing Data

Over the next several chapters, you build a Notebook that performs a data fit on climate data from a file. As you conclude each task, save the Notebook to use it in the next chapter. If you forget to save, you can start with a prebuilt Notebook at the beginning of each chapter.

1. Open a new file (**File»New**) and save the new notebook as `climate.HiQ` (**File»Save As**).
2. Select **Notebook»Import Wizard**, and click the **Browse** button.
3. Select the file `climate.dat` in the `Examples\Data` folder and click **Open**. This file contains a meteorological data set of yearly rainfall totals and temperatures.

The Preview window displays the raw, unformatted contents of the file, as shown in Figure 4-1.

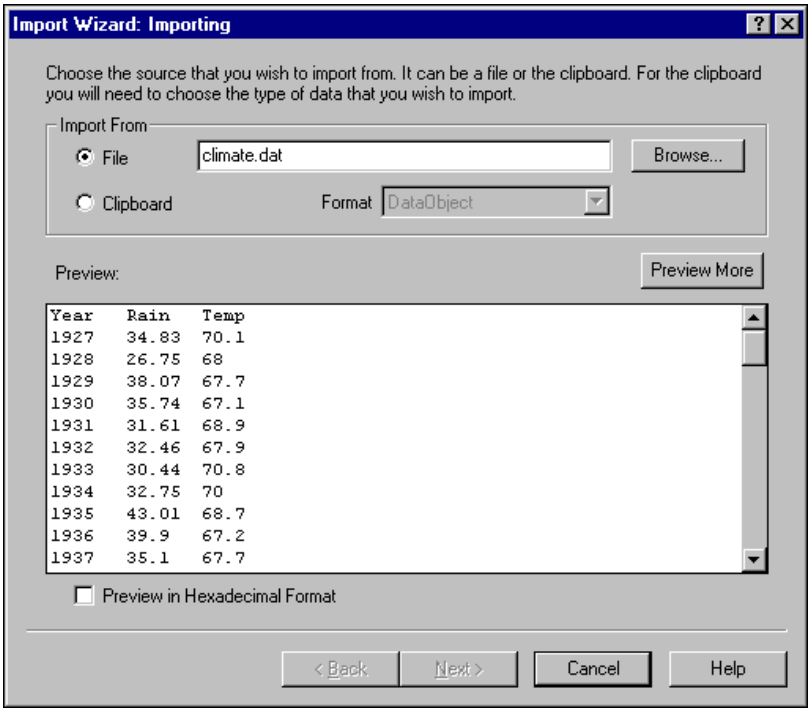



**Figure 4-1.** Importing Climate Data with the Import Wizard

In this example, the contents of the file are readable because the data was saved in ASCII format.

4. Click **Next**. On the second page of the Import Wizard, HiQ imports a portion of the file and displays the data in the Preview window.

   On this page, you can change the name of the object (HiQ uses the filename by default). You also can start importing the data on a row other than 1, which is useful if the data file contains a header that you do not want to import.

**Tip** The Import Wizard automatically parses and hides unneeded data. In this example, HiQ ignores the first line because it does not contain any numbers. Use the **Preview Original File** option to view the entire file.

5. Select **Generate Script** and name the script `ImportData`, as shown in Figure 4-2. Although this script is not required to import the data, later you use it to complete the Notebook as a problem solver.

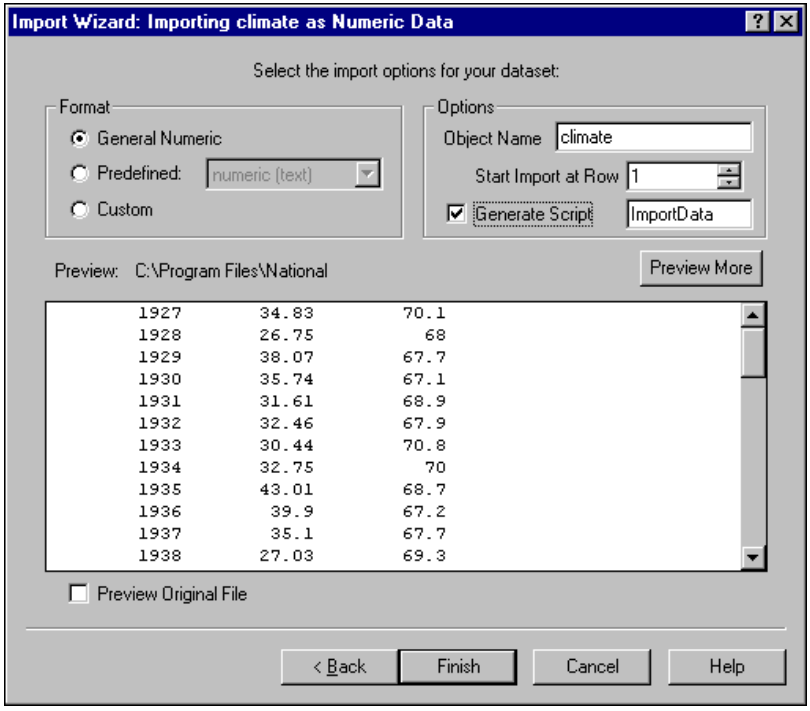



**Figure 4-2.** Generating a Script with the Import Wizard

6. Click **Finish** to import the data.
7. Save the Notebook.

**Tip** You can move, resize, and modify the appearance of any object on a Notebook page. For example, to quickly expand the matrix to display all of its data, right click on the matrix and select **Show All Elements**.

Figure 4-3 shows the Climate Notebook with the imported data. The climate data, which was imported as a matrix object, is visible on the Notebook page. The script generated as a result of the import is not visible on the Notebook page, but it is listed in the Explorer Object List.

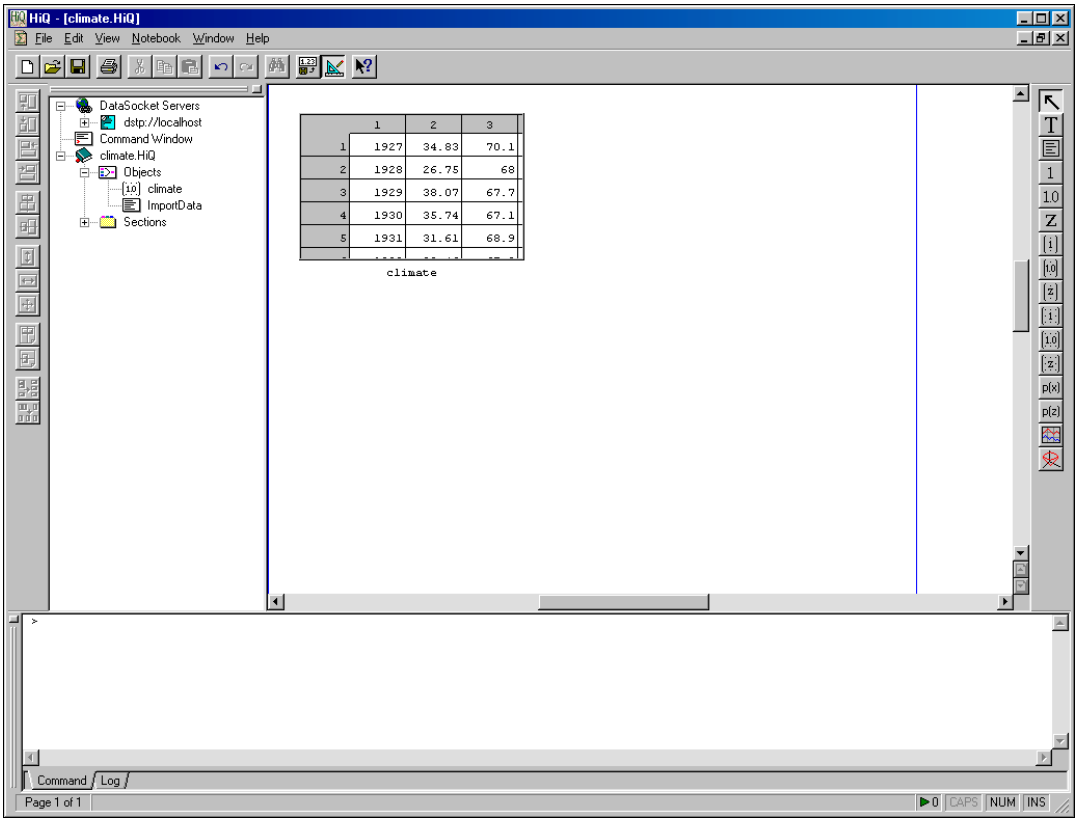



**Figure 4-3.** Climate Notebook with the Imported Data

# Seismic Notebook: Importing Data

Over the next several chapters, you create a Notebook that visualizes and analyzes seismic data from a file. As you conclude each task, save the Notebook to use it in the next chapter. If you forget to save, you can start with a prebuilt Notebook at the beginning of each chapter.

1. Open a new file (**File»New**) and save the new notebook as `seismic.HiQ`. (**File»Save As**).
2. Select **Notebook»Import Wizard** and press the **Browse** button.

3. Select the `Seismic.bin` file from the `Examples\Data` folder and click **Open**. This file contains binary data representing data collected from a seismic event. Because a raw binary data file contains no format information, you must know the structure of the data to successfully import it. In this example, the data represents a matrix of values with 100 rows and 100 columns and was written in IEEE Real 64 format.

   The Preview window displays the raw, unformatted contents of the file, as shown in Figure 4-4. Scroll through the Preview window to view the binary data.

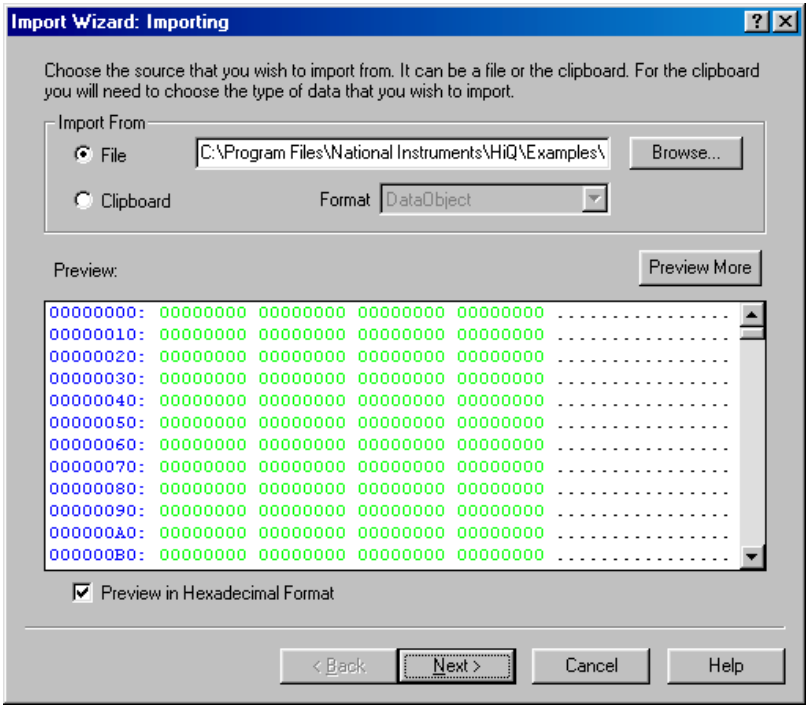
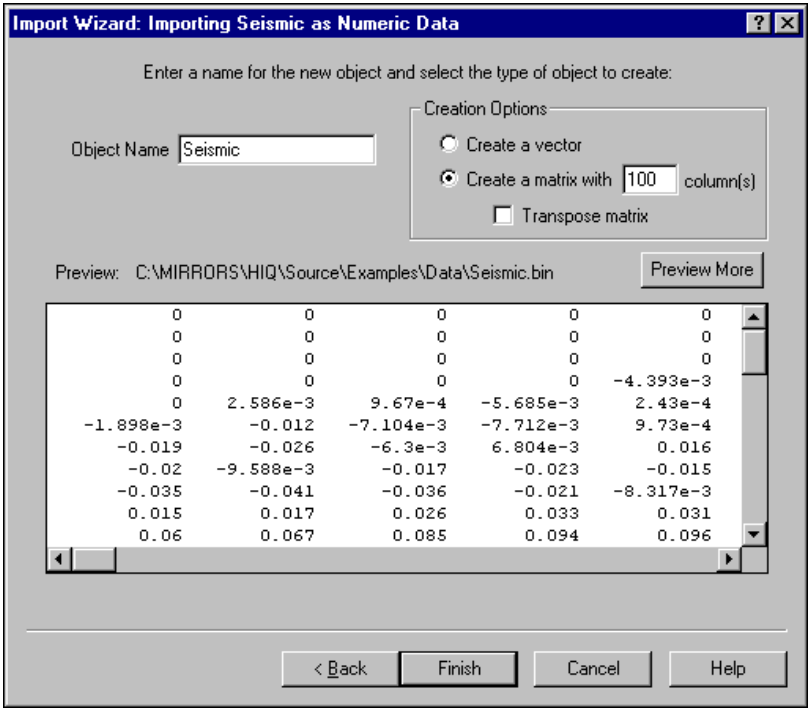



**Figure 4-4.** Importing Raw Seismic Data with the Import Wizard

   In this example, the file is not readable because the data is saved in an unreadable binary format. You must first tell HiQ how to format the data before the Preview window is readable.

4. Click **Next**. HiQ recognizes that the file is binary and automatically selects **Predefined: numeric (binary)** for the format.

5. Click **Next**. On the third page, HiQ imports a portion of the file and displays the data in the Preview window. To verify the data, scroll through the column of numbers. If the data had been written using a different binary format, you can set the correct **Data Type** and **Byte Ordering** options to specify the format.
6. Click **Next**. Select **Create a matrix** from the creation options and specify **100** columns, as shown in Figure 4-5, to import this data as a matrix with 100 columns.




| | | | | |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 0 | -4.393e-3 |
| 0 | 2.586e-3 | 9.67e-4 | -5.685e-3 | 2.43e-4 |
| -1.898e-3 | -0.012 | -7.104e-3 | -7.712e-3 | 9.73e-4 |
| -0.019 | -0.026 | -6.3e-3 | 6.804e-3 | 0.016 |
| -0.02 | -9.588e-3 | -0.017 | -0.023 | -0.015 |
| -0.035 | -0.041 | -0.036 | -0.021 | -8.317e-3 |
| 0.015 | 0.017 | 0.026 | 0.033 | 0.031 |
| 0.06 | 0.067 | 0.085 | 0.094 | 0.096 |

**Figure 4-5.** Importing Formatted Seismic Data

7. Click **Finish** to import the data.
8. Save the Notebook.

Figure 4-6 shows the Seismic Notebook with the imported data. The seismic data, which was imported as the matrix Seismic, is visible on the Notebook page.

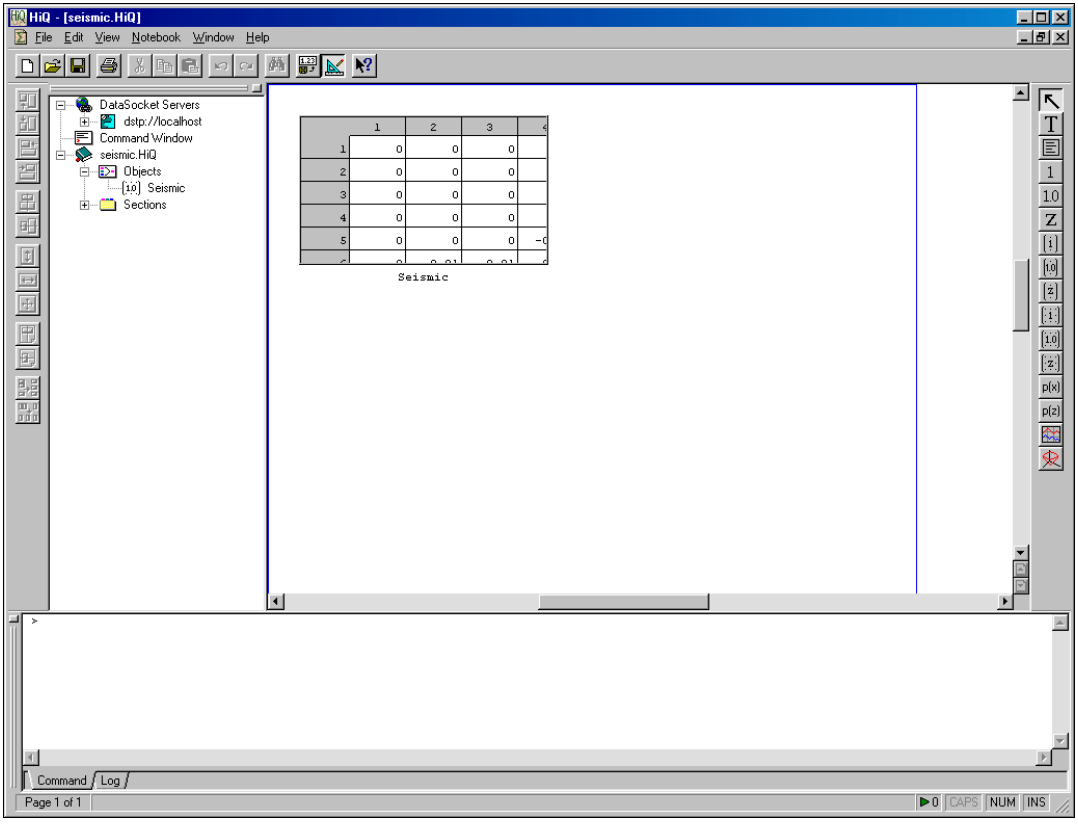



**Figure 4-6.** Seismic Notebook with Imported Data

**Tip** You can use the Import Wizard to get almost any type of data into HiQ. The Import Wizard can parse and import data as several different types of HiQ objects. If you have a unique data format, use the custom import option and the import built-in function.

If you need to export data from HiQ, use the Export Wizard, which works similarly to the Import Wizard. To access the Export Wizard, right click on the object you want to export and select **Export**.

# 5

# Publishing and Subscribing to Live Data

This chapter explains how you read and write live data from HiQ. Although the Import Wizard is the right tool for importing static data, such as historical weather or seismic information, sometimes you will need to access dynamic data that resides on a server somewhere on your network or on the Internet.

For connecting to live data, HiQ offers National Instruments DataSocket technology as a built-in feature. With DataSocket, HiQ provides interactive connectivity to several types of data sources (read access) and data destinations (write access). Through a URL, you can connect to files, HTTP/FTP servers, OLE for Process Control (OPC) servers, and National Instruments DataSocket servers on networked computers or over the Internet.

**Note** Although the examples in this chapter demonstrate DataSocket connectivity using DataSocket servers, you do not have to limit data publishing or subscribing to the DataSocket protocol. You can create connections from HiQ text, scalar, vector, matrix, and graph plot objects to different types of server protocols.

A DataSocket server is an application that accepts and stores information from data sources and relays it to other destinations. Because HiQ has built-in DataSocket connectivity, you can send or retrieve data from DataSocket servers on the same computer or other computers connected though a TCP network, such as the Internet.

In this chapter, you add to the Climate Notebook live data describing the current weather in Austin, Texas. With the Seismic Notebook, you learn how to make data from the Notebook available to others on the Internet.

# Climate Notebook: Subscribing to Live Temperature Measurements over the Internet

National Instruments publishes live weather measurements describing the weather conditions at its headquarters in Austin using a DataSocket server. In this example, you will view the data items published on the server and display current weather data on the Notebook page.

1. Open the `climate.HiQ` Notebook if you completed the Notebook example in Chapter 4, *Getting Your Data into HiQ*. Otherwise, open `climate5.HiQ` from the `Examples\Getting Results` folder.
2. Right click **DataSocket Servers** in the Explorer and select **Add/Remove DataSocket Servers**.
3. Click **Add** to add the name of the National Instruments DataSocket server that publishes the weather data, and type `weather.ni.com`
4. Click **OK**. Notice that the DataSocket server appears under DataSocket Servers in the Explorer as `dstp://weather.ni.com`

**Note** You must have a live Internet connection for this example to work. If you see a red X to the left of the DataSocket server name, make sure your Internet connection is active and then right click on **DataSocket Servers** and select **Refresh Explorer View**. If you do not have an Internet connection, continue to Chapter 6, *Visualizing Data with 2D and 3D Graphs*.

5. Expand the server view to see the data items published on the server.
6. Find the weather/current data item. This item contains information about the current weather conditions in Austin. Notice that HiQ identifies the data type of this item as a real vector. The vector contains the following information:

| | |
|---|---|
| weather_current[1] | timestamp (time in days elapsed since December 30, 1899) |
| weather_current[2] | temperature |
| weather_current[3] | windspeed |
| weather_current[4] | wind direction |
| weather_current[5] | humidity |

7. Drag-and-drop the weather/current item from the Explorer to the Notebook page. HiQ creates a real vector object with the same name as the data item and displays a view of the object on the Notebook page. Notice that this new object appears in the object list in the Climate Notebook Explorer.

**Tip** If you cannot see all elements in the vector, right click on the view and select **Show All Elements**.

8. To change the labels on the vector to document what each element contains, click on the label and type the new label.
9. Save the Notebook as `climate.HiQ`.

Figure 5-1 shows the Climate Notebook with the current temperature. When new data is published to the weather DataSocket server, HiQ automatically updates the real vector object view on the Notebook page. You can verify the temperature by visiting the National Instruments DataSocket weather page at `www.ni.com/datasocket` and following the links to the live weather station.

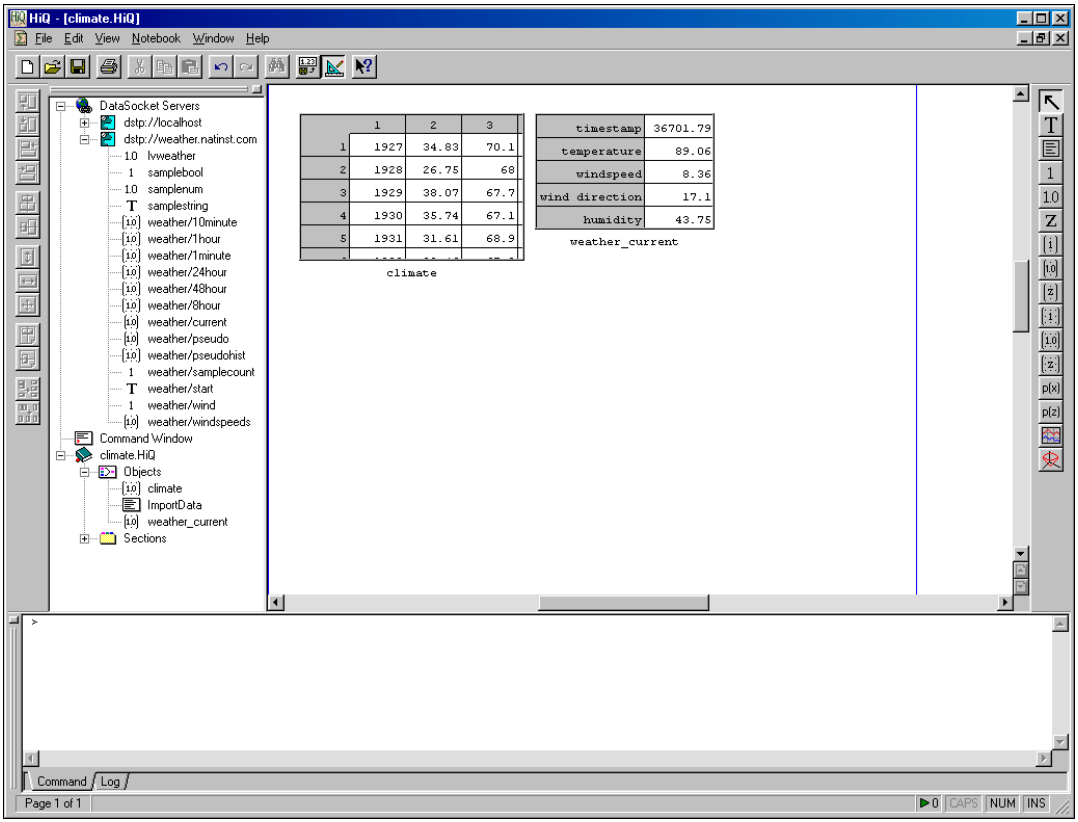



**Figure 5-1.** Climate Notebook with Live Weather Information from Austin, Texas

# Seismic Notebook: Publishing Data from HiQ

Imagine that you want to share seismic data from the Notebook with others in your office. You can publish data from a HiQ Notebook to a DataSocket server on your computer. The DataSocket server that resides on your computer is called `localhost`.

