

Mathcad

User's Guide

Mathcad 2000 Professional

Mathcad 2000 Standard

MathSoft, Inc.
101 Main Street
Cambridge
Massachusetts 02142
USA
<http://www.mathsoft.com/>

MathSoft
 $\Sigma + \sqrt{\quad} - = \times \int \div \delta$

MathSoft, Inc. owns both the Mathcad software program and its documentation. Both the program and documentation are copyrighted with all rights reserved by MathSoft. No part of this publication may be produced, transmitted, transcribed, stored in a retrieval system, or translated into any language in any form without the written permission of MathSoft, Inc.

U.S. Patent Numbers 5,526,475 and 5,468,538.

See the License Agreement and Limited Warranty for complete information.

English spelling software by Lernout & Haspie Speech Products, N.V.

MKM developed by Waterloo Maple Software.

VoloView Express technology, Copyright © 1999 Autodesk, Inc. All Rights Reserved.

The Mathcad Collaboratory is powered by O'Reilly WebBoard, Copyright © 1995-1999 Duke Engineering/O'Reilly & Associates, Inc.

Copyright © 1986-1999 MathSoft, Inc. All rights reserved.

MathSoft, Inc.
101 Main Street
Cambridge, MA 02142
USA

Mathcad, *Axum*, and *S-PLUS* are registered trademarks of MathSoft, Inc. *Electronic Book*, *QuickSheets*, *MathConnex*, *ConnexScript*, *Collaboratory*, *IntelliMath*, *Live Symbolics*, and the MathSoft logo are trademarks of MathSoft, Inc.

Microsoft, *Windows*, *IntelliMouse*, and the Windows logo are registered trademarks of Microsoft Corp. *Windows NT* is a trademark of Microsoft Corp.

OpenGL is a registered trademark of Silicon Graphics, Inc.

MATLAB is a registered trademark of The MathWorks, Inc.

SmartSketch is a registered trademark of Intergraph Corporation.

Other brand and product names referred to are trademarks or registered trademarks of their respective owners.

Printed in the United States of America. First printing: August 1999

Contents

How to Use This User's Guide	1
The Basics	
1: Welcome to Mathcad	3
What Is Mathcad?	3
Mathcad Editions	4
New in Mathcad 2000	4
System Requirements	5
Installation	5
Contacting MathSoft	6
2: Getting Started with Mathcad	7
The Mathcad Workspace	7
Regions	10
A Simple Calculation	12
Definitions and Variables	13
Entering Text	14
Iterative Calculations	15
Graphs	17
Saving, Printing, and Exiting	19
3: On-Line Resources	21
Resource Center and Electronic Books	21
Help	26
Internet Access in Mathcad	27
The Collaboratory	28
Other Resources	32
Creating Mathcad Worksheets	
4: Working with Math	33
Inserting Math	33
Building Expressions	39
Editing Expressions	43
Math Styles	51
5: Working with Text	55
Inserting Text	55
Text and Paragraph Properties	58
Text Styles	61
Equations in Text	63
Text Tools	64
6: Working with Graphics and Other Objects	67
Overview	67
Inserting Pictures	67

Inserting Objects	71
Inserting Graphics Computationally Linked to Your Worksheet	74
7: Worksheet Management	75
Worksheets and Templates	75
Rearranging Your Worksheet	79
Layout	83
Safeguarding an Area of the Worksheet	85
Hyperlinks	87
Creating an Electronic Book	89
Printing and Mailing	93
Computational Features	
8: Calculating in Mathcad	97
Defining and Evaluating Variables	97
Defining and Evaluating Functions	104
Units and Dimensions	106
Working with Results	110
Controlling Calculation	117
Animation	119
Error Messages	121
9: Operators	123
Working with Operators	123
Arithmetic and Boolean Operators	125
Vector and Matrix Operators	127
Summations and Products	130
Derivatives	133
Integrals	136
Customizing Operators	140
10: Built-in Functions	143
Inserting Built-in Functions	143
Core Mathematical Functions	145
Discrete Transform Functions	150
Vector and Matrix Functions	152
Solving and Optimization Functions	157
Statistics, Probability, and Data Analysis Functions	163
Finance Functions	173
Differential Equation Functions	176
Miscellaneous Functions	187
11: Vectors, Matrices, and Data Arrays	191
Creating Arrays	191
Accessing Array Elements	196
Displaying Arrays	198
Working with Arrays	201
Nested Arrays	204

12: 2D Plots	207
Overview of 2D Plotting	207
Graphing Functions and Expressions	209
Plotting Vectors of Data	212
Formatting a 2D Plot	216
Modifying Your 2D Plot's Perspective	220
13: 3D Plots	223
Overview of 3D Plotting	223
Creating 3D Plots of Functions	224
Creating 3D Plots of Data	228
Formatting a 3D Plot	234
Rotating and Zooming on 3D Plots	243
14: Symbolic Calculation	245
Overview of Symbolic Math	245
Live Symbolic Evaluation	246
Using the Symbolics Menu	254
Examples of Symbolic Calculation	255
Symbolic Optimization	265
15: Programming	267
Defining a Program	267
Conditional Statements	269
Looping	271
Controlling Program Execution	272
Error Handling	274
Programs Within Programs	276
16: Advanced Computational Features	279
Worksheet References	279
Exchanging Data with Other Applications	280
Scripting Custom OLE Automation Objects	292
Accessing Mathcad from Within Another Application	294
Appendices	297
Operators	298
Symbolic Transformation Functions	301
SI Units	303
CGS units	305
U.S. Customary Units	307
MKS Units	309
Predefined Variables	311
Suffixes for Numbers	312
Greek Letters	313
Arrow and Movement Keys	314
Function Keys	315
ASCII codes	316
Index	317

How to Use This *User's Guide*

This *User's Guide* is organized into the following parts:

τ The Basics

This section contains a quick introduction to Mathcad's features and workspace, including resources available in the product and on the Internet for getting more out of Mathcad. Be sure to read this section first if you are a new Mathcad user.

τ Creating Mathcad Worksheets

This section describes in more detail how to create and edit Mathcad worksheets. It leads you through editing and formatting equations, text, and graphics, as well as opening, editing, saving, and printing Mathcad worksheets and templates.

τ Computational Features

This section describes how Mathcad interprets equations and explains Mathcad's computational features: units of measurement, complex numbers, matrices, built-in functions, solving equations, programming, and so on. This section also describes how to do symbolic calculations and how to use Mathcad's two- and three-dimensional plotting features.

The *User's Guide* ends with reference appendices and a comprehensive index.

As far as possible, the topics in this guide are described independently of each other. This means that once you are familiar with the basic workings of Mathcad, you can just select a topic of interest and read about it.

The on-line Mathcad Resource Center (choose **Resource Center** from the **Help** menu) provides step by step tutorials, examples, and application files that you can use directly in your own Mathcad worksheets. Mathcad QuickSheets are templates available in the Resource Center that provide live examples that you can manipulate.

Notations and Conventions

This *User's Guide* uses the following notations and conventions:

Italics represent scalar variable names, function names, and error messages.

Bold Courier represents keys you should type.

Bold represents a menu command. It is also used to denote vector and matrix valued variables.

An arrow such as that in "**Graph⇒X-Y Plot**" indicates a pull-right menu command.

Function keys and other special keys are enclosed in brackets. For example, [↑], [↓], [←], and [→] are the arrow keys on the keyboard. [F1], [F2], etc., are function keys; [BkSp] is the Backspace key for backspacing over characters; [Del] is the Delete key for deleting characters to the right; [Ins] is the Insert key for inserting characters to the left of the insertion point; [Tab] is the Tab key; and [Space] is the space bar.

[Ctrl], [Shift], and [Alt] are the Control, Shift, and Alt keys. When two keys are shown together, for example, [Ctrl]V, press and hold down the first key, and then press the second key.

The symbol [↵] and [Enter] refer to the same key.

When this *User's Guide* shows spaces in an equation, you need not type the spaces. Mathcad automatically spaces the equation correctly.

Pro This *User's Guide* applies to Mathcad 2000 Professional and Mathcad 2000 Standard. If you're not using Mathcad 2000 Professional, certain features described in the *User's Guide* will not be available to you. The word Pro appears:

- In the page margin, as it does above, whenever a section in a chapter describes a feature or a function that is unique to Mathcad 2000 Professional.
- In the page footer, whenever all features described in that chapter are unique to Mathcad 2000 Professional.

This *User's Guide* also describes a few product features that are available only in add-on packages for Mathcad. For example, some numerical solving features and functions are provided only in the *Solving and Optimization Extension Pack (Expert Solver)*.

Chapter 2

Getting Started with Mathcad

- τ The Mathcad Workspace
- τ Regions
- τ A Simple Calculation
- τ Definitions and Variables
- τ Entering Text
- τ Iterative Calculations
- τ Graphs
- τ Saving, Printing, and Exiting

The Mathcad Workspace

For information on system requirements and how to install Mathcad on your computer, refer to Chapter 1, “Welcome to Mathcad.”

When you start Mathcad, you’ll see a window like that shown in Figure 2-1. By default the worksheet area is white. To select a different color, choose **Color**⇒**Background** from the **Format** menu.

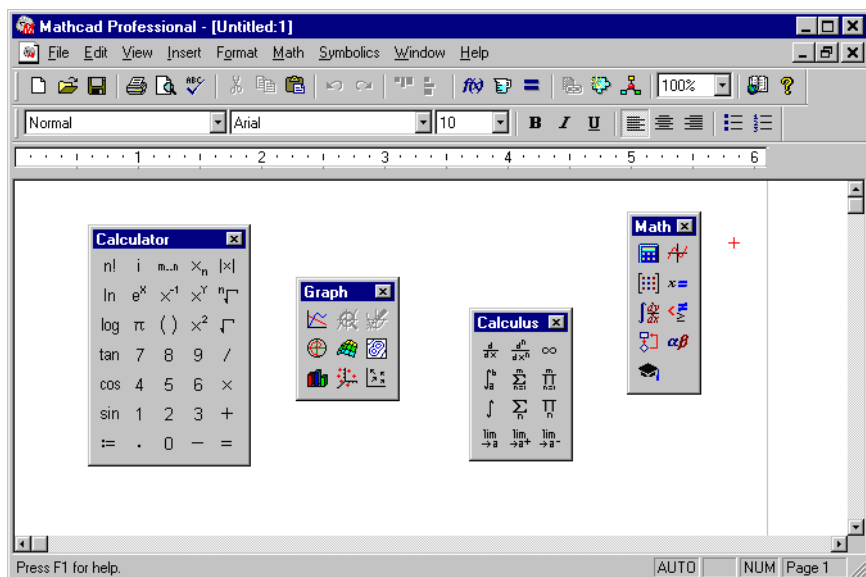


Figure 2-1: Mathcad Professional with various toolbars displayed.

Each button in the **Math toolbar**, shown in Figure 2-1, opens another toolbar of operators or symbols. You can insert many operators, Greek letters, and plots by clicking the buttons found on these toolbars:

Button Opens math toolbar...



Calculator—Common arithmetic operators.



Graph—Various two- and three-dimensional plot types and graph tools.



Matrix—Matrix and vector operators.



Evaluation—Equal signs for evaluation and definition.



Calculus—Derivatives, integrals, limits, and iterated sums and products.



Boolean—Comparative and logical operators for Boolean expression.



Programming—Programming constructs (*Mathcad Professional only*).



Greek—Greek letters.



Symbolic—Symbolic keywords.

The **Standard toolbar** is the strip of buttons shown just below the main menus in Figure 2-1:



Many menu commands can be accessed more quickly by clicking a button on the Standard toolbar.

The **Formatting toolbar** is shown immediately below the Standard toolbar in Figure 2-1. This contains scrolling lists and buttons used to specify font characteristics in equations and text.



Tip To learn what a button on any toolbar does, let the mouse pointer rest on the button momentarily. You'll see a tooltip beside the pointer giving a brief description.

To conserve screen space, you can show or hide each toolbar individually by choosing the appropriate command from the **View** menu. You can also detach and drag a toolbar around your window. To do so, place the mouse pointer anywhere other than on a button or a text box. Then press and hold down the mouse button and drag. You'll find that the toolbars rearrange themselves appropriately depending on where you drag them. And Mathcad remembers where you left your toolbars the next time you open the application.

Tip The Standard, Formatting, and Math toolbars are customizable. To add and remove buttons from one of these toolbars, click with the right mouse button on the toolbar and choose **Customize** from the pop-up menu to bring up the Customize Toolbar dialog box.

The **worksheet ruler** is shown towards the top of the screen in Figure 2-1. To hide or show the ruler, choose **Ruler** from the **View** menu. To change the measurement system used in the ruler, click on the ruler with the right mouse button, and choose **Inches**, **Centimeters**, **Points**, or **Picas** from the pop-up menu. For more information on using the ruler to format your worksheet, refer to “Using the worksheet ruler” on page 79.

Working with Windows

When you start Mathcad, you open up a window on a Mathcad *worksheet*. You can have as many worksheets open as your available system resources allow. This allows you to work on several worksheets at once by simply clicking the mouse in whichever document window you want to work in.

There are times when a Mathcad worksheet cannot be displayed in its entirety because the window is too small. To bring unseen portions of a worksheet into view, you can:

- Make the window larger as you do in other Windows applications.
- Choose **Zoom** from the **View** menu or click  on the Standard toolbar and choose a number smaller than 100%.

You can also use the scroll bars, mouse, and keystrokes to move around the Mathcad window, as you can in your other Windows applications. When you move the mouse pointer and click the mouse button, for example, the cursor jumps from wherever it was to wherever you clicked.

Tip Mathcad supports the Microsoft IntelliMouse and compatible pointing devices. Turning the wheel scrolls the window one line vertically for each click of the wheel. When you press **[Shift]** and turn the wheel, the window scrolls horizontally.

See “Arrow and Movement Keys” on page 314 in the Appendices for keystrokes to move the cursor in the worksheet. If you are working with a longer worksheet, choose **Go to Page** from the **Edit** menu and enter the page number you want to go to in the dialog box. When you click “OK,” Mathcad places the top of the page you specify at the top of the window.

Tip Mathcad supports standard Windows keystrokes for operations such as file opening, **[Ctrl]O**, saving, **[Ctrl]S**, printing, **[Ctrl]P**, copying, **[Ctrl]C**, and pasting, **[Ctrl]V**. Choose **Preferences** from the **View** menu and check “Standard Windows shortcut keys” in the Keyboard Options section of the General tab to enable all Windows shortcuts. Remove the check to use shortcut keys supported in earlier versions of Mathcad.

Regions

Mathcad lets you enter equations and text anywhere in the worksheet. Each equation, piece of text, or other element is a *region*. Mathcad creates an invisible rectangle to hold each region. A Mathcad worksheet is a collection of such regions. To start a new region in Mathcad:

1. Click anywhere in a blank area of the worksheet. You see a small crosshair. Anything you type appears at the crosshair. 
2. If the region you want to create is a math region, just start typing anywhere you put the crosshair. By default Mathcad understands what you type as mathematics. See “A Simple Calculation” on page 12 for an example.
3. To create a text region, first choose **Text Region** from the **Insert** menu and then start typing. See “Entering Text” on page 14 for an example.

In addition to equations and text, Mathcad supports a variety of plot regions. See “Graphs” on page 17 for an example of inserting a two-dimensional plot.

Tip Mathcad displays a box around any region you are currently working in. When you click outside the region, the surrounding box disappears. To put a permanent box around a region, click on it with the right mouse button and choose **Properties** from the pop-up menu. Click on the Display tab and click the box next to “Show Border.”

Selecting Regions

To select a single region, simply click it. Mathcad shows a rectangle around the region.

To select multiple regions:

1. Press and hold down the left mouse button to anchor one corner of the selection rectangle.
2. Without letting go of the mouse button, move the mouse to enclose everything you want to select inside the selection rectangle.
3. Release the mouse button. Mathcad shows dashed rectangles around regions you have selected.

Tip You can also select multiple regions anywhere in the worksheet by holding down the [Ctrl] key while clicking. If you click one region and [Shift]-click another, you select both regions and all regions in between.

Moving and Copying Regions

Once the regions are selected, you can move or copy them.

Moving regions

You can move regions by dragging with the mouse or by using **Cut** and **Paste**.

To drag regions with the mouse:

1. Select the regions as described in the previous section.
2. Place the pointer on the border of any selected region. The pointer turns into a small hand.

3. Press and hold down the mouse button.
4. Without letting go of the button, move the mouse. The rectangular outlines of the selected regions follow the mouse pointer.

At this point, you can either drag the selected regions to another spot in the worksheet, or you can drag them to another worksheet. To move the selected regions into another worksheet, press and hold down the mouse button, drag the rectangular outlines into the destination worksheet, and release the mouse button.

To move the selected regions by using **Cut** and **Paste**:

1. Select the regions as described in the previous section.
2. Choose **Cut** from the **Edit** menu (keystroke: **[Ctrl] X**), or click  on the Standard toolbar. This deletes the selected regions and puts them on the Clipboard.
3. Click the mouse wherever you want the regions moved to. Make sure you've clicked in an empty space. You can click either someplace else in your worksheet or in a different worksheet altogether. You should see the crosshair.
4. Choose **Paste** from the **Edit** menu (keystroke: **[Ctrl] V**), or click  on the Standard toolbar.

Note You can move one region on top of another. If you do, you can move a particular region to the top or bottom by clicking on it with the right mouse button and choosing **Bring to Front** or **Send to Back** from the pop-up menu.

