



Chapter 16

Advanced Computational Features

This chapter introduces some ways to extend the computational power of Mathcad and to integrate Mathcad with other applications on your computing desktop.

Worksheet references

How to include a reference to a worksheet inside another worksheet.

Exchanging data with other applications

Using application components to link Mathcad calculations dynamically to other computational objects, including MathSoft Axum and S-PLUS, Microsoft Excel, and MATLAB.

Scripting custom OLE automation objects

Using VBScript or JScript to configure custom OLE automation objects.

Using Mathcad as an OLE automation server

Communicating with Mathcad from within another Windows application.

Worksheet references

There may be times when you want to use formulas and calculations from one Mathcad worksheet inside another. You may also have calculations and definitions that you re-use frequently in your work. You can, of course, simply use **Copy** and **Paste** from the **Edit** menu to move whatever you need to move, or drag regions from one worksheet and drop them in another. However, when entire worksheets are involved, this method can be cumbersome or may obscure the main computations of your worksheet.

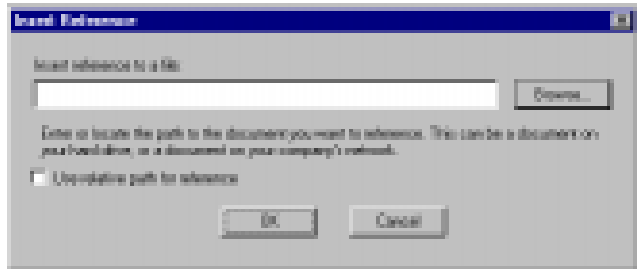
Mathcad therefore allows you to *reference* one worksheet from another—that is, to access the computations in the other worksheet without opening it or typing its equations or definitions directly in the current worksheet. When you insert a reference to a worksheet, you won't see the formulas of the referenced worksheet, but the current worksheet behaves as if you could.

Tip An alternative described in “Safeguarding an area of the worksheet” on page 109 is to create a collapsible *area* to hide calculations in your worksheet. This method, while it does not let you re-use calculations in the same way as a worksheet reference, does give you the option of password protecting or locking an area of calculations.

To insert a reference to a worksheet:

- Click the mouse wherever you want to insert the reference. Make sure you click in empty space and not in an existing region. The cursor should look like a crosshair.
- Choose **Reference** from the **Insert** menu.

- Click “Browse” to locate and select a worksheet. Alternatively, enter the path to a worksheet. You can also enter an Internet address (URL) to insert a reference to a Mathcad file that is located on the World Wide Web.



- Click “OK” to insert the reference into your worksheet.

To indicate that a reference has been inserted, Mathcad pastes a small icon wherever you had the crosshair. The path to the



referenced worksheet is to the right of the icon. All definitions in the referenced worksheet are available below or to the right of this icon. If you double-click this icon,

Mathcad opens the referenced worksheet in its own window for editing. You can move or delete this icon just as you would any other Mathcad region.

Note By default, the location of the referenced file is stored in the worksheet as an absolute system path (or URL). This means that if you move the main worksheet and the referenced worksheet to a different file system with a different directory structure, Mathcad cannot locate the referenced file. If you want the location of the referenced file on a drive to be stored relative to the Mathcad worksheet containing the reference, click “Use relative path for reference” in the Insert Reference dialog box. The reference is then valid even if you move the referenced file and the main worksheet to a different drive but keep the *relative* directory structure intact.

If you edit the contents of a referenced file so that any calculations change, you must re-open any worksheets that contain references to that file for calculations to update. The calculations in those worksheets do not update automatically.

Exchanging data with other applications

Components are specialized OLE objects that allow you to access the functions of other computational applications (such as MathSoft’s Axum and S-PLUS, Microsoft Excel, and MATLAB) within your Mathcad worksheet. Unlike other kinds of OLE objects you insert into a worksheet, as described in the section “Inserting objects” in Chapter 6, a component can receive data from Mathcad, return data to Mathcad, or both, linking the object dynamically to your Mathcad computations.

Tip As described in Chapter 11, “Vectors, Matrices, and Data Arrays,” Mathcad also provides the File Read/Write component for you to import and export *static* data files in a variety of formats compatible with other computational programs. And for linking dynamically to an object for which Mathcad does not have a dedicated component, see “Scripting custom OLE automation objects” on page 327.