1. Launch the DataSocket server from your Windows **Start** menu if it is not already running. In HiQ, right click on **DataSocket Servers** and select **Refresh Explorer View**. Browse through the data items listed under localhost. If you are not currently publishing data on the DataSocket server, you will find only sample data items.
2. Open the `seismic.HiQ` Notebook if you completed the Notebook example in Chapter 4, *Getting Your Data into HiQ*. Otherwise, open `seismic5.HiQ` from the `Examples\Getting Results` folder.
3. To publish the data in the Seismic matrix to your local DataSocket server, right click on the matrix view, select **Properties**, and view the DataSocket property page. In the Write to Destination section, enable updates and enter the URL to publish the data to the local DataSocket server. In this example, you dynamically create a data item named seismic:

   ```
   dstp://localhost/seismic
   ```

   The data item will exist on the DataSocket server only as long as the Seismic object exists and remains connected to the DataSocket server. If you delete the object or close the Notebook, the seismic data item will no longer exist on the DataSocket server.

**Tip** You also can drag the Seismic matrix from the Explorer and drop it on `dstp://localhost` in the DataSocket Servers browser. Your new data item is named with the same name as the HiQ object.

4. Click **OK**.
5. View the DataSocket server (as shown in Figure 5-2) to see the number of packets that have been transferred. A packet represents data that has been transferred to or from the DataSocket server. Now change the data in the Seismic matrix and view the DataSocket server again. Notice that the number of packets has increased, indicating that your new data has been transferred to the DataSocket server.

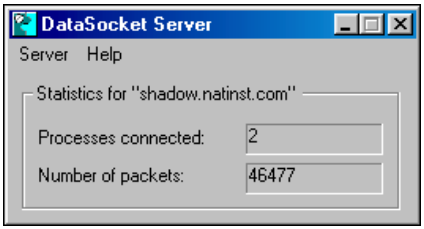



**Figure 5-2.** DataSocket Server

**Note** In Chapter 6, *Visualizing Data with 2D and 3D Graphs*, you will learn how to synchronize the seismic data between the seismic matrix and a three-dimensional graph.

6. You and anyone else connected to the network can access and retrieve the published data. For others to access the data items on your local DataSocket server, they can use the following URL:

   `dstp://MachineName/ItemName`

   where `MachineName` is the name of your computer and `ItemName` is the data item on the DataSocket server. For the DataSocket server shown in Figure 5-2, the URL is `dstp://shadow/seismic`

**Tip** National Instruments LabVIEW and Measurement Studio also include built-in DataSocket connectivity, so DataSocket data is easy to access from LabVIEW, LabWindows/CVI, Visual C++, and Visual Basic.

# 6

# Visualizing Data with 2D and 3D Graphs

This chapter describes data visualization, which is an indispensable tool for communicating results, interpreting your analysis, and gaining an intuitive understanding of what your data represents. In this chapter, you enhance the Climate and Seismic Notebooks with graphs displaying the data you imported in the previous chapter.

## Climate Notebook: Visualizing Rainfall and Temperature Data in a 2D Graph

You can graph most data in two dimensions, where the data is plotted against a single, independent variable. In this example, you plot temperature and rainfall data on a 2D graph.

### Creating a Graph

1. Open the `climate.HiQ` Notebook if you completed the Notebook example in Chapter 5, *Publishing and Subscribing to Live Data*. Otherwise, open `climate6.HiQ` from the `Examples\Getting Results` folder.
2. To add a 2D graph object to the Notebook, select the 2D graph tool in the HiQ Tools toolbar.

**Tip** You also can create objects using the **Notebook»Create** command. For this example, select **Notebook»Create»2D Graph**.

3. Click and drag on the Notebook page where you want the new graph to appear, as shown in Figure 6-1.

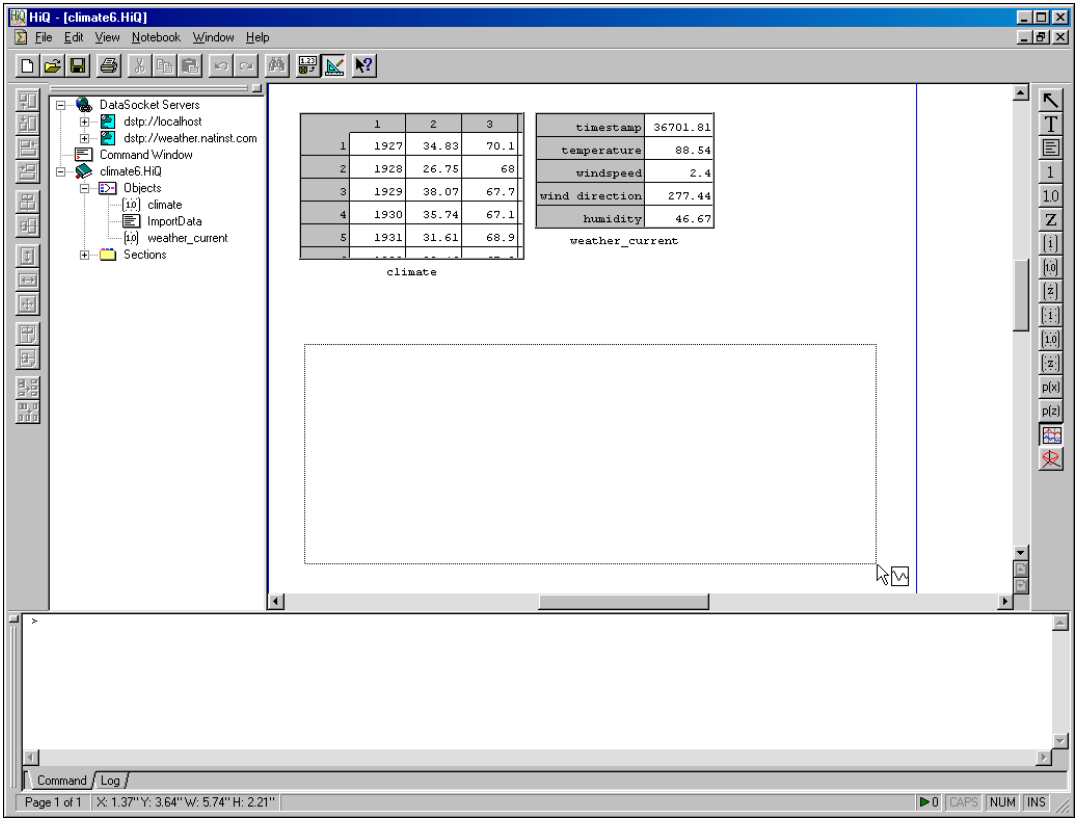



**Figure 6-1.** Clicking and Dragging to Place a 2D Graph Object on a Notebook Page

4. Change the name of the graph object in the Explorer. Select the object in the Object List, place the cursor in the object name, and replace `Graph2D_1` with `graph` to rename the object.

## Plotting a HiQ Data Object

Add a new plot to the graph as described in the following steps. After completing the first six steps, compare your New 2D Plot dialog box with Figure , which shows the correct settings.

1. Right click on the graph and select **New Plot**.
2. Select **Data** as the **Range** option for your graph and press the **Browse** button to the right. The range is the y data to plot.

3. Select `climate` in the **From Matrix** list, choose **Column**, and type `2`. Click **OK.**
4. Select **Data** for the **x Domain** option for your graph and press the **Browse** button to the right. The domain is the x data to plot.
5. Select `climate` in the **From Matrix** list, choose **Column**, and type `1`. Click **OK**.
6. Select the **Generate Script** option and name the script `Plot1`, as shown in Figure 6-2. Although this step is not required to plot the data, you can use this script later to complete the Notebook as a problem solver.

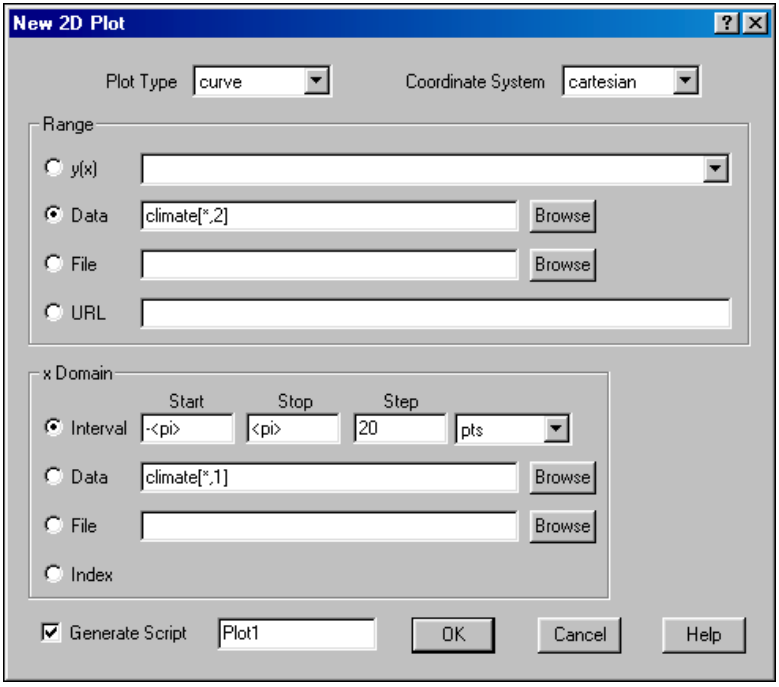



**Figure 6-2.** Adding a Plot to a Graph

7. Click **OK** to add the new plot.

**Tip** You can create a new plot with any valid DataSocket URL as well. Refer to Chapter 5, *Publishing and Subscribing to Live Data*, for information about connecting to external data sources.

Figure 6-3 shows the graph and new line plot, which represents the yearly rainfall for the last 71 years.

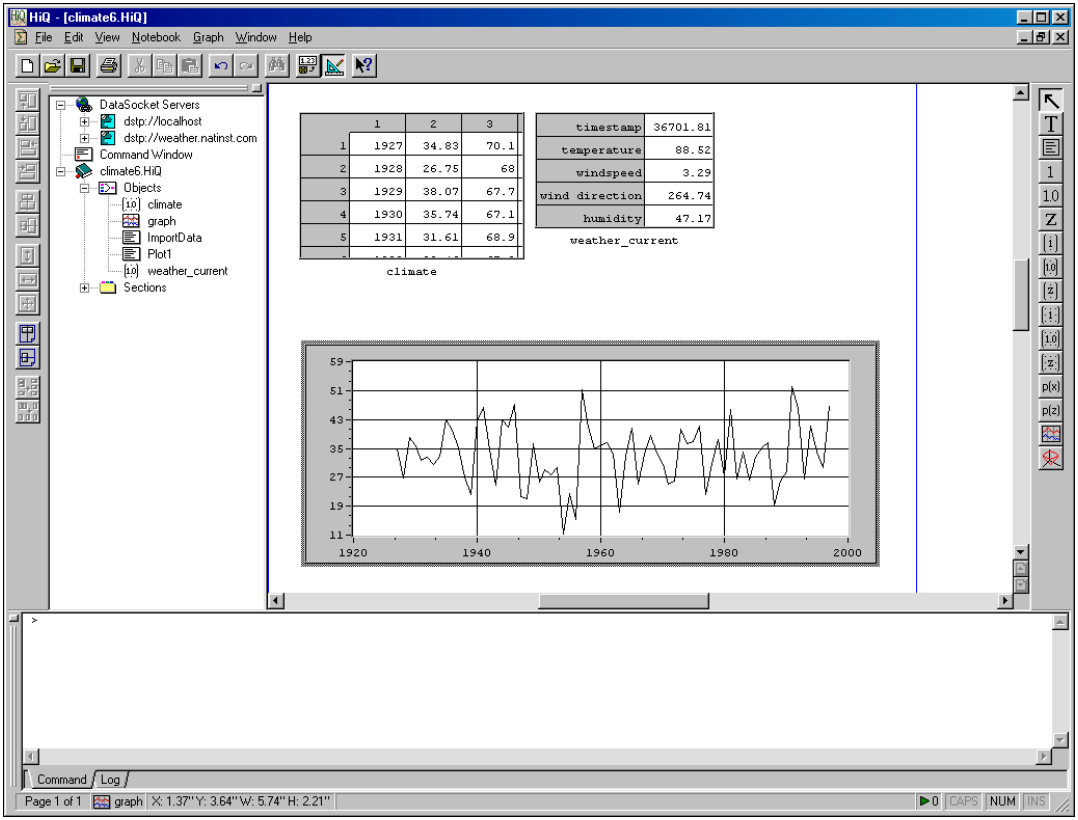



**Figure 6-3.** Plotting Climate Data on a 2D Graph

## Modifying the Graph and Plot

Many times the reports and presentations you create require graphs with special formatting. With HiQ, you can format your graphs to match your needs. In this example, rainfall data might better be represented by a bar graph. To modify the properties of a plot and graph, use the following procedure.

1. Right click on the graph and select **Properties** to access the 2D graph property pages, as shown in Figure 6-4.

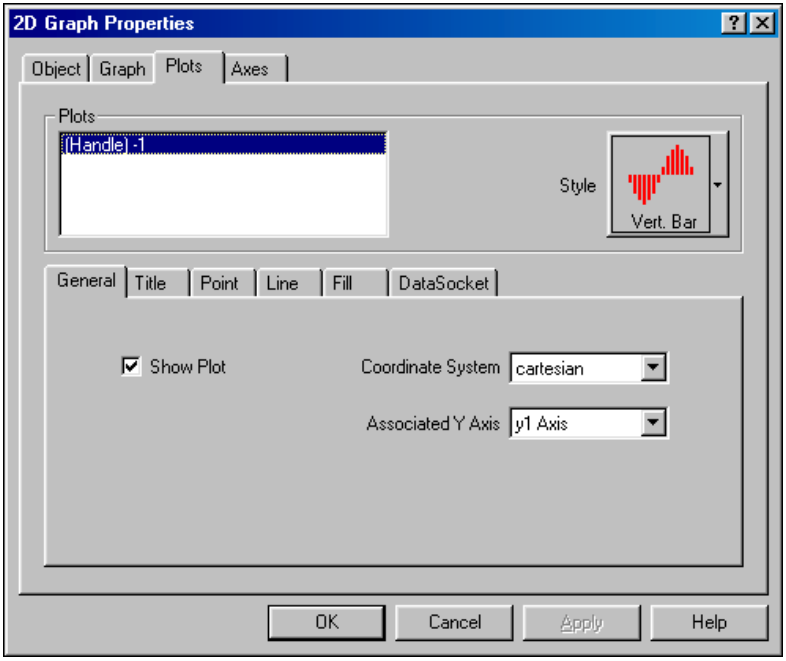



**Figure 6-4.** 2D Graph Property Pages

2. To modify plot properties, change the plot style to a vertical bar (**Vert. Bar**) with the **Style** property on the Plots page.
3. To modify graph properties, such as the caption of the graph, use the Graph page. Enter `Austin Climate` for the graph **Title**.
4. To modify axis properties, use the Axes page. In the **Axes** list box, select **x Axis** and enter `Year` as the **Title**.
5. Set **Axes** to **y1 Axis**. y1 denotes the first y-axis. Two-dimensional graphs can have up to eight y axes. You use multiple y-axes later in this chapter. Enter `Rainfall (inches)` for the **Title** of the y1 Axis.
6. Click **OK**.

Figure 6-5 shows the Climate Notebook with the modified plot. The bar graph effectively communicates the trend in rainfall totals for 71 years.

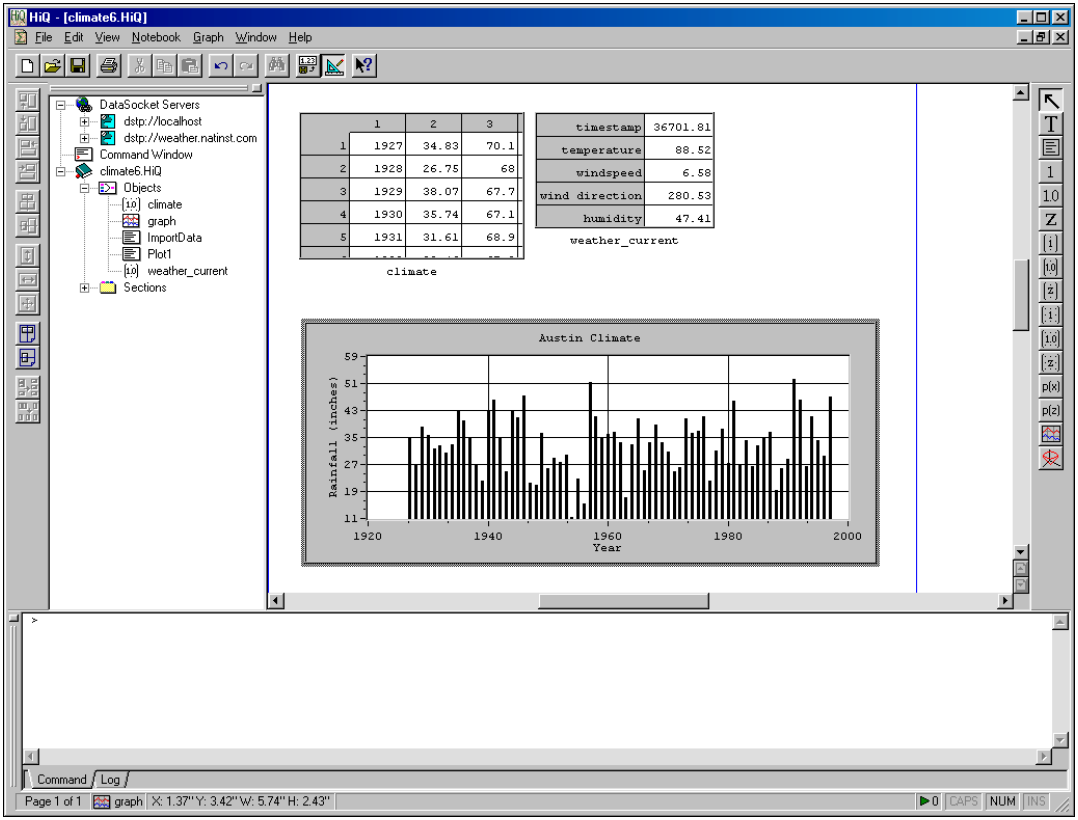



**Figure 6-5.** Climate Notebook After Modifying the Plot

You can add multiple plots to this graph to view the same data in different forms or to compare different data sets. In this example, you might want to add a new plot containing temperature data, as the next section explains.

## Working with Multiple Plots in a Graph

A HiQ graph can display multiple plots. In this section, you add a new plot of temperature data to the same graph that you created in the previous section.

1. Add a second plot of the rainfall data, specifying the third column for the range of data (climate[*,3]). Select the **Generate Script** option on the New 2D Plot dialog box and name the script `Plot2`. Figure 6-6 shows the correct settings for this plot.

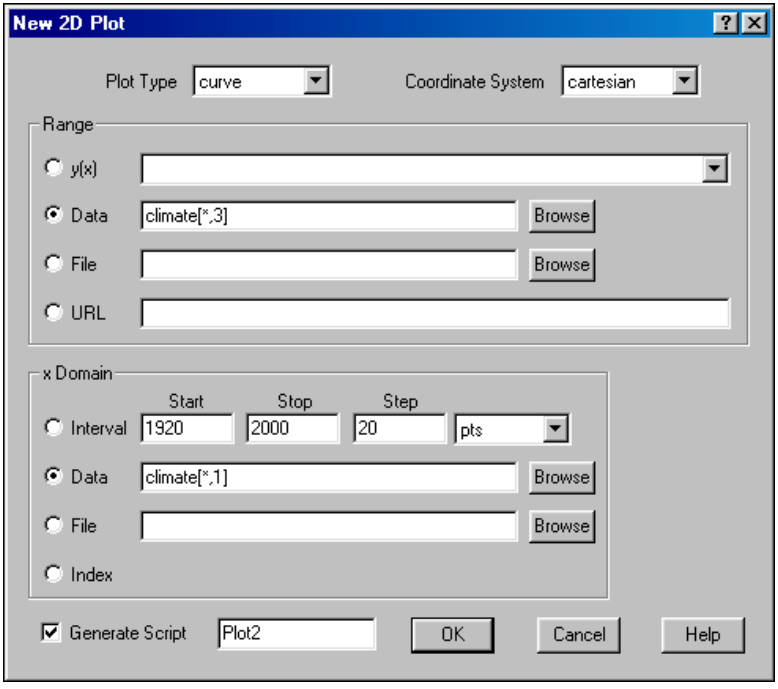



**Figure 6-6.** Adding a Second Plot to the Graph

Later you use this script to complete the Notebook. To review this procedure, see *Plotting a HiQ Data Object*.

The graph now has a second plot representing the average temperature over the same time period. Because the y-axis is labeled for the rainfall plot, we need to add a second y-axis for the temperature data. This second axis displays the temperature scale on the right side of the graph.

2. Right click on the graph and select **Properties**.
3. On the Plots page, select the rainfall plot `(Handle)-2` from the list of available plots.
4. On the General page on the Plots property page, choose **y2 Axis** as the **Associated Y Axis**, as shown in Figure 6-7.

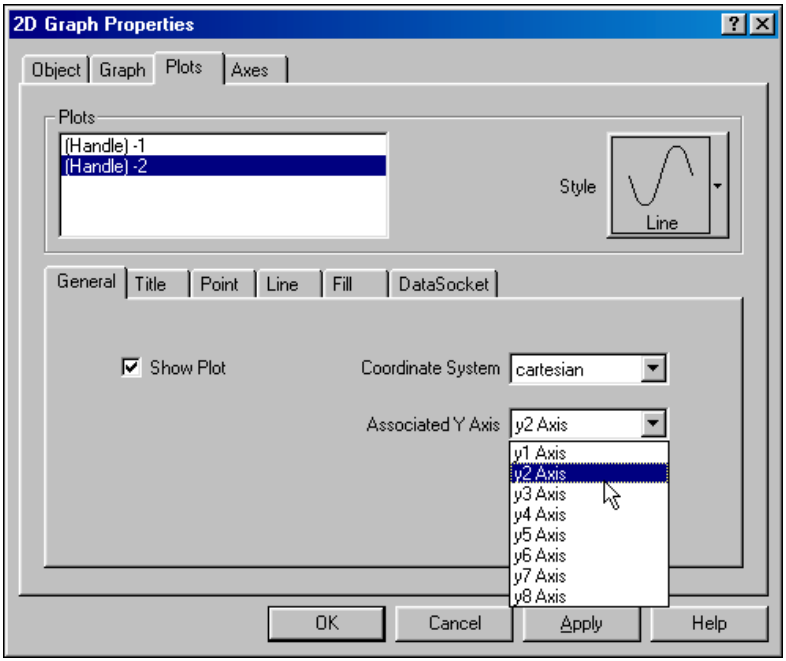



**Figure 6-7.** Adding a Second Y-Axis

5. On the Axes property page, select the **y2 Axis**.
6. On the Title page, name the y2 axis `Temperature (degrees F)`.

7. On the Range tab, set the **Mode** to **Manual** with a **Minimum** of 50 and a **Maximum** of 80, as shown in Figure 6-8.

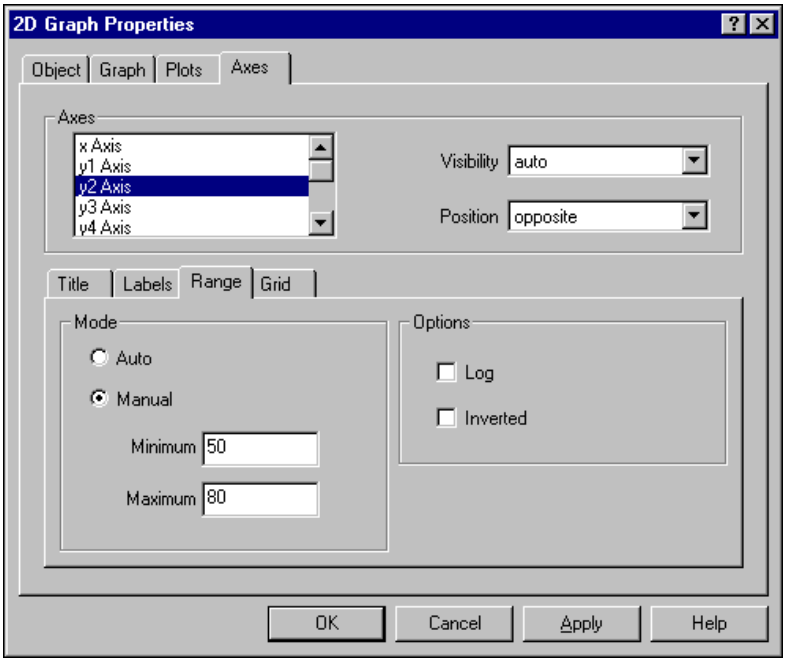



**Figure 6-8.** Setting Range Properties for a Y-Axis

8. Click **OK**.
9. Save the Notebook as `climate.HiQ`.

Figure 6-9 shows the Climate Notebook with the second plot representing temperature data. This Notebook effectively presents two related data sets with different ranges using multiple y axes on a single graph.