Copying Regions

You copy regions by using the **Copy** and **Paste** commands:

1. Select the regions as described in “Selecting Regions” on page 10.
2. Choose **Copy** from the **Edit** menu (keystroke: **[Ctrl] C**), or click  on the Standard toolbar. This copies the selected regions to the Clipboard.
3. Click the mouse wherever you want to place a copy of the regions. You can click either someplace else in your worksheet or in a different worksheet altogether. Make sure you've clicked in an empty space. You should see the crosshair.
4. Choose **Paste** from the **Edit** menu (keystroke: **[Ctrl] V**), or click  on the Standard toolbar.

Tip If the regions you want to copy are coming from a locked area (see “Safeguarding an Area of the Worksheet” on page 85) or an Electronic Book, you can copy them simply by dragging them with the mouse into your worksheet.

Deleting Regions

To delete one or more regions:

1. Select the regions as described in “Selecting Regions” on page 10.

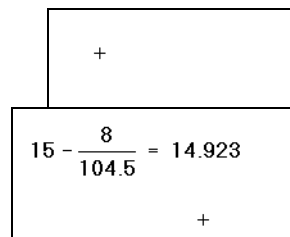
2. Choose **Cut** from the **Edit** menu (keystroke: **[Ctrl] X**), or click  on the Standard toolbar.

Choosing **Cut** removes the selected regions from your worksheet and puts them on the Clipboard. If you don't want to disturb the contents of your Clipboard, or if you don't want to save the selected regions, choose **Delete** from the **Edit** menu (Keystroke: **[Ctrl] D**) instead.

A Simple Calculation

Although Mathcad can perform sophisticated mathematics, you can just as easily use it as a simple calculator. To try your first calculation, follow these steps:

1. Click anywhere in the worksheet. You see a small crosshair. Anything you type appears at the crosshair.
2. Type **15 - 8/104.5 =** . When you type the equal sign or click  on the Evaluation toolbar, Mathcad computes and shows the result.


$$15 - \frac{8}{104.5} = 14.923$$

This calculation demonstrates the way Mathcad works:

- Mathcad shows equations as you might see them in a book or on a blackboard, expanded fully in two dimensions. Mathcad sizes fraction bars, brackets, and other symbols to display equations the same way you would write them on paper.
- Mathcad understands which operation to perform first. In this example, Mathcad knew to perform the division before the subtraction and displayed the equation accordingly.
- As soon as you type the equal sign or click  on the Evaluation toolbar, Mathcad returns the result. Unless you specify otherwise, Mathcad processes each equation as you enter it. See the section “Controlling Calculation” in Chapter 8 to learn how to change this.
- As you type each operator (in this case, – and /), Mathcad shows a small rectangle called a *placeholder*. Placeholders hold spaces open for numbers or expressions not yet typed. As soon as you type a number, it replaces the placeholder in the expression. The placeholder that appears at the end of the expression is used for unit conversions. Its use is discussed in “Displaying Units of Results” on page 113.

Once an equation is on the screen, you can edit it by clicking in the appropriate spot and typing new letters, digits, or operators. You can type many operators and Greek letters by clicking in the Math toolbars introduced in “The Mathcad Workspace” on page 7. Chapter 4, “Working with Math,” explains in detail how to edit Mathcad equations.

Definitions and Variables

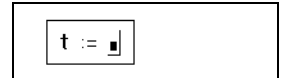
Mathcad's power and versatility quickly become apparent once you begin using *variables* and *functions*. By defining variables and functions, you can link equations together and use intermediate results in further calculations.

The following examples show how to define and use several variables.

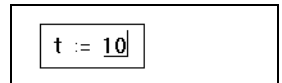
Defining Variables

To define a variable t , follow these steps:

1. Type **t** followed by a colon **:** or click  on the Calculator toolbar. Mathcad shows the colon as the definition symbol **:=**.
2. Type **10** in the empty placeholder to complete the definition for t .



A screenshot of the Mathcad interface showing the definition of a variable t . The text $t :=$ is entered, followed by a small rectangular placeholder box containing a cursor.



A screenshot of the Mathcad interface showing the completed definition of a variable t . The text $t := 10$ is entered, with the number 10 inside the placeholder box.

If you make a mistake, click on the equation and press **[Space]** until the entire expression is between the two editing lines, just as you did earlier. Then delete it by choosing **Cut** from the **Edit** menu (keystroke: **[Ctrl] X**). See Chapter 4, “Working with Math,” for other ways to correct or edit an expression.

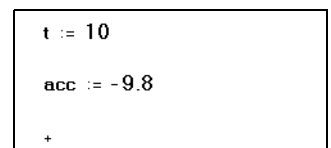
These steps show the form for typing any definition:

1. Type the variable name to be defined.
2. Type the colon key **:** or click  on the Calculator toolbar to insert the definition symbol. The examples that follow encourage you to use the colon key, since that is usually faster.
3. Type the value to be assigned to the variable. The value can be a single number, as in the example shown here, or a more complicated combination of numbers and previously defined variables.

Mathcad worksheets read from top to bottom and left to right. Once you have defined a variable like t , you can compute with it anywhere *below and to the right* of the equation that defines it.

Now enter another definition.

1. Press **[↵]**. This moves the crosshair below the first equation.
2. To define acc as -9.8 , type: **acc : -9.8**. Then press **[↵]** again. Mathcad shows the crosshair cursor below the last equation you entered.

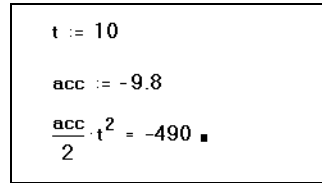


A screenshot of the Mathcad interface showing two definitions. The first line is $t := 10$. The second line is $acc := -9.8$. Below the second line, there is a small rectangular placeholder box containing a crosshair cursor.

Calculating Results

Now that the variables acc and t are defined, you can use them in other expressions.

1. Click the mouse a few lines below the two definitions.
2. Type **acc/2[Space]*t^2**. The caret symbol (^) represents raising to a power, the asterisk (*) is multiplication, and the slash (/) represents division.
3. Press the equal sign (=).



The screenshot shows three lines of text in a Mathcad worksheet. The first line is $t := 10$. The second line is $acc := -9.8$. The third line is $\frac{acc}{2} \cdot t^2 = -490$. The result -490 is displayed in a larger font and bolded.

This equation calculates the distance traveled by a falling body in time t with acceleration acc . When you enter the equation and press the equal sign (=), or click



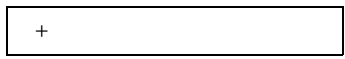
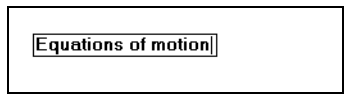
on the Evaluation toolbar, Mathcad returns the result.

Mathcad updates results as soon as you make changes. For example, if you click on the 10 on your screen and change it to some other number, Mathcad changes the result as soon as you press [↵] or click outside of the equation.

Entering Text

Mathcad handles text as easily as it does equations, so you can make notes about the calculations you are doing.

Here's how to enter some text:

1. Click in the blank space to the right of the equations you entered. You'll see a small crosshair.
2. Choose **Text Region** from the **Insert** menu, or press " (the double-quote key), to tell Mathcad that you're about to enter some text. Mathcad changes the crosshair into a vertical line called the insertion point. A box surrounds the insertion point, indicating you are now in a text region. This box is called a text box. It grows as you enter text.
3. Type **Equations of motion**. Mathcad shows the text in the worksheet, next to the equations.

Note If **Ruler** under the **View** menu is checked when the cursor is inside a text region, the ruler resizes to indicate the size of your text region. For more information on using the ruler to set tab stops and indents in a text region, see "Changing Paragraph Properties" on page 59.

Tip If you click in blank space in the worksheet and start typing, which creates a math region, Mathcad automatically converts the math region to a text region when you press [Space].

To enter a second line of text, just press [↵] and continue typing:

1. Press [↵].

2. Then type **for falling body under gravity**.

3. Click in a different spot in the worksheet or press **[Ctrl][Shift][↵]** to move out of the text region. The text box disappears and the cursor appears as a small crosshair.

Equations of motion
for falling body under gravity.

+

Note Use **[Ctrl][Shift][↵]** to move out of the text region to a blank space in your worksheet. If you press **[↵]**, Mathcad inserts a line break in the *current* text region instead.

You can set the width of a text region and change the font, size, and style of the text in it. For more information on how to do these things, see Chapter 5, “Working with Text.”

Iterative Calculations

Mathcad can do repeated or iterative calculations as easily as individual calculations. Mathcad uses a special variable called a *range variable* to perform iteration.

Range variables take on a range of values, such as all the integers from 0 to 10. Whenever a range variable appears in a Mathcad equation, Mathcad calculates the equation not just once, but once for each value of the range variable.

This section describes how to use range variables to do iterative calculations.

Creating a Range Variable

To compute equations for a range of values, first create a range variable. In the problem shown in “Calculating Results” on page 14, for example, you can compute results for a range of values of t from 10 to 20 in steps of 1. To do so, follow these steps:

1. First, change t into a range variable by editing its definition. Click on the **10** in the equation **$t := 10$** . The insertion point should be next to the 10 as shown on the right.

$t := 10|$

2. Type **, 11**. This tells Mathcad that the next number in the range will be 11.

$t := 10, 11|$

3. Type **;** for the range variable operator, or click  on the Calculator toolbar, and then type the last number, **20**. This tells Mathcad that the last number in the range will be 20. Mathcad shows the range variable operator as a pair of dots.

$t := 10, 11 .. 20|$

- Now click outside the equation for t . Mathcad begins to compute with t defined as a range variable. Since t now takes on eleven different values, there must also be eleven different answers. These are displayed in an *output table* as shown at right. You may have to resize your window or scroll down to see the whole table.

$\frac{\text{acc}}{2} \cdot t^2 =$
-490
-592.9
-705.6
-828.1
-960.4
-1.103·10 ³
-1.254·10 ³
-1.416·10 ³
-1.588·10 ³
-1.769·10 ³
-1.96·10 ³

Defining a Function

You can gain additional flexibility by defining functions. Here's how to add a function definition to your worksheet:

- First delete the table. To do so, drag-select the entire region until you've enclosed everything between the two editing lines. Then

choose **Cut** from the **Edit** menu (keystroke: [Ctrl]X) or click  on the Standard toolbar.

- Now define the function $d(t)$ by typing **d(t) :**

$d(t) :=$
$d(t) := 1600 + \frac{\text{acc}}{2} \cdot t^2$

- Complete the definition by typing this expression:
1600+acc/2[Space]*t^2[↵]

The definition you just typed defines a function. The function name is d , and the argument of the function is t . You can use this function to evaluate the above expression for different values of t . To do so, simply replace t with an appropriate number. For example:

- To evaluate the function at a particular value, such as 3.5, type **d(3.5)=**. Mathcad returns the correct value as shown at right.

$d(3.5) = 1.54 \times 10^3$

- To evaluate the function once for each value of the range variable t you defined earlier, click below the other equations and type **d(t)=**. As before, Mathcad shows a table of values, as shown at right.

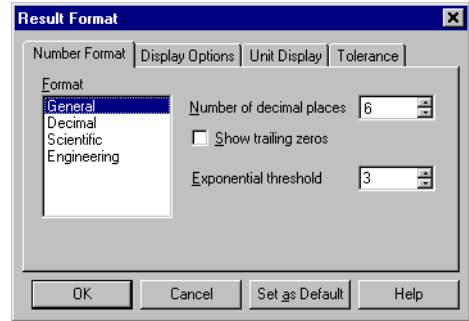
$d(t) =$
1.11·10 ³
1.007·10 ³
894.4
771.9
639.6
497.5
345.6
183.9
12.4
-168.9
-360

Formatting a Result

You can set the display format for any number Mathcad calculates and displays. This means changing the number of decimal places shown, changing exponential notation to ordinary decimal notation, and so on.

For example, in the example above, the first two values, $1.11 \cdot 10^3$ and $1.007 \cdot 10^3$, are in exponential (powers of 10) notation. Here's how to change the table produced above so that none of the numbers in it are displayed in exponential notation:

1. Click anywhere on the table with the mouse.
2. Choose **Result** from the **Format** menu. You see the Result Format dialog box. This box contains settings that affect how results are displayed, including the number of decimal places, the use of exponential notation, the radix, and so on.



3. The default format scheme is General which has Exponential Threshold set to 3. This means that only numbers greater than or equal to 10^3 are displayed in exponential notation. Click the arrows to the right of the 3 to increase the Exponential Threshold to 6.
 4. Click "OK." The table changes to reflect the new result format.
- For more information on formatting results, refer to "Formatting Results" on page 110.

d (t) =
1110
1007.1
894.4
771.9
639.6
497.5
345.6
183.9
12.4
-168.9
-360

Note When you format a result, only the display of the result is affected. Mathcad maintains full precision internally (up to 15 digits).

Graphs

Mathcad can show both two-dimensional Cartesian and polar graphs, contour plots, surface plots, and a variety of other three-dimensional graphs. These are all examples of *graph regions*.

This section describes how to create a simple two-dimensional graph showing the points calculated in the previous section.

Creating a Graph

To create an X-Y plot in Mathcad, click in blank space where you want the graph to appear and choose **Graph**⇒**X-Y Plot** from the **Insert** menu or click  on the Graph toolbar. An empty graph appears with placeholders on the x-axis and y-axis for the expressions to be graphed. X-Y and polar plots are ordinarily driven by range variables you define: Mathcad graphs one point for each value of the range variable used in the

graph. In most cases you enter the range variable, or an expression depending on the range variable, on the x-axis of the plot. For example, here's how to create a plot of the function $d(t)$ defined in the previous section:

1. Position the crosshair in a blank spot and type **$d(t)$** . Make sure the editing lines remain displayed on the expression.

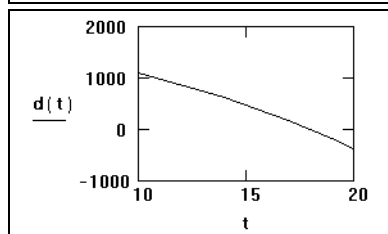
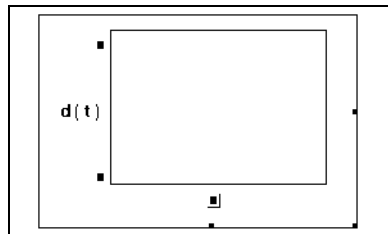
$d(t)$

2. Now choose **Graph**⇒**X-Y Plot** from the

Insert menu, or click  on the Graph toolbar. Mathcad displays the frame of the graph.

3. Type **t** in the bottom middle placeholder on the graph.

4. Click anywhere outside the graph. Mathcad calculates and graphs the points. A sample line appears under the “ $d(t)$.” This helps you identify the different curves when you plot more than one function. Unless you specify otherwise, Mathcad draws straight lines between the points and fills in the axis limits.



For detailed information on creating and formatting graphs, see Chapter 12, “2D Plots.” In particular, refer to Chapter 12 for information about the *QuickPlot* feature in Mathcad which lets you plot expressions even when you don’t specify the range variable directly in the plot.

Resizing a graph

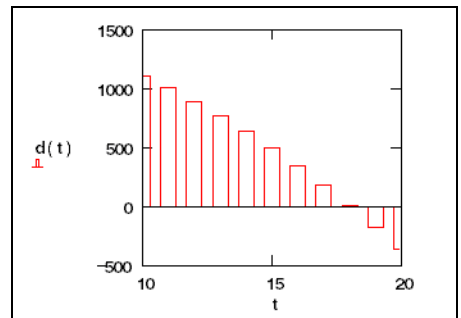
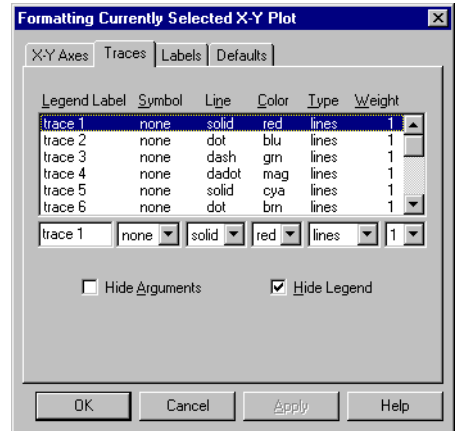
To resize a plot, click in the plot to select it. Then move the cursor to a handle along the edge of the plot until the cursor changes to a double-headed arrow. Hold the mouse button down and drag the mouse in the direction that you want the plot’s dimension to change.

Formatting a Graph

When you first create a graph it has *default* characteristics: numbered linear axes, no grid lines, and points connected with solid lines. You can change these characteristics by *formatting* the graph.

To format the graph created previously, follow these steps:

1. Click on the the graph and choose **Graph⇒X-Y Plot** from the **Format** menu, or double-click the graph to bring up the formatting dialog box. This box contains settings for all available plot format options. To learn more about these settings, see Chapter 12, “2D Plots.”
2. Click the Traces tab.
3. Click “trace 1” in the scrolling list under “Legend Label.” Mathcad places the current settings for trace 1 in the boxes under the corresponding columns of the scrolling list.
4. Click the arrow under the “Type” column to see a drop-down list of trace types. Select “bar” from this drop-down list.
5. Click “OK” to show the result of changing the setting. Mathcad shows the graph as a bar chart instead of connecting the points with lines. Note that the sample line under the $d(t)$ now has a bar on top of it.
6. Click outside the graph to deselect it.



Saving, Printing, and Exiting

Once you’ve created a worksheet, you will probably want to save or print it.