The available components in Mathcad include:

- Axum, for creating Axum graphs
- S-PLUS Graph component, for creating S-PLUS graphs
- S-PLUS Script component, for accessing the programming environment of S-PLUS
- Excel, for accessing cells and formulas in a Microsoft Excel spreadsheet
- MATLAB, for accessing the programming environment of MATLAB

Pro

Note To use an application component, you must have the application for that component installed, but not necessarily running, on your system.

Tip See the SAMPLES folder of the location where you installed Mathcad for a variety of example files that use components.

How to use components

Components are designed to receive *input* from one or more Mathcad variables, do something you specify with the data, and in most cases return *output* to other Mathcad variables. An “input variable” is a scalar, vector, or matrix you have already defined in your Mathcad worksheet. It contains the data that is passed into a component. Output from a component (again, either a scalar, vector, or matrix) is then assigned to a Mathcad variable. This variable is referred to as an “output variable.”

The basic steps for using a component are as follows:

- Insert the component.
- Specify the input variable(s) and output variable(s).
- Configure the component to handle inputs from and return outputs to Mathcad.

Since some components only take input or only send output, these steps differ slightly for each component. The ideas presented in the steps provide an overview.

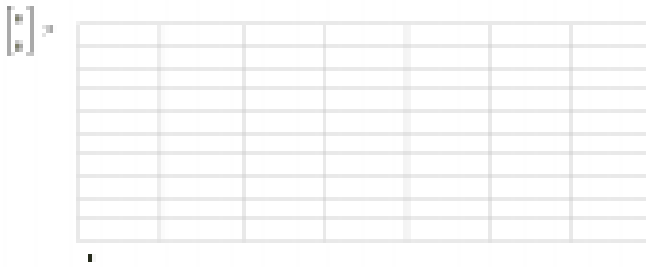
Step 1: Inserting a component

To insert a component into a Mathcad worksheet:

- Click in a blank spot of your Mathcad worksheet where you want the component to appear. Click below or to the right of definitions for any variables that will become inputs to the component.
- Choose **Component** from the **Insert** menu. This launches the Component Wizard.
- Choose one of the components from the list and click “Next.” Depending on the component you choose, you may see an additional dialog box that lets you specify properties of the component before it is inserted. When you click “Finish,” the component is inserted into your worksheet.

If you don’t see a Wizard when you choose one of the components from the Insert Component dialog box, you’ll immediately see the component, with some default properties, inserted into your worksheet.

Each component has its own particular appearance, but all components have one or more placeholders to the left of the := and/or at the bottom of the component. For example, the Excel component looks like this when inserted into your worksheet:



The placeholder(s) at the bottom of the component are for the names of previously defined input variables. The placeholder(s) you see to the left of the := are for the output variables.

Note To add an input or output variable (up to four each), click with the right mouse button on the component and choose **Add Input Variable** or **Add Output Variable** from the pop-up menu. To eliminate an input or output, choose **Remove Input Variable** or **Remove Output Variable** from the menu.

Step 2: Configuring a component

Once you've inserted a component into a worksheet, you configure its properties so that the component knows how to handle any inputs it receives from Mathcad and what to send as output. To configure the properties for a component:

- Click on the component once to select it.
- Click on it with the right mouse button to see a pop-up menu.
- Choose **Properties** from the pop-up menu.

The settings in the Properties dialog box differ for each component. For example, the Properties dialog box for the Excel component in Mathcad Professional lets you specify the cells in which the input values are stored and the cells from which the output is sent.

Tip When you insert an application component, you see a small window on that application's environment embedded in your Mathcad worksheet. When you *double-click* the component, the component is activated in place and Mathcad's menus and toolbars change to those of the other application, if it is installed on your computer. This gives you access to the features of that application without leaving the Mathcad environment.

Step 3: Exchanging data

Once you've configured the component, click outside it. At that point, the data exchange takes place: data passes from the input variable(s) into the component, the component processes the data, and the output variable(s) receive output from the component. This

exchange happens whenever you click on the component and press [F9], when the input variables change, or when you choose **Calculate Worksheet** from the **Math** menu.

Axum component

Axum is a technical graphing and data analysis application available from MathSoft that gives you access to over 75 two- and three-dimensional graph types with sophisticated formatting options. If you have Axum 5.03 or higher installed on your system, the Axum component brings some of this graphing power to your Mathcad worksheet.