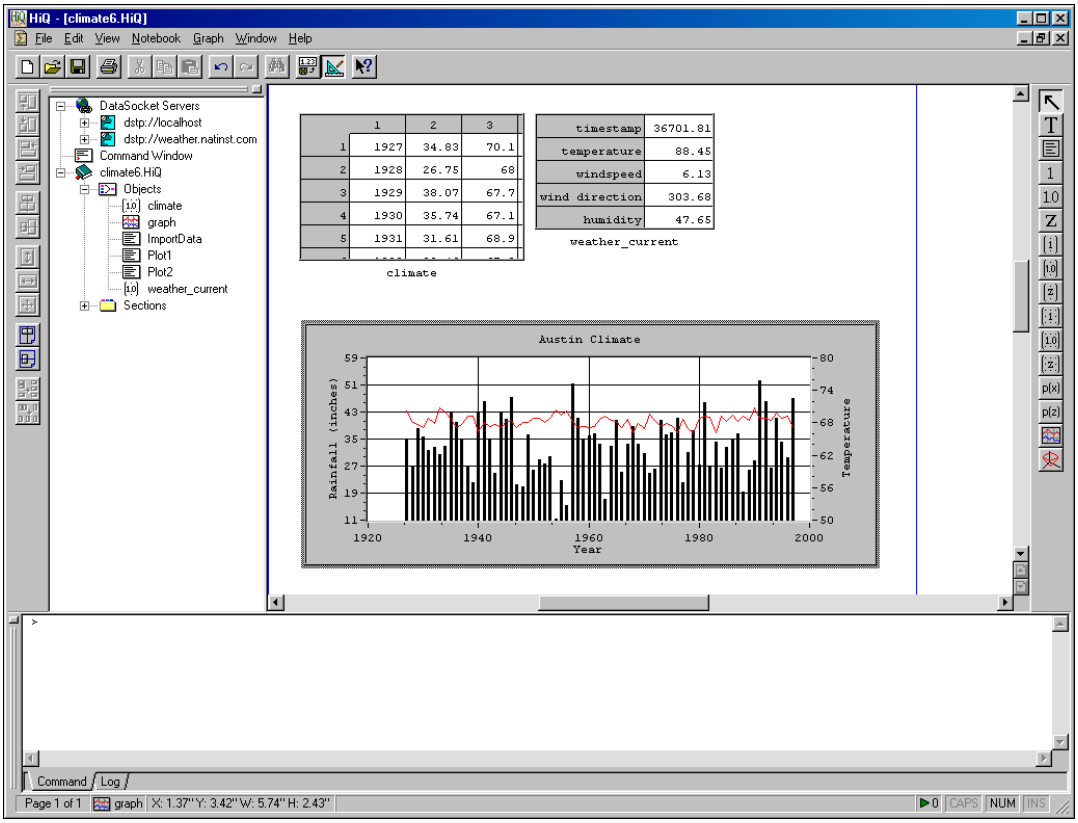



**Figure 6-9.** Climate Notebook with Two Related Data Plots

# Seismic Notebook: Visualizing Seismic Data in a 3D Graph

For many real-world data sets—such as terrain contours, the motion of an aircraft in three dimensions, the temperature distribution on a surface, and joint time-frequency analysis—you need to visualize data in three dimensions.

## Visualizing Live Data in 3D

You can drag-and-drop any compatible numeric object from your Notebook on a graph to visualize the data in the object. You can visualize DataSocket data just as easily. In this part of the example, you visualize the data stored in the seismic matrix and learn how to change seismic data without having to manually update the plot.

1. If you just completed the `seismic.HiQ` Notebook example in Chapter 5, *Publishing and Subscribing to Live Data*, continue to Step 2. Otherwise, open `seismic4.HiQ` from the `Examples\Getting Results` folder, and drag-and-drop the Seismic matrix from the Explorer to the `localhost` DataSocket server.

**Note** To learn about publishing your data using DataSocket, refer to Chapter 5, *Publishing and Subscribing to Live Data*.

2. To add a 3D graph object to the Notebook, select the 3D graph tool in the HiQ Tools toolbar, and click and drag on the Notebook page where you want to place the new graph.

3. Drag the seismic data from the localhost server to the graph on the Notebook page. When your cursor looks like the one in Figure 6-10, you can drop the data on the graph.

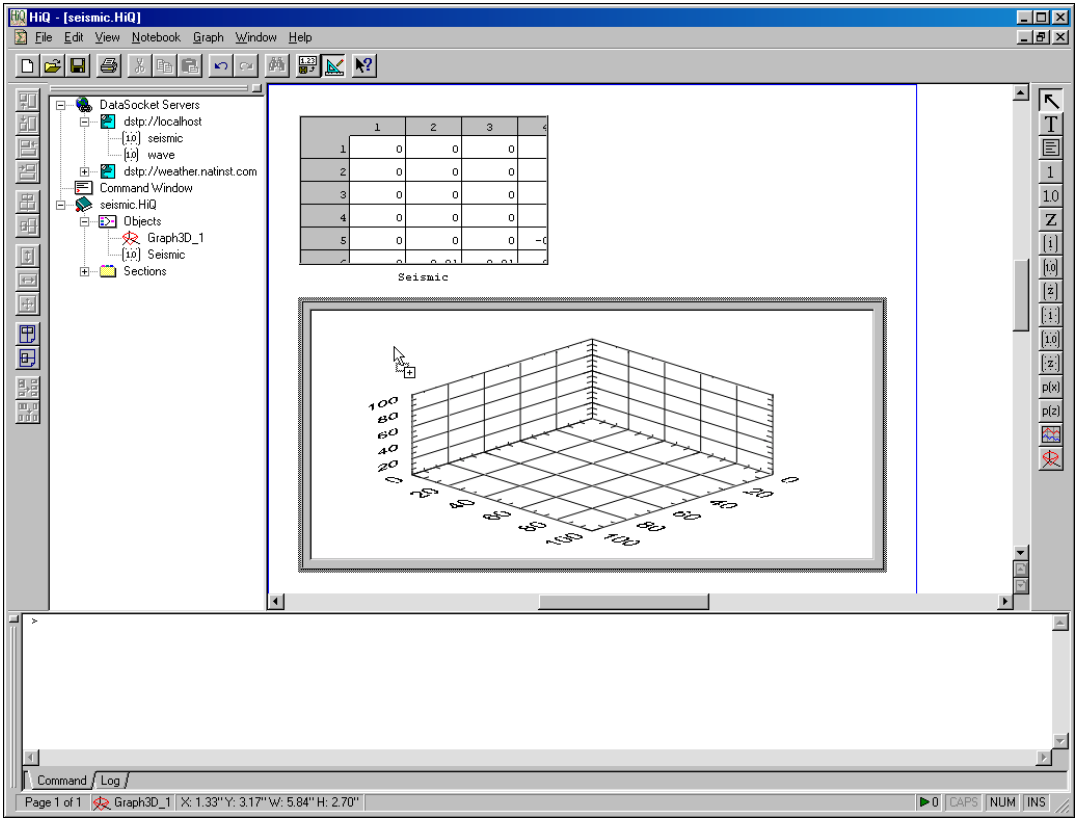



**Figure 6-10.** Drag-and-Drop to Plot Seismic Data

The data in the original Seismic matrix and the 3D plot are linked through the local DataSocket server. When you change the data in the matrix, DataSocket automatically updates the plot.

**Tip** If you want to share data between two or more HiQ objects in a Notebook, publish the data to your local DataSocket server. Create your new objects and specify the shared data as the source. When you change the published HiQ object, all connected objects will update automatically.

## Modifying a 3D Graph and Plot

Three-dimensional graphs have all 2D graph properties plus several other properties of their own. You can access all properties from the 3D graph property pages. For example, you might want to change the color of the surface plot and project the data to both the XZ and YZ planes.

1. Right click on the graph and select **Properties**.
2. On the Graph page, enter `Seismic Activity` for the graph **Title**.
3. On the Axes page, select the axis and enter the appropriate title:

| Axis | Title |
|---|---|
| x | `Time (sec)` |
| y | `Frequency (Hz)` |
| z | `Magnitude` |

4. From the Plots–General page, change the **Color Map** to **color spectrum**, as shown in Figure 6-11.

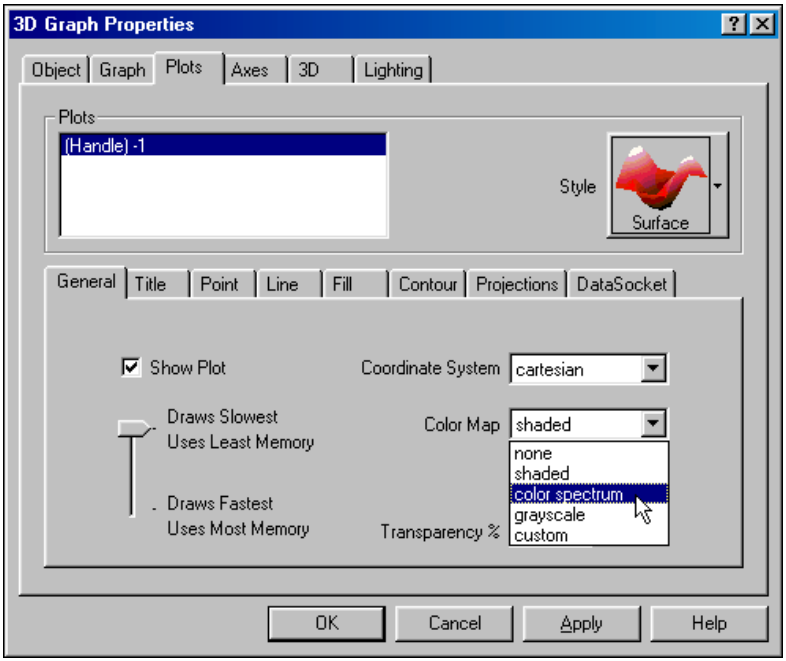



**Figure 6-11.** Setting the Plot Color Map

5. Click **Apply** and **OK** and examine the plot. Often you can visualize much more detail in a three-dimensional plot by customizing the color map. Return to the Plots property page and click the **Customize** button to create your own color map.
6. To design a custom color map, add new color breakpoints to the color map and select various colors for each breakpoint to highlight the details of the plot, as shown in Figure 6-12. You do not need many breakpoints to create an effective color map. Click **OK** to apply the color map.

**Tip** Experiment with different breakpoints, colors, and options. Try rotating the graph and viewing the plot from the side to determine if your color map provides enough color contouring.

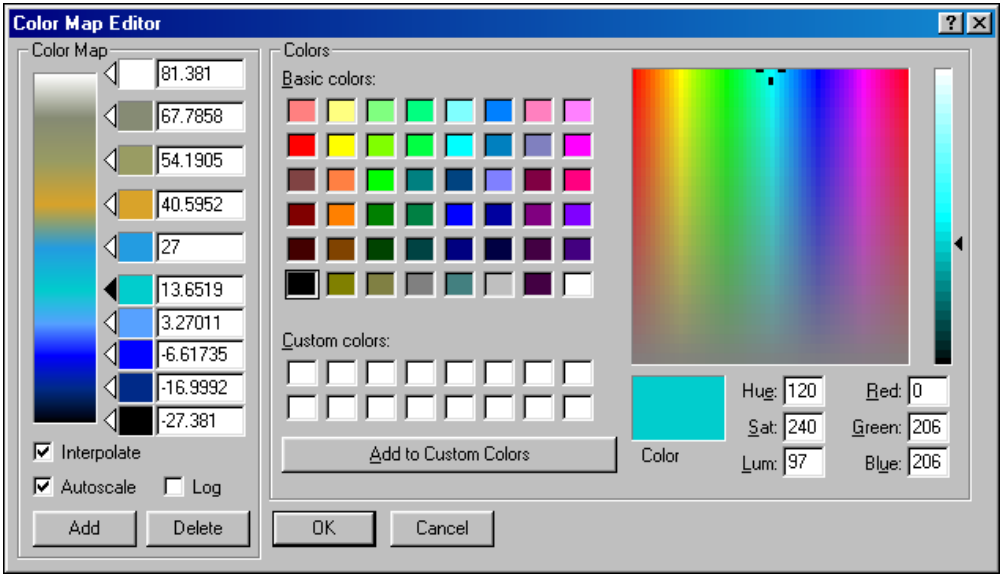



**Figure 6-12.** Creating a Custom Color Map

7. On the Projections page, select **Project to X-Z Plane** and **Project to Y-Z Plane**.
8. Save the Notebook as `seismic.HiQ`.

Figure 6-13 shows a graph similar to the one you have created that displays data regarding seismic activity as a function of time and frequency. You can experiment with the other 3D property pages—3D and Lighting—to become more familiar with the flexibility of display formats for 3D graphs.

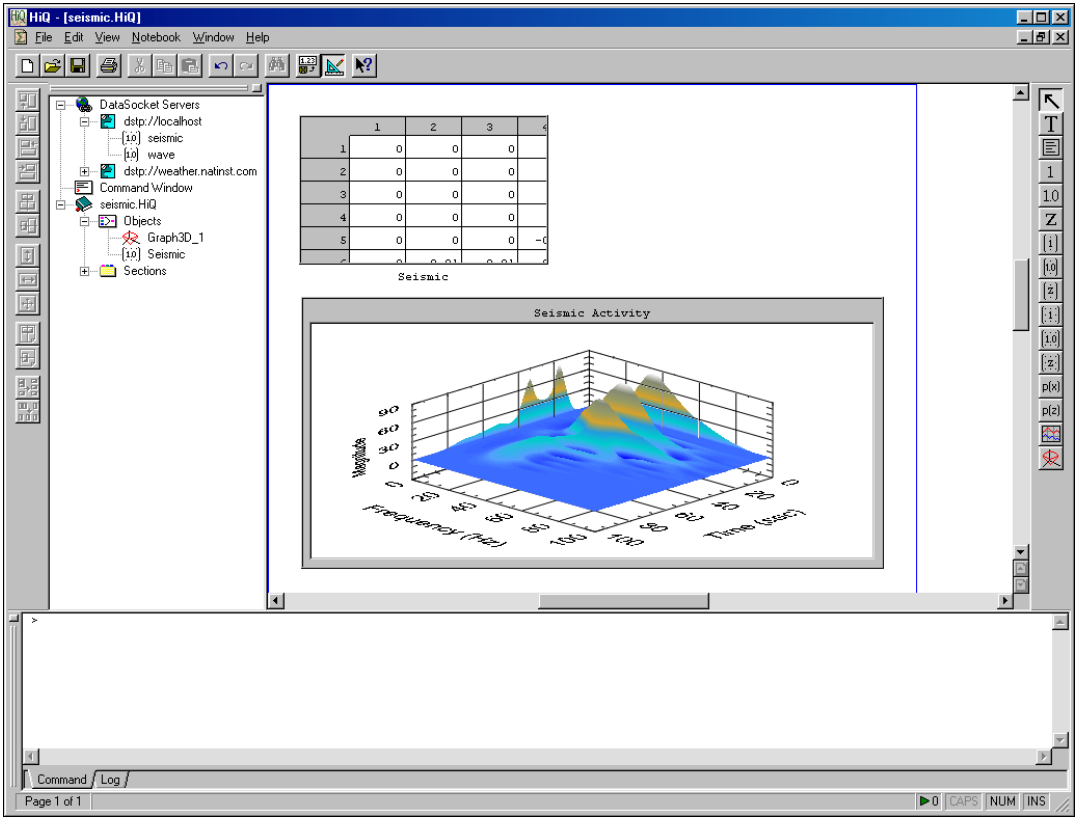



**Figure 6-13.** Seismic Notebook with Data Visualized in a 3D Graph

## Rotating, Zooming, and Panning a 3D Graph

After putting a 3D graph into active mode, you can change the viewing position of it using combinations of the mouse, the <Alt> key, and the <Shift> key.

**Note** Click on a graph object to put it into active mode (the mode for editing the graph); a shaded border appears around the object. If you do not see the shaded border, the object is probably in selected mode (the mode for sizing and moving the object as a whole). Deselect the object by clicking on a blank space in the Notebook page and then click on the object again to put it into active mode.

To rotate the graph, click and hold anywhere in the graph area and drag the mouse up, down, left, or right to rotate the graph up, down, left, or right respectively.

To zoom the graph, click and hold anywhere in the graph area while pressing the <Alt> key. Dragging the mouse down zooms the graph in and dragging the mouse up zooms the graph out.

To pan the graph, click and move the mouse anywhere in the graph area while pressing the <Shift> key.

# 7

# Analyzing Data with HiQ-Script

This chapter introduces HiQ-Script as an analysis tool. In this chapter, you analyze the data in the Climate and Seismic Notebooks with HiQ-Script from the Command Window or the Notebook.

HiQ-Script, a programming language built for analysis and visualization, has many advanced features to make programming easier. With HiQ-Script, you have access to all HiQ built-in analysis functions, plus you have the power to create your own functions and programs using the HiQ-Script programming environment. With a script, HiQ Notebooks emerge as documentation tools and interactive analysis environments. You can build a Notebook documenting what you have done and bring the analysis to life for your audience.

You can access HiQ-Script in the Command Window or from a script object on a Notebook page. The Command Window returns immediate results. The script object encapsulates a series of HiQ-Script statements to extend the analysis and visualization capabilities of a Notebook.

## Climate Notebook: Analyzing Rainfall Data

In this data analysis example, your goal is to add analysis to an existing Notebook that contains climate data to determine specific statistics, such as the average yearly rainfall totals and the average temperatures over the past 71 years.

1. Open `climate.HiQ` if you completed the Notebook example in Chapter 6, *Visualizing Data with 2D and 3D Graphs*. Otherwise, open `climate7.HiQ` from the `Examples\Getting Results` folder.
2. To quickly compute the average rainfall and temperature using the Command Window, type the following two commands.

```
> averageRain = mean(climate[*,2])

averageRain = 33.042958

> averageTemp = mean(climate[*,3])

averageTemp = 68.302817
```

HiQ creates two new objects that contain the average of the second and third columns of the climate matrix. The brackets denote the desired indices of the matrix, the asterisk denotes all row elements, and the 2 and 3 denote the second and third columns of the matrix. You can continue to analyze the data from the Command Window. Although obtaining results from the Command Window satisfies your information needs, you can quickly and easily document the results on the Notebook page.

3. Check the Object List in the Explorer. As shown in Figure 7-1, the objects you created—`averageRain` and `averageTemp`—are now listed. To place any object from the Object List on the Notebook page, you can drag-and-drop the object from the Explorer to the Notebook page.

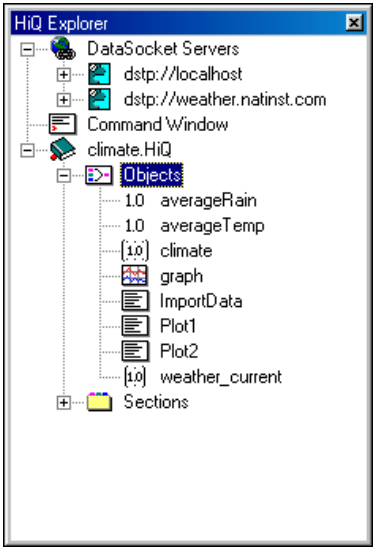



**Figure 7-1.** Climate Object List

4. Drag-and-drop the object `averageRain` on the Notebook page.

   You can resize this object and change any properties. For example, you might want to view exactly three decimal places of precision, including any trailing zeros.

5. Right click on the `averageRain` view, and select **Properties**. On the Numbers page, set the **Decimal Places** to 3 and uncheck **Remove Trailing Zeros**. Click **OK**.

## Setting Default View Properties

You can modify the properties of an object and make them the default properties so that any new views of an object of that same type have those properties.

1. Right click on the `averageRain` object view and choose **Defaults»Update Default Properties**.
2. Drag-and-drop the `averageTemp` object from the Explorer to the Notebook page near the `averageRain` object. Notice that this new view has the same number of decimal places as `averageRain`.

Figure 7-2 shows the Climate Notebook with the calculated average temperature and rainfall. You can print this Notebook to document the climate statistics over the last 71 years.

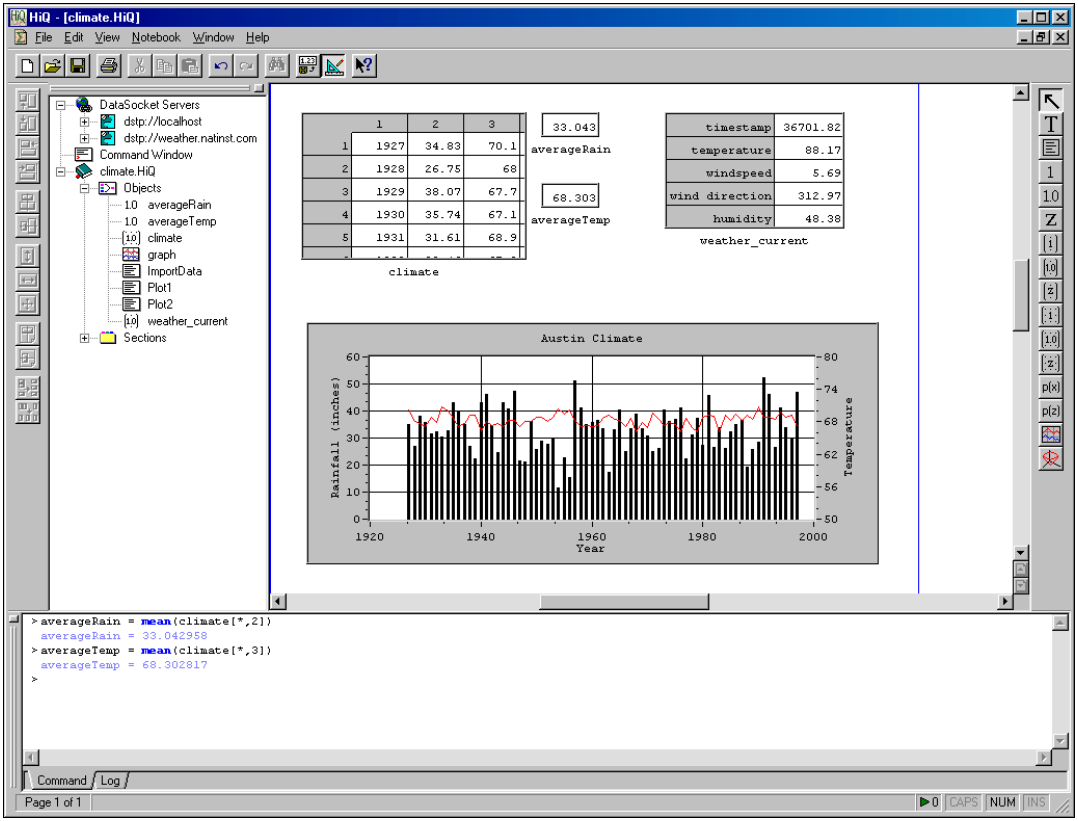



**Figure 7-2.** Climate Notebook After Analyzing Rainfall and Temperature Data

## Looking for Trends

Finally, it might interest you to determine if there was a trend in the yearly rainfall data over the past 71 years. You can detect a simple trend by performing a linear data fit of the rainfall data. With the following steps, you can perform a data fit and graph the results.

The built-in functions in HiQ-Script are intuitively named. For example, you used the built-in function `mean` to compute the mean (or average) of a data set. The built-in function for performing a data fit is named `fit`, and the function for adding a plot to a graph is named `addPlot`.

1. At the Command Window prompt, type `fit` without pressing the <Enter> key. Notice that `fit` turns blue, indicating that HiQ recognizes this name as a built-in function.
2. With the cursor placed in `fit`, press the <F1> key. HiQ displays the online help for the `fit` function.