Saving a Worksheet

To save a worksheet:

1. Choose **Save** from the **File** menu (keystroke: **[Ctrl] S**) or click  on the Standard toolbar. If the file has never been saved before, the **Save As** dialog box appears. Otherwise, Mathcad saves the file with no further prompting.
2. Type the name of the file in the text box provided. To save to another folder, locate the folder using the Save As dialog box.

By default Mathcad saves the file in Mathcad (MCD) format, but you have the option of saving in other formats, such as RTF and HTML, as a template for future Mathcad worksheets, or in a format compatible with earlier Mathcad versions. For more information, see Chapter 7, “Worksheet Management.”

Printing

To print, choose **Print** from the **File** menu or click  on the Standard toolbar. To

preview the printed page, choose **Print Preview** from the **File** menu or click  on the Standard toolbar.

For more information on printing, see Chapter 7, “Worksheet Management.”

Exiting Mathcad

When you’re done using Mathcad, choose **Exit** from the **File** menu. Mathcad closes down all its windows and returns you to the Desktop. If you’ve made any changes in your worksheets since the last time you saved, a dialog box appears asking if you want to discard or save your changes. If you have moved any toolbars, Mathcad remembers their locations for the next time you open the application.

Note To close a particular worksheet while keeping Mathcad open, choose **Close** from the **File** menu.

Chapter 4

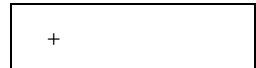
Working with Math

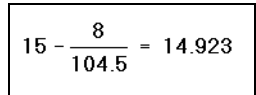
- τ Inserting Math
- τ Building Expressions
- τ Editing Expressions
- τ Math Styles

Inserting Math

You can place math equations and expressions anywhere you want in a Mathcad worksheet. All you have to do is click in the worksheet and start typing.

1. Click anywhere in the worksheet. You see a small crosshair. Anything you type appears at the crosshair.
2. Type numbers, letters, and math operators, or insert them by clicking buttons on Mathcad’s math toolbars, to create a *math region*.





You’ll notice that unlike a word processor, Mathcad by default understands anything you type at the crosshair cursor as math. If you want to create a *text region* instead, follow the procedures described in Chapter 5, “Working with Text.”

You can also type math expressions in any math *placeholder*, which appears when you insert certain operators. See Chapter 9, “Operators,” for more on Mathcad’s mathematical operators and the placeholders that appear when you insert them.

The rest of this chapter introduces the elements of math expressions in Mathcad and describes the techniques you use to build and edit them. See the chapters in the **Computational Features** section of this *User’s Guide* for details on numerical and symbolic calculation in Mathcad.

Numbers and Complex Numbers

This section describes the various types of numbers that Mathcad uses and how to enter them into math expressions. A single number in Mathcad is called a *scalar*. For information on entering groups of numbers in *arrays*, see “Vectors and Matrices” on page 35.

Types of numbers

In math regions, Mathcad interprets anything beginning with one of the digits 0–9 as a number. A digit can be followed by:

- other digits
- a decimal point
- digits after the decimal point

- one of the letters **b**, **h**, or **o**, for binary, hexadecimal, and octal numbers, or **i** or **j** for imaginary numbers. These are discussed in more detail below. See “Suffixes for Numbers” on page 312 in the Appendices for additional suffixes.

Note Mathcad uses the period (.) to signify the decimal point. The comma (,) is used to separate values in a range variable definition, as described in “Range Variables” on page 101. So when you enter numbers greater than 999, do not use either a comma or a period to separate digits into groups of three. Simply type the digits one after another. For example, to enter ten thousand, type “**10000**”.

Imaginary and complex numbers

To enter an imaginary number, follow it with *i* or *j*, as in **1i** or **2.5j**.

Note You cannot use *i* or *j* alone to represent the imaginary unit. You must always type **1i** or **1j**. If you don’t, Mathcad thinks you are referring to a variable named either *i* or *j*. When the cursor is outside an equation that contains *1i* or *1j*, however, Mathcad hides the (superfluous) 1.

Although you can enter imaginary numbers followed by either *i* or *j*, Mathcad normally displays them followed by *i*. To have Mathcad display imaginary numbers with *j*, choose **Result** from the **Format** menu, click on the Display Options tab, and set “Imaginary value” to “j(J).” See “Formatting Results” on page 110 for a full description of the result formatting options.

Mathcad accepts complex numbers of the form $a + bi$ (or $a + bj$), where *a* and *b* are ordinary numbers.

Binary numbers

To enter a number in binary, follow it with the lowercase letter **b**. For example, **11110000b** represents 240 in decimal. Binary numbers must be less than 2^{31} .

Octal numbers

To enter a number in octal, follow it with the lowercase letter **o**. For example, **25636o** represents 11166 in decimal. Octal numbers must be less than 2^{31} .

Hexadecimal numbers

To enter a number in hexadecimal, follow it with the lowercase letter **h**. For example, **2b9eh** represents 11166 in decimal. To represent digits above 9, use the upper or lowercase letters **A** through **F**. To enter a hexadecimal number that begins with a letter, you must begin it with a leading zero. If you don’t, Mathcad will think it’s a variable name. For example, use **0a3h** (delete the implied multiplication symbol between **0** and **a**) rather than **a3h** to represent the decimal number 163 in hexadecimal.

Hexadecimal numbers must be less than 2^{31} .

Exponential notation

To enter very large or very small numbers in exponential notation, just multiply a number by a power of 10. For example, to represent the number $3 \cdot 10^8$, type **3*10^8**.

Vectors and Matrices

A column of numbers is a *vector*, and a rectangular array of numbers is called a *matrix*. The general term for a vector or matrix is an *array*.

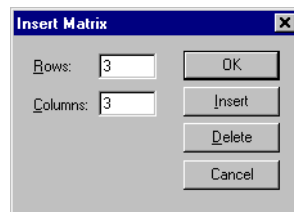
There are a number of ways to create an array in Mathcad. One of the simplest is by filling in an array of empty placeholders as discussed in this section. This technique is useful for arrays that are not too large. See Chapter 11, “Vectors, Matrices, and Data Arrays,” for additional techniques for creating arrays of arbitrary size.

Tip You may wish to distinguish between the names of matrices, vectors, and scalars by font. For example, in many math and engineering books, names of vectors are set in bold while those of scalars are set in italic. See “Math Styles” on page 51 for a description of how to do this.

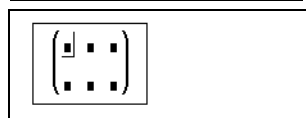
Creating a vector or matrix

To create a vector or matrix in Mathcad, follow these steps:

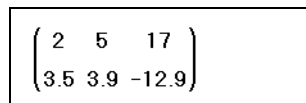
1. Choose **Matrix** from the **Insert** menu or click  on the Matrix toolbar. The dialog box shown on the right appears.



2. Enter a number of rows and a number of columns in the appropriate boxes. In this example, there are two rows and three columns. Then click “OK.” Mathcad inserts a matrix of placeholders.
3. Fill in the placeholders to complete the matrix. Press [**Tab**] to move from placeholder to placeholder.



You can use this matrix in equations, just as you would a number.



Tip The **Insert Matrix** dialog box also allows you to insert or delete a specified number of rows or columns from an array you have already created. See “Changing the size of a vector or matrix” on page 192.

Note Throughout this *User’s Guide*, the term “vector” refers to a *column vector*. A column vector is simply a matrix with one column. You can also create a *row vector* by creating a matrix with one row and many columns.

Strings

Although in most cases the math expressions or variables you work with in Mathcad are numbers or arrays, you can also work with *strings* (also called *string literals* or *string variables*). Strings can include any character you can type at the keyboard, including letters, numbers, punctuation, and spacing, as well as a variety of special symbols as listed in “ASCII codes” on page 316 in the Appendices. Strings differ from variable names or numbers because Mathcad always displays them between double

quotes. You can assign a string to a variable name, use a string as an element of a vector or matrix, or use a string as the argument to a function.

To create a string:

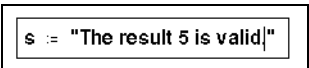
1. Click on an empty math placeholder in a math expression, usually on the right-hand side of a variable definition.
2. Type the double-quote (") key. Mathcad displays a pair of quotes and an insertion line between them.
3. Type any combination of letters, numbers, punctuation, or spaces. Click outside the expression or press the right arrow key (→) twice when you are finished.



A screenshot of a Mathcad worksheet showing a variable definition. The text is `$ := "` followed by a vertical insertion line.



A screenshot of a Mathcad worksheet showing a variable definition. The text is `$ := " | "` followed by a vertical insertion line.



A screenshot of a Mathcad worksheet showing a variable definition. The text is `$ := "The result 5 is valid|"` followed by a vertical insertion line.

To enter a special character corresponding to one of the ASCII codes, do the following:

1. Click to position the insertion point in the string.
2. Hold down the **[Alt]** key, and type the number “0” followed immediately by the number of the ASCII code *using the numeric keypad at the right of the keyboard* in number-entry mode.
3. Release the **[Alt]** key to see the symbol in the string.

For example, to enter the degree symbol (°) in a string, press **[Alt]** and type “0176” using the numeric keypad.

Note The double-quote key (") has a variety of meanings in Mathcad, depending on the exact location of the cursor in your worksheet. When you want to enter a string, you must *always* have a blank placeholder selected.

Valid strings include expressions such as “The Rain in Spain Falls Mainly on the Plain,” “Invalid input: try a number less than -5,” and “Meets stress requirements.” A string in Mathcad, while not limited in size, always appears as a single line of text in your worksheet. Note that a string such as “123,” created in the way described above, is understood by Mathcad to be a string of characters rather than the number 123.

Tip Strings are especially useful for generating custom error messages in programs, as described in Chapter 15, “Programming.” Other string handling functions are listed in “String Functions” on page 187. Use strings also to specify system paths for arguments to some Mathcad built-in functions; see “File Access Functions” on page 188.

Names

A *name* in Mathcad is simply a sequence of characters you type or insert in a math region. A name usually refers to a variable or function that you use in your computations. Mathcad distinguishes between two kinds of names:

- Built-in names, which are the names of variables and functions that are always available in Mathcad, and which you can use freely in building up math expressions.
- User-defined names, which are the names of variables and functions you create in your Mathcad worksheets.

Built-in names

Because Mathcad is an environment for numerical and symbolic computation, a large number of names are built into the product for use in math expressions. These built-in names include built-in *variables* and built-in *functions*.

- Mathcad includes several variables that, unlike ordinary variables, are already defined when you start Mathcad. These *predefined* or *built-in* variables either have a conventional value, like π (3.14159...) or e (2.71828...), or are used as system variables to control how Mathcad calculates. See “Built-in Variables” on page 98 for more information.
- In addition to these predefined variables, Mathcad treats the names of all built-in *units* as predefined variables. For example, Mathcad recognizes the name “A” as the ampere, “m” as the meter, “s” as the second, and so on. Choose **Unit** from the

Insert menu or click  on the Standard toolbar to insert one of Mathcad’s predefined units. See “Units and Dimensions” on page 106 for more on built-in units in Mathcad.

- Mathcad includes a large number of built-in functions that handle a range of computational chores ranging from basic calculation to sophisticated curve fitting, matrix manipulation, and statistics. To access one of these built-in functions, you can simply type its name in a math region. For example, Mathcad recognizes the name “mean” as the name of the built-in *mean* function, which calculates the arithmetic mean of the elements of an array, and the name “eigenvals” as the name of the built-in *eigenvals* function, which returns a vector of eigenvalues for a matrix.

You can also choose **Function** from the **Insert** menu or click  on the Standard toolbar to insert one of Mathcad’s built-in functions. See Chapter 10, “Built-in Functions,” for a broad overview of Mathcad’s built-in functions.

User-defined variable and function names

Mathcad lets you use a wide variety of expressions as variable or function names.

Names in Mathcad can contain any of the following characters:

- Uppercase and lowercase letters.
- The digits 0 through 9.
- The underscore (_).
- The prime symbol ('). Note that this is not the same as an apostrophe. You’ll find the prime symbol on the same key as the tilde (~) or press **[Ctrl][F7]** to insert it.
- The percent symbol (%).
- Greek letters. To insert a Greek letter, click a button on the Greek toolbar or type the equivalent roman letter and press **[Ctrl]G**. The section “Greek letters” on page 38 gives more details.
- The infinity symbol ∞ , which you insert by clicking  on the Calculus toolbar or by typing **[Ctrl][Shift]Z**.

The following are examples of valid names:

a lpha
xyz700
F1 '

b
A1_B2_C3_D4%%%
a%%

The following restrictions apply to variable names:

- A name cannot start with one of the digits 0 through 9. Mathcad interprets anything beginning with a digit as either an imaginary number ($2i$ or $3j$), a binary, octal, or hexadecimal number (e.g., $5o$, $7h$), or as a number *times* a variable ($3 \cdot x$).
- The infinity symbol, ∞ , can only appear as the first character in a name.
- Any characters you type after a period (.) appear as a subscript. This is discussed in “Literal subscripts” on page 39.
- All characters in a name must be in the same font, have the same point size, and be in the same style (italic, bold, etc.). Greek letters can, however, appear in any variable name. See “Math Styles” on page 51.
- Mathcad does not distinguish between variable names and function names. Thus, if you define $f(x)$, and later on you define the variable f , you will find that you cannot use $f(x)$ anywhere below the definition for f .
- Although you can redefine Mathcad’s names for built-in functions, constants, and units, keep in mind that their built-in meanings will no longer exist after the definition. For example, if you define a variable *mean*, Mathcad’s built-in function *mean(v)* can no longer be used.

Note Mathcad distinguishes between uppercase and lowercase letters. For example, *diam* is a different variable from *DIAM*. Mathcad also distinguishes between names in different fonts, as discussed in “Math Styles” on page 51. Thus, *Diam* is also a different variable from ***Diam***.

Tip To type symbols such as \$ in a name, press **[Ctrl][Shift]K**, type the symbol(s), and type **[Ctrl][Shift]K** again.

Greek letters

There are two ways to enter a Greek variable name in Mathcad:

- Click on the appropriate letter on the Greek toolbar. To see this toolbar, click  on the Math toolbar, or choose **Toolbars**⇒**Greek** from the **View** menu.
- Type the *Roman equivalent* of the Greek symbol and then press **[Ctrl]G**. For example, to enter ϕ , press **f[Ctrl]G**. See “Greek Letters” on page 313 in the Appendices for a table of Greek letters and their Roman equivalents.

Note Although many of the uppercase Greek letters look like ordinary capital letters, they are *not* the same. Mathcad distinguishes between Greek and Roman letters, even if they appear visually equivalent.

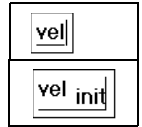
Tip Because it is used so frequently, the Greek letter π can also be typed by pressing **[Ctrl][Shift]P**.

Literal subscripts

If you include a period in a variable name, Mathcad displays whatever follows the period as a subscript. You can use these *literal subscripts* to create variables with names like vel_{init} and u_{air} .

To create a literal subscript, follow these steps:

1. Type the portion of the name that appears before the subscript.
2. Type a period (.), followed by the portion of the name that is to become the subscript.



Note Do not confuse literal subscripts with *array* subscripts, which are generated with the left bracket

key (**[**) or by clicking  on the Calculator toolbar. Although they appear similar—a literal subscript appears below the line, like an array subscript, but with a slight space before the subscript—they behave quite differently in computations. A literal subscript is simply a cosmetic part of a variable name. An array subscript represents a reference to an array element. See Chapter 11, “Vectors, Matrices, and Data Arrays,” for a description of how to use subscripts with arrays.

Operators

As described in the previous section, certain characters, like letters and digits, make up parts of names and numbers. Other characters, like ***** and **+**, represent “operators.”

Operators are symbols like “+” and “–” that link variables and numbers together to form *expressions*. The variables and numbers linked together by operators are called *operands*. For example, in an expression like:

$$a^{x+y}$$

the operands for the “+” are x and y . The operands for the *exponent* operator are a and the expression $x + y$.

You type the common arithmetic operators using the standard keystrokes, like ***** and **+**, used in your spreadsheet and other applications. But all of Mathcad’s operators can be entered with keystrokes or by clicking buttons in the Math toolbars. For example,

you insert Mathcad’s derivative operator by typing **?** or by clicking  on the Calculus toolbar. See “Operators” on page 298 in the Appendices for a complete list of operators. Mathcad’s operators are also discussed in detail in Chapter 9, “Operators.”

Building Expressions

You can create many mathematical expressions by simply typing in a stream of characters, or by inserting appropriate operators from the Math toolbars.

For example, if you type the characters

$$3/4+5^2=$$

you get the result shown at the right.

$$\frac{3}{4+5^2} = 0.103$$

On the surface, Mathcad's equation editor seems very much like a simple text editor, but there's more to it than this. Mathematical expressions have a well-defined structure and Mathcad's equation editor is designed specifically to work within that structure. In Mathcad, mathematical expressions are not so much typed-in as they are built.

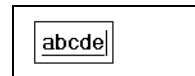
Mathcad automatically assembles the various parts that make up an expression using the rules of precedence and some additional rules that simplify entering denominators,

exponents, and expressions in radicals. For example, when you type $/$ or click  on the Calculator toolbar to create a fraction, Mathcad stays in the denominator until you press **[Space]** to select the entire expression.