Inserting an Axum component

To insert an Axum component into a Mathcad worksheet:

- Create the arrays which will provide input to the Axum component and which Axum will display in a graph. For information on the number and type of arrays required for each type of graph, see the Axum on-line Help.
- Click in a blank spot in your worksheet. Click below or to the right of the arrays.
- Choose **Component** from the **Insert** menu.
- Select Axum from the list and click “Next.” Choose the type of graph and the plot type on successive pages of the Wizard. When you click “Finish,” the Axum component is inserted into your worksheet.
- Enter the appropriate array variables in the placeholders that appear.

When you click outside the component, the array data are displayed in the Axum plot type you selected in the Wizard. See Figure 16-1 for an example.

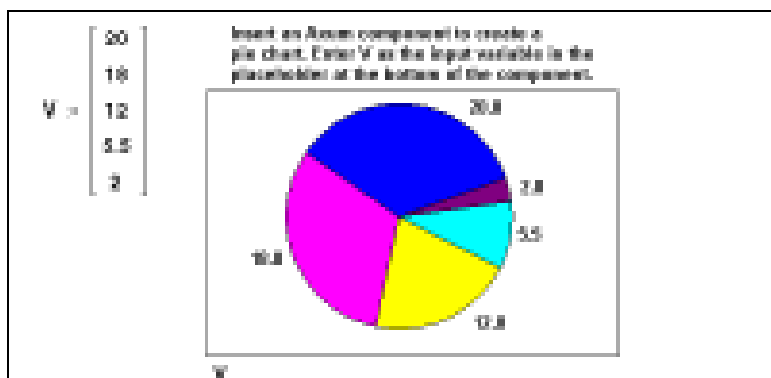


Figure 16-1: Creating an Axum graph via the Axum component.

Editing an Axum graph

After inserting an Axum component into a Mathcad worksheet, you can format the graph using Axum’s formatting options. To do so:

- Double-click the Axum graph in the Mathcad worksheet. The menus and toolbars change to Axum’s menus and toolbars.

- Edit the Axum graph using available options in the Axum environment.
- Click back in the Mathcad worksheet to recalculate the component and to resume working in Mathcad.

S-PLUS Graph component

S-PLUS is a sophisticated exploratory data analysis and statistical application available from MathSoft. The S-PLUS Graph component is an embedded S-PLUS graph sheet that allows you to integrate a wide range of two- and three-dimensional graphs, including Trellis plots, into your Mathcad worksheet. You must have S-PLUS 4.5 or higher installed on your computer to use this component.

Inserting an S-PLUS Graph component

To insert an S-PLUS Graph component into a Mathcad worksheet:

- Create the array(s) which will provide input to the S-PLUS Graph component. See the documentation that accompanied your copy of S-PLUS for details about the number and type of arrays required for each type of graph.
- Click in a blank spot in your worksheet. Make sure you click below or to the right of inputs you have defined in your Mathcad worksheet.
- Choose **Component** from the **Insert** menu.
- Select S-PLUS Graph from the list and click “Finish.” The S-PLUS Graph component is inserted into your worksheet. By default the component is inserted with a placeholder for a single input variable.

Tip The only way to send meaningful values to output variables is to set the graph type to “Call Graph Script” (see below) and specify the outputs within the script.

Note By default, the data in the Mathcad input variables are sent into S-PLUS variables named **in0**, **in1**, **in2**, and so on. The S-PLUS variables **out0**, **out1**, **out2**, and so on define the data to be passed to the Mathcad output variables, if any. To change these names, choose **Properties** from the component’s pop-up menu, and enter new names in the Input Variables and Output Variables tabs.

Types of graphs

To access the different graph types for the component, click with the right mouse button on the component and choose **Properties** from the pop-up menu. The Graph Type tab in the Properties dialog box allows you to select one of the following:

- **Line Plot.** This default graph type shows one to four traces on a single graph. Each input is interpreted as a matrix with either one column (interpreted as y-values to plot against $x = 1 \dots n$), or two columns (interpreted as x- and y-values).

- **Gui Graph.** This graph type supports any of the S-PLUS graphs available through the S-PLUS graphical user interface. For this component, the inputs are interpreted as arrays whose data columns are passed to the graph-creation routines. The graph type rendered is specified in the “Axis Type” and “Plot Type” fields. Click “Choose Axis/Plot Type” to open the S-PLUS dialog box for choosing plot types and viewing thumbnail images of the different types.