   This function can perform many kinds of data fits, from simple linear regression to complex multi-dimensional nonlinear fits. Because you are interested in the actual fitted data points in this example, you perform a line fit, which is the first usage listed for `fit`.

   For information about the value of each parameter, scroll down to the *Parameters* sections. Notice that the fifth return value returns the actual fitted data points.
3. To compute the fitted data, type the following script at the Command Window and press <Enter>.

   ```
   [,,,,fitData]=fit(climate[*,1],climate[*,2],<line>);
   ```

   The four commas on the left indicate that you do not want the first four values returned from the function.

**Note** HiQ-Script is case sensitive. Make sure you type the script exactly as it appears.

4. Close the help window.
5. To add a plot of the fitted data to your graph, type the following script at the Command Window and press <Enter>.

   ```
   fitPlot=addPlot(graph,fitPlot,climate[*,1],fitData);
   ```

   The parameter `fitPlot` is a plot handle you can use to refer to the plot itself when you want to manipulate it later.
6. Save the Notebook as `climate.HiQ`.

Figure 7-3 shows the Climate Notebook, which now displays a plot representing the trend in rainfall over the last 71 years.

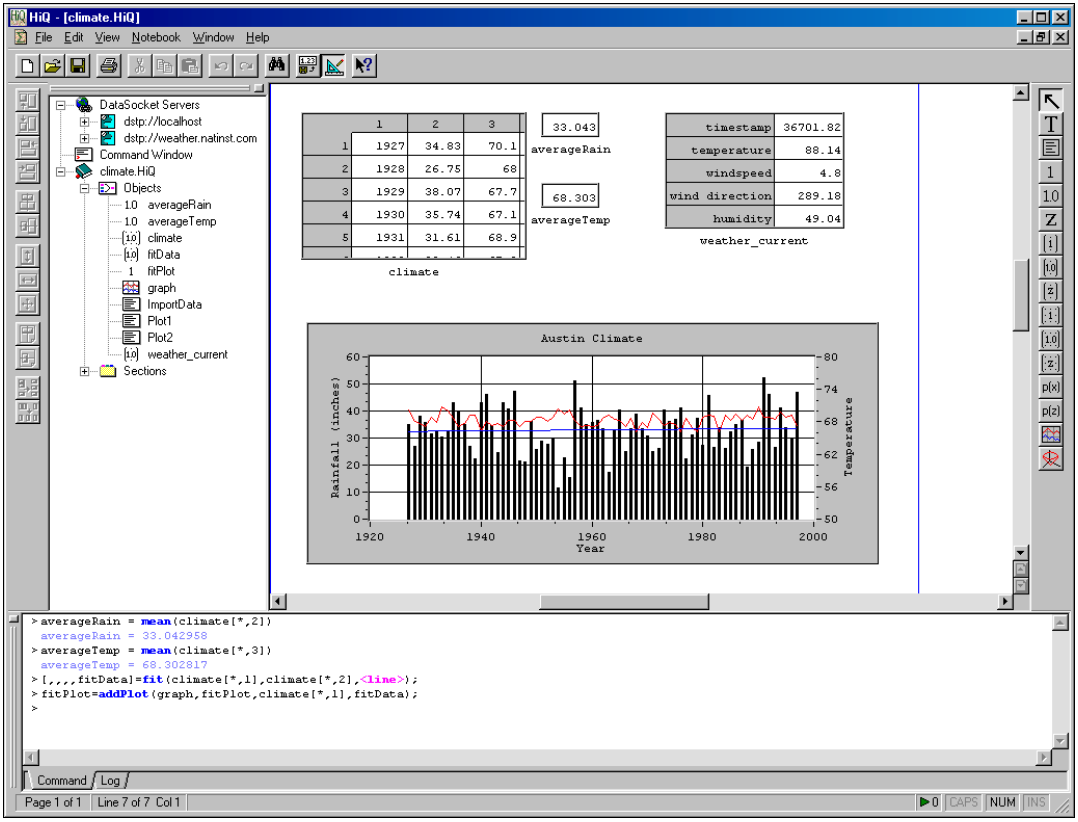



**Figure 7-3.** Climate Notebook Showing Rainfall Trends

## Maintaining the Notebook

This Notebook now communicates the rainfall history over the past several years, as well as the average rainfall and average temperature. You can use this Notebook as a report for documentation or as a visual aid for a presentation. If you repeat your analysis only once a year as new data becomes available, the steps you performed while creating this Notebook might suffice; however, if you need to repeat the analysis on a daily basis on many different sets of data, these steps might be inefficient. Because you can automate analysis in HiQ, you can create a Notebook that analyzes data when you push a button. You automate this analysis in Chapter 9, *Converting a Notebook to a Problem Solver*.

# Seismic Notebook: Computing the Energy in the Signal

In this example, your goal is to compute the energy present in the seismic data at a particular point in time. Because the seismic data represents the power of the measured signal versus time and frequency, the energy in the signal at a specific time is the integral of the frequency signal at that time.

1. Open `seismic.HiQ` if you completed the Notebook example in Chapter 6, *Visualizing Data with 2D and 3D Graphs*. Otherwise, open `seismic7.HiQ from the Examples\Getting Results` folder.
2. Generate a vector representing the actual frequencies in the seismic data. Each column in the Seismic matrix represents the time domain data at a specific frequency. The first frequency point (the first column in the Seismic matrix) is 4 hertz, and the frequencies (columns) are separated by 4 hertz increments.

   Type the following line of HiQ-Script in the Command Window to create a vector of frequencies starting at four, incrementing by four, and containing as many elements as the Seismic matrix has columns.

   ```
   f = 4 * seq(Seismic.columns);
   ```

**Note** HiQ-Script is case sensitive. Make sure you type the script exactly as it appears.

3. Press <Enter>.
4. To compute the energy in the signal at time 30, type the following script in the Command Window and press <Enter>.

   ```
   energy = 1/(2*<pi>) * integrate(f, Seismic[30,*])
   ```

The answer is echoed back in the Command Window, as shown in Figure 7-4.

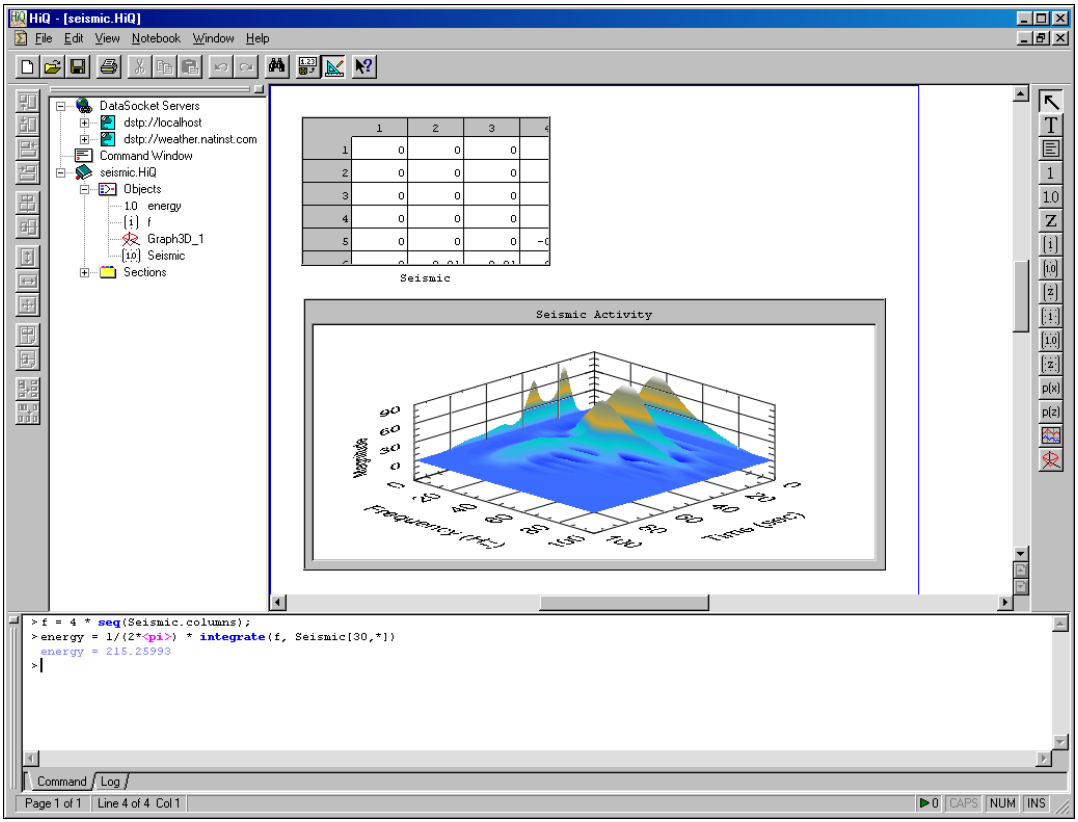



**Figure 7-4.** Seismic Notebook After Computing the Energy in a Signal

5. To find information about the `integrate` function, place the cursor in the function name and press the <F1> key.

6. Click on the first usage for `integrate`. The help for this topic appears, as shown in Figure 7-5.

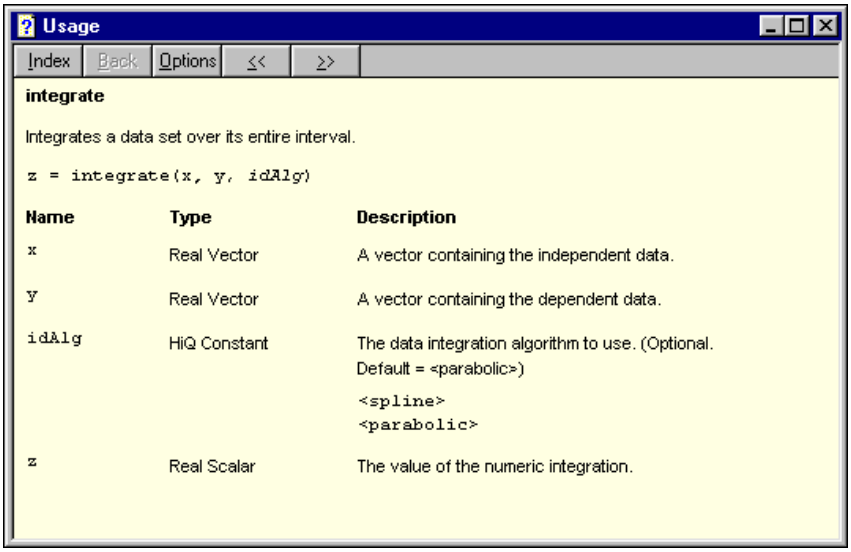



**Figure 7-5.** HiQ Online Help Describing the integrate Function

In this usage, `integrate` is integrating a data set specified by the second parameter with respect to the first parameter.

The data set is row 30 of the seismic data, which represents the frequency content of the seismic signal at 30 seconds. The `integrate` function allows you to optionally specify the algorithm to use for integrating the data. By default, the parabolic method is used.

7. Close the help window.

## Automating the Analysis

To repeat this analysis for any other time, you can recall the line of HiQ-Script and change the value 30 to the new time, or you can automate this analysis so that you can open the Seismic Notebook and compute the results by pressing a button.

1. Place a new script object on the Notebook page using the Script tool on the HiQ Tools toolbar.
2. To compute the energy at time 30, type the following script in the Script object as shown in Figure 7-6.

```
energy = 1/(2*<pi>) * integrate(f, Seismic[30,*]);
```

**Tip** You can copy and paste this line from the Command Window to the Script object.

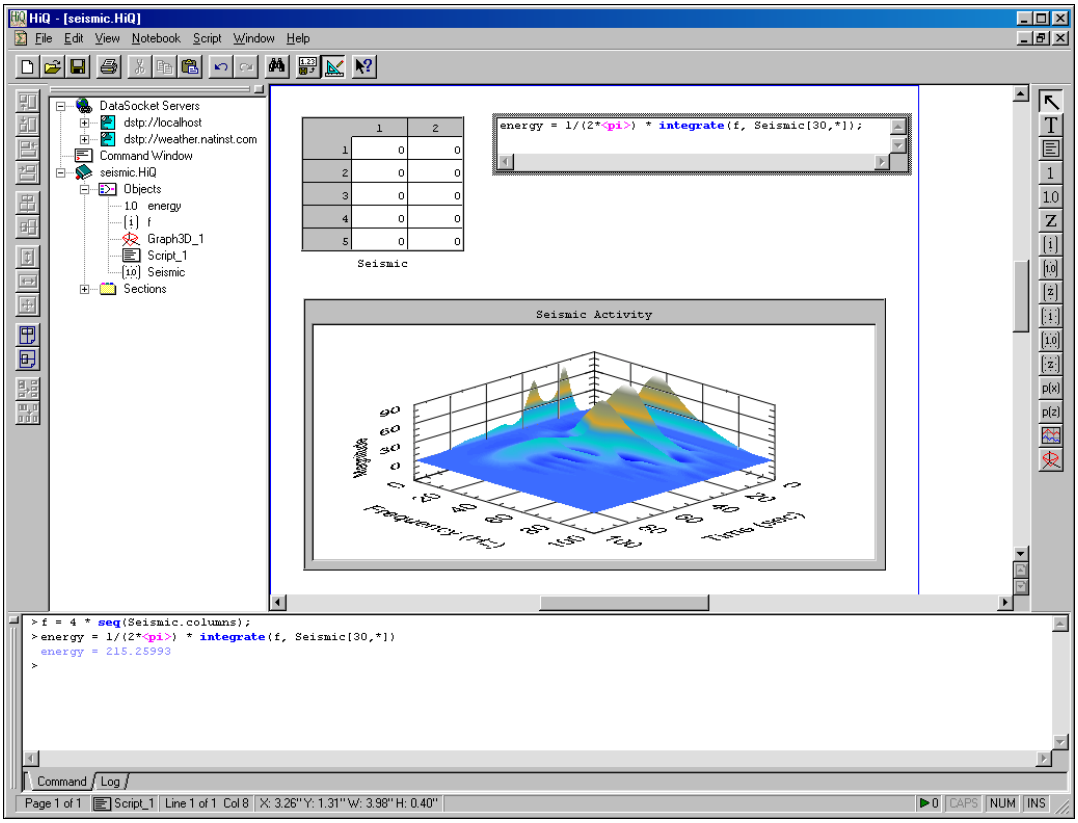



**Figure 7-6.** Adding Script to the Seismic Notebook to Automate Analysis

3. Right click on the script view and select **Properties**.
4. On the View page, select **Run View** and type `Compute Energy` for the **Caption**.
5. Click **OK**.
6. Drag-and-drop a view of the `energy` object on the Notebook page.
7. Run the script by clicking on the **Compute Energy** run button.
8. Save the Notebook as `seismic.HiQ`.

Figure 7-7 shows the Seismic Notebook, which now computes and displays the energy contained in the signal at the thirtieth second.

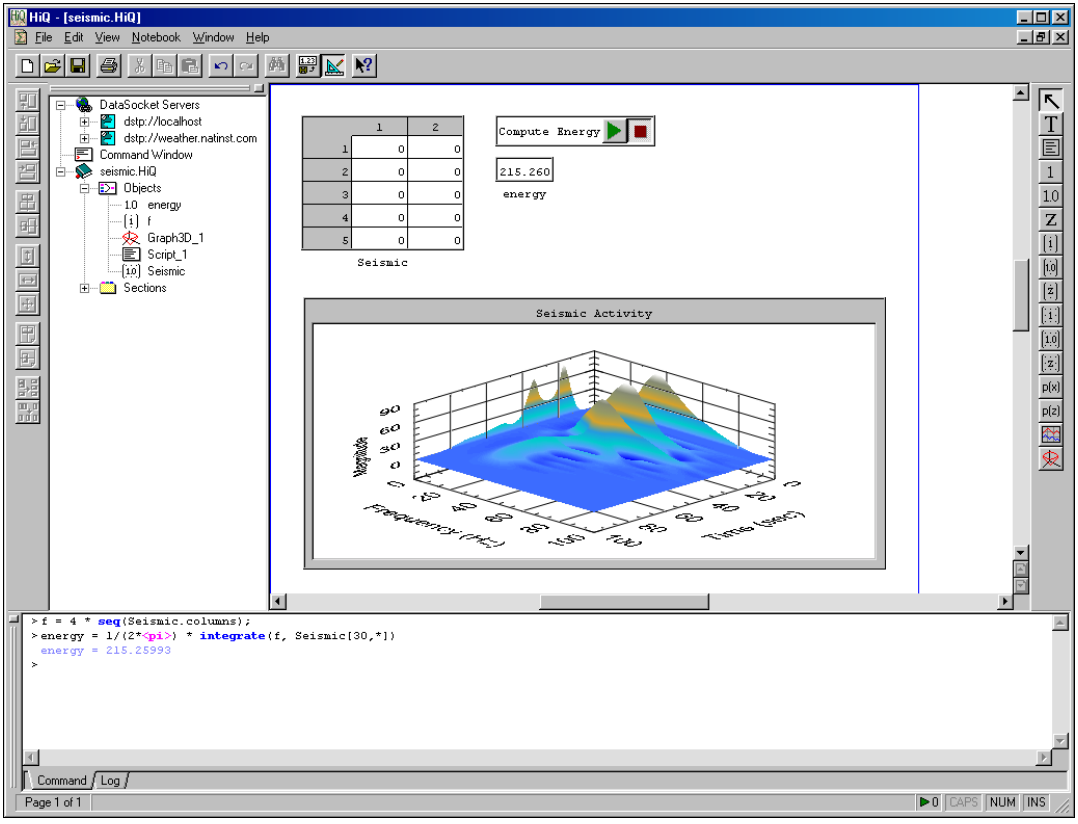



**Figure 7-7.** Seismic Notebook with Compute Energy Run Button

# 8

# Taking Advantage of ActiveX

This chapter explains how you can use different ActiveX technologies in HiQ. In this chapter, you enhance the Climate and Seismic Notebooks with ActiveX components. ActiveX technologies add flexibility and versatility to your project. HiQ offers the following ActiveX technologies.

- **ActiveX Document Container**—HiQ allows you to embed objects from other applications directly into your HiQ Notebook. For example, you can embed a Microsoft Word file, an Excel spreadsheet, or a PowerPoint presentation directly in your HiQ Notebook. These embedded objects are editable within the HiQ environment and can be stored within a HiQ Notebook or linked to an external file on disk.
- **ActiveX Document Server**—HiQ allows you to embed a HiQ Notebook into any other application that is an ActiveX document container. For example, you can embed a HiQ Notebook directly into a Microsoft Word document, an Excel spreadsheet, or a PowerPoint presentation. The embedded HiQ Notebook is editable within the other application and can be stored within the container document.
- **ActiveX Automation Client**—HiQ allows you to run and control other applications from the HiQ environment. For example, you can create a HiQ Notebook that automatically launches Microsoft Word or Excel and then shares HiQ data with the Word document or Excel spreadsheet to produce an automated report.
- **ActiveX Automation Server**—HiQ allows you to run and control a HiQ Notebook from within any other application that is an ActiveX automation client. For example, you can create a program in another application, such as Microsoft Visual Basic, LabVIEW, or LabWindows/CVI, that automatically launches HiQ, opens a HiQ Notebook, and sends data to HiQ for automated analysis, visualization, and report generation.
- **ActiveX Controls Container**—HiQ allows you to embed ActiveX controls directly into your HiQ Notebook. These embedded controls can be accessed both interactively and programmatically from within the HiQ environment. For example, you can embed a Microsoft Web Browser or National Instruments ComponentWorks control directly into your HiQ Notebook and then access that control to automatically gather data from the Internet or a physical measurement device.

# Climate Notebook: Adding ActiveX Objects

In this example, you add a button to the Notebook that indicates whether a script performs a data fit on the rainfall data. When the button is on, the script performs the data fit and plots the data fit. When the button is off, the script removes the data fit plot.

**Note** To complete this example, you must have ComponentWorks installed on your computer. HiQ Professional installs ComponentWorks, and HiQ installs the ComponentWorks demonstration version. You also can download the ComponentWorks demo version from the National Instruments Web site at `www.ni.com`

## Adding a ComponentWorks ActiveX Control

1. Open `climate.HiQ` if you completed the Notebook example in Chapter 7, *Analyzing Data with HiQ-Script*. Otherwise, open `climate8.HiQ` from the `Examples\Getting Results` folder.
2. Select **Edit»Insert Control**, then **CWButton Control** from the Insert Control dialog box, as shown in Figure 8-1, and click **OK**.

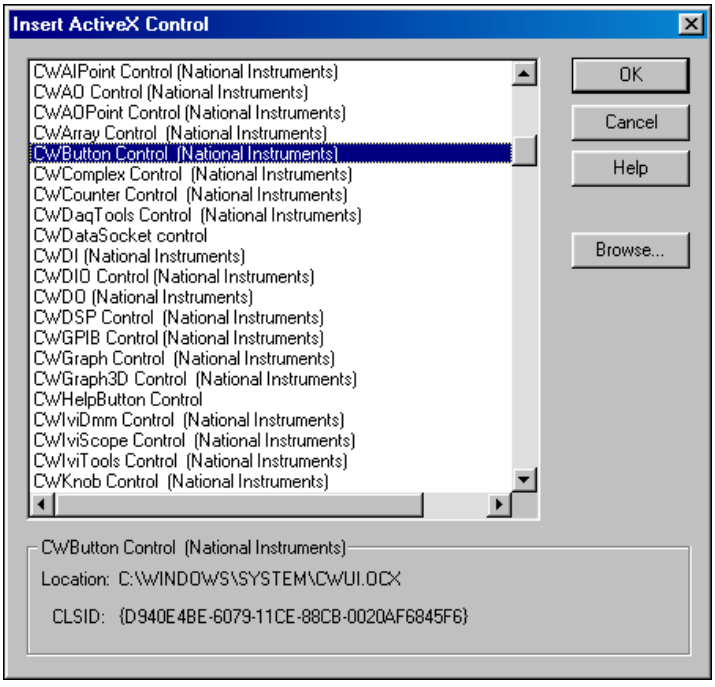



**Figure 8-1.** Insert Control Dialog Box

3. Resize the control as needed.
4. Right click on the new control and drag it if you need to reposition the control on the page.
5. Right click on the control and choose **Rename** to name the object `switch`.

**Note** When you place an ActiveX control in a Notebook, the Notebook enters ActiveX Design Mode, which allows you to set properties for the control. Some controls only operate when ActiveX Design Mode is off. To test or operate a control, turn off ActiveX Design Mode by unchecking **Notebook»ActiveX Design Mode**.

6. Right click on the button control and choose **CWButton Control (National Instruments) Object»Properties**.

   The property pages that you see vary depending on the control. HiQ does not create the property pages for ActiveX controls. The property pages of the ActiveX control should clearly document its properties and methods.
7. On the Style page, choose the On/Off Toggle button, as shown in Figure 8-2. Notice that the preview window displays how the button will look when you place it on the Notebook.

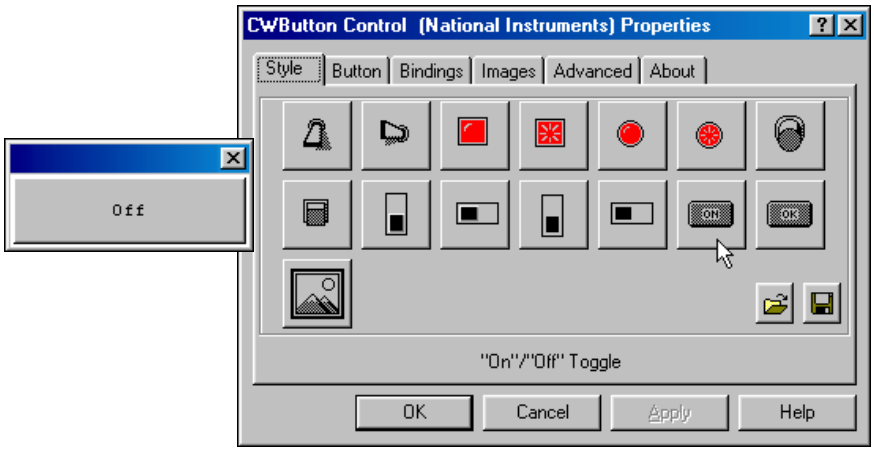



**Figure 8-2.** Selecting a Style for the CWButton Control

8. On the Button page, type `Perform Data Fit` for the **Caption** property.
9. Click **OK**.

## Programmatically Accessing the ActiveX Control

You can use the controls's properties and methods to programmatically access its capabilities from HiQ-Script. Use the HiQ ActiveX Object Browser to explore the methods and properties available in the control, as explained in following steps.