Typing in Names and Numbers

When you type in names or numbers, Mathcad behaves very much like a standard word processor. As you type, you see the characters you type appear behind a vertical *editing line*. The left and right arrow keys move this vertical editing line to the left or to the right a character at a time, just as they would in a word processor. There are, however, two important differences:

- As it moves to the right, the vertical editing line leaves behind a trail. This trail is a “horizontal editing line.” Its importance becomes apparent when you begin working with operators.
- Unless the equation you've clicked in already has an operator in it, pressing **[Space]** turns the math region into a text region. It is not possible to turn a text region back into a math region.



Typing in Operators

The key to working with operators is learning to specify what variable or expression is to become an *operand*. There are two ways to do this:

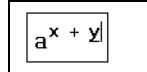
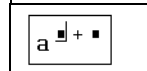
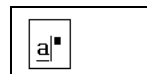
- You can type the operator first and fill in the placeholders with operands, or
- You can use the editing lines to specify what variable or expression you want to turn into an operand.

The first method feels more like you're building a skeleton and filling in the details later. This method may be easier to use when you're building very complicated expressions, or when you're working with operators like summation that require many operands but don't have a natural typing order.

The second method feels more like straight typing and can be much faster when expressions are simple. In practice, you may find yourself switching back and forth as the need arises.

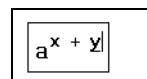
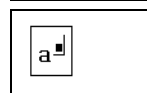
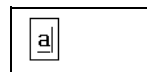
Here's how to create the expression a^{x+y} using the first method:

1. Press **^** to create the exponent operator, or click  on the Calculator toolbar. You see two placeholders. The editing lines "hold" the exponent placeholder.
2. Click in the lower placeholder and type **a**.
3. Click in the upper placeholder.
4. Type **+**.
5. Click in the remaining placeholders and type **x** and **y**.



To use the editing lines to create the expression a^{x+y} , proceed as follows:

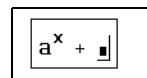
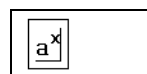
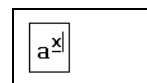
1. Type **a**. The editing lines hold the *a* indicating that *a* becomes the first operand of whatever operator you type next.
2. Press **^** to create the exponent operator. As promised, *a* becomes the first operand of the exponent. The editing lines now hold another placeholder.
3. Type **x+y** in this placeholder to complete the expression.



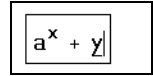
Note that in this example, you could type the expression the same way you'd say it out loud. However, even this simple example already contains an ambiguity. When you say "*a* to the *x* plus *y*" there's no way to tell if you mean a^{x+y} or $a^x + y$. For more complicated expressions, the number of ambiguities increases dramatically.

Although you can always resolve ambiguities by using parentheses, doing so can quickly become cumbersome. A better way is to use the editing lines to specify the operands of whatever operator you type. The following example illustrates this by describing how to create the expression $a^x + y$ instead of a^{x+y} .

1. Enter **a^x** as you did in the previous example. Note how the editing lines hold the *x* between them. If you were to type **+** at this point, the *x* would become the first operand of the plus.
2. Press [**Space**]. The editing lines now hold the entire expression a^x .
3. Now type **+**. Whatever was held between the editing lines now becomes the first operand of the plus.



4. In the remaining placeholder, type **y**.



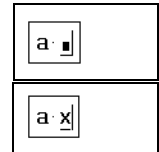
Multiplication

A common way to show multiplication between two variables on a piece of paper is to place them next to each other. For example, expressions like ax or $a(x + y)$ are easily understood to mean “ a times x ” and “ a times the quantity x plus y ,” respectively.

This cannot be done with Mathcad variables for the simple reason that when you type **ax**, Mathcad has no way of knowing whether you mean “ a times x ” or “the variable named ax .” Similarly, when you type **a(x+y)**, Mathcad cannot tell if you mean “ a times the quantity x plus y ” or whether you mean “the function a applied to the argument $x + y$.”

To avoid ambiguity in your everyday work, we recommend that you always press ***** explicitly to indicate multiplication, as shown in the following example:

1. Type **a** followed by *****. Mathcad inserts a small dot after the “ a ” to indicate multiplication.
2. In the placeholder, type the second factor, **x**.



Note In the special case when you type a numerical constant followed immediately by a variable name, such as **4x**, Mathcad interprets the expression to mean the constant multiplied by the variable: $4 \cdot x$. Mathcad displays a space between the constant and the variable to indicate that the multiplication is implied. In this way, you can produce math notation that closely approximates the notation you see in textbooks and reference books. However, Mathcad reserves certain letters, such as “ i ” for the imaginary unit and “ o ” for octal, as suffixes for numbers, and in these cases does not attempt to multiply the number by a variable name but rather treats the expression as a single number with a suffix.

Tip You can change the display of the multiplication operator to an X, a thin space, or a large dot. To do so, click on the multiplication operator with the right mouse button and choose **View Multiplication As...** Or to change all the multiplication operators in a worksheet, choose **Options** from the **Math** menu, click on the Display tab, and choose from the selections next to “Multiplication.” See “Changing the Display of an Operator” on page 124 for additional information.

An Annotated Example

When it comes to editing equations, knowing how to use the editing lines assumes an importance similar to knowing where to put the flashing vertical bar (insertion point) you see in most word processors. A word processor can use a simple vertical bar because text is inherently one-dimensional, like a line. New letters go either to the left or to the right of old ones. An equation, on the other hand, is really *two-dimensional*, with a structure more like a tree with branches than like a line of text. As a result, Mathcad has to use a *two-dimensional* version of that same vertical bar. That’s why there are two editing lines: a vertical line and a horizontal line.

Suppose, for example, that you want to type the slightly more complicated expression

$$\frac{x - 3 \cdot a^2}{-4 + \sqrt{y + 1} + \pi}$$

Watch what happens to the editing lines in the following steps:

1. Type **x-3*a^2**. Since the editing lines contain just the “2,” only the “2” becomes the numerator when you press the **/**.
Since we want the whole expression, $x - 3 \cdot a^2$, to be the numerator, we must make the editing lines hold that entire expression.
2. To do so, press **[Space]**. Each time you press **[Space]**, the editing lines hold more of the expression. You need to press **[Space]** three times to enclose the entire expression.
3. Now press **/** to create a division bar. Note that the numerator is whatever was enclosed between the editing lines when you pressed **/**.
4. Now type **-4+** and click  on the Calculator toolbar. Then type **y+1** under the radical to complete the denominator.
5. To add something *outside* the radical sign, press **[Space]** twice to make the editing lines hold the radical. For example, to add the number π to the denominator, press **[Space]** twice.
6. Press **+**. Since the editing lines are holding the entire radical, it is the entire radical that becomes the first operand when you press **+**.
7. Click  on the Calculator toolbar or press **[Ctrl][Shift]P**. This is one of Mathcad’s built-in variables.

$$x - 3 \cdot a^2$$

$$x - 3 \cdot a^2$$

$$\frac{x - 3 \cdot a^2}{\quad}$$

$$\frac{x - 3 \cdot a^2}{-4 + \sqrt{y + 1}}$$

$$\frac{x - 3 \cdot a^2}{-4 + \sqrt{y + 1}}$$

$$\frac{x - 3 \cdot a^2}{-4 + \sqrt{y + 1} + \quad}$$

$$\frac{x - 3 \cdot a^2}{-4 + \sqrt{y + 1} + \pi}$$

Editing Expressions

This section describes how to make changes to an existing expression.

Changing a Name or Number

To edit a name or number:

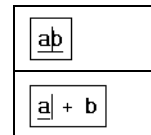
1. Click on it with the mouse. This places the vertical editing line where you clicked the mouse.
2. Move the vertical editing line if necessary by pressing the **[→]** and **[←]** keys.
3. If you type a character, it appears just to the left of the vertical editing line. Pressing **[Bksp]** removes the character to the left of the vertical editing line. Pressing **[Delete]** removes the character to the right of the vertical editing line.

If you need to change several occurrences of the same name or number, you may find it useful to choose **Replace** from the **Edit** menu. To search for a sequence of characters, choose **Find** from the **Edit** menu. These commands are discussed further in “Text Tools” on page 64.

Inserting an Operator

The easiest place to insert an operator is between two characters in a name or two numbers in a constant. For example, here’s how to insert a plus sign between two characters:

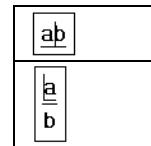
1. Place the editing lines where you want the plus sign to be.
2. Press the **+** key, or click  on the Calculator toolbar.



Note You never need to insert a space when typing an equation. Mathcad inserts spaces automatically around operators wherever doing so is appropriate. If you do try to insert a space, Mathcad assumes you meant to type text rather than math and converts your math region into a text region accordingly.

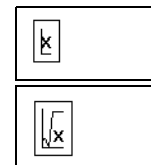
Operators such as division and exponentiation result in more dramatic formatting changes. For example, when you insert a divide sign, Mathcad moves everything that comes after the divide sign into the denominator. Here’s how you insert a divide sign:

1. Place the editing lines where you want the divide sign to be.
2. Press the **/** key or click  on the Calculator toolbar. Mathcad reformats the expression to accommodate the division.



Some operators require only one operand. Examples are the square root, absolute value, and complex conjugate operators. To insert one of these, place the editing lines on either side of the operand and press the appropriate keystroke. Many of these operators are available on the Calculator toolbar as well. For example, to turn x into $\sqrt{x}$ do the following:

1. Place the editing lines around the “ x ,” either preceding or following the character.
2. Press **** to insert the square root operator, or click  on the Calculator toolbar.



Applying an Operator to an Expression

The methods described in the previous section work most predictably when you want to apply an operator to a variable or a number. If, however, you want to apply an operator to an *entire expression*, there are two ways to proceed:

- Surround that expression in parentheses and proceed as described in the previous section, or

- Use the editing lines to specify the expression to which you want to apply the operator.

Although the first method may be more intuitive, it is slower since you need to type a pair of parentheses. The more efficient, second method is the subject of this section. The sections “Inserting Parentheses” on page 48 and “Deleting Parentheses” on page 49 describe ways to work with parentheses more efficiently.

The editing lines consist of a horizontal line and a vertical line that moves left to right along the horizontal line. To make an operator apply to an expression, select the expression by placing it between the two editing lines. The following examples show how typing ***c** results in completely different expressions depending on what was selected.

- Here, the two editing lines hold only the numerator. This means any operator you type will apply only to the numerator.

$$\frac{a + b}{x + d}$$

$$\left(\frac{a + b}{x + d} \right) \cdot c$$

- Typing ***c** results in this expression. Note how the expression held between the editing lines became the first operand of the multiplication.

- Here, the editing lines hold the entire fraction. This means any operator you type will apply to the entire fraction.

$$\frac{a + b}{x + d}$$

$$\frac{a + b}{x + d} \cdot c$$

- Typing ***c** results in this expression. Note how everything between the editing lines became the first operand of the multiplication.

- Here, the editing lines hold the entire fraction as they did in the previous example. However, this time the vertical editing line is on the *left* side instead of on the right side.

$$\frac{a + b}{x + d}$$

- Typing ***c** results in this expression. Note how the expression enclosed by the editing lines became the *second* rather than the first operand of the multiplication. This happened because the vertical editing line was on the *left* side rather than the right side.

$$c \cdot \frac{a + b}{x + d}$$

Controlling the editing lines

You use the following techniques to control what’s between the editing lines:

- Click on an operator. Depending on where on the operator you click, you’ll find the vertical editing line either on the left or on the right of the operator, with the horizontal line selecting an operand of the operator. If you want to move the vertical editing line from one side to the other of the currently selected expression, press **[Insert]**.
- Use the left and right arrow keys to move the vertical editing line one character at a time. The horizontal editing line selects an operand of the nearest operator. If your expression contains built-up fractions, you can also use the up and down arrow keys to move the editing lines.
- Press **[Space]** to select progressively larger parts of the expression with the editing lines. Each time you press **[Space]**, the editing lines enclose more and more of the

expression, until eventually they enclose the entire expression. Pressing **[Space]** one more time brings the editing lines back to where they were when you started.

Tip You can also *drag-select* parts of an expression to hold it between the editing lines. When you do this, the selected expression is highlighted in reverse video. Note that whatever you type next overwrites the highlighted expression.

The following example walks you through a short cycle of using **[Space]**:

1. This is the starting position. The two editing lines hold just the single variable “*d*.”
2. Pressing **[Space]** makes the editing lines grow so that they now hold the entire denominator.
3. Pressing **[Space]** once makes the editing lines grow again so that they now hold the entire expression.
4. At this point, the editing lines can’t become any longer. Pressing **[Space]** brings the editing lines back to the starting point of the cycle.

$\frac{a + b}{x + d}$
$\frac{a + b}{x + d}$
$\frac{a + b}{x + d}$
$\frac{a + b}{x + d}$

You’ll notice that in stepping through the previous cycle there was never an intermediate step in which the editing lines held just the numerator. Nor was there ever a step in which the editing lines held just the *a* or just the *b* in the numerator. That’s because the sequence of steps the editing lines go through as you press **[Space]** depends on the starting point of the cycle.

To set the starting point of the cycle, either click on the appropriate part of the expression as described earlier, or use the arrow keys to move around the expression. The arrow keys walk the editing lines through the expression in the indicated direction. Keep in mind, however, that the idea of “up” and “down” or “left” and “right” may not always be obvious, particularly when the expression becomes very complicated or if it involves summations, integrals, and other advanced operators.

Note Editing of strings differs from editing of other math expressions because you must use the arrow keys or click outside the string to move out of a string. Pressing **[Space]**, which can be used in other expressions to change the position of the editing lines, is interpreted as just another character in a string.

Deleting an Operator

To delete an operator connecting two variable names or constants:

1. Place the vertical editing line after the operator.
2. Press **[BkSp]**.

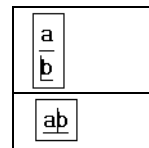
Now you can easily insert a new operator to replace the one you deleted just by typing it in.

$a + b$
ab

Tip You can also delete an operator by placing the editing lines *before* it and pressing **[Delete]**.

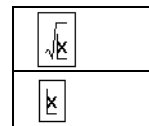
In the above examples, it is easy to see what “before” and “after” mean because the expressions involved naturally flow from left to right, the same way we read. Fractions behave the same way. Since we naturally say “ a over b ,” putting the editing lines “after” the division bar means putting them just before the b . Similarly, putting the editing lines “before” the division bar means putting them immediately after the a . The following example illustrates this:

1. Place the vertical editing lines *after* the division bar.
2. Press [**BkSp**].



To delete an operator having only one operand (for example, $\sqrt{x}$, $|x|$ or $x!$):

1. Position the editing lines just after the operator.
2. Press [**BkSp**].



For certain operators, it may not be clear where to put the editing lines.

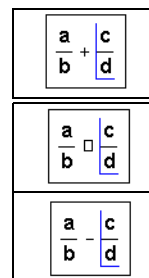
For example, it is not clear when looking at $|x|$ or $\bar{x}$ what “before” and “after” mean. When this happens, Mathcad resolves the ambiguity by referring to the spoken form of the expression. For example, since you read $\bar{x}$ as “ x conjugate,” the bar is treated as being *after* the x .

Replacing an Operator

To replace an operator after deleting it between two variables or constants or on a single variable, as shown in the steps above, simply type the new operator after pressing [**BkSp**].

To replace an operator between two expressions:

1. Position the editing lines just after the operator.
2. Press [**BkSp**]. An operator placeholder appears.
3. Type the new operator.

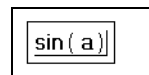


Inserting a Minus Sign

The minus sign that means “negation” uses the same keystroke as the one that means “subtract.” To determine which one to insert, Mathcad looks at where the vertical editing line is. If it’s on the left, Mathcad inserts the “negation” minus sign. If it’s on the right, Mathcad inserts the “subtract” minus sign. To move the vertical editing line from one side to the other, use [**Insert**].

The following example shows how to insert a minus sign in front of “ $\sin(a)$.”

1. Click on the $\sin(a)$. If necessary, press [**Space**] to select the entire expression.



2. If necessary, press [**Insert**] to move the vertical editing line all the way to the left.

$$\sin(a)$$

3. Type -, or click  on the Calculator toolbar, to insert a minus sign.

$$-\sin(a)$$

If what you really want to do is turn $\sin(a)$ into $1 - \sin(a)$, insert another operator (say, “+”) as described in the section “Inserting an Operator” on page 44. Then replace the operator with a minus sign as described in the section “Deleting an Operator” on page 46. Notice that in Mathcad the unary negation symbol in the expression $-\sin(a)$ appears smaller than the minus sign in expressions such as $1 - \sin(a)$.

Note When you are replacing an operator and the operator placeholder is showing, select an expression, rather than a single variable, to the right of the operator placeholder and type - in order to put a subtraction minus sign in the placeholder. Otherwise Mathcad inserts a negation sign.

Inserting Parentheses

Mathcad places parentheses automatically as needed to maintain the precedence of operations. There may be instances, however, when you want to place parentheses to clarify an expression or to change the overall structure of the expression. You can either insert a matched pair of parentheses all at once or insert the parentheses one at a time. We recommend you insert a matched pair since this avoids the possibility of unmatched parentheses.

To enclose an expression with a matched pair of parentheses:

1. Select the expression by placing it between the editing lines. Do this by clicking on the expression and pressing [**Space**] one or more times.

$$\frac{a+b}{a-b} c$$

2. Type the single-quote key (‘), or click  on the Calculator toolbar. The selected expression is now enclosed by parentheses.

$$\left(\frac{a+b}{a-b} \right) c$$

It is sometimes necessary to insert parentheses one at a time using the (and) keys.