Note On Gui graphs, particular inputs can be treated as conditioning variables in order to produce Trellis plots (conditioned, multipanel plots). To do this, click the checkboxes in the Input Variables tab of the Properties dialog box.

- **Call Graph Script.** This graph type supports any commands that generate “traditional” (version 3.3-style) S-PLUS graphics. To enter commands, open the Script Editor by choosing **Edit Script** from the component’s pop-up menu. The script you enter can contain any S-PLUS code, with calls to `plot()`, `points()`, etc. to generate one or more graphs. The inputs and outputs and static variables are specified on the Input Variables and Output Variables tabs of the Properties dialog box, just as they are in the S-PLUS Script component (see below). If the script is totally empty, by default the script is defined as `plot(in0)`. The variable `graphsheet.name` can be accessed within the script to get the name of the graph sheet. When you are finished editing the script, choose **Close & Return** from the Script Editor’s **File** menu.

Note Normally, the plot is cleared before each call to the script to update the graph. If the check “Clear plot before each call” on the Graph Type page of the Properties dialog box is removed, this is not done. In this way you can collect multiple plots in different pages.

S-PLUS Script component

Pro The S-PLUS Script component allows you to write and execute S-PLUS 4.5 or higher programs and link them to other computations in Mathcad Professional.

Inserting an S-PLUS Script component

To insert an S-PLUS Script component into a Mathcad worksheet:

- Click in a blank spot in your worksheet. If you want to send values to the component from a Mathcad variable, click below or to the right of the variable definition.
- Choose **Component** from the **Insert** menu.
- Select S-PLUS Script from the list and click “Finish.” The S-PLUS script component appears as an empty box containing the words “S-PLUS Script.”

- In the placeholder that appears at the bottom, enter the name of the Mathcad input variable to pass into the component. In the placeholder that appears to the left of the component, enter the name of the Mathcad output variable to be defined.
-

Note By default, inputs are referred to in your script as variables named **in0**, **in1**, **in2**, and so on, and outputs as **out0**, **out1**, **out2**, and so on. To change these names, choose **Properties** from the component's pop-up menu and type in new names in the Input Variables and Output Variables tabs.

Editing the script

To enter or edit your S-PLUS code, open the Script Editor by clicking the right mouse button on the component and choosing **Edit Script** from the component's pop-up menu.

Your S-PLUS code, which can include multiple statements, is inserted within an automatically created function to run the component. This code can use temporary variables, but any temporary variables are not visible outside of the component (unless **assign** is called to change the S-PLUS databases). For example, a script could contain **out0<-sin(in0)** to set the output to the sine of the input.

Note For details on the script syntax, see the documentation and on-line Help accompanying S-PLUS.

Tip To declare static variables, which keep their values between calls to the script, add the variable names (separated by commas) to the Static Variable Names field on the Input Variables tab of the component's Properties dialog box. All static variables are reset whenever the script itself is changed, however.

The following variables can be accessed within a script:

- **input.var.names** and **output.var.names** are vectors of strings that give the input and output variable names for the component. You can use these variables to write scripts that handle different numbers of inputs and outputs.
- **first.call** has a value of T if this is the first time this script has been executed. This variable can be used to initialize static variables.

When you are finished editing the script, choose **Close & Return** from the Script Editor's **File** menu. When you click outside the component, the script is executed. If the text of an S-PLUS program cannot be parsed (for example, if it contains an unmatched parenthesis), or if an error occurs when it is executed, a popup window appears.

Note If “Show Captured Results” on the S-PLUS Script component’s pop-up menu is checked, the component displays the text produced by the last execution of the S-PLUS program, including any printing done by the script, as if it had been executed at the S-PLUS command line. Capturing this text can slow down the execution. Error messages are always captured, however, so after an error occurs the display can be switched to see the error message.

Excel component

Pro The Excel component allows you to exchange data with and access the features of Microsoft Excel (version 7 or higher), if it is installed on your system.

Tip If you only need to import or export a static data file in Excel format, use the File Read/Write component as described in Chapter 11, “Vectors, Matrices, and Data Arrays.” While an Excel component may be derived from an existing Excel file, it cannot change the content of that file.