1. From the Command Window, type `browse switch` and press <Enter>. You also can right click on the switch object and select **Browse**.

   The ActiveX Object Browser is displayed with two lists: the left pane displays the list of controls in a library with the particular control highlighted, and the right pane lists the methods and properties for the highlighted control.

2. Scroll down the methods and properties list in the right pane and select the `Value` property, which you will use to determine the position of the switch. Notice the syntax help and the property description for `Value` at the bottom of the ActiveX Object Browser.

3. Place a Script object on the Notebook.

4. To perform the data fit on the rainfall data when the switch is on and remove the plot when the switch is off, add the following HiQ-Script:

Script_1 in climate.HiQ

```
//If switch is on, calculate the fit and
// plot it on the graph.

if switch.value == 1 then
    [,,,,fitData] = fit(climate[*,1],climate[*,2]);
    fitPlot = addPlot(graph,fitPlot,climate[*,1],fitData);
else
    //If switch is off, remove the fit plot.
    removePlot(graph, fitPlot);
end if;
```

5. Right click on the script and select **Properties**. On the View page, select **Run View** and type `Compute Fit` for the **Caption**, as shown in Figure 8-3.

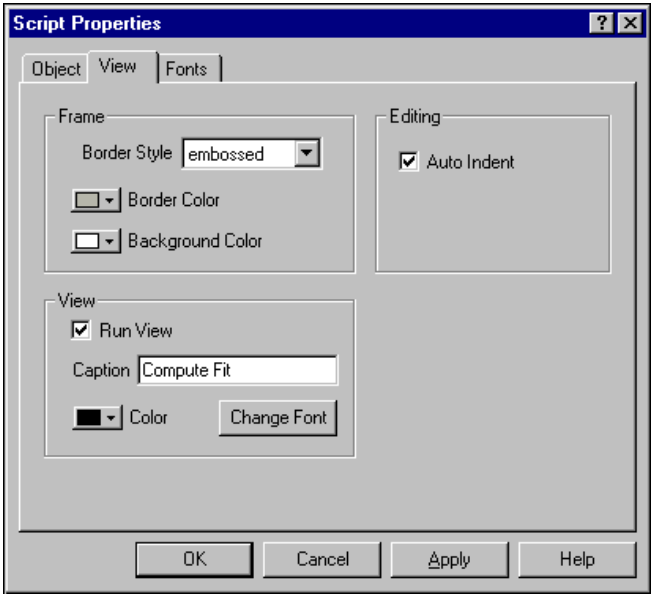



**Figure 8-3.** Setting View Properties for a Script Object

6. Click **OK**.
7. Run the script to plot and analyze the data. Notice the linear data fit on the graph. You can remove the plot by clicking on the switch object and running the script again.
8. Save the Notebook as `climate.HiQ`.

Figure 8-4 shows the Climate Notebook with the CWButton control.

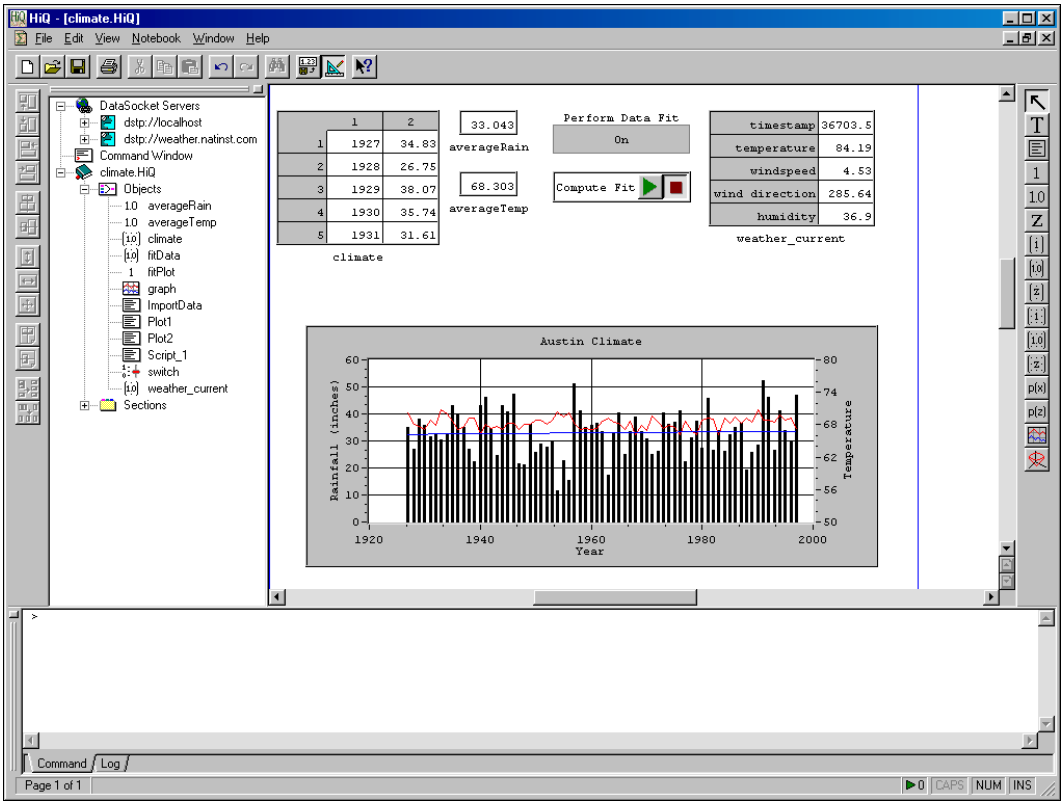



**Figure 8-4.** Climate Notebook with a ComponentWorks ActiveX Control

## Adding a Microsoft Word Document

You also can embed ActiveX documents such as a Word document, an Excel spreadsheet, or PowerPoint slides in your Notebook. For this Notebook, add a Word document to summarize conclusions about this Notebook.

**Note** To complete this example, you must have Microsoft Word installed on your computer.

1. To place an ActiveX object in a Notebook, use the **Edit»Insert Object** menu item.

   When you insert an ActiveX object, you can either link the object to an existing file or embed a new object directly in HiQ. In both cases, the object appears on the Notebook page. While an embedded object stores its data in a Notebook file, a linked object does not. When a linked object file changes, the object displayed in the HiQ Notebook automatically updates. However, if the linked object file outside of the Notebook is moved or deleted, the object within the Notebook is no longer visible in HiQ.

2. Select **Microsoft Word Document** and the **Create New** option, as shown in Figure 8-5.

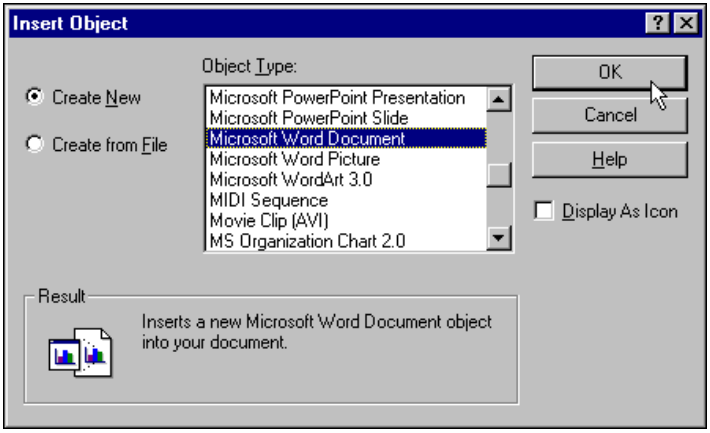



**Figure 8-5.** Insert Object Dialog Box Lists ActiveX Objects You Can Insert in HiQ

3. Click **OK** to insert a new Microsoft Word document.

4. Use the power of the embedded Word document to enter and format text. For this example, type the following text:

   The average yearly rainfall is ***increasing***, while the average temperature is remaining ***steady***.

5. When you finish entering and formatting your text, click on the Notebook page (outside of the Microsoft Word Document object).

6. Save the Notebook as `climate.HiQ`.

Figure 8-6 shows the Climate Notebook with the embedded Microsoft Word Document.

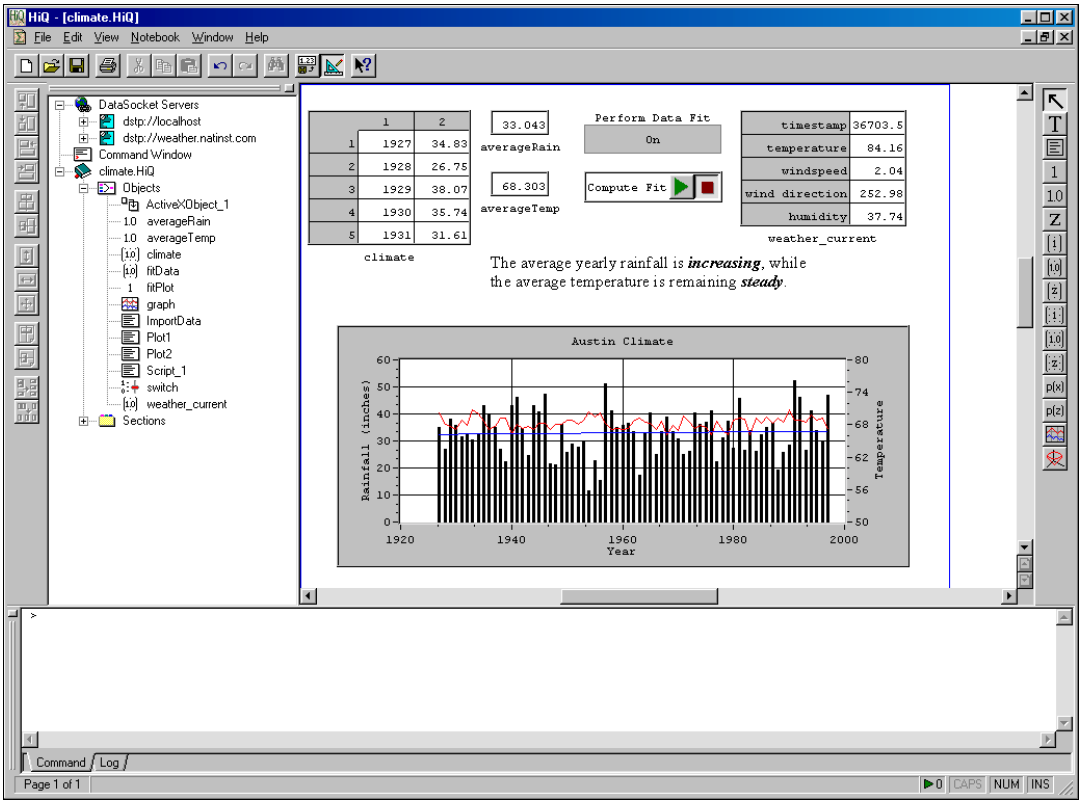



**Figure 8-6.** Climate Notebook with an Embedded Microsoft Word Document

# Seismic Notebook: Adding a ComponentWorks Control

In this example, you add a ComponentWorks Slide control that specifies which algorithm to use for computing the energy in the signal.

**Note** To complete this example, you must have ComponentWorks installed on your computer.

## Adding an ActiveX Control

1. Open `seismic.HiQ` if you completed the Notebook example in Chapter 7, *Analyzing Data with HiQ-Script*. Otherwise, open `seismic8.HiQ` from the `Examples\Getting Results` folder.
2. To place an ActiveX control in a Notebook, use the **Edit»Insert Control** command.
3. Select **CWSlide control**, and click **OK**.
4. Right click on the slide and choose **Rename**. Rename the object `method`.
5. Right click on the slide and drag it if you need to reposition the control on the page.

You can specify which numerical integration algorithm you want to use with the Slide control. HiQ uses two methods to compute the integral of a data set: `spline` and `parabolic`. Refer to the *integrate* topic in the online help for more information about these methods. Complete the following steps to modify the Slide control properties.

6. Right click on the slide and choose **CWSlide Control (National Instruments) Object»Properties**. The control generates these property pages so you can customize the control.
7. On the Style page, choose **Vertical Slide** and **Value pairs only** for the Style, as shown in Figure 8-7.

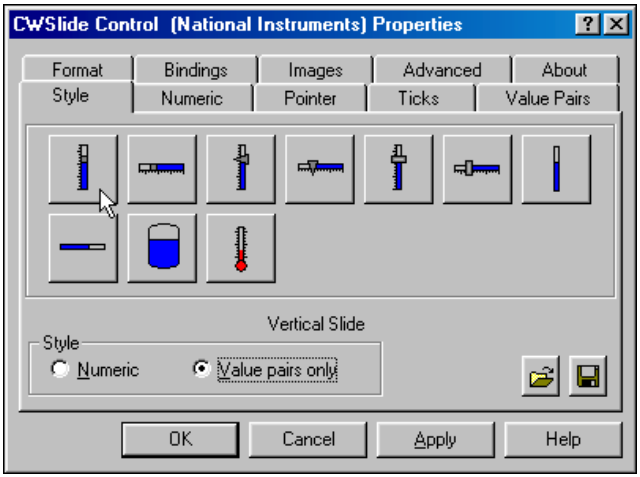



**Figure 8-7.** Setting Style Properties for the ComponentWorks Slide Control

8. On the Pointer page, choose **None** for **Fill Style**.
9. On the Value Pairs page, delete all but two value pairs in the list. Change one of the pairs' name to `spline` and value to `0`, and change the other's name to `parabolic` and value to `1`.
10. Click **OK**.

## Programmatically Accessing the ActiveX Control

To access the value of the slide, use the `Value` property. Because the `integrate` function expects a HiQ constant as the third parameter to specify the method, you need to write a short user function to return the appropriate HiQ constant based on the actual value of the slide.

1. Right click on the **Compute Energy** run button and choose **View Source**. The object containing the HiQ-Script is now visible in its own window.
2. To determine the HiQ constant for the integration method using the slide, add a user function to the end of the script object as follows.

```
function getMethod()
   //Project declares that the object is defined
   //outside this function.
   project method;
   //Choose the appropriate HiQ constant based on the
   //value of the method object.
   select method.value from
   case 0:
      return <spline>;
   case 1:
      return <parabolic>;
   end select;
end function;
```

To extend your analysis, you can include user functions in your HiQ-Scripts. For example, the user function `getMethod` returns the correct HiQ constant given the position of the slide. This function is not automatically executed when you run the script. It is executed only when called as a function from elsewhere in your script. In fact, you can write this function in a separate script object if you do not want to include it in the script that computes the energy. The next step references this function in the call to `integrate`.

3. To compute the energy with the specified integration method, modify the call to the built-in function `integrate` at the top of the HiQ-Script.

```
energy = 1/(2*<pi>) * integrate(f, Seismic[30,*],
  getMethod());
```

The third parameter to the function `integrate` is a call to the user function `getMethod`. In the previous step, you defined this user function to return the HiQ constant `<spline>` or `<parabolic>`, depending on the position of the slide. Figure 8-8 shows the completed script.

Script_1 in seismic.HiQ

```
energy = 1/(2*<pi>) * integrate(f, Seismic[30,*], getMethod());

function getMethod()
   //Project declares that the object is defined outside this function.
   project method;
   //Choose the appropriate HiQ constant based on the value
   //of the method slide.
   select method.value from
   case 0:
      return <spline>;
   case 1:
      return <parabolic>;
   end select;
end function;
```

**Figure 8-8.** HiQ-Script Accessing the User Function getMethod

4. Close the Script window.
5. Execute the script twice, using each integration algorithm to see how the algorithms affect the energy computation.

   To display more precision in the answer, change the **Decimal Places** property in the property pages for the energy object.
6. Save the Notebook as `seismic.HiQ`.

Figure 8-9 shows the Seismic Notebook with the ComponentWorks Slide control, which controls the integration method used.

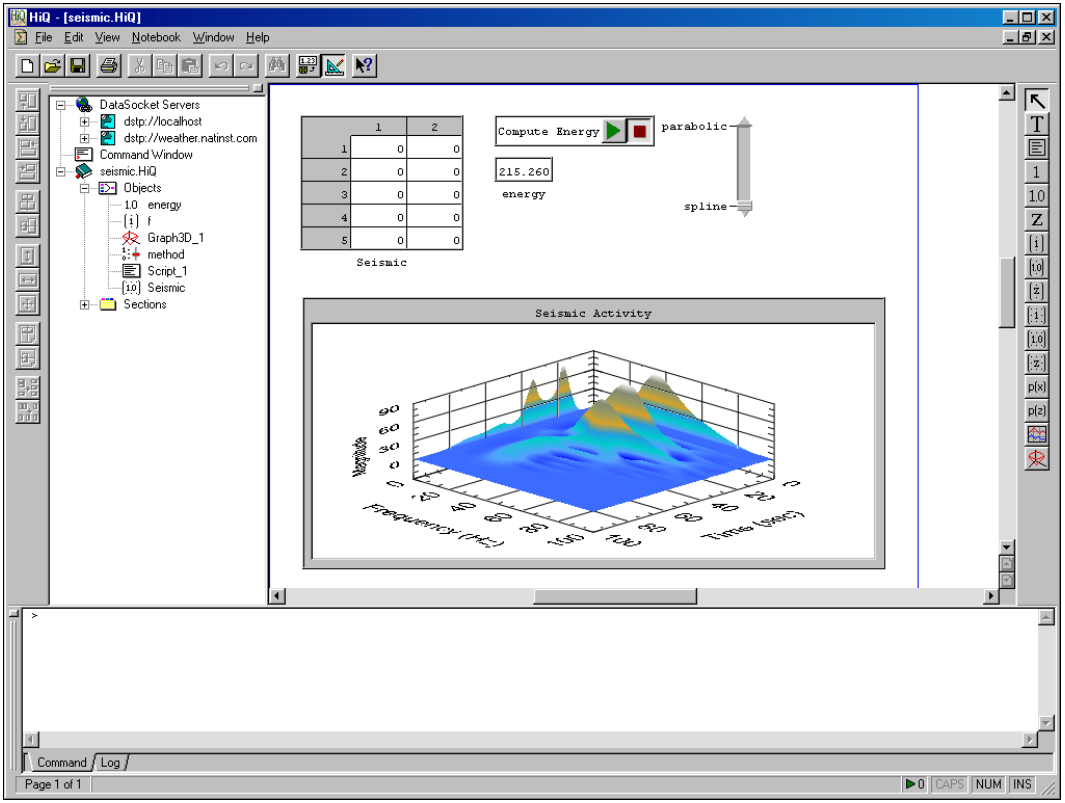



**Figure 8-9.** Seismic Notebook with the ComponentWorks Slide Control

# 9

# Converting a Notebook to a Problem Solver

This chapter introduces problem solving Notebooks, which are useful tools for analyzing different data or performing different analysis on a specific data set. In this chapter, you complete the Climate and Seismic Notebooks by converting them to Problem Solvers that can analyze a variety of different data sets.

A *Problem Solver* is a Notebook that is flexible enough to work with different data or perform different analysis on a specific data set when you press a button. For example, a custom HiQ Notebook might be able to compute the best exponential data fit to the growth of a specific stock price in a specific file, but you would need to modify the Notebook if you wanted a different type of fit or a different data file. A Problem Solver Notebook allows you to choose the data file and the type of data fit to perform without having to redesign the Notebook.

## Climate Notebook: Converting to a Problem Solver

In Chapter 8, *Taking Advantage of ActiveX*, you completed a Notebook that analyzed specific rainfall data. In the following Notebook example, you convert that Notebook to a Problem Solver that allows you to choose the data file, graph the data, and compute the mean and standard deviation when you press a button.

1. Open `climate.HiQ` if you completed the Notebook example in Chapter 8, *Taking Advantage of ActiveX*. Otherwise, open `climate9.HiQ` from the `Examples\Getting Results` folder.

   In previous chapters, you imported data, graphed the data, and finally analyzed the data. However, you can automate this process with HiQ-Script so you can quickly repeat your analysis with another data file. This Notebook already contains a script object that computes a data fit. You can modify this script to import a data file and compute averages as well.

2. To edit the script represented by the **Compute Fit** run button, right click on the button and choose **View Source**.

A floating window containing the script appears, as shown in Figure 9-1. You can position and size this window.

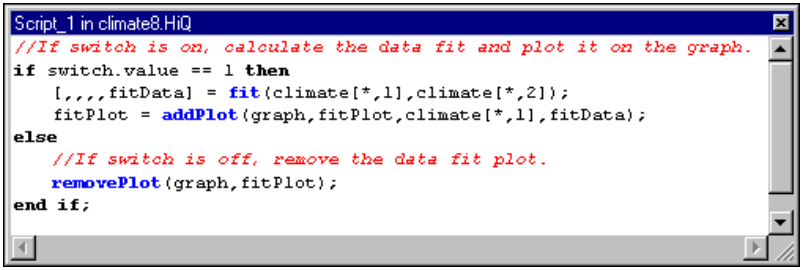



**Figure 9-1.** Climate Notebook Script Displayed in a Floating Window

3. Select **Notebook»Compile All Scripts** to compile all the HiQ-Scripts in this Notebook. This step compiles the previously generated scripts, `Plot1`, `Plot2`, and `ImportData`, and creates three new functions, `Plot1_Run`, `Plot2_Run`, and `ImportData_Run`.

4. Edit the script object to import and graph the data by typing the following lines at the beginning of the script:

```
//Import the climate data file.
ImportData_Run();

//Remove all plots from the graph.
removePlot(graph);

//Plot the rainfall data in the second column.
Plot1_Run();

//Plot the temperature data in the third column.
Plot2_Run();
```

5. Change the rainfall plot to a vertical bar by typing the following lines:

```
//Change the rainfall plot to a vertical bar.
graph.plot(-1).style = <verticalbar>;
```

6. Associate the temperature plot with the second y-axis and change the y-axis range:

```
//Associate the temperature plot with
//the second y axis.
graph.plot(-2).yaxis = 2;

//Set the range of the second y axis.
graph.axis.y(2).range.min = 50;
graph.axis.y(2).range.max = 80;
```

7. Compute the mean of the rainfall and temperatures by typing the following lines:

```
//Compute the average rainfall and temperature.
averageRain = mean(climate[*,2]);
averageTemp = mean(climate[*,3]);
```

Figure 9-2 shows the final script in the script window.

Script_1 in climate8.HiQ

```
//Import the climate data file.
ImportData_Run();

//Remove all plots from the graph.
removePlot(graph);

//Plot the rainfall data in the second column.
Plot1_Run();

//Plot the temperature data in the third column.
Plot2_Run();

//Change the rainfall plot to a vertical bar.
graph.plot(-1).style = <verticalBar>;

//Associate the temperature plot with the second y axis.
graph.plot(-2).yaxis = 2;

//Set the range of the second y axis.
graph.axis.y(2).range.min = 50;
graph.axis.y(2).range.max = 80;

//Compute the average rainfall and temperature.
averageRain = mean(climate[*,2]);
averageTemp = mean(climate[*,3]);


//If switch is on, calculate the data fit and plot it on the graph.
if switch.value == 1 then
    [,,,,fitData] = fit(climate[*,1],climate[*,2]);
    fitPlot = addPlot(graph,fitPlot,climate[*,1],fitData);
else
    //If switch is off, remove the data fit plot.
    removePlot(graph,fitPlot);
end if;
```

**Figure 9-2.** Climate Notebook Final Script

8. Close the script window.

9. To test the Notebook, run the script by clicking on the **Compute Fit** run button and selecting the `climate.dat` data file.

10. Save the Notebook as `climate.HiQ`.

Figure 9-3 shows the Climate Notebook, which now imports a data file, graphs the data, computes the averages, and optionally performs a data fit when you press the **Compute Fit** run button.