For example, to change $a - b + c$ to $a - (b + c)$ do the following:

1. Move the editing lines just to the left of the b . Make sure the vertical editing line is on the left as shown. Press [**Insert**] if necessary to move it over.
2. Type (and click to the right of the c . Make sure the vertical editing line is to the right as shown. Press [**Insert**] if necessary to move it over.
3. Type).

$$a - \underline{b} + c$$

$$a - (\underline{b} + \underline{c})$$

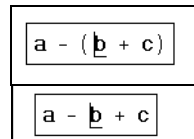
$$a - (b + c)$$

Deleting Parentheses

You cannot delete one parenthesis at a time. Whenever you delete one parenthesis, Mathcad deletes the matched parenthesis as well. This prevents you from inadvertently creating an expression having unmatched parentheses.

To delete a matched pair of parentheses:

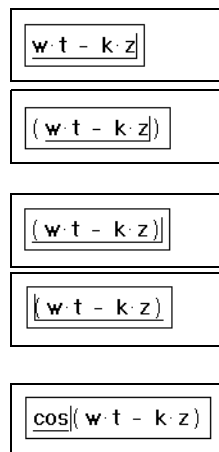
1. Move the editing lines to the right of the “(”.
2. Press [**BkSp**]. Note that you could also begin with the editing lines to the left of the “)” and press [**Delete**] instead.



Applying a Function to an Expression

To turn an expression into the argument of a function, follow these steps:

1. Click in the expression and press [**Space**] until the entire expression, $w \cdot t - k \cdot z$, is held between the editing lines.
2. Type the single-quote key ('), or click  on the Calculator toolbar. The selected expression is enclosed by parentheses.
3. Press [**Space**]. The editing lines now hold the parentheses as well.
4. If necessary, press the [**Insert**] key so that the vertical editing line switches to the left side. If the vertical editing line is already on the left side, skip this step.
5. Now type the name of the function. If the function you wish to use is a built-in function, you can also choose **Function**



from the **Insert** menu or click  on the Standard toolbar and double-click the name of the function.

Moving Parts of an Expression

The menu commands **Cut**, **Copy**, and **Paste** from the **Edit** menu are useful for editing complicated expressions. They function as follows:

- **Cut** ( on the Standard toolbar or [**Ctrl**][**X**] on the keyboard) deletes whatever is between the editing lines and copies it to the Clipboard.
- **Copy** ( on the Standard toolbar or [**Ctrl**][**C**] on the keyboard) takes whatever is between the editing lines and copies it to the Clipboard.
- **Paste** ( on the Standard toolbar or [**Ctrl**][**V**] on the keyboard) takes whatever is on the Clipboard and places it into your worksheet, either into a placeholder or into the blank space between other regions.

The **Copy** and **Paste** commands use the Clipboard to move expressions from one place to another. You can, however, bypass the Clipboard by using Mathcad's *equation drag and drop* feature.

Suppose you want to build the expression

$$\cos(\omega t + x) + \sin(\omega t + x)$$

1. Drag-select the argument to the cosine function so that it is highlighted in reverse video.

2. Press and hold down **[Ctrl]** and the mouse button. The pointer changes to indicate that it carries the selected expression with it. It continues to carry the selected expression until you release the mouse button.

3. With the mouse button still held down, drag the pointer over the placeholder.

4. Release the mouse button. The pointer drops the expression into the placeholder. It then recovers its original form.

Tip You can drag and drop expressions, or even entire math regions, into placeholders in other expressions or into any blank space in your worksheet. Just be sure you don't let go of the mouse button before you've dragged the expression to wherever you want to drop it. If you're trying to drop the expression into a placeholder, be sure to position the pointer carefully over the placeholder.

Deleting Parts of an Expression

You can delete part of an expression by using either the **[Delete]** key or the **[BkSp]** key. If you use this method, whatever you delete is *not* placed on the Clipboard. This is useful when you intend to replace whatever you delete with whatever is currently on the Clipboard.

To delete part of an expression *without* placing it on the Clipboard:

1. Drag-select the part of the expression (in this case, the numerator) so that it is highlighted in reverse video.
2. Press **[Delete]** or **[BkSp]**. This removes the numerator and leaves behind a placeholder.

Note If you select an expression with the editing lines instead of drag-selecting as shown above, you must press **[Bksp]** or **[Delete]** *twice* to remove it. In this case, **[Bksp]** removes the expression to the left of the editing lines, and **[Delete]** removes to the right.

Math Styles

You may already have encountered *styles* in your other applications to determine the appearance of text or other elements. By making changes to text styles rather than to individual text elements in a word processing document, you can make sweeping and strikingly uniform changes in the way that documents looks. (See Chapter 5, “Working with Text,” for an explanation of Mathcad’s text styles) You can get this same kind of leverage by using *math styles* to assign particular fonts, font sizes, font styles and effects, and colors to the elements of your math expressions.

Mathcad has predefined math styles that govern the default appearance of all the math in your worksheet, but you can define and apply additional styles to enhance the appearance of your equations.

Mathcad’s predefined math styles are:

- **Variables**, which governs the default appearance of all variables.
- **Constants**, which governs the default appearance of all numbers you type in math regions as well as all numbers that appear in results.

Whenever you type a variable name, Mathcad:

- Assigns to it a math style named “Variables.”
- Displays the variable name using the characteristics associated with the style named “Variables.”

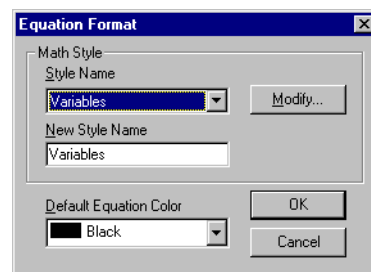
Similarly, when you type a number or when a result is calculated, Mathcad:

- Assigns to it a math style named “Constants.”
- Displays the number using the characteristics associated with the style named “Constants.”

Editing Math Styles

To change Mathcad’s default style for all variables and plots:

1. Click on a variable name in your worksheet.
2. Choose **Equation** from the **Format** menu.
The style name “Variables” is selected.
3. Click “Modify” to change the font associated with the “Variables” style. You’ll see a dialog box for changing fonts.
4. Make any changes using the dialog box and click “OK.” Mathcad changes the font of all variables in the worksheet.



If you change the Variables style, you may also want to change the style used for numbers so that the two look good together. To do so:

1. Click on a number.
2. Choose **Equation** from the **Format** menu to see the Equation Format dialog box.
The style name “Constants” is now selected.

3. Follow the procedure given above for modifying the Variables style.

You can also use the Formatting toolbar to change the font, font size, or font style associated with a math style. For example, to use the Formatting toolbar to modify some of the settings for the Variables math style, click on a variable, then click on the appropriate Formatting toolbar button to make variables bold, italic, or underlined or to specify the font or point size in the drop-down lists.



Note Mathcad's line-and-character grid does not respond automatically to changes in the font sizes used in text and math. Changing font characteristics, particularly font sizes, may cause regions to overlap. You can separate these regions by choosing **Separate Regions** from the **Format** menu.

You may wish to have your equations display in a different color than your default text regions to avoid confusing the two. To change the default color of all equations in your worksheet,

1. Choose **Equation** from the **Format** menu.
2. Select a color in the "Default Equation Color" drop-down list.
3. Click "OK."

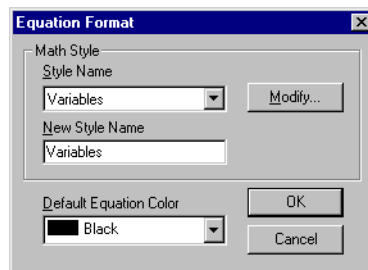
Applying Math Styles

The "Variables" and "Constants" styles govern the default appearance of all math in your worksheet. These two style names cannot be changed. You may, however, create and apply additional math styles, named as you choose, in your worksheets and templates.

To see what math style is currently assigned to a name or number, simply click in the name or number, and look at the style window on the Formatting toolbar.

Alternatively, click the name or number and choose **Equation** from the **Format** menu. The math style associated with whatever you clicked on appears in the drop-down list in the Equation Format dialog box.

If you click on the button to the right of "Variables" in either the Formatting toolbar or the Equation Format dialog box, you'll see a drop-down list of available math styles. If you now choose "User 1" and click "OK," a new math style is applied to the selected element and its appearance changes accordingly.



In this way you can apply any of a variety of math styles to:

- individual variable names in an expression, or
- individual numbers in a math expression (but not in computed results, which always display in the "Constants" style).

For example, many math books show vectors in a bold, underlined font. If you want to use this convention, do the following:

1. Choose **Equation** from the **Format** menu.
2. Click the down arrow beside the name of the current math styles to see a drop-down list of available math styles.
3. Click on an unused math style name like “User 1” to select it. The name “User 1” should now appear in the “New Style Name” text box. Click in this text box and change the name to something like “Vectors.”
4. Click “Modify” to change this style to a bold, underlined font.

This creates a math style called “Vectors” with the desired appearance. When you’re done defining the style, click “OK.”

Now rather than individually changing the font, font size, and font style for names of vectors, you can simply change their math styles.

Note All names, whether function names or variable names, are font sensitive. This means that x and x refer to different variables, and $f(x)$ and $f(x)$ refer to different functions. In deciding whether two variable names are the same, Mathcad actually checks *math styles* rather than fonts. To avoid having distinct variables that look identical, don’t create a math style with exactly the same font, size, and other characteristics as another math style.

Saving Math Styles

Once you’ve completed a set of math styles that you like, you need not repeat the process for other worksheets. You can save math style information by saving a worksheet as a template. Choose **Save As** from the **File** menu and select Mathcad Template (*.mct) as the file type in the Save As dialog box.

To apply math style information to another worksheet, open your template from the **File** menu and copy the contents of the worksheet to the template. For more information about worksheet templates, see Chapter 7, “Worksheet Management.”

Chapter 5

Working with Text

- τ Inserting Text
- τ Text and Paragraph Properties
- τ Text Styles
- τ Equations in Text
- τ Text Tools

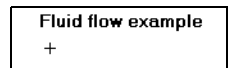
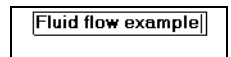
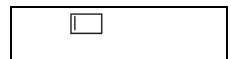
Inserting Text

This section describes how to create text regions in Mathcad. Text regions are useful for inserting any kind of text into your worksheets and templates: comments around the equations and plots in your worksheet, blocks of explanatory text, background information, instructions for the use of the worksheet, and so on. Mathcad ignores text when it performs calculations, but you can insert working math equations into text regions as described in “Equations in Text” on page 63.

Creating a Text Region

To create a text region, follow these steps. First, click in a blank space in your worksheet to position the crosshair where you want the text region to begin. Then:

1. Choose **Text Region** from the **Insert** menu, or press the double-quote (") key. Mathcad begins a text region. The crosshair changes into an insertion point and a text box appears.
2. Now begin typing some text. Mathcad displays the text and surrounds it with a text box. As you type, the insertion point moves and the text box grows.
3. When you finish typing the text, click outside the text region. The text box disappears.



Note You cannot leave a text region simply by pressing [↵]. You must leave the text region by clicking outside the region, by pressing [Ctrl][Shift][↵], or by repeatedly pressing one of the arrow keys until the cursor leaves the region.

To insert text into an existing text region:

- Click anywhere in a text region. A text box now surrounds your text. Anything you type gets inserted at the insertion point.

To delete text from an existing text region, click in the text region and:

1. Press [BkSp] to delete the character to the left of the insertion point, or

2. Press [**Delete**] to delete the character to the right of the insertion point.

To overtype text:

1. Place the insertion point to the left of the first character you want to overtype.
2. Press [**Insert**] to begin typing in *overtyp*e mode. To return to the default *insert* mode, press [**Insert**] again.

You can also overtype text by first selecting it (see “Selecting Text” on page 56). Whatever you type next replaces your selection.

Tip To break a line or start a new line in a text region, press [↵]. Mathcad inserts a hard return and moves the insertion point down to the next line. Press [**Shift**][↵] to start a new line in the same paragraph. When you rewrap the text by changing the width of the text region, Mathcad maintains line breaks at these spot in the text.

Moving the Insertion Point

In general, you move the insertion point within text regions by clicking with the mouse wherever you want to put the insertion point. However, you can also use the arrow keys to move the insertion point.

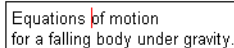
The arrow keys move the insertion point character by character or line by line within text. Pressing [**Ctrl**] and an arrow key moves the insertion point word by word or line by line. These and other ways of moving the insertion point are summarized below.

Key	Action
[→]	Move right one character.
[←]	Move left one character.
[↑]	Move up to the previous line.
[↓]	Move down to the next line.
[Ctrl][→]	Move to the end of the current word. If the insertion point is already there, move to the end of the next word.
[Ctrl][←]	Move to the beginning of the current word. If the insertion point is already there, move to the beginning of the previous word.
[Ctrl][↑]	Move to the beginning of the current line. If the insertion point is already there, move to the beginning of the previous line.
[Ctrl][↓]	Move to the end of the current line. If the insertion point is already there, move to the end of the next line.
[Home]	Move to the beginning of the current line.
[End]	Move to the end of the current line.

Selecting Text

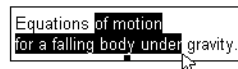
One way to select text within a text region is:

1. Click in the text region so that the text box appears.
2. Drag across the text holding the mouse button down.



Equations of motion
for a falling body under gravity.

Mathcad highlights the selected text, including any full lines between the first and last characters you selected.



On-line Help

You can also select text using arrow keys and multiple clicks of the mouse button, just as you can in most word processing applications. For more information, refer to the topic “Selecting text” in the on-line Help.

Once text is selected, you can delete it, copy it, cut it, check the spelling, or change its font, size, style, or color.

Tip

Once you’ve cut or copied text to the Clipboard, you can paste it back into any text region or into an empty space to create a new text region.

To select and move an entire text region or group of regions, follow the same steps that you would use with math regions, described on “Moving and Copying Regions” on page 10. To perform other editing actions, select the regions, and then choose **Cut**, **Delete**, **Paste**, or **Copy** from the **Edit** menu, or click the corresponding buttons on the Standard toolbar.

Greek Letters in Text

To type a Greek letter in a text region, use one of these two methods:

- Click on the appropriate letter on the Greek toolbar. To see this toolbar, click  on the Math toolbar, or choose **Toolbars**⇒**Greek** from the **View** menu, or
- Type the *Roman equivalent* of the Greek symbol and then press **[Ctrl]G**. For example, to enter ϕ , press **f[Ctrl]G**. See “Greek Letters” on page 313 in the Appendices for a table of Greek letters and their Roman equivalents.

Tip

As discussed in the section “Inserting Math” in Chapter 4, typing **[Ctrl]G** after a letter in a math region also converts it to its Greek equivalent. In addition, **[Ctrl]G** converts a nonalphabetic character to its Greek symbol equivalent. For example, typing **[Shift]2[Ctrl]G** in a text region produces the “ Ξ ” character.

To change a text *selection* into its Greek equivalent, select the text and then:

1. Choose **Text** from the **Format** menu.
2. From the Font list select the Symbol font.

You can also change the font of a text selection by using the Formatting toolbar.

Changing the Width of a Text Region

When you start typing in a text region, the region grows as you type, wrapping only when you reach the right margin or page boundary. (The location of the right margin is determined by the settings in the Page Setup dialog box, which you can modify by choosing **Page Setup** from the **File** menu.) Press **[↵]** whenever you want to start a new line. To set a width for your whole text region and have lines wrap to stay within that width as you type. To do this:

1. Type normally until the first line reaches the width you want.
2. Type a space and press **[Ctrl][↵]**.

All other lines break to stay within this width. When you add to or edit the text, Mathcad rewraps the text according to the width set by the line at the end of which you pressed **[Ctrl][Enter]**.

To change the width of an existing text region, do the following:

1. Click anywhere in the text region. A selection box encloses the text region.
2. Move the pointer to the middle of the right edge of the text region until it hovers over the “handle” on the selection rectangle. The pointer changes to a double-headed arrow. You can now change the size of the text region the same way you change the size of any window—by dragging the mouse.

Tip You can specify that a text region occupies the full page width by clicking on the region and choosing **Properties** from the **Format** menu. Click the Text tab and check “Occupy Page Width.” As you enter more lines of text into a full-width text region, any regions that are below are automatically pushed down in the worksheet.

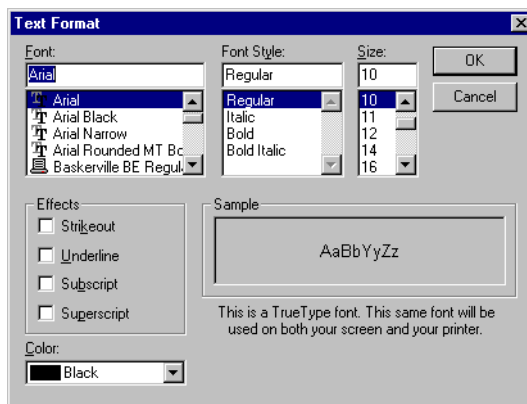
Text and Paragraph Properties

This section describes changing various font properties and changing the alignment and indenting of *paragraphs* within a text region.

Changing Text Properties

To change the font, size, style, position, or color of a portion of the text within a text region, first select the text. (See “Selecting Text” on page 56.) Then choose **Text** from the **Format** menu to access the Text Format dialog box. The Text Format dialog box also appears when you click with the right mouse button on selected text and choose **Font** from the pop-up menu.