Inserting an Excel component

To insert an Excel component into a Mathcad worksheet:

- Click in a blank spot in your worksheet. If you want to send values to the component from a Mathcad variable defined in your worksheet, click below or to the right of the variable definition.
- Choose **Component** from the **Insert** menu.
- Select Excel from the list and click “Next.” To create an object based on a file you’ve already created, choose “Create from file,” and type the path name in the text box or use the Browse button to locate the file; then click “Open.” Otherwise, choose “Create an empty Excel Worksheet.”
- Click Display as Icon if you want to see an icon in your Mathcad worksheet rather than a portion of the Excel file.

Successive pages of the Wizard allow you to specify:

- **The number of input and output variables.** Supply between 0 and 4 input variables and between 0 and 4 output variables.
- **Input ranges.** The cells in which the values of each input variable from Mathcad will be stored. Enter the starting cell, which is the cell that will hold the element in the upper left corner of an input array. For example, for an input variable containing a 3×3 matrix of values, you can specify A1 as the starting cell, and the values will be placed in cells A1 through C3.
- **Output ranges.** The cells whose values will define the output variables in Mathcad. For example, enter C2:L11 to extract the values in cells C2 through L11 and create a 10×10 matrix.

When you finish using the Wizard, the Excel component appears in your worksheet with placeholders for the input and output variables. Enter the names of input variables in the bottom placeholders. Enter the output variables into the placeholders to the left of the :=. When you click outside the component, input variables are sent to Excel from Mathcad and a range of cells are returned to Mathcad.

Figure 16-2 shows an example of an Excel component in a Mathcad worksheet.

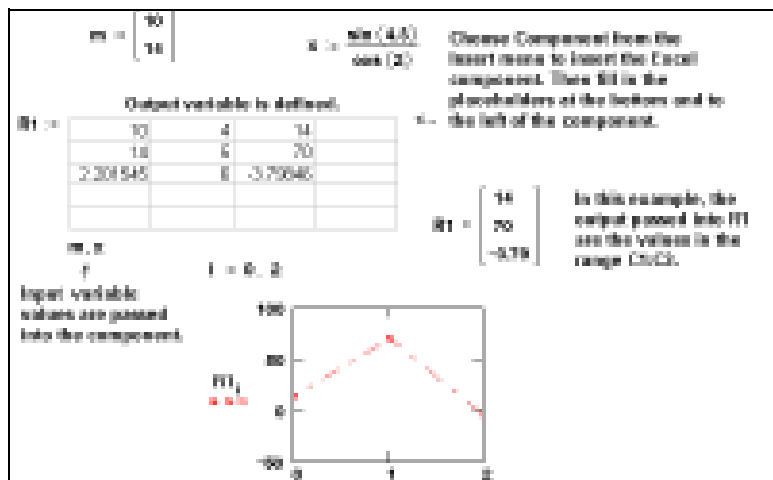


Figure 16-2: Linking an Excel spreadsheet and a Mathcad worksheet.

Note By default, the Excel component displays only some of the rows and columns of the underlying spreadsheet. To see more or fewer rows and columns, double-click the component so that you see handles along the sides of the component. Then resize the component by dragging a handle.

Changing the inputs and outputs

If you add input or output variables, you need to specify which cells in the component will store the new input and which will provide the new output. To do so:

- Click on the component with the right mouse button and choose **Properties** from the pop-up menu.
- Choose the Inputs tab or the Outputs tab and specify a range of cells for each input and each output.

You should also follow these steps if you want to change the cell ranges for inputs and outputs you initially specified in the Setup Wizard.

Accessing Excel

After inserting an Excel component into a Mathcad worksheet, you can use the component to perform calculations in Excel, provided you have installed Excel on your system. To do so:

- Double-click the Excel component in the Mathcad worksheet. The Excel component opens and the menus and toolbars change to Excel's menus and toolbars.
- Edit the Excel component however you'd like.
- Click back in the Mathcad worksheet to have the component recalculate and to resume working in Mathcad.

MATLAB component

Pro The MATLAB component allows you to exchange data with and access the programming environment of The MathWorks' MATLAB Professional 4.2c or higher, if it is installed on your system.

Tip If you only need to import or export a static data file in MATLAB format, use the File Read/Write component as described in Chapter 11, "Vectors, Matrices, and Data Arrays."