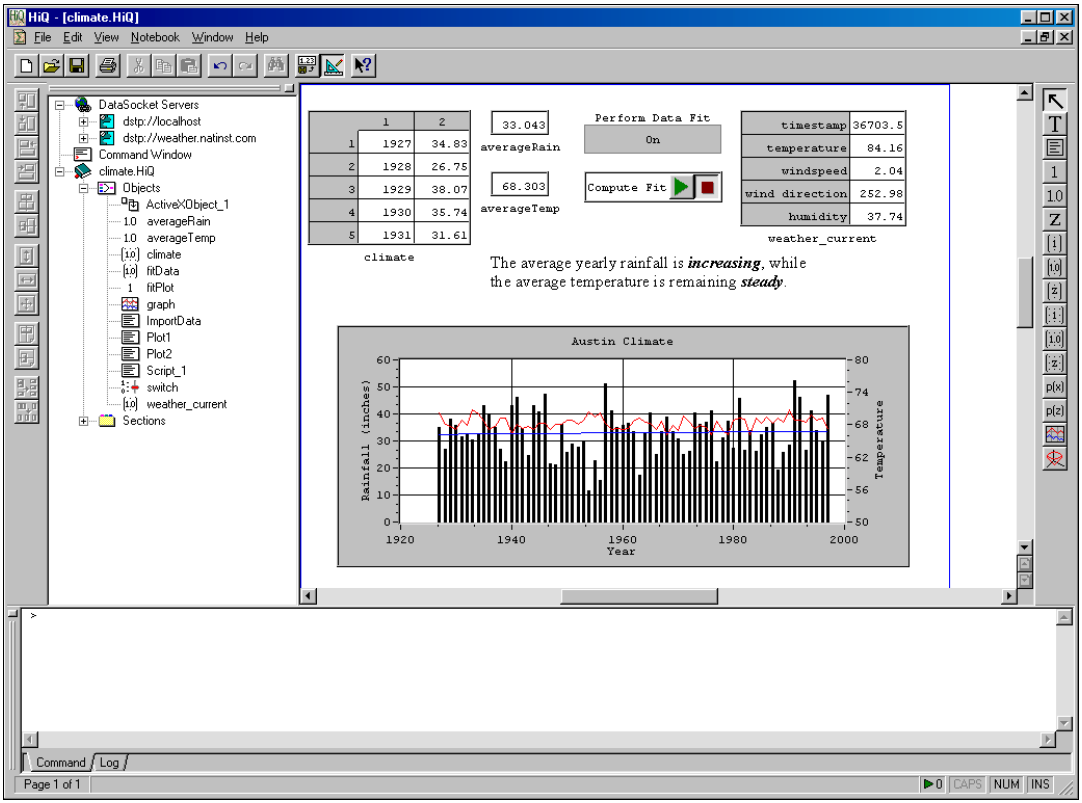



**Figure 9-3.** Climate Notebook as a Problem Solver

# Seismic Notebook: Converting to a Problem Solver

In Chapter 8, *Taking Advantage of ActiveX*, you completed a Notebook that computed the energy contained in a joint time-frequency representation of a seismic data set at a particular time. In the following example, you convert that Notebook to a problem solver that allows you to choose the time value to compute the energy when you press the **Compute Energy** run button.

1. Open `seismic.HiQ` if you completed the Notebook example in Chapter 8, *Taking Advantage of ActiveX*. Otherwise, open `seismic9.HiQ` from the `Examples\Getting Results` folder.
2. Place a new integer scalar object on the Notebook page and rename it `t`.
3. Right click on the **Compute Energy** run button and select **View Source**.
4. To compute the energy at the time specified by the integer scalar object, modify the first line of the script by replacing `30` with `t`.

   ```
   energy = 1/(2*<pi>) * integrate(f, Seismic[t,*],
   getMethod());
   ```

5. Close the script window.
6. To recompute the energy at the specified time, enter a time in the `t` object. For this example, enter `50` and press the **Compute Energy** run button.

   The energy object updates its value according to the time you specify in `t`.
7. Save the Notebook as `seismic.HiQ`.

Figure 9-4 shows the Seismic Notebook, which serves as a Problem Solver that computes the energy at any time `t` using either of the integration methods.

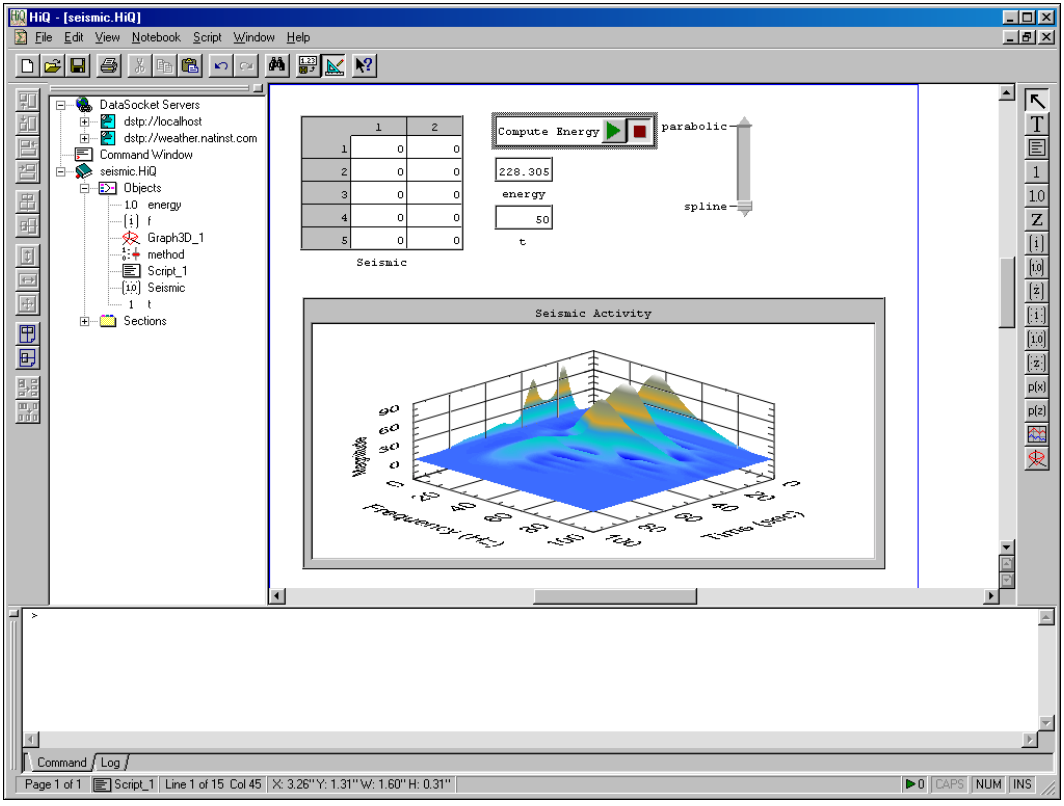



**Figure 9-4.** Seismic Notebook as a Problem Solver

# 10

# Sharing Notebooks with Others

This chapter describes how you can distribute HiQ Notebooks to effectively communicate your data analysis and visualization with others.

If you want to share the information in your HiQ Notebooks, you can freely distribute your Notebooks and the HiQ Reader. The HiQ Reader enables others to view your Notebooks and run scripts but protects your Notebook from changes. You also can create and distribute help files for individual Notebooks.

**Note** You must use the HiQ Professional version to create Notebooks that others can view in the HiQ Reader.

## About the HiQ Reader

The HiQ Reader enables users to interact with a HiQ Professional Notebook, run scripts, and print and save the Notebook. The HiQ Reader does not allow users to create new objects, delete objects, or modify the format of the Notebook. Accordingly, the HiQ Reader does not have a Command Window and several options have been disabled or removed from the menus. You also will notice a limited Notebook Explorer window.

You can distribute the HiQ Reader for free from either your HiQ Professional CD or from the National Instruments Web site at `www.ni.com`

If you are creating a Notebook for distribution, you can preview the file in the HiQ Reader to make sure it looks and functions as you expect it to.

### Climate Notebook: Previewing the Notebook in the HiQ Reader

In the following example, you test the Climate Notebook in the HiQ Reader and save it as a HiQ Reader Notebook.

1. Open `climate.hiq` if you completed the Notebook example in Chapter 9, *Converting a Notebook to a Problem Solver*. Otherwise, open `climate10.hiq` from the `Examples\Getting Results` folder.

2. Choose **File»Open in HiQ Reader**. The HiQ Reader launches and the Notebook is opened.

**Note** You must save Notebooks with HiQ Professional before the **Open in HiQ Reader** option is available.

3. Verify that the Notebook performs as expected. Try running the script and viewing data.
4. Save the Notebook as `climate.hqr`. The `.hqr` extension indicates that the Notebook is a HiQ Reader Notebook. You can still open and modify the Notebook in HiQ Professional; however, the new extension will launch HiQ Reader when you double click the Notebook.

# Distributing ActiveX Controls in Your Notebooks

If you take advantage of ActiveX controls in distributable Notebooks, remember that users need those ActiveX controls installed on their computers to use the Notebooks as you have designed them. If the ActiveX controls are not installed on their computers, the Notebook will open but will not be behave as intended.

For example, the Climate Notebook uses a ComponentWorks ActiveX Button control. For users to view the Climate Notebook correctly, they need the ComponentWorks ActiveX user interface controls installed on their computers. You can distribute the demonstration version of the ComponentWorks ActiveX controls with the Notebook if users do not have ComponentWorks.

**Note** The HiQ Reader automatically installs a demonstration version of ComponentWorks. The demonstration version of the ComponentWorks user interface controls is freely distributable and includes a run-time license. You also can freely download the demonstration version from the National Instruments Web site at `www.ni.com`

Refer to the license agreements of all ActiveX controls that you intend to use and/or distribute with your Notebooks to verify that you have the correct run-time and design-time licenses and whether you have permission to distribute the controls to others.

# Automatically Executing Functions

To make your Notebooks easier to use and prevent users from having to perform the same steps every time they open, save, or close your Notebook, you can include user functions that automatically execute as users open, save, or close your Notebook. For example, if you create a Notebook that uses a combobox control to hold a collection of options, you need to write a function to initialize the combobox with the appropriate options every time the Notebook is opened (combobox values are not saved when the Notebook is saved or closed).

To automatically initialize the controls, you can write a user function called `onOpen`. The following example initializes a Microsoft Forms 2.0 combobox control every time the Notebook is loaded.

```
function onOpen()
   assume project;
   myComboBox.additem("Select One…");
   myComboBox.additem("Low Pass");
   myComboBox.additem("Band Pass");
   myComboBox.additem("High Pass");
end function;
```

HiQ automatically executes three user functions: `onOpen`, `onSave`, and `onClose`. When these functions exist in your Notebook, they are executed when the Notebook is opened, saved, or closed, respectively.

# Distributing Help for a Notebook

You can create and distribute help for individual Notebooks to provide general information about the Notebook, such as its background and purpose. The help can contain any information that you do not want to include in the Notebook itself.

If there is a file in the same directory as the active Notebook with the same name but with an extension of `.htm`, `.chm`, or `.hlp`, you can access this help from the **Help** menu by selecting **Help for Current Notebook**. HiQ supports several types of help formats:

- HTML Help (`.htm` or `.html`)—Use any editor or Web page creation software to build the HTML file. When you distribute help as a Web page, remember to include all graphics files and secondary pages that you link to from the main help page.

- Microsoft Compiled HTML Help (`.chm` and `.chh`, if you include a Contents file)—Use a tool such as Microsoft HTML Help Workshop to create a collection of Web pages and compile them into a single distributable file.
- Windows Help (`.hlp` and `.cnt`, if you include a Contents file)—Use a WinHelp development tool such as Microsoft Help Workshop to compile several RTF files into a single distributable file.

**Note** When you distribute Notebooks that have help, be sure to include all necessary files, including graphics files. Users can access your help file through the HiQ Reader **Help** menu (**Help»Help For Current Notebook**).

## Climate Notebook: Maintaining the HTML Help File

In Chapter 9, *Converting a Notebook to a Problem Solver*, you completed a Notebook that analyzed specific rainfall data. As a Problem Solver, this Notebook can be reused to analyze new data files. In the following example, you modify an HTML file that describes how to use this Notebook.

1. Open `climate.hiq` if you completed the Notebook example in Chapter 9, *Converting a Notebook to a Problem Solver*. Otherwise, open `climate10.hiq` from the `Examples\Getting Results` folder and save it as `climate.hiq`.
2. Choose **Help»Help for Current Notebook**. HiQ opens the `climate.htm` file located in the same directory as the `climate.hiq` Notebook. This Web page briefly describes the purpose of the Notebook and the format of the input data file.
3. Click **data file** to view the actual input data file. This shows an example of the correct data file format used by the Notebook.
4. Open `climate.htm` in a text editor or Web authoring tool.
5. To add a new paragraph of text with a link to the HiQ help file, type the following text just before the </body> tag:

```
<p>Refer to the <a href="file:///C.../Program
Files/National Instruments/HiQ Pro 4.1/
Program/HiQ.hlp">HiQ help</a> to learn more about HiQ.
```

**Note** You need to specify the correct path to `HiQ.hlp` file in the above link.

6. Save and close the file.
7. Choose **Help»Help for Current Notebook**. The text and a new link are now visible at the bottom of the HTML help page.

8. Click on **HiQ help** to access the HiQ online help.

You can add any information to this Web page that you feel is important to users of this Notebook.

## Seismic Notebook: Viewing Compiled HTML Help

In Chapter 9, *Converting a Notebook to a Problem Solver*, you completed a Notebook that computed the energy contained in a joint time-frequency representation of a seismic data set at a particular time. As a Problem Solver, this Notebook can be reused to analyze new data files. In the following example, you inspect a compiled HTML help file that describes how to use this Notebook.

1. Open `seismic.hiq` if you completed the Notebook example in Chapter 9, *Converting a Notebook to a Problem Solver*. Otherwise, open `seismic10.hiq` from the `Examples\Getting Results` folder and save it as `seismic.hiq`.
2. Choose **Help»Help for Current Notebook**. HiQ opens the `seismic.chm` file located in the same folder.

   The help system for this Notebook was created using Microsoft HTML Help Workshop. Using the workshop, you can create a series of HTML help files and compile them into a single file for distribution with your Notebook.

# 11

# Getting Results as a MATLAB User

This chapter summarizes the MATLAB to HiQ transitional tools—such as the HiQ Command Window, the automated M-file to HiQ-Script translator, and HiQ-Script for MATLAB users—and describes how MATLAB users can use these HiQ features to leverage work they have already done in MATLAB.

## About HiQ

HiQ is designed specifically for scientists, engineers, and technical professionals. It offers flexible graphics for data visualization and includes a mature analysis library and scripting language that handles the rigorous requirements of current problem solving. The intuitive, natural user interface encourages the use of all the powerful capabilities in HiQ without the difficulties of traditional programming packages. In fact, this ease-of-use approach is the defining factor that separates HiQ from other analysis and data visualization packages.

Traditionally, when you changed from one analysis package to another, you encountered incredible obstacles, such as learning the features in the new package, transferring all work previously done in the prior package, fixing problems created by the transfer, and implementing functionality available in the old version that is not available in the new version. With HiQ 4.0, you will find transitioning to the HiQ environment easier than ever.

HiQ offers a transitional approach that allows you to continue working while learning the HiQ environment. Features that support this approach include the following:

- HiQ Command Window—The HiQ Command Window provides two interface modes: a HiQ mode that exploits all HiQ features and a 100% MATLAB 5 compatibility mode using your installed version of MATLAB 5 or greater.

- Automated M-file to HiQ-Script Translator—The translator and translation library are integral in supporting your progress in moving from MATLAB to HiQ.
- HiQ-Script—The HiQ-Script language is a programming language specifically designed for analysis. HiQ-Script is comparable to MATLAB M-file programming but contains additional programming constructs and built-in data types.
- HiQ Online Help for MATLAB Users—The online help is always available by pressing the <F1> key.

The goal of this chapter is to present ways in which HiQ and MATLAB can be used together. This chapter offers MATLAB-to-HiQ transition avenues to help you migrate at your own pace while still making optimal use of your previous work and valuable time.

# HiQ Command Window

In MATLAB, you use the command line interface to interact with your analysis results. Because you can use the HiQ Command Window exactly as you do in MATLAB, you can get acquainted with HiQ while you continue to use the data and M-files you have already created in MATLAB.

## Accessing the HiQ Command Window

After starting HiQ, access the command line feature by opening the Command Window. Verify that the **View»Command Window** option is checked. If not, select it.

If you are running HiQ for the first time, the Command Window appears in the lower section of the HiQ window. If you have used HiQ or moved the Command Window before, it appears in the last position it was placed.

At the top of the Command Window, you should see a > prompt, which is the prompt used by the HiQ command line while in HiQ mode. In the next section, you switch to MATLAB mode.

## Accessing and Using MATLAB 5 Compatibility Mode

**Note** To complete this example, you must have MATLAB 5 or greater installed on your computer.

At the HiQ Command Window prompt, type the following command:

```
> matlab
```

After a few seconds, the MATLAB prompt, », appears.

**Note** Because the installation of MATLAB version 5.0 does not automatically register the application with the operating system, you might see the following error instead of the MATLAB prompt:

```
Error:Unable to launch MATLAB. Verify that MATLAB is
properly installed.
```

If you receive this error, start MATLAB from the operating system with the following option:

```
MATLAB.exe /regserver
```

This option registers MATLAB with the operating system so other applications, including HiQ, can communicate with it.

Exit MATLAB and return to HiQ. Execute the `matlab` command again in the Command Window to get the MATLAB prompt.

Once you enter MATLAB mode, you can work as if you were in MATLAB:

- Execute your existing M-files by typing in their name and associated parameters.
- View the data objects by typing in their name and pressing return.
- Create and manipulate any data objects with the MATLAB programming language.
- Run any command defined in the MATLAB command line interface.

For example, try the following commands:

```
» x = eye(2)
» x
» size(x)
» x(2,2) = 5;
» cd
» help inv
» y=inv(x);
» sqrt(y)
```

Notice the MATLAB browser in the HiQ Explorer. The MATLAB browser lists all variables created in the MATLAB session. To access any MATLAB variable, select it from the browser and drag-and-drop it on the Notebook page to create a HiQ object, as shown in Figure 11-1. You also can access the MATLAB variables from the Command Window or a Script object.

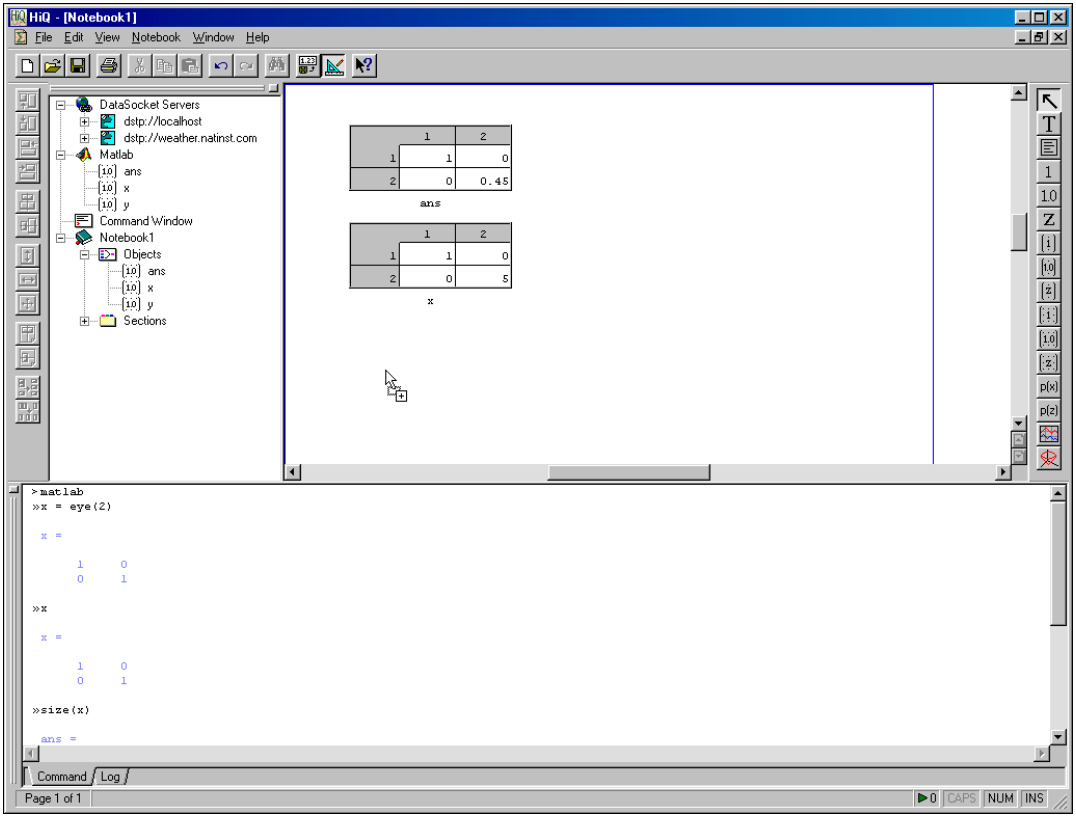



**Figure 11-1.** Browsing and Accessing MATLAB Variables in HiQ

In the command window, notice that information is displayed exactly as it is in MATLAB. The `inv` and `sqrt` functions used in this example are also HiQ built-in functions. In HiQ mode, the names of these functions are highlighted in blue to clearly indicate that you are calling a HiQ built-in function.

Try experimenting with various MATLAB commands. When the HiQ Command Window is in MATLAB compatibility mode, you have access to all MATLAB commands and programs from within HiQ.

## Sharing Data and Functionality

You have already seen an example of HiQ using MATLAB functionality with the `inv` and `sqrt` functions. However, this capability is not limited to functions shipping with MATLAB. Any M-file that you or your colleagues create is accessible using the same approach. In addition to running programs, you also can share data between HiQ and MATLAB.

Table 11-1 lists the additional commands available in MATLAB mode that you can use to share any numeric or textual data created by either application. Remember that you also can drag-and-drop data between the MATLAB browser and the Notebook browser or page.

**Table 11-1.** MATLAB Mode Commands

| Command | Explanation |
|---|---|
| `get` *x y* | Copies the variable named *x* from the current MATLAB workspace to an object named y in the active HiQ Notebook. If you omit *y*, HiQ names the new HiQ object *x*. |
| `getAll` | Copies all objects from the current MATLAB workspace to the active HiQ Notebook. |
| `put` *x y* | Copies the HiQ object named *x* from the active HiQ Notebook to the current MATLAB workspace and names it *y*. If you omit *y*, MATLAB names the variable *x*. |
| `putAll` | Copies all objects in the active HiQ Notebook to the current MATLAB workspace. |

At the MATLAB prompt, type the following commands.

```
» r = rand(50);
» h = surf(r)
```

**Note** When using MATLAB handle graphics, the `get(h)` function still operates as expected.

A new window has been generated by MATLAB and named `Figure 1`. It contains a static representation of the random surface. Now, get the data from MATLAB to HiQ and graph it in HiQ with the following commands.

```
» get r
» hiq
```

The `hiq` command switches the interface mode from MATLAB to HiQ. Notice that the prompt changes (from » to >) when you switched modes. The remaining commands in this example are native HiQ commands.

```
> graph = createGraph(r)
> view graph
```

The result is a HiQ graph object that contains a dynamic representation of the random surface. For comparison, change the color map for the plot to color spectrum using the Graph property page. You can access the property page by right clicking on the graph and selecting **Properties**. To rotate the HiQ graph, click on the graph and drag.

Look at the data used to generate the graphs. In MATLAB mode, you type in the name of the variable, `r`, and press <Enter>. If you want to see only a portion of `r`, you index the variable for the range of interest. Unfortunately, this creates a new temporary object (in `ans`) and takes up MATLAB workspace. In HiQ mode, however, you have two options:

- Use the command line view, which provides results similar to MATLAB.

  ```
  > r
  ```

- Create a view of the object, which creates a window `r` that looks similar to an Excel spreadsheet. You can scroll through the elements of the matrix `r` using the scroll bars at the right and bottom. To edit individual values, highlight the cell and input a new value. For example, use this matrix view to change the corner elements of the matrix to 5.

  ```
  > view r
  ```

After editing the matrix, send the changes back to MATLAB and regraph the matrix to view the edits.

```
> matlab
» put r
» surf(r)
```

Notice that the four corners of the surface are spiked because the larger data values were added to the matrix.

## Accessing and Using HiQ Mode

To change the HiQ Command Window from MATLAB mode to HiQ mode, type the following at the command prompt:

```
» hiq
```

Several commands, such as `cd`, `pwd`, `ls`, and `dir`, are identical in the HiQ Command Window and MATLAB. MATLAB makes no distinction in M-files between command line commands and built-in functions. In HiQ, commands are not built-in functions. HiQ commands are valid only in the HiQ Command Window.

Use HiQ commands as you would operating system commands. For example, `delete` *`objectName`* frees the resources used by the object with the given name.