Many of the options of the Text Format dialog box are also available via the buttons and drop-down lists on the Formatting toolbar:



When you first insert text, its properties are determined by the worksheet or template defaults for the style called “Normal.” See “Text Styles” on page 61 to find out about applying and modifying existing text styles and creating new ones for governing the default appearance of entire text paragraphs or regions. Any properties that you change for selected text as described here *override* the properties associated with the style for that text region.

Tip If you simply place the insertion point in text and then change the text properties through the Text Format dialog box or the Formatting toolbar, any text you now type at that insertion point will have the new properties you selected.

You can change the following properties of selected text:

- Font
- Font style
- Font size
- Effects such as subscripts and superscripts
- Color

Font sizes are in points. Note that some fonts are available in many sizes and others aren't. Remember that if you choose a bigger font, the text region you're in may grow and overlap nearby regions. Choose **Separate Regions** from the **Format** menu if necessary.

Tip You can specify that a text region automatically pushes following regions down as it grows by clicking on the region and choosing **Properties** from the **Format** menu. Click the "Text" tab and select "Push Regions Down As You Type."

Tip As a shortcut for creating subscripts and superscripts in text, use the **Subscript** and **Superscript** commands on the pop-up menu that appears when you click with the right mouse button on selected text.

Changing Paragraph Properties

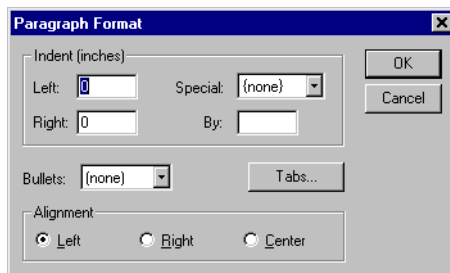
A paragraph in a text region is any stream of characters followed by a hard return, which is created when you type [↵]. You can assign distinct properties to each paragraph in a text region, including *alignment*, *indenting* for either the first or all lines in the paragraph, *tab stops*, and *bullets* or *sequential numbering* to begin the paragraph.

When you first create a text region, its paragraph properties are determined by the worksheet or template defaults for the style called "Normal." See "Text Styles" on page 61 to find out about text styles for governing the default appearance of entire text regions or paragraphs. Any paragraph properties that you change as described here *override* the paragraph properties associated with the style for that text region.

Note When you type [Shift][↵] Mathcad inserts a new line within the current paragraph; it does not create a new paragraph.

You can change the properties for a paragraph within a text region by doing the following:

1. Select the paragraph by clicking in it to place the insertion point, by drag-selecting it, or by triple-clicking it.
2. Choose **Paragraph** from the **Format** menu, or click with the right mouse button and choose **Paragraph** from the pop-up menu. Mathcad displays the Paragraph Format dialog box.
3. Change the appropriate properties in the dialog box and click “OK.”



You can change the following paragraph properties:

Indent

To indent every line in the paragraph the same amount, enter numbers in the “Left” and “Right” text boxes. To indent the *first* line of the paragraph a different amount than the rest of the lines, as for a conventional or hanging indent, select “First Line” or “Hanging” from the “Special” drop-down list and enter a value below.

You can also set indents using the text ruler. Click in a paragraph and choose **Ruler** from the **View** menu. Move the top or bottom arrow in the ruler to set a different indent for the first line, or move both arrows to indent all the lines in the paragraph.

Bullets and numbered lists

To begin the paragraph with a bullet, select “Bullets” from the “Bullets” drop-down list. Select “Numbers” from the drop-down list to have Mathcad number successive

paragraphs in the region automatically. Alternatively, click  or  on the Formatting toolbar.

Alignment

To align the paragraph at either the left or right edge of the text region, or to center the text within the text region, use the three alignment buttons in the dialog box. Alterna-

tively, click one of the three alignment buttons on the Formatting toolbar: , , or .

Tab stops

To specify tabs, click the “Tabs” button in the Paragraph Format dialog box to open the Tabs dialog box. Enter numbers into the “Tab stop position” text box. Click “Set” for each tab stop then click “OK.”

Alternatively, you can set tab stops using the text ruler. Click in a paragraph and choose **Ruler** from the **View** menu. Click in the ruler where you want a tab stop to be. A tab stop symbol appears. To remove a tab stop, click on the tab stop symbol, hold the mouse button down, and drag the cursor away from the ruler.

Tip To change the measurement system used in the Paragraph Format dialog box or in the text ruler, choose **Ruler** from the **View** menu to show the text ruler if it is not already showing, click on the ruler with the right mouse button, and choose **Inches**, **Centimeters**, **Points**, or **Picas** from the pop-up menu.

Text Styles

Mathcad uses *text styles* to assign default text and paragraph properties to text regions. Text styles give you an easy way to create a consistent appearance in your worksheets. Rather than choosing particular text and paragraph properties for each individual region, you can apply an available text style, setting a range of text and paragraph properties at once.

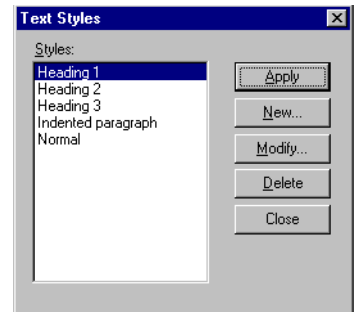
Every worksheet has a default “normal” text style with a particular choice of text and paragraph properties. Depending on your worksheet and the template from which the worksheet is derived, you may have other predefined text styles to which you can apply to existing or new text regions. You can also modify existing text styles, create new ones of your own, and delete ones you no longer need.

This section describes the procedures for applying, modifying, creating, and deleting text styles. See the previous section, “Text and Paragraph Properties,” for details on the available text and paragraph properties.

Applying a Text Style to a Paragraph in a Text Region

When you create a text region in your worksheet, the region is tagged by default with the “Normal” style. You can, however, apply a different style to each paragraph—each stream of characters followed by a hard return—within the text region:

1. Click in the text region on the paragraph where you want to change the style.
2. Choose **Style** from the **Format** menu, or click with the right mouse button and choose **Style** from the pop-up menu, to see a list of the available text styles. Available text styles depend on the worksheet template used.
3. Select one of the available text styles and click “Apply.” The default text in your paragraph acquires the text and paragraph properties associated with that style.



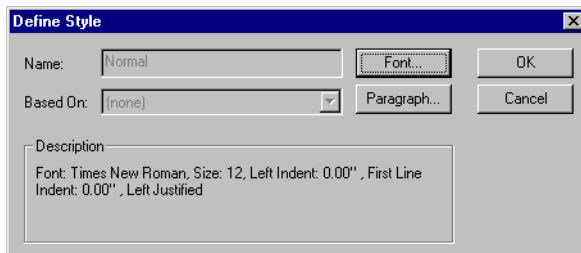
Tip As an alternative to choosing **Style** from the **Format** menu, you can apply a text style to a text paragraph simply by clicking in the paragraph and choosing a style from the left-most drop-down list in the Formatting toolbar. To apply a text style to an entire text region, first select all the text in the region. For information on selecting text, refer to “Selecting Text” on page 56.

Modifying an Existing Text Style

You can change the definition of a text style—its text and paragraph properties—at any time.

To modify a text style:

1. Choose **Style** from the **Format** menu. Mathcad brings up the Text Styles dialog box showing the currently available text styles.
2. Select the name of the text style you want to modify and click “Modify.”
3. The Define Style dialog box displays the definitions of that text style.
4. Click “Font” to modify text formats such as the font, font size, font styling, special effects, and color. Click “Paragraph” to modify the indenting and alignment and other properties for paragraphs. See “Text and Paragraph Properties” on page 58 for details about the available text and paragraph formatting options.
5. Click “OK” to save your changes.



Any new text regions to which you apply the modified text style will reflect the new definition for that text style. In addition, any text regions previously created with the text style will be modified accordingly.

Creating and Deleting Text Styles

You can modify the list of available text styles in your worksheet by creating new ones and deleting ones you no longer use; any text style changes are saved with your worksheet. You can base a new text style on an existing text style, such that it inherits text or paragraph properties, or you can create an entirely new style. For example, you may want to base a new “Subheading” style on an existing “Heading” style, but choose a smaller font size, keeping other text and paragraph properties the same.

Creating a text style

To create a new text style:

1. Choose **Style** from the **Format** menu. Mathcad brings up the Text Styles dialog box showing the currently available text styles.
2. Click “New” to bring up the Define Style dialog box.
3. Enter a name for the new style in the “Name” text box. If you want to base the new style on one of the existing styles in the current worksheet or template, select a style from the “Based on” drop-down list.
4. Click the “Font” button to make your choices for text formats for the new style. Click the “Paragraph” button to choose paragraph formats for the new style.
5. Click “OK” when you have finished defining the new style.

Your new style now appears in the Text Styles dialog box and can be applied to any text region as described in “Applying a Text Style to a Paragraph in a Text Region” on page 61. When you save the worksheet, the new text style is saved with it. If you want

to use the new text style in your future worksheets, save your worksheet as a template as described in Chapter 7, “Worksheet Management.” You may also copy the text style into another worksheet simply by copying and pasting a styled region into the new worksheet.

Note If you base a new text style on an existing text style, any changes you later make to the original text style will be reflected in the new text style as well.

Deleting a text style

You may delete a text style at any time. To do so:

1. Choose **Style** from the **Format** menu. Mathcad brings up the Text Styles dialog box showing the currently available text styles.
2. Select one of the available text styles from the list.
3. Click “Delete.”

The text style is removed from the list of available text styles. However, any text regions in your worksheet whose text and paragraph properties were defined in terms of that text style will continue to display the properties of that style.

Equations in Text

This section describes how to insert equations into your text regions. Equations inserted into text have the same properties as those in the rest of your worksheet. You can edit them using the methods described in Chapter 4, “Working with Math.”

Inserting an Equation into Text

Place an equation into text either by creating a new equation inside a text region or by pasting an existing equation into a text region.

To add a new equation into a text region or a paragraph, follow these steps:

1. Click in the text region or paragraph to place the insertion point where you want the equation to start.
2. Choose **Math Region** from the **Insert** menu. A placeholder appears.
3. Type in the equation just as you would in a math region.
4. When you’ve finished typing in the equation, click on any text to return to the text region. Mathcad adjusts the line spacing in the text region to accommodate the embedded math region.

The universal gravitational constant, G, has the value | and can be used to determine the acceleration of a less massive object toward a more massive object.

The universal gravitational constant, G, has the value $\frac{6.67259 \cdot 10^{-11} \cdot \text{m}^3}{\text{kg} \cdot \text{s}^2}$ and can be used to determine the acceleration of a less massive object toward a more massive object.

The universal gravitational constant, G, has the value $G := 6.67259 \cdot 10^{-11} \cdot \frac{\text{m}^3}{\text{kg} \cdot \text{s}^2}$ and can be used to determine the acceleration of a less massive object toward a more massive object.

To paste an existing equation into a text region, follow these steps:

1. Select the equation you want to paste into the text.
2. Choose **Copy** from the **Edit** menu, or click  on the Standard toolbar.
3. Click in the text region to place the insertion point where you want the equation to start.
4. Choose **Paste** from the **Edit** menu, or click  on the Standard toolbar.

Disabling Embedded Equations

When you first insert an equation into text, it behaves just like an equation in a math region; it affects calculations throughout the worksheet. If you want the equation to be purely cosmetic, you can disable it so that it no longer calculates. To do so:

1. Click on the equation you want to disable.
2. Choose **Properties** from the **Format** menu. Click on the Calculation tab.
3. Click the “Disable Evaluation” check box.
4. Click “OK.”

Once you have done so, the equation can neither affect nor be affected by other equations in the worksheet. To turn it back on, remove the check next to “Disable Evaluation” in the Properties dialog box.

For a more general discussion of disabling and locking equations, see “Disabling Equations” on page 118.

Text Tools

Mathcad has tools for finding and replacing text as well as checking the spelling of text.

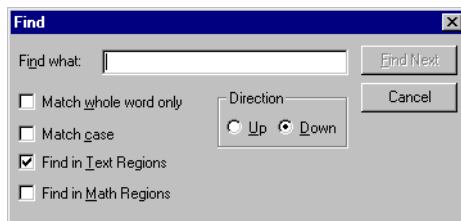
Find and Replace

Mathcad’s **Find** and **Replace** commands on the **Edit** menu are capable of working in both text and math regions. By default, however, Mathcad finds and replaces text in text regions only.

Searching for text

To find a sequence of characters:

1. Choose **Find** from the **Edit** menu. Mathcad brings up the Find dialog box.
2. Enter the sequence of characters you want to find.
3. Click “Find Next” to find the occurrence of the character sequence immediately after the current insertion point location. Use the available options in the dialog box to search upward or downward in the worksheet, to match whole words



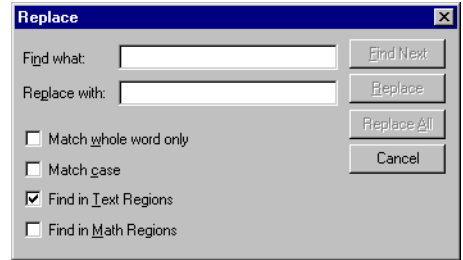
only, to match the case exactly of the characters you entered, and to specify whether Mathcad should search in text or math regions or both.

On-line Help The Help topic “Characters You Can Find and Replace” details the characters you can find in math and text regions, including Greek symbols. Many special characters, including punctuation and spaces, can be located only in text or math strings.

Replacing characters

To search and replace text:

1. Choose **Replace** from the **Edit** menu to bring up the Replace dialog box.
2. Enter the string you want to find (the target string) in the “Find what” box.
3. Enter the string you want to replace it with in the “Replace with” box. Check the appropriate boxes to match whole words only, to match the case exactly of the characters you entered, and to specify whether Mathcad should search in text or math regions or both.



You now have the following options:

- Click “Find Next” to find and select the next instance of your target string.
- Click “Replace” to replace the currently selected instance of the string.
- Click “Replace All” to replace all instances of the string.

Spell-Checking

After creating text, you can have Mathcad search the text for misspelled words and suggest replacements. You can also add words that you commonly use to your personal dictionary.

Note Mathcad spell-checks text regions only, not math or graphics regions.

To begin spell-checking, specify the portion of the worksheet to spell-check. There are two ways to do this:

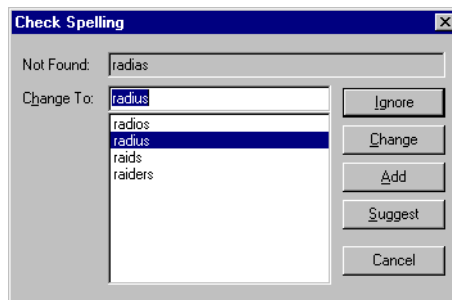
- Click at the beginning of wherever you want to spell-check. Mathcad spell-checks starting from this point and continues to the end of the worksheet. You can then either continue the spell-check from the beginning of the worksheet or quit spell-checking.
- Alternatively, select the text you want to spell-check.

Once you've defined a range over which to check spelling:

1. Choose **Check Spelling** from the **Edit**

menu, or click  on the Standard toolbar.

2. When Mathcad finds a misspelled word, it opens the Check Spelling dialog box. The misspelled word is shown along with a suggested replacement or a list of possible replacements. If Mathcad has no immediate suggestions, it shows only the misspelled word.



Tip To determine whether a word is misspelled, Mathcad compares it with the words in two dictionaries: a general dictionary of common English words supplemented by mathematical terms and a personal dictionary. If there are certain correctly spelled words throughout your worksheet which Mathcad nevertheless shows as being misspelled, you may want to add them to your personal dictionary.

After the Check Spelling dialog box appears, you have several options:

- To change the word to the suggested replacement, or to another word you select from the list of possible replacements, click “Change.”
- Click “Suggest” to see additional but less likely replacements. If Mathcad can offer no additional suggestions, “Suggest” is grayed.
- To change the word to one not listed, type the replacement into the “Change to” box and click “Change.”
- To leave the word as is, click “Ignore” or “Add.” If you click “Ignore,” Mathcad leaves the word alone, continues spell-checking, and ignores all future occurrences of the word. If you click “Add,” the word is added to your personal dictionary.

Note To choose a dialect associated with the English dictionary, choose **Preferences** from the **View** menu, click on the General tab, and choose an option below “Spell Check Dialect.”

Chapter 7

Worksheet Management

- τ Worksheets and Templates
- τ Rearranging Your Worksheet
- τ Layout
- τ Safeguarding an Area of the Worksheet
- τ Hyperlinks
- τ Creating an Electronic Book
- τ Printing and Mailing

Worksheets and Templates

As you use Mathcad and save your work for later use, you typically create a *worksheet* that contains unique text, math, and graphic regions. Mathcad uses .MCD as the default file extension for worksheets.

When you create a new worksheet in Mathcad, you can start with Mathcad's default choices for formats and layout, or you can use a *template* that contains customized information for laying out and formatting the worksheet. When you create a worksheet based on a template, all of the formatting information and any text, math, and graphic regions from the template are copied to the new worksheet. The new worksheet therefore inherits the appearance and formatting instructions of the template, allowing you to maintain consistency in the appearance of multiple worksheets.

Mathcad comes with a variety of predefined templates for you to use as you create new Mathcad worksheets. You extend the collection of templates by saving any of your Mathcad worksheets as a template. Mathcad uses .MCT as the default file extension for templates.