Inserting a MATLAB component

To insert a MATLAB component into a Mathcad worksheet:

- Click in a blank spot in your worksheet. If you want to send values to the MATLAB component from a Mathcad variable, click below or to the right of the variable definition.
- Choose **Component** from the **Insert** menu.
- Select MATLAB from the list and click "Next." The MATLAB component is inserted into your worksheet.
- In the placeholder that appears at the bottom, enter the name of the Mathcad input variable to pass into the MATLAB component. In the placeholder that appears to the left of the component, enter the name of the Mathcad output variable to be defined.

Note By default, the data in the Mathcad input variables are sent into MATLAB variables named **in0**, **in1**, **in2**, and so on. The MATLAB variables **out0**, **out1**, **out2**, and so on define the data to be passed to the Mathcad output variables. To change these names, choose **Properties** from the component's pop-up menu and type in new names in the Inputs and Outputs tabs.

Accessing MATLAB

To use the MATLAB component to perform calculations in MATLAB:

- Double-click the MATLAB component in the Mathcad worksheet. The MATLAB component opens a text window for entering MATLAB commands.
- Edit the MATLAB script however you'd like. Be sure to use the appropriate MATLAB variable names to take the input from Mathcad and provide the output.


When you click outside the component, input variables from Mathcad are sent to MATLAB, and arrays from MATLAB are assigned to output variables in Mathcad.

Note Some versions of MATLAB support multidimensional arrays and other complex data structures. While you may use these structures within the MATLAB component, you may pass only scalars, vectors, and two-dimensional arrays from Mathcad to the MATLAB component and vice versa.

MathConnex

Pro The components available in Mathcad are used to connect a Mathcad worksheet to other data sources and applications. If you have Mathcad Professional, you can use the MathConnex application to connect these data sources and applications to *each other* as well as to Mathcad.

In addition to the components available in Mathcad Professional, MathConnex contains a number of other components for manipulating data, such as a Mathcad component for connecting to a Mathcad worksheet. The MathConnex environment lets you connect any of one of the available components to any other component. MathConnex is therefore a tool for controlling data as it flows from one data source or application to another. You can visually design systems of data flow to analyze projects which involve a variety of applications and data sources.

To run MathConnex, click  on the Standard toolbar, or exit Mathcad and run MathConnex as you would any application. For more information about using MathConnex, refer to the *MathConnex User's Guide*.

Scripting custom OLE automation objects

Pro As described in the previous section, Mathcad has several specialized components for using the functionality of other technical computing environments within your Mathcad worksheet. However, you can dynamically exchange data between a Mathcad worksheet and any other application that supports OLE Automation, even if Mathcad does not have a specific component to do so. To do so, use the *Scriptable Object component*. You can create a custom scriptable object from any object you can insert into a Mathcad worksheet.

To create a Scriptable Object component, you must:

- Be proficient in a supported scripting language, such as Microsoft VBScript or JScript, that is installed on your system.
- Have some knowledge of the way the other application has implemented OLE.
- Have the other application installed on your system.

Scripting languages

To use a Scriptable Object component, you must have a supported scripting language installed on your system. As this *User's Guide* goes to press, the following two scripting languages are supported: Microsoft VBScript (Visual Basic Scripting Edition) and Microsoft JScript (an implementation of JavaScript). Both of these scripting languages are included with Microsoft Internet Explorer, which can be installed from the Mathcad installation media. These scripting languages can also be downloaded at no charge from Microsoft, Inc. at:

<http://www.microsoft.com/scripting>

Note VBScript is a strict *subset* of the Visual Basic for Applications language used in Microsoft Excel, Project, Access, and the Visual Basic development system. VBScript is designed to be a lightweight interpreted language, so it does not use strict types (only Variants). Also, because VBScript is intended to be a safe subset of the language, it does not include file input/output or direct access to the underlying operating system. JScript is a fast, portable, lightweight interpreter for use in applications that use ActiveX controls, OLE automation servers, and Java applets. JScript is directly comparable to VBScript (not Java). Like VBScript, JScript is a pure interpreter that processes source code rather than producing stand-alone applets. The syntax and techniques used in the scripting language you choose are beyond the scope of this *User's Guide*.