By treating commands differently from built-in functions, you can create user-defined functions in your scripts using the same names as the commands without limiting the use of either one in the Command Window. If the command name is the same as an object name, command overwriting occurs as it does in MATLAB. To continue using the command in this situation, precede the command with the pound sign (#). The first two lines treat `cd` as a scalar, while the last executes the `cd` command.

```
> cd = 5;
> y = 3 * cd;
> #cd
```

Many commands in the Command Window control other features of HiQ, such as the HiQ Notebook. The HiQ Notebook, the primary interface element in HiQ, provides a framework for analyzing, visualizing, and documenting solutions. You can think of the HiQ Notebook as an extension of the project notebook kept by an engineer or scientist with sections, pages, drawings, and algorithms.

In HiQ, you can have multiple notebooks open at one time solving multiple problems. Each notebook has its own collection of objects and results. In this type of environment, the Command Window must attach to a particular notebook to allow the command line interface to alter the objects in that notebook. You must detach the Command Window to perform work independent of a notebook.

**Tip** You can learn more about detaching and attaching the Command Window in the online help (**Help»HiQ Help Topics**).

## Frequently Asked Questions

The following questions address issues you might encounter while in HiQ mode. For a complete description of the HiQ Command Window and its commands, see Chapter 3, *Getting Results with the HiQ Command Window*.

### How can I stop variables from displaying their contents when I execute a statement?

HiQ offers two display modes: terse and verbose. The Command Window is in verbose mode by default. If you prefer the MATLAB approach, you can set the **Trailing semicolon implies terse** mode option for the Command Window. Right click in the Command Window, select **Properties**, and set this property.

### Where can I get help on a built-in function?

As in MATLAB mode, type `help` followed by the function name, or place the cursor in the text of the function name and press the <F1> key.

### How do I clear a variable from my workspace?

To remove an object created by the Command Window, you can delete it from the Object List in the Explorer or use the `delete` command in the Command Window.

### How can I display the variables that were affected by the last command line execution?

The `whatChanged` command provides a list of all HiQ objects changed by the execution of the last line in the Command Window.

### How can I prevent the Command Window from creating variables in my current workspace?

If you use the `detach` command, all HiQ objects created in the Command Window from that point forward are not associated with the active notebook.

### How do I save variables created in the Command Window?

If the HiQ objects were created while the Command Window was attached to an active notebook, save the notebook, and all objects are saved with it.

### How do I change the precision at which variables are displayed?

Right click on the Command Window, select **Properties**, and view the Numbers page. In the entry for **Decimal Places**, verify that the **Auto** option is not checked and set the precision required. If you prefer trailing zeros, uncheck the **Remove Trailing Zeros** option.

### Where can I find a list of available commands for the Command Window?

Search for Command Window in the online help (**HiQ»HiQ Help Topics**).

### Where can I find a list of available built-in functions for use in the Command Window?

Type `help` to evoke the HiQ online help. Select *Built-In Functions* on the Contents tab to display a list of all available HiQ functions listed by name or category.

### How do I view a variable in its own window?

Use the `view` command followed by the name of the HiQ object.

### Where can I find a list of variables accessible in my workspace?

Look at the Object List in the Explorer, or type the `objects` command to get a list of available HiQ objects.

### How can I tell if I am using the correct name of a built-in function?

By default, syntax highlighting is on. Syntax highlighting displays built-in function names in bright blue letters. If the cursor is in the text of the function name, you can press <F1> to open the online help topic for that function.

### How can I keep a log of results generated by the Command Window?

To save the entire contents of the Command Window, right click on the window, select **Save As**, and choose the name of the file in which you want the contents to be stored. To log specific messages to the Log Window, use the `logMessage()` function. To view this log, click on the Log tab in the lower left corner of the Command Window.

**How do I reuse previous command line statements?**

Use the up and down arrow keys to access the history of commands. If you type the initial part of the command, the arrow keys cycle through only the commands in the history that start with what you typed.

## Comparing HiQ and MATLAB Commands

Many commands in MATLAB mode are defined in HiQ mode as well. Other commands are handled more efficiently or flexibly by other HiQ user interfaces. Table 11-2 contains an alphabetical collection of general commands in MATLAB mode and the best way to achieve the same results in HiQ.

**Table 11-2.** HiQ and MATLAB Command Comparisons

| MATLAB Command | HiQ Equivalent |
|---|---|
| `cd` | Type `cd` in the Command Window. |
| `clear` *`objectName`* | Type `delete` *`objectName`* in the Command Window, where *`objectName`* is the name of the object you want to delete. |
| `diary` | Right click on the Command Window and select **Save As** to save the contents of the current Command Window. Type `clear` to clear the Command Window contents before beginning the session you want to record. |
| `dir` | Type `dir` or `ls` in the Command Window. |
| `echo` | Use the built-in function `logMessage` to record the supplied text in the Log Window. |
| `edit` | Edit the contents of a HiQ-Script through a view of the script object. Think of this view as a built-in editor provided by HiQ specifically designed for editing HiQ-Script. Use one of the following methods to create a view of a HiQ-Script.<br>• Type `view` *`scriptName`* on the command line, where *`scriptName`* is the name of an existing HiQ-Script you want to edit.<br>• In the Explorer, find the name of the script in the tree structure under the Objects branch. You can double click on it or press <Shift-Enter> to create a view of the script in its own window.<br>• Type `place` *`scriptName`* on the command line, where *`scriptName`* is the name of the script you want to edit. A view of the script is placed on the current notebook page. |

**Table 11-2.** HiQ and MATLAB Command Comparisons (Continued)

| MATLAB Command | HiQ Equivalent |
|---|---|
| `format` | Use the HiQ property pages. Right click on the Command Window, select **Properties**, and click on the Numbers page to access the formatting properties for displaying numbers in the Command Window. |
| `helpwin` | Press the <F1> key anywhere in HiQ to bring up the HiQ online help. From there, you have access to all help resources. You also can type `help` from the HiQ Command Window to activate the help system. |
| `helpdesk` | Select **Help»Visit HiQ Web Page** to retrieve online product and technical information. |
| `load` | In HiQ, all objects are stored within the Notebook. When you open the Notebook, all objects associated with that Notebook are automatically opened. You can open a Notebook with the **File»Open** command. |
| `more` | Because views of HiQ objects are scrollable, creating a view of a particular object allows you to inspect any portion of the object dynamically.<br><br>To specify where you want object views to appear, right click on the Command Window and select **Properties**. From the property pages, you can control which objects are displayed in the Command Window, which are viewed in their own windows, and the maximum size of a matrix or vector viewable in the Command Window. Matrices and vectors larger than the specified size are displayed in individual windows. |
| `quit` | Type `quit` in the Command Window or select **File»Exit**. |
| `save` | In HiQ, all objects are stored within the Notebook. Saving a Notebook saves all of the objects associated with it. To save a Notebook, use **File»Save** or **File»Save As**. |

**Table 11-2.** HiQ and MATLAB Command Comparisons (Continued)

| MATLAB Command | HiQ Equivalent |
|---|---|
| `type` | List the contents of a HiQ-Script through a view of the Script object. Think of this view as a built-in editor provided by HiQ specifically designed for inspecting and editing HiQ-Script. Use one of the following methods to create a view of a HiQ-Script.<br>• Type `view` *`scriptName`* on the command line, where *`scriptName`* is the name of an existing HiQ-Script you want to view.<br>• In the Explorer, find the name of the script in the tree structure under the Objects branch. You can double click on it or press <Shift-Enter> to create a view of the script in its own window.<br>• Type `place` *`scriptName`* on the command line where *`scriptName`* is the name of the script you want to view. A view of the script is placed on the current Notebook page. |
| `ver` | Select **Help»About HiQ** to get version and copyright information. |
| `what` | Select **File»Open** to display the current directory storing HiQ notebooks. By default, the file filter is set to the Notebook extension `.HiQ` so that only HiQ Notebooks show in the directory listing. |
| `whatsnew` | Select **Help»What's New** to read about new features in this version of HiQ. |
| `who` | Type `objects` in the Command Window or look at the Objects section in the Explorer. |
| `whos` *`objectName`* | Type `whatis` *`objectName`* in the Command Window or look at the Objects section in the Explorer. |
| `web` | Select **Help»Visit HiQ Web Page** to visit the HiQ Web Page. |

If you do not find a command in Table 11-2, it might be because that command is unique to the MATLAB environment. For example, path commands such as `rmpath` and `addpath` do not have meaning in the HiQ environment because a HiQ Notebook contains all of the objects and scripts associated with it.

# Automated M-file to HiQ-Script Translator

The M-file to HiQ-Script translator is a tool you can use to convert your M-file programs to HiQ. Its primary goal is to automate the change in language syntax, including loops (such as while and for), conditional execution (such as if and switch), and structure subranging. The translator can map functions if the usage of the function is unambiguous. For example, some MATLAB functions map directly to built-in HiQ functions allowing the translator to replace the MATLAB function name with the HiQ function name. Many MATLAB functions can be emulated by a user-defined function in HiQ. In these cases, the MATLAB functions are prefixed with `M__`. When a script executes, the appropriate HiQ functions from the translation library execute to produce equivalent results.

The translation maintains the readability of the script in both function calls and common language syntax. Some language syntax used in M-files does not match any syntax in HiQ-Script, in which case HiQ functions are used in place of the language syntax to produce the proper results.

## Using the Translator

The Import Wizard is the standard interface in HiQ for importing any kind of data, including M-files. Use the following procedure to import M-files with the HiQ Import Wizard.

1. Access the Import Wizard by selecting **Notebook»Import Wizard** or select the Import Wizard icon in the toolbar.
2. In the **Import From** section of the Import Wizard, select the **File** option and press **Browse** to select the M-file you want to translate.

   After selecting the file, notice that the contents of the beginning of the M-file appear in the Preview window.
3. If the Preview window is displaying the M-file you want to translate, click the **Next** button at the bottom of dialog box to continue.

   If the file has the default MATLAB extension, `.m`, the Import Wizard automatically detects it as an M-file, and the **Predefined** option in the Format section selects the description **M-file to HiQ-Script**. In the upper right, notice the name of the script object that will hold the results of the translation. To see the results of the translation, look in the Preview window.

4. Click the **Next** button again to view the Translation Output Options dialog, as shown in Figure 11-2.

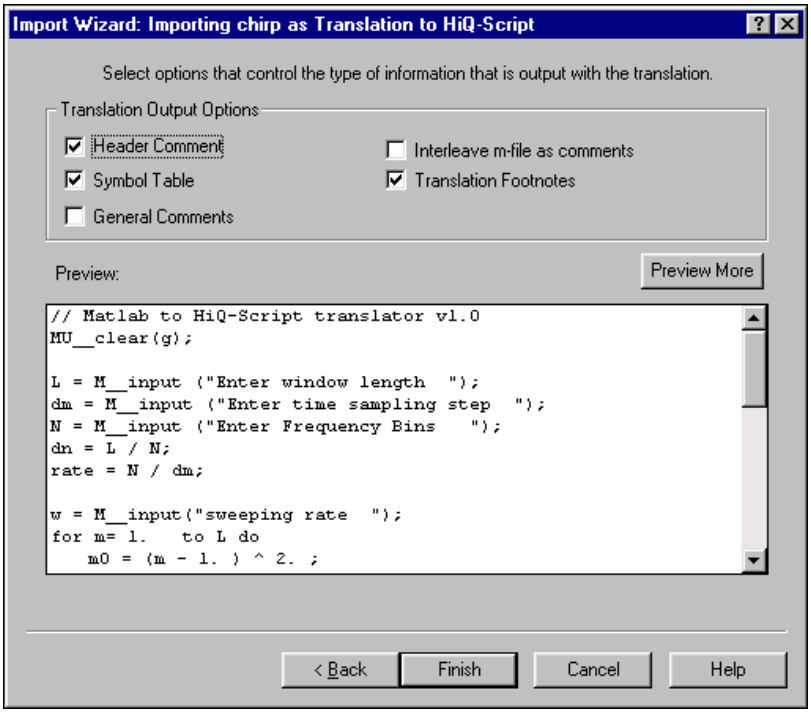



**Figure 11-2.** Import Wizard—Translator Output Options

5. Select the translation options that you want. The following list explains the options in more detail. Notice that the Preview window is automatically updated to reflect the new options.
   - **Header Comments**—Inserts a header comment at the beginning of the translated file.
   - **Symbol Table**—Includes a list of all symbols used in the translation. Use this option to verify the type descriptions of the translated objects and functions.
   - **General Comments**—Includes explanations and general comments about the details of the translation.

- **Interleave M-file as Comments**—Interleaves each line of the original M-file with the corresponding translated line in HiQ-Script. Use this option to compare M-file script code with the automatically generated HiQ-Script code.
- **Translation Footnotes**—Appends additional translation footnotes to the end of the translation file. Use this option to obtain additional details about any limitations of the current translation.

6. Press **Finish** to complete the translation.

You also can translate M-files through the view of a script object. Although this approach has fewer options than the first, it provides a faster and more automated way to translate an M-file. The results of both procedures are the same.

1. To translate a new M-file, place a new view of a new script object on the notebook page.
2. Right click on the view and select **File»Import MATLAB M-file**.
3. Select the M-file you want to import. After you select the file, the translation begins and the resulting translation is stored in the current script object.

## Working with the Results

If you translated your M-file using the Import Wizard, a view of that object is automatically placed on the Notebook page.

If you translated your M-file into an existing HiQ-Script view, the contents of the HiQ-Script are replaced by the translated output.

Table 11-3 compares how common scripting errors are reported in MATLAB and HiQ.

**Table 11-3.** Common Error Messages in MATLAB and HiQ

| M-file Error Message | HiQ-Script Error Message |
|---|---|
| **Assignment using a variable that does not exist: y = x;** | |
| `??? Undefined function or variable 'x'.` | `The object "x" hasn't yet been assigned a value.` |
| **Calling a function that is not defined: y = x(5);** | |
| `Warning: Reference to uninitialized variable 'x'.` | `"x" is not a function or polynomial.` |

**Table 11-3.** Common Error Messages in MATLAB and HiQ (Continued)

| M-file Error Message | HiQ-Script Error Message |
|---|---|
| **Performing a language operation with incompatible object types:** | |
| `??? Function '<op>' not defined for variables of class '<variableType>'.` | `Invalid operand types.` |

To successfully execute a script, all functions in the script must be defined and compiled in the notebook. If you are translating an M-file that calls functions in other M-files, you need to translate those files to HiQ-Script and compile them to make them available to all scripts in the current Notebook.

# HiQ-Script for MATLAB Users

Working in a new scripting language involves learning new syntax as well as new functionality. Table 11-4 compares M-files and HiQ-Script to help you learn the HiQ syntax and functionality in the following categories:

- Script Comments
- Initializing Matrices
- Subscripting Matrices
- Expressions (for example, x = a + b/2; v = u.*w; E = B-C;)
- Operators (for example, .^, -, .\)
- Flow Control (for example, `if`, `for`, and `while`)
- Function Definitions
- Function Calls

**Table 11-4.** HiQ-Script for MATLAB Users

| MATLAB | HiQ-Script |
|---|---|
| **Script Comments** | |
| % A comment in an M-file | // A comment in HiQ-Script |
| **Initializing Matrices** | |
| `M1 = [1];`<br>`M2 = [1 2 3 4 5];`<br>`M3 = [1; 2; 3; 4; 5];`<br>`M4 = [1 2 3; 4 5 6; 7 8 9];` | `M1 = {1};`<br>`M2 = {1, 2, 3, 4, 5};`<br>`M3 = {1; 2; 3; 4; 5};`<br>`M4 = {1, 2, 3; 4, 5, 6; 7, 8, 9};` |

**Table 11-4.** HiQ-Script for MATLAB Users (Continued)

| MATLAB | HiQ-Script |
|---|---|
| **Subscripting Matrices** | |
| `x = A(1,1);` | `x = A[1,1];` |
| `v = A(1,:);` | `v = A[1];` |
| `u = A(:,1);` | `u = A[:,1];` |
| `y = A(5);` | `y = (toVector(A'))[5];` |
| `p = A(1:5,4);` | `p = A[1:5,4];` |
| `q = A(2,3:7);` | `q = A[2,3:7];` |
| `M = A(3:5,4:9);` | `M = A[3:5,4:9]` |
| **Expressions** | |
| `r = 1+sqrt(5)/2;` | `r = 1+sqrt(5)/2;` |
| `s = abs(5-7i);` | `s = abs(5-7*<i>);` |
| `z = (4+2i)*(3-8i);` | `z = (4,2)*(3,-8);` |
| `x = exp(log(1e10*r));` | `x = exp(ln(1e10*r));` |
| `y = log(exp(1e10*s));` | `y = ln(exp(1e10*s));` |
| **Operators** | |
| `s = 5;` | `s = 5;` |
| `v = 1:.5:5;` | `v = seq(1, 5, .5);` |
| `M = [v; v; v; v];` | `M = {v; v; v; v};` |
| `r = s+s;` | `r = s+s;` |
| `t = s.*r;` | `t = s.*r;` |
| `u = v.*v;` | `u = v.*v;` |
| `A = M./M;` | `A = M./M;` |
| `B = M.\A;` | `B = M.\A;` |
| `C = A.^B;` | `C = A.^B;` |
| `D = C.';` | `D = trans(C);` |
| `E = D';` | `E = D';` |
| `w = u - v;` | `w = u - v;` |
| `n = v';` | `n = v';` |
| `p = [n, n .^2, 2 .^n];` | `p = {n, n .^2, 2 .^n};` |

**Table 11-4.** HiQ-Script for MATLAB Users (Continued)

| MATLAB | HiQ-Script |
|---|---|
| **Flow Control** | |
| `if errCode < 0`<br>`  disp('Invalid error code.');`<br>`elseif errCode == 0`<br>`  disp('No errors occurred.');`<br>`else`<br>`  warning('An error has`<br>`  occurred.');`<br>`end` | `if errCode < 0 then`<br>`  message("Invalid error code.");`<br>`else if errCode == 0 then`<br>`  message("No errors occurred.");`<br>`else`<br>`  warning("An error has`<br>`  occurred.");`<br>`end if;` |
| `switch errCode`<br>`  case 0`<br>`  disp('No errors reported.');`<br>`  case 1`<br>`  error('Error 1 occurred.');`<br>`  case 2`<br>`  error('Error 2 occurred.');`<br>`  otherwise`<br>`  error('Unhandled error code.');`<br>`end` | `select errCode from`<br>`  case 0:`<br>`  error("No errors reported.");`<br>`  case 1:`<br>`  error("Error 1 occurred.");`<br>`  case 2:`<br>`  error("Error 2 occurred.");`<br>`  default:`<br>`  error("Unhandled error code.");`<br>`end select;` |
| `i = 1;`<br>`absSum = 0;`<br>`while i <= size(A, 1)`<br>`  j = 1;`<br>`  while j <= size(A, 2)`<br>`  absSum = absSum + abs(A(i, j));`<br>`  j = j + 1;`<br>`  end`<br>`  i = i + 1;`<br>`  if absSum == inf`<br>`  break;`<br>`  end`<br>`end` | `i = 1;`<br>`absSum = 0;`<br>`while i <= A.rows do`<br>`  j = 1;`<br>`  while j <= A.columns do`<br>`  absSum = absSum + abs(A[i, j]);`<br>`  j = j + 1;`<br>`  end while;`<br>`  i = i + 1;`<br>`  if absSum == <inf> then`<br>`  exit while;`<br>`  end if;`<br>`end while;` |

**Table 11-4.** HiQ-Script for MATLAB Users (Continued)

| MATLAB | HiQ-Script |
|---|---|
| `continue = 1;`<br>`absSum = 0;`<br>`for i = 1:size(A, 1)`<br>`  for j = 1:size(A, 2)`<br>`  absSum = absSum + abs(A(i, j));`<br>`  if absSum == inf`<br>`  continue = 0;`<br>`  break;`<br>`  end`<br>`  end`<br>`  if ~continue`<br>`  break;`<br>`  end`<br>`end` | `// multiple exits avoid 'continue'`<br>`absSum = 0;`<br>`for i = 1 to A.rows do`<br>`  for j = 1 to A.columns do`<br>`  absSum = absSum + abs(A(i, j));`<br>`  if absSum == <inf> then`<br>`  // multiple-exit support`<br>`  exit 2 fors;`<br>`  end if;`<br>`  end for;`<br>`end for;` |
| **Function Definitions** | |
| `function m = slope(P1, P2)`<br>`% SLOPE Slope between two points.`<br>`% SLOPE(P1) computes the slope`<br>`% between two points stored in`<br>`% the columns of P.`<br>`% SLOPE(P1, P2) computes the`<br>`% slope between two points stored`<br>`% in column vectors P1 and P2.`<br>`if nargin == 1`<br>`  m = P1(2,2)-P1(2,1);`<br>`  m = m/(P1(1,2)-P1(1,1));`<br>`else`<br>`  m = P2(2,1)-P1(2,1)`<br>`  m = m/(P2(1,1)-P1(1,1));`<br>`end` | `function slope(P1, P2)`<br>`// Compute the slope of two points`<br>`// where the points are given in`<br>`// columns of P1 or in two vectors,`<br>`// P1 and P2.`<br>`  if __ins == 1 then`<br>`  m = P1[2,2]-P1[2,1];`<br>`  m = m/(P1[1,2]-P1[1,1]);`<br>`  else`<br>`  m = P2[2]-P1[2];`<br>`  m = m/(P2[1]-P1[1]);`<br>`  end if;`<br>`  return m;`<br>`end function;` |
| `function [xmin, xmax] = minmax(x)`<br>`% MINMAX Compute the minimum and`<br>`% and maximum values in x.`<br>`xmin = min(min(x));`<br>`if nargout == 2`<br>`  xmax = max(max(x));`<br>`end` | `function minmax(x)`<br>`// Compute the minimum and`<br>`// maximum values in x.`<br>`  xmin = min(x);`<br>`  if __outs == 2 then`<br>`  xmax = max(x);`<br>`  end if;`<br>`  return xmin, xmax;`<br>`end function;` |

**Table 11-4.** HiQ-Script for MATLAB Users (Continued)

| MATLAB | HiQ-Script |
|---|---|
| **Function Calls** | |
| matrix = abs(matrix); | matrix = abs(matrix); |
| matrix = chol(matrix); | matrix = choleskyD(matrix); |
| scalar = cond(matrix); | scalar = cond(matrix); |
| scalar = cond(matrix, inf); | scalar = cond(matrix, <Li>); |
| vector = eig(matrix); | vector = eigen(matrix); |
| vector = eig(matrix, matrix); | vector = eigen(matrix, matrix); |
| [V,D] = eig(matrix); | [d,V] = eigen(matrix); D = diag(d); |
| [v,d] = eigs(matrix, 1); | [d,v] = eigenDom(matrix); |
| [v,d] = eigs(matrix, 1, scalar); | [d,v] = eigenSel(matrix, scalar); |
| matrix = eye(scalar); | matrix = ident(integer); |
| scalar = fmin('cos', 3, 4); | [,scalar] = optimize(cos, toVector(3.1)); |
| scalar = fmins('fct', vector); | [,scalar] = optimize(fct, vector); |
| scalar = fzero('sin', [a b]); | scalar = root(sin, a, b); |
| scalar = fzero('sin', 3); | scalar = root(sin, toVector(3)); |
| matrix = inv(matrix); | matrix = inv(matrix); |
| scalar = max(max(matrix)); | scalar = max(matrix); |
| scalar = mean(mean(matrix)); | scalar = mean(toVector(matrix)); |
| scalar = norm(matrix); | scalar = norm(matrix); |
| scalar = norm(matrix, 'fro'); | scalar = norm(matrix, <frob>); |
| scalar = norm(matrix, 1); | scalar = norm(matrix, <L1>); |
| scalar = plot(vector,vector); | [graph, integer] = createGraph(vector, vector); |
| R = qr(matrix); | R = QRD(matrix); |
| [Q,R] = qr(matrix); | [R,Q] = QRD(matrix); |
| vector = rand(6, 1); | vector = createVector(6, <random>); |

**Table 11-4.** HiQ-Script for MATLAB Users (Continued)

| MATLAB | HiQ-Script |
|---|---|
| `matrix = rand(5, 4);` | `matrix = createMatrix(5, 4, <random>);` |
| `matrix = randn(3);` | `matrix = createMatrix(3, 3, <random>, <normal>);` |
| `scalar = rank(matrix);` | `integer = rank(matrix);` |
| `scalar = rank(matrix, 1e-4);` | `integer = rank(matrix, 1e-4);` |
| `vector = roots([1 2 3 4]);` | `vector = roots({poly: "x^3+2x^2+3x+4"});` |
| `[vector, ivector] = sort(vector);` | `[vector, integerVector] = sort(vector);` |
| `matrix = sqrt(matrix);` | `matrix = sqrt(matrix);` |
| `scalar = surf(matrix);` | `graph = createGraph(matrix);` |
| `vector = svd(matrix);` | `vector = sv(matrix);` |
| `[U,S,V] = svd(matrix);` | `[U,S,V] = svd(matrix);` |

# HiQ Online Help for MATLAB Users

HiQ offers online help specifically designed to help MATLAB users work in HiQ. You can access the online help in either of the following ways:

- Press <F1> and select any of the MATLAB topics from the Index tab.
- Type `help matlab` in the Command Window.