Other saving options are available in Mathcad. You can save a worksheet in rich-text format (RTF), so that it can be opened by most word processors, or in Hypertext Markup Language (HTML), so that the file can be viewed through a Web browser. You can also save a worksheet in a format that can be read by earlier versions of Mathcad.

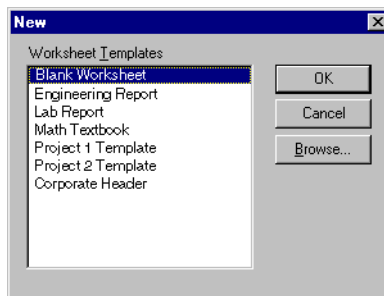
Creating a New Worksheet

When you first open Mathcad or click  on the Standard toolbar, you see an empty worksheet based on the *blank worksheet template* (normal.mct). You can enter and format equations, graphs, text, and graphics in this space, as well as modifying worksheet attributes such as the page margins, numerical format, headers and footers, and text and math styles. The blank worksheet is only one of the templates Mathcad provides. Other built-in templates are specific to types of worksheets you might be

creating. For example, there is a template for Electronic Books, one for engineering reports, and one for specification sheets.

To create a new worksheet based on a template:

1. Choose **New** from the **File** menu. Mathcad displays a list of available worksheet templates. The exact templates available differ depending on the templates you have developed.
2. Choose a template other than “Blank Worksheet.” By default Mathcad displays worksheet templates saved in the TEMPLATE folder of the directory you used to install Mathcad. Click “Browse” to find a template in another directory or on another drive.
3. Click “OK.”



Saving Your Worksheet

When you want to save the worksheet, choose either **Save** or **Save As** from the **File** menu and enter a file name with the extension .MCD. After the first time you save the worksheet, simply choose **Save** from the **File** menu or click  on the Standard toolbar to update the saved copy of the worksheet.

Tip To work on a worksheet you saved before, choose **Open** from the **File** menu or click  on the Standard toolbar. Mathcad prompts you for a name by displaying the Open dialog box. You can locate and open a Mathcad worksheet from other directories or drives just as you would in any other Windows application. At the bottom of the **File** menu, Mathcad maintains a list of the most recently used worksheets, which you can choose directly, if you wish.

Saving your worksheet in RTF format

To save a worksheet so you can open it in a word processor capable of reading an RTF file with embedded graphics:

1. Scroll to the bottom of your worksheet to update all calculated results.
2. Choose **Save As** from the **File** menu.
3. In the Save As dialog box, choose “Rich Text Format File” from the “Save as type” drop-down list.
4. Enter a file name and then click “Save.”

When you open an RTF file with a word processor such as Microsoft Word, you’ll find all the Mathcad regions lined up one above the other at the left edge of the document. You may have to move regions in the word processors to make them look like your original Mathcad worksheet. Once the Mathcad regions have been loaded into a word processor, you will be able to edit the text. However, you’ll no longer be able to edit math regions and graphs, which have become embedded graphics. To embed Mathcad worksheets or regions in a word processing document in a form that allows

you to continue to edit the original Mathcad worksheets, see “Inserting Objects” on page 71.

Tip Mathcad’s text supports Microsoft’s “Rich Text Format” (RTF) specification. This means you can easily export text from Mathcad text regions to most word processing programs via the Clipboard. Simply select text in a Mathcad text regions, copy the text to the Clipboard by

choosing **Copy** from the **Edit** menu or clicking  on the Standard toolbar, and choose **Paste** from the **Edit** menu in your word processing application.

Saving your worksheet in HTML format

To save a worksheet so that a Web browser can open it:

1. Scroll to the bottom of your worksheet to update all calculated results.
2. Choose **Save As** from the **File** menu.
3. In the Save As dialog box, choose “HTML File” from the “Save as type” drop-down list.
4. Enter a file name and then click “Save.”

Note When you save a worksheet as HTML, regions other than text regions are saved as individual JPEG files. If your worksheet contains many regions other than text, save the worksheet in a new folder so that all the files will be accessed in a single location.

Saving your worksheet in an earlier format

In general, worksheets created in an earlier version of Mathcad open in the current version, but files created in the current version of Mathcad *do not* open in earlier versions. Mathcad 2000, however, allows you to save a worksheet as a Mathcad 8, 7, or 6 worksheet.

Note Features in your worksheet available only in Mathcad 2000 will not be recognized in earlier versions of Mathcad. Regions or features that won’t work in an earlier version are rendered as bitmaps.

To save a worksheet in a form that can be read by an earlier version of Mathcad:

1. Choose **Save** or **Save As** from the **File** menu.
2. In the “Save as type” drop-down list, select “Mathcad 8 Worksheet,” “Mathcad 7 Worksheet,” or “Mathcad 6 Worksheet” and provide a file name.
3. Click “Save.” A message appears warning you that certain features available only in Mathcad 2000 will not work in earlier versions.

Creating a New Template

You can extend the collection of templates by creating your own. A template you create can have equations, text, and graphics in places you determine, as well as customized information in the headers and footers (see “Layout” on page 83). The template also specifies:

- Definitions of all math styles (Chapter 4).
- Definitions of all text styles (Chapter 5).

- Margins for printing (see “Layout” on page 83).
- Numerical result formats and values for Mathcad’s built-in variables (Chapter 8).
- Names of Mathcad’s basic units and the default unit system (Chapter 8).
- The default calculation mode (Chapter 8).
- Ruler visibility and measurement system (see “Aligning Regions” on page 79).

To create a new template, first create a new worksheet having the options listed above set the way you want. The worksheet can also contain any equations, text, and graphics that you want in the template. The next step is to save this worksheet as a template. To do so:

1. Choose **Save As** from the **File** menu.
2. Double-click the TEMPLATE folder in the Save As dialog.
3. In the “Save as type” drop-down list, select “Mathcad Templates (*.mct).”
4. Type a name for the template in the “File name” box.
5. Click “Save.”

Your template is now added to the list of templates available in the dialog box that appears when you choose **New** from the **File** menu. To make a new worksheet based on a template you’ve created, simply choose **New** from the **File** menu and select your template from the list. If you did not save your template to the TEMPLATE folder, you need to browse to find the template.

Modifying a Template

To modify an existing worksheet template:

1. Choose **Open** from the **File** menu or click  on the Standard toolbar.
2. In the “Files of type” drop-down list, select “All Files.”
3. Type the name of the template in the “File name” box, or browse to locate it in the dialog box. Worksheet templates are saved by default in the TEMPLATE folder.
4. Click “Open.” The template opens in the Mathcad window.

You may now edit the template as you would modify any Mathcad worksheet. To save your changes under the current template name, choose **Save** from the **File** menu or click  on the Standard toolbar. If you want to give a new name to the modified template, choose **Save As** from the **File** menu and enter a new name for the template.

Tip To modify the default template for a blank worksheet, modify the template file NORMAL.MCT.

Note When you modify a template, your changes affect only new files created from the modified template. The changes do not affect any worksheets created with the template before the template was modified.

Rearranging Your Worksheet

This section describes how to rearrange math, graphics, and text in your worksheets. See the section “Regions” on page 10 for the basics on selecting, copying, moving, and deleting regions.

Note You can get an overall view of how your worksheet looks by choosing **Zoom** from the **View** menu or clicking  on the Standard toolbar and choosing a magnification. Choose a magnification less than 100% to zoom out of the worksheet, or use a magnification greater than 100% to zoom in. Alternatively, use the **Print Preview** command as described under “Print Preview” on page 94.

Aligning Regions

Once you’ve inserted regions into your worksheet, you can align them vertically or horizontally using menu commands or you can align them using the worksheet ruler.

Using commands

To align regions horizontally or vertically using commands:

1. Select regions as described on page 10.
2. Choose **Align Regions**⇒**Across** (to align horizontally) or **Align Regions**⇒**Down** (to align vertically) from the Format menu. Or choose these commands by clicking

 and  on the Standard toolbar.

When you choose **Align Regions**⇒**Down** from the pull-right menu or click  on the Standard toolbar, Mathcad does the following:

- Mathcad draws an invisible vertical line halfway between the right edge of the right-most selected region and the left edge of the left-most selected region.
- All selected regions to the right of this line are moved left until their left edges are aligned with this line.
- All selected regions to the left of this line are moved right until their left edges are aligned with this line.

Choosing **Align Regions**⇒**Across** or clicking  on the Standard toolbar works in much the same way. Mathcad draws an invisible horizontal line halfway between the top edge of the uppermost region and the bottom edge of the lowest region. Selected regions below and above this line are moved up and down respectively until the midpoints of their left edges are on this line.

Note Aligning regions may inadvertently cause regions to overlap. Mathcad warns you when this will occur, but you can separate overlapping regions as described in “Separating Regions” below.

Using the worksheet ruler

When you choose **Ruler** from the **View** menu while the cursor is in a blank spot or in a math region, you see the worksheet ruler at the top of the window. You can use

alignment guidelines on the ruler to align regions at particular measurements along the worksheet.

To set an alignment guideline on the ruler:

1. Click on the ruler wherever you want the alignment guideline to appear. A tab stop symbol appears on the ruler.
2. Click on the tab stop symbol with the right mouse button and choose **Show Guideline** from the pop-up menu. A check appears next to the command.

The alignment guideline appears as a green vertical line. Select and move regions to the guideline. Figure 7-1 shows how you can use an alignment guideline to align math regions.

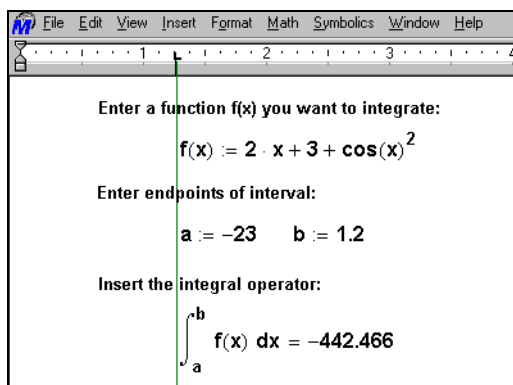


Figure 7-1: Using an alignment guideline to align regions vertically.

Note The tab stops you insert on the ruler specify where the cursor should move when you press the **[TAB]** key. To remove a tab stop, click on its symbol, hold the mouse button down, and drag the cursor away from the ruler.

To remove an alignment guideline, click on the ruler with the right mouse button where the guideline is located and choose **Show Guideline** from the menu to uncheck it.

Tip You can change the measurement system used in the ruler by clicking on the ruler with the right mouse button, and choosing **Inches**, **Centimeters**, **Points**, or **Picas** from the pop-up menu. To change the ruler measurement for all documents, make this change to normal.mct.

Inserting or Deleting Blank Lines

You can easily insert one or more blank lines into your worksheet:

1. Click on the blank line below which you want to insert one or more blank lines. Make sure the cursor looks like a crosshair.
2. Press **[Enter]** to insert a blank line and move the cursor to the left margin. Do this as many times as you want to insert lines.

To delete one or more blank lines from your worksheet:

1. Click above the blank lines you want to delete. Make sure the cursor looks like a crosshair and that there are no regions to the right or left of the cursor.
2. Press [**Delete**] as many times as there are lines you want to delete. Mathcad deletes blank lines below your cursor. Alternatively, press [**BkSp**] as many times as there are lines you want to delete. Mathcad deletes blank lines *above* your cursor.

If you press either [**Delete**] or [**BkSp**] and nothing seems to be happening, check to make sure that the cursor is on a line all by itself. If any region in your worksheet extends into the line you are trying to delete, Mathcad won't be able to delete that line.

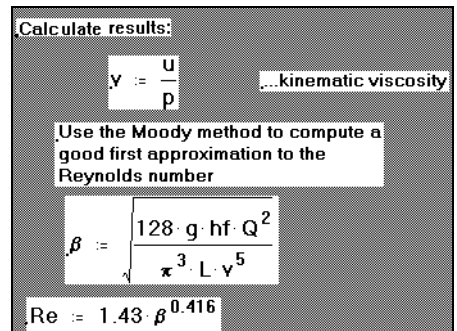
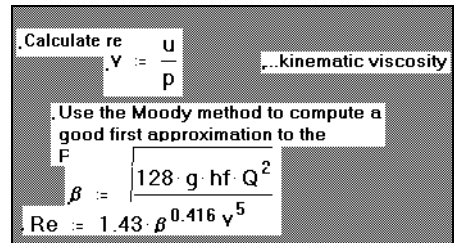
Tip To quickly insert or delete a *specific number* of lines from your worksheet, click in a blank part of the worksheet with the right mouse button, choose **Insert Lines** or **Delete Lines** from the pop-up menu, and enter the number of lines in the dialog box.

Separating Regions

As you move and edit the regions in a Mathcad worksheet, they may end up overlapping one another. Overlapping regions don't interfere with each other's calculations, but they may make your worksheet hard to read.

A good way to determine whether regions overlap is to choose **Regions** from the **View** menu. As shown at right, Mathcad displays blank space in gray and leaves the regions in your default background color. To turn the blank space back into the default background color, choose **Regions** from the **View** menu again.

To separate all overlapping regions, choose **Separate Regions** from the **Format** menu. Wherever regions overlap, this command moves the regions in such a way as to avoid overlaps while preserving the order of the calculations, as shown at right.



Note Be careful with the **Separate Regions** menu command since not only can it have far-reaching effects, it also cannot be undone. Regions are moved around and the order of calculation can change. As an alternative, you can drag regions individually, add lines by pressing [**Enter**], or cut and paste the regions so they don't overlap.

Highlighting Regions

Mathcad allows you to highlight regions so that they stand out from the rest of the equations and text in your worksheet:

To apply a background highlight color to a region:

1. Click in the region you want to highlight.
2. Choose **Properties** from the **Format** menu.
3. Click the Display tab.
4. Check “Highlight Region.” Click “Choose Color” to choose a highlight color other than the default choice.
5. Click “OK.”

Mathcad fills a box around the equation with either the default background highlight color or the color you chose. This is a purely cosmetic change with no effect on the equation other than making it more conspicuous.

Note The appearance of a highlighted region on printing depends very much on the capabilities of your printer and the choice of highlight color. Some black and white printers render a color as black, obscuring the equation in the process. Others render just the right gray to highlight the equation without obscuring it.

To change the default background color of highlighted regions, do the following:

1. Choose **Color** from the **Format** menu.
2. Pull right and choose **Highlight** to bring up a dialog box containing a palette of colors. Click the appropriate color.
3. Click “OK.”

Changing the worksheet background color

Mathcad allows you to change the color of the background of your worksheet. To do so:

1. Choose **Color** from the **Format** menu.
2. Pull right and choose **Background** to bring up a dialog box containing a palette of colors. Click the appropriate color.
3. Click “OK.”

Layout

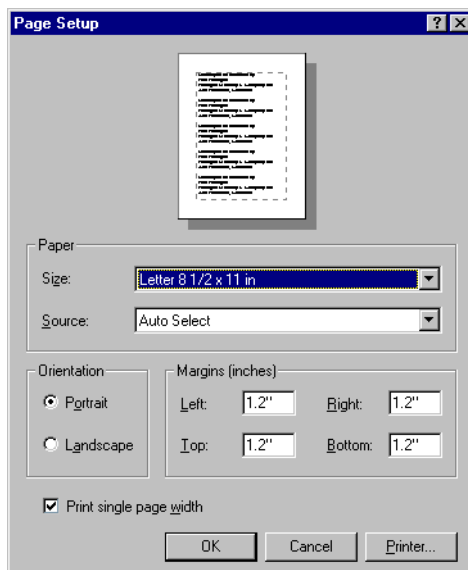
Before printing a worksheet, you may need to adjust the margins, paper options, page breaks, and headers and footers so that pages of the worksheet are printed appropriately.

Setting Margins, Paper Size, Source, and Orientation

Mathcad worksheets have user-specifiable margins at the left, right, top, and bottom of the worksheet. To set these margins, choose **Page Setup** from the **File** menu.

Use the four text boxes in the lower right of the Page Setup to specify the distances from the margin to the corresponding edge of the actual sheet of paper on which you are printing.

You can also use settings in the Page Setup dialog box to change the size, source, or orientation of the paper on which you print your worksheet. See “Printing and Mailing” on page 93 for more about printing your Mathcad worksheets.



Tip If you want the margin and other page setup settings in the current worksheet to be used in other worksheets, save the worksheet as a template as described in “Creating a New Template” on page 77.

Page Breaks

Mathcad provides two kinds of page breaks:

- **Soft page breaks:** Mathcad uses your default printer settings and your top and bottom margins to insert these page breaks automatically. These show up as dotted horizontal lines, and you see them as you scroll down in your worksheet. You cannot add or remove soft page breaks.
- **Hard page breaks:** You can insert a hard page break by placing the cursor at the appropriate place in your worksheet and choosing **Page Break** from the **Insert** menu. Hard pagebreaks display as solid horizontal lines in your worksheets.

When Mathcad prints your worksheet, it begins printing on a new page whenever it encounters either a soft or a hard page break.