Inserting a Scriptable Object

To insert a Scriptable Object component into a Mathcad worksheet:

- Click in a blank spot in your worksheet. If you want to send values to the object from a Mathcad variable, click below or to the right of the variable definition.
- Choose **Component** from the **Insert** menu.
- Select Scriptable Object from the list in the Wizard and click “Next.”

This launches the Scripting Wizard. The Object to Script scrolling list shows the available server applications on your system. Choose an application that supports the OLE 2 automation interface (consult the documentation for the application for details).

You must specify:

- Whether the component is a new file or whether you will insert an existing file.
- Whether you will see the actual file or an icon in your Mathcad worksheet.

In the remaining pages of the Wizard you specify: the scripting language you are using, the type of object you want to script, the name of the object, and the number of inputs and outputs the object will accept and provide.

A Scriptable Object component appears in your worksheet with placeholders for the input and output variables. Enter the input variables in the bottom placeholders. Enter the output variables into the placeholders to the left of the :=.

Object model

The Scriptable Object component has the following predefined objects, properties, and methods that enable you to configure it to work as a component in Mathcad.

Collections

- **Inputs** and **Outputs** are predefined *collections* of *DataValue* objects (see below) containing the Scriptable Object's inputs and the outputs, respectively.
- The **Count** property can be used to query the total number of elements in the collection. For example, **Outputs.Count** returns the number of output variables.
- The **Item** method is used to specify an individual element in the collection. To refer to a particular input or output, use the notation **Inputs.Item(*n*)** or **Outputs.Item(*n*)**, where *n* is the index of the input or output. The index *n* always starts from 0. Since **Item** is the default method, languages such as VBScript and JScript let you drop the method name to imply the default method. For example, **Inputs(0)** is equivalent to **Inputs.Item(0)** and references the first input.

DataValue objects

- The **Value** property accesses a *DataValue*'s real part. For example, in VBScript or JScript **Inputs(0).Value** returns the real part of the first input.
- The **IValue** property accesses a *DataValue*'s imaginary part. For example, in VBScript or JScript **Outputs(1).IValue** returns the imaginary part of the second output. If there is no imaginary part, the **IValue** portion returns "NIL."
- The **IsComplex** property returns "TRUE" if a *DataValue* has a valid imaginary part; this property returns "FALSE" otherwise. For example, the expression (**inputs(0).IsComplex**) returns "FALSE" if the first input has only a real part.
- The **Rows** and **Cols** properties yield the number of rows and columns.

Global methods

- The **alert** function takes a single string parameter that is presented to the user as a standard modal Windows message box with an "OK" button.
- The **errmsg** function takes a string parameter that appears as an error message from within the script and causes the script to stop execution. A second, optional parameter is a string used to display the source of the error.

Note In JScript, the names of functions, methods, objects, and properties are case sensitive, while in VBScript they are not.

Scripting the object

To script an object, click once on the component to select it. Then click on the component with the right mouse button and choose **Edit Script** from the pop-up menu.

You'll see a Script Editor window containing three subroutine stubs in which you insert your own scripting code.

The script you write will usually contain at a minimum the following three subroutines:

- A *starting* routine, called once when execution of the component begins. This is a good place to initialize variables, open files for reading and writing, etc.
- An *execution* routine that by default takes as arguments the collections **Inputs** and **Outputs**.
- A *stopping* routine, called once when execution of the component stops.

The commands in each section are executed in sequence whenever the Mathcad worksheet is calculated. What you include in these subroutines is determined largely by the properties of the OLE object you are scripting; consult the documentation for the server or control.

Choose **Close & Return** from the Script Editor's **File** menu when you have completed your script and want to resume working in the Mathcad worksheet.

Using Mathcad as an OLE automation server

The previous section describes how to script a custom OLE object in Mathcad. Mathcad's OLE automation interface provides a mechanism for the complementary process of using Mathcad as an automation server from *within* another Windows application. Using Mathcad's OLE automation interface, you can send data dynamically to Mathcad from another application, use Mathcad to perform calculations or other data manipulations, and send results back to the original application.

Note The OLE automation interface is supported in Mathcad 7.02 and higher and supersedes the DDE interface supported in Mathcad 5 and 6. For current information on and examples of the interface, visit the MathSoft Web site at <http://www.mathsoft.com/support/>.