# A

# Technical Support Resources

## Web Support

National Instruments Web support is your first stop for help in solving installation, configuration, and application problems and questions. Online problem-solving and diagnostic resources include frequently asked questions, knowledge bases, product-specific troubleshooting wizards, manuals, drivers, software updates, and more. Web support is available through the Technical Support section of `www.ni.com`

## NI Developer Zone

The NI Developer Zone at `zone.ni.com` is the essential resource for building measurement and automation systems. At the NI Developer Zone, you can easily access the latest example programs, system configurators, tutorials, technical news, as well as a community of developers ready to share their own techniques.

## Customer Education

National Instruments provides a number of alternatives to satisfy your training needs, from self-paced tutorials, videos, and interactive CDs to instructor-led hands-on courses at locations around the world. Visit the Customer Education section of `www.ni.com` for online course schedules, syllabi, training centers, and class registration.

## System Integration

If you have time constraints, limited in-house technical resources, or other dilemmas, you may prefer to employ consulting or system integration services. You can rely on the expertise available through our worldwide network of Alliance Program members. To find out more about our Alliance system integration solutions, visit the System Integration section of `www.ni.com`

# Worldwide Support

National Instruments has offices located around the world to help address your support needs. You can access our branch office Web sites from the Worldwide Offices section of `www.ni.com`. Branch office web sites provide up-to-date contact information, support phone numbers, e-mail addresses, and current events.

If you have searched the technical support resources on our Web site and still cannot find the answers you need, contact your local office or National Instruments corporate. Phone numbers for our worldwide offices are listed at the front of this manual.

# Glossary

| Prefix | Meanings | Value |
|---|---|---|
| p- | pico- | $10^{-12}$ |
| n- | nano- | $10^{-9}$ |
| µ- | micro- | $10^{-6}$ |
| m- | milli- | $10^{-3}$ |
| k- | kilo- | $10^{3}$ |
| M- | mega- | $10^{6}$ |
| G- | giga- | $10^{9}$ |
| t- | tera- | $10^{12}$ |

## A

ActiveX (Microsoft ActiveX) — A programming system and user interface that lets you work with interactive objects. Formerly called OLE.

ActiveX control — A standard software tool that adds additional functionality to any compatible ActiveX container.

ActiveX Control object — An object in HiQ that represents an inserted ActiveX control.

ActiveX embedded object — An object placed into a container and unconnected to any other object or application. *See also* embedded.

ActiveX Interface object — An object in HiQ that represents an interface to an ActiveX automation server application. See the `createInterface` function in the online help.

ActiveX linked object — An object placed into a container and connected to another object or application in the same container or in a separate container. *See also* linked.

ActiveX object — An object in HiQ that represents an inserted ActiveX object.

argument — *See* parameter.

assignment — A script statement that sets a value to a variable.

attribute — *See* property.

## B

built-in function — One of many programming utilities in HiQ-Script that perform analysis, graphical, or utility operations.

## C

caret — *See* insertion point.

cell — In a matrix or vector, the intersection of a row and column that contains a numerical value.

Color object — A HiQ object that sets the color attributes of graphs and plots.

Command Window — *See* HiQ Command Window.

comment — An explanatory line or portion of a line in HiQ-Script.

compiled script — A HiQ object containing HiQ-Script language that has been converted to code that a computer uses to execute the program.

ComponentWorks — A collection of 32-bit ActiveX controls designed for building virtual instrumentation systems.

constant — A predefined value in HiQ-Script.

cursor — The pointer or other image that displays on screen to show the location of your mouse, trackball, or other pointing device.

## D

| | |
|---|---|
| data type | *See* numeric type. |
| DataSocket | Technology that simplifies data exchange between an application and other applications, files, FTP servers, and Web servers. It provides one common API to a number of different communication protocols. You can specify DataSocket sources and targets (connections) using URLs (uniform resource locators) that adhere to the familiar URL model.<br><br>DataSocket uses an enhanced data format for exchanging measurement data, as well as the attributes of the data. Data attributes might include information such as an acquisition rate, test operator name, time stamp, quality of data, and so on. |
| debug | To check and correct invalid code in a HiQ script in order to eliminate errors during the compilation or execution of the script. |
| declaration | A HiQ-Script statement that defines the scope of variables. Can be either project or local. |
| dialog box | A window containing a message, interactive options, and buttons. |
| dock | To attach a detachable window or toolbar to the HiQ window. |
| drag-and-drop | To move an object to a specific location, using the mouse to click on, drag, and release the object. Depending on the location of release, a specific action can occur. |

## E

| | |
|---|---|
| embedded | Inserted into a container object and unconnected to any other object or application. Compare this term to linked. *See also* ActiveX embedded object. |
| error message | An information box that appears when HiQ cannot complete an action due to an internal error, compile error, run-time error, or user error. |
| Explorer | *See* HiQ Explorer. |
| expression | A mathematical operator and its operands. |

## F

| | |
|---|---|
| Font object | A HiQ object type used to set the font attributes of graphs and plots. |
| function | A block of code that performs a specific task in HiQ-Script; can be a HiQ built-in function or a user-defined function. |
| function call | Specific HiQ-Script syntax that calls a user or built-in function with given parameters. |
| Function Object | An object that represents the executable pseudocode of a complied Script object. |

## G

| | |
|---|---|
| grab handle | A site on a selected object that you click on and drag to move, size, or reshape the object. |
| Graph object | The HiQ object that contains a two-dimensional or three-dimensional collection of plots. *See* Plot object. |

## H

| | |
|---|---|
| handle | *See* grab handle. |
| HiQ Command Window | An intuitive window where you type HiQ-Script to get immediate results. |
| HiQ Constant object | A HiQ object that represents a constant in HiQ-Script. |
| HiQ Explorer | An interactive window displaying the objects, sections, and pages of an open Notebook. |
| HiQ Log Window | A window to which you can post messages from HiQ-Script. |
| HiQ Object Browser | A browser window in which you can view all the interfaces for ActiveX servers, objects, and controls installed on your computer. |
| HiQ Tools toolbar | Part of the HiQ user interface that contains icons for objects you can place in a Notebook. Depending on your preference, the toolbar can appear on any edge of the interface, or as a floating palette. |
| HiQ-Script | The intuitive HiQ programming language for mathematics. |

## I

IEEE — Institute of Electrical and Electronic Engineers.

initialization — In HiQ-Script, a designated syntax which creates a particular type of object, for example, vector, matrix, color, and others.

insertion point — The location where text will be inserted (also referred to as the *caret*).

## K

keyword — A reserved word in HiQ Script, such as `if`, `then`, or `while`, that is used for constructing specific types of programming statements.

## L

LabVIEW — Laboratory Virtual Instrument Engineering Workbench. Program development application based on the programming language G used commonly for test and measurement applications.

LabWindows\CVI — An integrated ANSI C environment designed for engineers and scientists creating virtual instrumentation applications.

linked — Inserted into a container object and connected with another object or application. Compare this term to embedded. *See also* ActiveX linked object.

local — Describes the scope of an object. An object with local scope is not associated with the Notebook.

Log Window — *See* HiQ Log Window.

loop — A statement in HiQ-Script consisting of keywords and nested statements that performs a repetitive function. Also known as an iteration statement.

## M

| | |
|---|---|
| MATLAB | Third-party software product for programming and working with data. |
| Matrix object | A HiQ numeric object containing an array of elements with rows and columns. *See* Vector object. |
| message box | A secondary window that appears containing information about the status of a HiQ operation. *See* dialog box. |

## N

| | |
|---|---|
| Notebook | The workspace in HiQ that stores, organizes, and displays all the components of an analysis and visualization problem. |
| Numeric object | A HiQ object type that is defined in terms of a numeric type (integer, real, or complex) and an object type (matrix, vector, polynomial, or scalar). |
| numeric type | One of three types of a numeric object (integer, real, or complex). |

## O

| | |
|---|---|
| object | An entity in a HiQ Notebook that contains data of a specific type, for example, numeric, graphic, text, or HiQ-Script. Objects work together in a Notebook to generate and display solutions to analysis and visualization problems. Objects are always stored in a Notebook, but are not always visible on a Notebook page. *See* object view. |
| Object Browser | *See* HiQ Object Browser. |
| Object List | A window in HiQ that displays all the objects in a HiQ Notebook. |
| object view | A HiQ object that is visible on a Notebook page. |
| OLE (Microsoft OLE) | Object Linking and Embedding. *See* ActiveX (Microsoft ActiveX). |
| operand | An object (or objects) modified by an operator. |
| operator | Code in HiQ-Script that is specific to basic mathematical operations and structure of HiQ-Script language. Often represented as a symbol, for example, "/" represents division. |

## P

| | |
|---|---|
| parameter | An independent variable passed to a user-defined or built-in function call in a parameter list. |
| Plot object | A HiQ object type that graphically represents a 2D or 3D function or data set used in conjunction with a graph object. *See* Graph object. |
| Polynomial object | A HiQ numeric object represented by an equation in the polynomial form $ax^n + bx^{n-1}+cx^{n-2} + ...$ |
| pop-up menu | A context-sensitive menu that you can access by right clicking with your mouse on an object or on the Notebook. |
| Problem Solver | A HiQ Notebook containing objects, including HiQ-Script, that allows you to interactively perform analysis of data and display results for a broad class of problems. For example, the expression evaluator problem solver can display the results of any expression. |
| project | Describes the scope of an object. An object with project scope is saved with the Notebook. |
| Properties dialog box | A window in HiQ with tabbed pages (property pages) where you can quickly set a wide variety of attributes for a given object. |
| property | Attributes of a HiQ object. Examples include the color of a plot, the size of a matrix, and the type of any object. |
| property page | A tabbed subsection of a property dialog box containing a collection of object attributes. |

## S

| | |
|---|---|
| Scalar object | A HiQ numeric object represented as a number. *See* vector and matrix. |
| scope | In HiQ-Script, a HiQ-object declaration that specifies whether the object is available to the entire Notebook (project) or is temporarily available (local). |
| script | A block of code that performs a certain task. *See* compiled script and HiQ-Script. |
| Script object | A HiQ object type containing HiQ-Script and from which you compile and execute your program. |

section tabs — An organizing tool in a HiQ Notebook that lets you label and quickly access different parts of your Notebook.

selection handle — A graphical control point of an object that provides direct manipulation support for operations of that object, such as moving, sizing, or scaling.

selection tool — The mouse cursor in HiQ shaped like a standard pointer arrow that lets you select, move, and manipulate objects on the Notebook.

shortcut key — A key or combination of keys that you press to invoke a command.

Standard toolbar — Site on the user interface of HiQ that contains basic utility tools, for example, **Save**, **Open**, **Cut**, **Paste**, and, **Print**. Depending on your preference, the toolbar can appear on any edge of the interface, or as a floating window.

statement — In HiQ-Script, a line of code consisting of various keywords, functions, and/or operators that performs a certain task.

status bar — A region, usually the bottom of a window, containing information about HiQ and any selected object.

symbol — The name for *objects* in HiQ for the Macintosh.

## T

Text object — A HiQ object type where you can enter and edit text.

toolbar — Site on an application interface that contains various buttons and other controls. Depending on your preference, a toolbar can appear on any edge of the interface, or a floating window.

tooltip — A small, descriptive pop-up window that appears when you position the mouse cursor over a toolbar icon.

type — A classification of an object based on its characteristics, behavior, and attributes.

## U

| | |
|---|---|
| Untyped object | A HiQ object that has not been assigned a value. |
| user-defined function | A function that a user creates in HiQ-Script to perform customized analysis, graphical, or utility operations. |

## V

| | |
|---|---|
| Vector object | A HiQ numeric object containing an array with one row or column. Compare with matrix. |

# Index

## Symbols


> prompt, 3-2, 11-2
» prompt, 11-3


## Numerics


2D graphs, 6-1 to 6-10
- automatically generating HiQ-Script with new plots, 6-3
- axis properties, 6-5
- axis ranges, 6-9
- captions, 6-5
- Climate Notebook example, 6-1 to 6-10
- creating, 6-1 to 6-2
- modifying graph and plot, 6-4 to 6-6
- plot styles, 6-5
- plotting data, 6-2 to 6-4
- properties, 6-4 to 6-6
- visualizing data, 6-1 to 6-4
- working with multiple plots, 6-7 to 6-10

3D graphs, 6-11 to 6-16
- captions and axes titles, 6-13
- color maps, 6-13 to 6-14
- creating, 6-11
- modifying graph and plot, 6-13 to 6-15
- panning, 6-16
- projections, 6-14
- properties, 6-13 to 6-14
- rotating, 6-16
- Seismic Notebook example, 6-11 to 6-15
- visualizing data, 6-11
- zooming, 6-16


## A


ActiveX
- automation client, 8-1
- automation server, 8-1
- control container, 8-1
- controls, 8-2 to 8-5, 8-8 to 8-12, 10-2
- design mode, 8-3
- document container, 8-1
- document server, 8-1
- Microsoft Word Document objects, 8-6 to 8-8
- Object Browser, 8-4

analyzing data with HiQ-Script, 7-1 to 7-10
attach, 3-14
automatic generation of HiQ-Script, 4-3, 6-3
automatically executing functions, 10-3
automating HiQ-Script analysis, 7-9 to 7-10


## B


backslash operator, 3-5
browse, 3-15


## C


cd, 3-15
clear, 3-14
clearHistory, 3-14
Climate Notebook
- adding ActiveX objects, 8-2 to 8-8
  - ComponentWorks ActiveX controls, 8-2 to 8-6
  - Microsoft Word Document, 8-6 to 8-8

analyzing data with HiQ-Script, 7-1 to 7-5
converting to Problem Solver, 9-1 to 9-4
importing data, 4-1 to 4-4, 5-2 to 5-3
maintaining help file for, 10-4 to 10-5
previewing in HiQ Reader, 10-1 to 10-2
subscribing to live measurements over the Internet, 5-2 to 5-3
visualizing 2D data, 6-1 to 6-10
color maps, 6-13 to 6-14
Command Window, 3-1 to 3-13
> prompt, 3-2
adjusting the window, 3-1
changing default name for objects, 3-3
commands, 3-14 to 3-15
copying and pasting, 3-6
default position, 3-1, 11-2
floating Command Window, 3-1 to 3-2
getting help, 3-9
hiding results using semicolon, 3-2
MATLAB compatibility. *See* MATLAB compatibility.
multiline commands, 3-4
navigating, 3-1 to 3-4
recalling and editing commands, 3-4, 3-7
saving objects, 3-8
syntax highlighting, 3-3
using with Notebooks, 3-10 to 3-11, 3-12 to 3-13
ComponentWorks ActiveX controls
adding to Climate Notebook, 8-2 to 8-6
adding to Seismic Notebook, 8-8 to 8-12
distributing with HiQ Notebooks, 10-2
property pages, 8-3
connecting to DataSocket data, 5-1 to 5-5
conventions used in the manual, *iv*
Customer Education, A-1


## D


data fitting Notebooks, 2-1 to 2-4
data importing. *See* importing data.
data visualization. *See* visualizing data
DataSocket, 5-1 to 5-5
accessing data with LabVIEW or Measurement Studio, 5-5
adding/removing DataSocket Servers, 5-2
browsing data, 5-1 to 5-5
packet, 5-4
publishing HiQ data, 5-4 to 5-5, 6-11 to 6-12
refreshing Explorer View, 5-2
subscribing to live data, 5-2 to 5-3, 6-11 to 6-12
DataSocket server, 5-1 to 5-5
delete, 3-14
destinations (data), 5-1 to 5-5
detach, 3-14
detach command, 11-8
dir, 3-15
distributing Notebooks, 10-1 to 10-5
dstp, 5-1 to 5-5


## E


examples
addPlot function, 3-8, 7-4, 8-4
arithmetic mean, 3-11, 7-1, 9-3
associating plots with second y-axis, 9-2
automatically executing functions, 10-3
backslash operator, 3-5
comments, 11-16
converting existing script to Problem Solver script, 9-5
createGraph function, 3-3, 3-8, 3-11

creating a matrix, 3-2, 3-5, 3-8
creating a polynomial, 3-3
creating a vector, 3-5, 3-11, 7-6
Expression Evaluator Problem Solver, 2-4 to 2-7
expressions, 11-17
fct function, 3-7
fit function, 7-4, 8-4
flow control, 11-18 to 11-19
function calls, 11-20 to 11-21
function definitions, 11-19
importing and graphing data, 9-2
initializing an ActiveX control, 10-3
initializing matrices, 11-16
integrate function, 7-6, 8-11
MATLAB, 11-16 to 11-21
modifying plot styles, 9-2
ODEIVP function, 3-8
onOpen function, 10-3
operators, 11-17
placing objects on the Notebook, 3-12
Population Fit Notebook, 2-1 to 2-4
programmatically accessing ComponentWorks ActiveX controls, 8-4, 8-10
removePlot function, 8-4, 9-2
solve function, 3-5
solving a linear system of equations, 3-4 to 3-5
solving a system of ordinary differential equations, 3-5 to 3-9
subrange operator, 7-1
subscripting matrices, 11-17
user functions, 8-10, 11-19
viewing objects, 3-11
Export Wizard, 4-1, 4-7
Expression Evaluator Problem Solver Notebook, 2-4 to 2-7
entering data, 2-6 to 2-7
opening, 2-5
running scripts, 2-6 to 2-7

## F

functions
*See also* HiQ-Script.
automatically executing, 10-3
built-in, 3-3, 7-4
calling, 3-6
reference, 3-6 to 3-7

## G

getting started, 1-2 to 1-3
graphs. *See* 2D graphs *and* 3D graphs.

## H

help, 3-14
help. *See* online help.
HiQ
ActiveX Object Browser. *See* ActiveX Object Browser.
Command Window. *See* Command Window.
environment, 1-3 to 1-5
getting started, 1-2 to 1-3
installing, 1-1 to 1-2
launching, 1-2
Notebook. *See* Notebooks.
system requirements, 1-1
HiQ Reader, 10-1 to 10-2
HiQ-Script
*See also* examples.
analyzing data with HiQ-Script, 7-1 to 7-10
automated M-file to HiQ-Script translator, 11-13 to 11-16
translation options, 11-14
using the translator, 11-13 to 11-15
working with translated results, 11-15 to 11-16
built-in functions, 7-4
case sensitivity, 7-4

compiling all scripts in a Notebook, 9-2
generating with Import Wizard, 4-3
generating with new plots, 6-3
run view, 7-10
syntax comparison for MATLAB users, 11-16 to 11-21
View Source, 2-3

## I

Import Wizard, 4-7
about, 4-1
Climate Notebook example, 4-1 to 4-4
generating HiQ-Script, 4-3
importing M-files, 11-13 to 11-15
Seismic Notebook example, 4-4 to 4-6
importing data, 4-1 to 4-7, 5-2 to 5-3
as live data, 5-1 to 5-5
Climate Notebook example, 4-1 to 4-4, 5-2 to 5-3
custom import mode option, 4-1
M-files, 11-13 to 11-15
Seismic Notebook example, 4-4 to 4-7
installing HiQ, 1-1 to 1-2

## L

launching HiQ, 1-2
localhost, 5-4 to 5-5
ls, 3-15

## M

MATLAB compatibility, 3-15, 11-1 to 11-21
» prompt, 11-3
accessing MATLAB compatibility mode, 11-3 to 11-5
automated M-file to HiQ-Script translator, 11-13 to 11-16
translation options, 11-14
using the translator, 11-13 to 11-15
working with the translated results, 11-15 to 11-16
browsing MATLAB session variables, 11-4
comparison of HiQ and MATLAB commands, 11-10 to 11-12
creating HiQ objects, 11-4
frequently asked questions, 11-8 to 11-10
get and getall commands, 11-5
getting help, 11-8, 11-21
hiq command, 11-7
HiQ-Script syntax, 11-16 to 11-21
leveraging MATLAB work in HiQ, 11-1 to 11-2
matlab command, 11-3
put and putall commands, 11-5
sharing MATLAB data and functionality with HiQ, 11-5 to 11-7
using the Command Window, 11-2 to 11-12
working in MATLAB compatibility mode, 11-3 to 11-7
Microsoft Word document
embedding as ActiveX object, 8-6 to 8-8
multiline commands in the Command Window, 3-4
Multiple Document Interface (MDI), 1-6

## N

National Instruments Web support, A-1
navigating the Command Window, 3-1 to 3-4
NI Developer Zone, A-1

Notebooks, 2-1 to 2-7
*See also* Problem Solvers.
creating help files for, 10-3 to 10-5
definition
distributing to others, 10-1 to 10-5
examples, 2-1 to 2-7
help for current Notebook, 10-3 to 10-5
interacting with, 1-7
modifying from Command Window, 3-12 to 3-13
opening, 4-1, 4-4
as organizational tools, 1-4
sharing with others, 10-1 to 10-5
working with multiple windows, 1-6

## O

Object List, 2-4
objects, 1-5 to 1-6, 3-14
changing default name in Command Window, 3-3
definition, 1-4 to 1-5
deleting, 1-6
deleting views, 1-6
placing on Notebook page
from Command Window, 3-12
from DataSocket Servers, 5-2
from the Object List, 7-2
saving objects created in Command Window, 3-8
setting default view properties, 7-3
synchronizing data in multiple objects, 6-11 to 6-12
views, 1-6
online help
accessing, 1-7
Command Window help, 3-9
distributing with Notebooks, 10-3 to 10-5
MATLAB compatibility, 11-8, 11-21
opening Notebooks, 2-2
openNotebook, 3-14

## P

panning 3D graphs, 6-16
place, 3-14
plots. *See* graphs.
Population Fit Notebook, 2-1 to 2-4
opening a Notebook, 2-2
running a script, 2-3
viewing the Object List, 2-4
previewing Notebooks, 10-1 to 10-2
Problem Solvers
Climate Notebook example, 9-1 to 9-4
definition, 2-4, 9-1
entering data, 2-6 to 2-7
Expression Evaluator Problem Solver, 2-4 to 2-7
Seismic Notebook example, 9-5 to 9-6
projections, 6-14
properties
2D graph, 6-4 to 6-6, 6-8 to 6-10
3D graph, 6-13 to 6-14
general object properties, 4-3
scripts, 7-10
setting default view, 7-3
publishing HiQ data, 5-4 to 5-5
pwd, 3-15

## Q

quit, 3-14

## R

Reader. *See* HiQ Reader.
reading data, 5-2 to 5-3
recalling and editing commands, 3-4
requirements. *See* system requirements.
rotating 3D graphs, 6-16
run, 3-15
running scripts, 1-7, 2-3

## S

script. *See* HiQ-Script.
Seismic Notebook
adding ComponentWorks ActiveX controls, 8-8 to 8-12
analyzing data with HiQ-Script, 7-6 to 7-10
automating analysis, 7-9 to 7-10
converting to Problem Solver, 9-5 to 9-6
importing data, 4-4 to 4-7
publishing HiQ data over the Internet, 5-4 to 5-5
viewing help for, 10-5
visualizing 3D data, 6-11 to 6-15
sharing Notebooks with others, 10-1 to 10-5
sources (data), 5-1 to 5-5
starting HiQ, 1-2
subscribing to live data, 5-2 to 5-3
system integration, A-1
system requirements, 1-1

## T

technical support resources, A-1
terse, 3-14
terse mode, 11-8
three-dimensional graphs. *See* 3D graphs.
two-dimensional graphs. *See* 2D graphs.

## V

verbose, 3-14
verbose mode, 11-8
view, 3-15
View Source, 2-3
viewing the Object List, 2-4
visualizing data, 6-1 to 6-16

## W

Web support, A-1
whatChanged, 3-14
whatis, 3-15
worldwide technical support, A-2
writing data, 5-4 to 5-5

## Z

zooming 3D graphs, 6-16