Mathcad Document object model

To use the OLE automation interface, you must write a program in Visual Basic 5.0 or higher or in an application that can serve as an OLE automation client, such as Microsoft Excel 5.0 and higher. (Check with the application's documentation to find out whether it is an OLE automation client.) You use the program to define and retrieve variables in *Mathcad Document objects*. The variables being defined in Mathcad must be named **in0**, **in1**, **in2**, etc. The variables being retrieved from Mathcad must be named **out0**, **out1**, **out2**, etc.

Automation methods

- **GetComplex**(*Name*, *RealPart*, *ImagPart*) retrieves complex data (real and imaginary parts) from Mathcad variable *Name*, where *Name* is one of **out0**, **out1**, **out2**, etc.
- **SetComplex**(*Name*, *RealPart*, *ImagPart*) assigns complex data (real and imaginary parts) to Mathcad variable *Name*, where *Name* is one of **in0**, **in1**, **in2**, etc.
- **Recalculate** triggers recalculation of the Mathcad document.
- **SaveAs**(*Name*) saves the Mathcad document as a file whose path is given by the string *Name*.

Scripting the object

The specific procedures required to script a Mathcad OLE automation object differ slightly depending on the application that serves as an OLE automation client, but the main steps are as follows:

- Provide or create a Mathcad OLE object with which to communicate.
- Set up the client application with the data to send to Mathcad and/or with locations to put the data retrieved from Mathcad.
- Write code to specify the data to send to Mathcad and/or the data to retrieve.

The following example shows how to use Microsoft Excel's VBA environment to assign a complex number to the variable **in0** in a Mathcad OLE object, trigger a calculation in Mathcad, and return the answer to Excel:

- Insert an OLE object into an Excel worksheet. For details, see Excel's documentation and on-line Help.
- Set up Excel so that there is data to pass to Mathcad and/or available cells to store data returned from Mathcad.
- Write a Visual Basic macro module. Refer to Excel's on-line Help for information.
- Double-click the Mathcad object in the Excel worksheet to activate the linked object. Then run the Excel macro.

Figure 16-3 shows how data stored in cells G7 through H8 are passed into the Mathcad variable **in0**. Mathcad performs a calculation (in this simple example, it trivially adds 1 to the values), and the results stored in the Mathcad variable **out0** are returned to cells G12 through H13. The VBA macro looks like this:

```
Sub UpdateWorksheet()  
    Dim MathcadObject As Object  
    Dim outRe, outIm As Variant  
    Dim inRe, inIm As Variant  
    ' Get a reference to the Mathcad object  
    Set MathcadObject = ActiveSheet.OLEObjects(1).Object  
    ' Read in values to be passed from Excel to Mathcad  
    inRe = ActiveSheet.Range("G7:G8").Value
```

```

inIm = ActiveSheet.Range("H7:H8").Value
' Send the values over to Mathcad, assign them to variable in0, and recalculate
Call MathcadObject.SetComplex("in0", inRe, inIm)
Call MathcadObject.Recalculate
' Place the result values into the chosen Excel cells
Call MathcadObject.GetComplex("out0", outRe, outIm)
ActiveSheet.Range("G12:G13").Value = outRe
ActiveSheet.Range("H12:H13").Value = outIm
End Sub

```

Note To pass real data into Mathcad, you should have cell(s) containing zeros for the imaginary part.

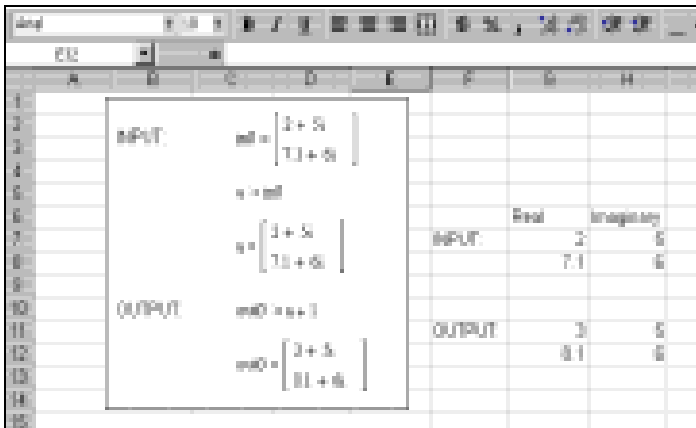


Figure 16-3: Mathcad as an OLE automation server within Microsoft Excel.