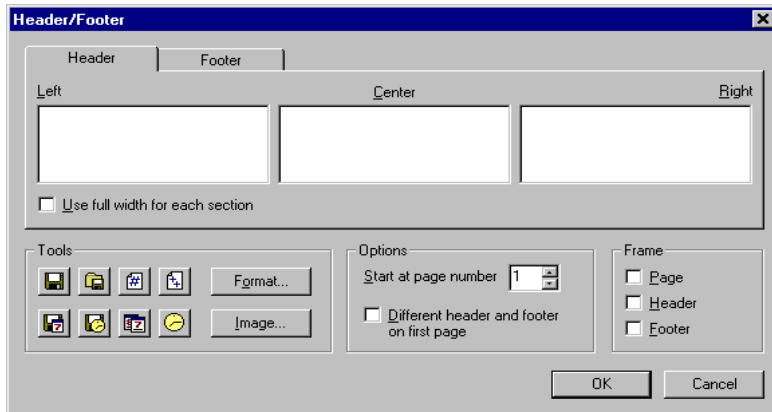
To delete a hard page break:

1. Drag-select the hard page break as you would select any other region in your Mathcad worksheet. A dashed selection box appears around the page break.
2. Choose **Delete** from the **Edit** menu.

Tip Because Mathcad is a WYSIWYG environment, any region that overlaps a soft or hard page break prints by default in pieces on successive pages. To separate a region from a hard page break, choose **Separate Regions** from the **Format** menu. However, this command does not separate regions from any overlapping *soft* page breaks. Choose **Repaginate Now** from the **Format** menu to force Mathcad to insert a soft page break above any region that otherwise would print in pieces on successive pages.

Headers and Footers

To add a header or a footer to every printed page, to create a different header or footer for the first page of a worksheet, or to modify an existing header or footer, choose **Headers/Footers** from the **Format** menu. The Header/Footer dialog box appears:



To add or edit a header or footer:

1. Click the Header or Footer tab to modify the header or footer for the worksheet. To create a different header or footer for the first page of your worksheet, check the “Different header and footer on first page” option and click the Header–Page 1 or Footer–Page 1 tab.
2. Type the header or footer information into one or more of the text boxes. Whatever you type into the Left, Center, and Right text boxes will appear in these positions on the page. Click “Format” in the Tools group to change the header or footer font, font style, size, or alignment. Click “Use full width for each section” if you want text in any of the boxes to extend beyond the width of the text box.
3. Click one or more of the buttons in the Tools group to insert items such as the file name, page number, current date, or time automatically wherever the insertion point is. To insert an image, click “Image” in the Tools group and browse to locate a bitmap (.BMP format) file.

Tip Mathcad by default begins numbering at page 1. You can set a different starting page number in the Options group in the Header/Footer dialog box.

Safeguarding an Area of the Worksheet

The ease with which you can alter a Mathcad worksheet can present a problem. It is all too easy to alter a worksheet and to change things which are not meant to be changed. For example, if you've developed and thoroughly tested a set of equations, you may want to prevent readers of your worksheet from tampering with them. To avoid unintended edits to your worksheet, you can safeguard an area of your worksheet by locking it such that you can still edit it even though nobody else can.

Pro You can use Mathcad Professional to lock an area of your worksheet. To do so:

1. You create an *area* in your worksheet to contain the regions to be protected.
2. You place the regions that you want to safeguard into that area.
3. You lock the area. Optionally you can password protect and collapse the area.

Once a region is safely inside a locked area, nobody can edit it. Any math regions inside a locked area continue, however, to affect other equations in the document. For example, if you define a function inside a locked area, you can still use that function anywhere below and to the right of its definition. You cannot, however, change the function's definition itself unless you unlock the area first.

Inserting an Area

To insert a lockable area into your worksheet:

1. Choose **Area** from the **Insert** menu.
Mathcad inserts a pair of lines into the worksheet. These mark the boundaries of the lockable area.
2. Select either of these boundary lines just as you'd select any region: by dragging the mouse across the line or by clicking the line itself.
3. Once you've selected the boundary line, drag it just as you'd drag any other region to move it.



You should position the boundaries so that there's enough space between them for whatever regions you want to lock. You can have any number of lockable areas in your worksheet. The only restriction is that you cannot have one lockable area inside another.

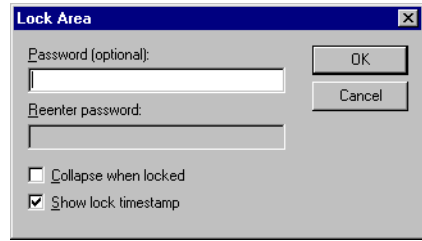
Tip To name an area in your worksheet, click on an area boundary, choose **Properties** from the **Format** menu, and enter a name on the Area tab. The Area tab also lets you modify other display attributes of an area, such as whether a border or icon appears.

Locking and Collapsing an Area

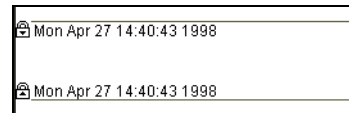
Once you've placed whatever regions you want inside an area, you can lock the area. You can choose to lock an area with a password to prevent unauthorized editing of the regions in it. You can also collapse the area, either with or without locking it, so that the regions are hidden from view.

To lock an area:

1. Click in the area.
2. Choose **Area⇒Lock** from the **Format** menu.
3. In the Lock Area dialog box, enter a password if you want to lock the area with a password. Type any combination of letters, numbers, and other characters. You must re-enter the password to confirm it.
4. Check “Collapse when locked” to hide the locked regions from view. Check “Show lock timestamp” to display the date and time the area was last locked above and below the boundary lines.
5. Click “OK.”

The 'Lock Area' dialog box has a title bar with a close button. It contains two text input fields: 'Password (optional):' and 'Reenter password:'. To the right of these fields are 'OK' and 'Cancel' buttons. At the bottom, there are two checkboxes: 'Collapse when locked' (unchecked) and 'Show lock timestamp' (checked).

The area is now locked and by default shows padlocks on the boundaries and a timestamp.



Note If you choose to password protect an area, make sure you remember your password. If you forget it, you will find yourself permanently locked out of that area. Keep in mind also that the password is case sensitive.

To collapse an area without locking it first:

1. Click in the area.
2. Choose **Area⇒Collapse** from the **Format** menu.

A collapsed area appears by default as a single line in your worksheet.

Unlocking and Expanding an Area

If you want to make changes to a region inside a locked area, you have to unlock it. If the area is collapsed, you must also expand it.

To unlock a locked area:

1. Click in the area you want to unlock.
2. Choose **Area⇒Unlock** from the **Format** menu.
3. If a password is required, you are prompted for the password.

To expand a collapsed area:

1. Click on the boundary line.
2. Choose **Area⇒Expand** from the **Format** menu.

Once an area is unlocked and expanded, you can make whatever changes you want to just as freely as you would elsewhere in your worksheet.

Tip When you lock an area without a password, anyone can unlock it by simply choosing **Area⇒Unlock** from the **Format** menu.

Deleting an Area

You can delete a lockable area just as you would any other region. To do so:

1. Make sure the area is unlocked. You cannot delete a locked area.
2. Select either of the two lines indicating the extent of the locked area by dragging the mouse across it.
3. Choose **Cut** from the **Edit** menu or click  on the Standard toolbar.

Hyperlinks

Mathcad allows you to create *hyperlinks*—that is, to create “hotspots” in your Mathcad worksheets that, when double-clicked, open Mathcad worksheets or other files.

Creating Hyperlinks Between Worksheets

You can create a hyperlink from any Mathcad region, such as a text region or a graphic element, to any Mathcad worksheet. When you double-click the hyperlink, Mathcad opens the Mathcad worksheet designated by the hyperlink. In this way you can connect groups of related worksheets in a form similar to Mathcad’s Electronic Books, or simply cross-reference a related Mathcad worksheet from within the current worksheet.

You have two options for the appearance of the linked worksheet when you double-click the hyperlink:

- The hyperlinked worksheet can open in a full-sized Mathcad worksheet window that overlays the current worksheet window and allows you to edit its contents.
- The hyperlinked worksheet can open in a small *pop-up window* that displays the contents of the worksheet, but does not allow you to edit its contents. This type of hyperlink is called a *Popup*.

Mathcad can follow a hyperlink to any worksheet, whether it is stored on a local drive, a network file system, or the World Wide Web.

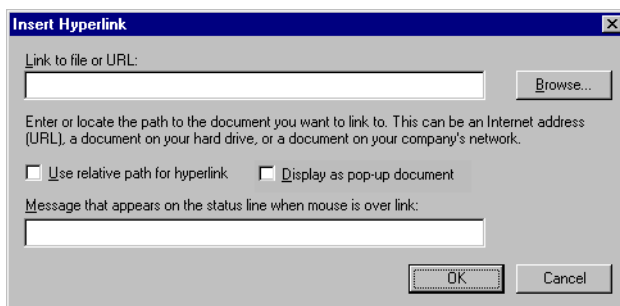
To create a hyperlink, first specify the hyperlink by:

1. Selecting a piece of text, or
2. Clicking anywhere in an equation or graphics region, or
3. Placing the insertion point anywhere within an entire text region.

Tip In general, either selected text or an embedded graphic works best as a hyperlink to another worksheet.

The next step is to specify the target. To do so:

1. Choose **Hyperlink** from the **Insert** menu. Mathcad opens the Insert Hyperlink dialog box.



2. Click “Browse” to locate and select the target worksheet. Alternatively, enter the complete path to a worksheet in the empty text box at the top of the dialog box, or enter an Internet address (URL) to create a hyperlink to a file on the World Wide Web.
3. Click “Use relative path for hyperlink” to store the location of the target worksheet relative to the Mathcad worksheet containing the hyperlink. This allows the hyperlink to be valid even if you move the target file and the worksheet containing the hyperlink, but keep the relative directory structure between the two the same.

Note In order for “Use relative path for hyperlink” to be available, you must first save the worksheet in which you are inserting the hyperlink.

4. Check “Display as pop-up document” if you want the target worksheet to open in a small pop-up window.
5. If you want a message to appear on the status line at the bottom of the window when the mouse hovers over the hyperlink, type the message in the text box at the bottom of the dialog box.
6. Click “OK.”

When you double-click a hyperlink, Mathcad opens the target worksheet in the kind of window (either pop-up or full) you specified. Close a pop-up window by clicking on the close box in the upper right corner.

To change any aspects of a hyperlink—for example, if you move the target worksheet and still want the hyperlink to work—click the hyperlink and choose **Hyperlink** from the **Insert** menu. Make any changes you wish in the Edit Hyperlink dialog box.

To remove a hyperlink, click the hyperlink and choose **Hyperlink** from the **Insert** menu. Click “Remove Link” in the dialog box. Mathcad removes all traces of the link.

Note If you launch a hyperlink from selected text, Mathcad underlines the text and makes it bold to indicate the existence of a hyperlink. Mathcad changes the mouse pointer to a “hand” cursor when you hover over any hyperlink, and any message you specified appears on the status line at the bottom of the window when the cursor is over the hyperlink.

Creating Hyperlinks to Other Files

The methods described in the previous section can create a hyperlink not only from one Mathcad worksheet to another, but also from a Mathcad worksheet to any other file type, either on a local or network file system or on the World Wide Web. Use this

feature to create Electronic Books, as described in “Creating an Electronic Book,” or compound documents that contain not only Mathcad worksheets but word processing files, animation files—any file type that you want.

Note When you double-click a hyperlink to a file other than a Mathcad worksheet, you launch either the application that created the file or an application associated with a file of that type in the Windows Registry. You cannot display such hyperlinked files within a pop-up window.

Creating an Electronic Book

As described in Chapter 3, “On-Line Resources,” an Electronic Book is a hyperlinked collection of Mathcad worksheets. When you open an Electronic Book in Mathcad, it opens in its own window. The Electronic Book has a table of contents and an index as well as other browsing features which you can access using the buttons on the toolbar in the window. The worksheets in an Electronic Book are live so a reader can experiment directly within the book.

If you have several Mathcad worksheets which you want to collect together and you have Mathcad Professional, you can create your own Electronic Book. The steps to creating an Electronic Book are easy; they include:

1. Creating individual Mathcad files
2. Preparing a Table of Contents
3. Adding hyperlinks between appropriate files
4. Creating an .HBK file to specify the order of the files in the book
5. Developing an index

Each step is explained in more detail below. The process of creating a sample Electronic Book called “Explorations in Algebra” is described. After you’ve created an Electronic Book, you or others can open it in Mathcad and navigate through it using the toolbar buttons on the Electronic Book window.

Step 1: Creating Mathcad files

Create as many Mathcad worksheets as you would like to include in your book. Put all the worksheets into a special folder designated for your Electronic Book.

Tip If you want to include data from a data file in a worksheet, you should insert an input table and import the data into it. This will ensure your book is readable even if it is on a CD or on a Web site. Refer to “Importing Once from a Data File” on page 196 for information on importing data to an input table.

For example, for the Electronic Book titled “Explorations in Algebra,” you have created three Mathcad worksheets called FRACTIONS.MCD, PRIMES.MCD, and SQUARE.MCD. Store these files in a folder called ALGEBRA.

To create a consistent look to your worksheets, you can create and use templates and text styles. Refer to “Worksheets and Templates” on page 75 and “Text Styles” on page 61 for more information.

Steps 2 and 3: Preparing a Table of Contents and adding hyperlinks

Prepare a file called CONTENTS.MCD containing a list of the topics that describe the worksheets in your Electronic Book. Then insert hyperlinks from the list of topics to the worksheets. Follow the steps on page 87 for inserting hyperlinks.

For example, the “Explorations in Algebra” book, CONTENTS.MCD could contain the following text:

	Explorations in Algebra
Continued Fractions	
Searching for Prime Numbers	
Journey Around the Square	

You would create hyperlinks from “Continue Fractions,” “Searching for Prime Numbers,” and from “Journey Around the Square” to the files FRACTIONS.MCD, PRIMES.MCD, and SQUARE.MCD, respectively.

The table of contents allows readers to browse to the sections as they prefer.

Step 4: Creating an HBK file

An HBK file for a Mathcad Electronic Book serves the same function as a traditional book binding; it specifies the order for the individual files in the book as the reader browses through it. To create an HBK file:

1. Use any text editor or word processing application.
2. Enter the following four required keywords at the beginning of the HBK file:
.version 2000
.title
SPLASH
TOC

Note that SPLASH and TOC must be capitalized. In the second column of these lines, follow .title with the title of the book; follow SPLASH and TOC with the names of .MCD files. See Figure 7-2. Use tabs to separate the columns.

3. Enter a line for each file in the book with three columns. Use tabs to separate columns. In the first column, type the logical name of the file (the filename without the extension). In the second column, type the filename. In the third column, type a colon (:), a space, and the title of the file that should appear in the title bar of the book window. See Figure 7-2.
4. Save the file as a text-only file with an HBK extension and a name that is the same as the name of your folder containing the Mathcad worksheets that are included in the book. For example, if the folder called ALGEBRA contains all the worksheets for the Electronic Book “Explorations in Algebra,” save the HBK file as ALGEBRA.HBK.

5. Look for a subfolder called HANDBOOK located under the Mathcad Professional folder. If you do not have a subfolder called HANDBOOK, create one.
6. Place the HBK file and the folder containing your worksheets in the HANDBOOK folder. For example, place ALBEGRA.HBK and the ALGEBRA folder in the HANDBOOK folder.

Tip You can create an HBK file using any text editor or word processing application. Just be sure to save the HBK file as text-only.

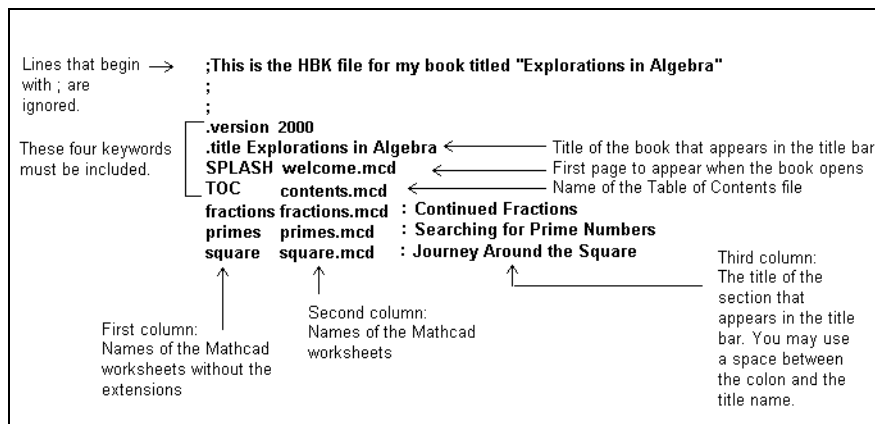


Figure 7-2: Sample HBK file.

Tips for creating an HBK file:

- Limit the worksheet filenames to eight characters to avoid problems some networks may have reading long filenames.
- If the titles of your files are very long, abbreviate the title so it is not cut off when it appears at the top of the window. Think of this as the running head or header that appears at the top of book page.
- You may not include any blank lines in the HBK file.
- Files to which hyperlinks popup rather than jump and files linked to from subdirectories do not have to be listed in the HBK file.
- Any lines beginning with a semicolon is ignored.
- If you include a space and the keyword “skip” following the name of a file in the second column of the HBK file, a reader of the book will not be able to browse to that file. This is helpful if you want people to link to a file but not browse to it.

Step 5 (optional): Developing a searchable index

You can create a searchable phrase index for your Electronic Book much as the author of a printed book creates an index. To see an example of a searchable index, choose

Resource Center from the Help menu and click on the  button.

Note To create an index, use a DOS utility called NEWDICTIONARY.EXE located in your Mathcad directory. You should only create a searchable index if you are comfortable working in DOS.

To create a searchable index:

1. Create a text file with any name you wish and the extension .TXT.
2. Type entries into the text file. For example, for the “Explorations in Algebra” book type entries such as those shown in Figure 7-3. Each line consists of an index entry, a region number enclosed in | | characters, and a filename.

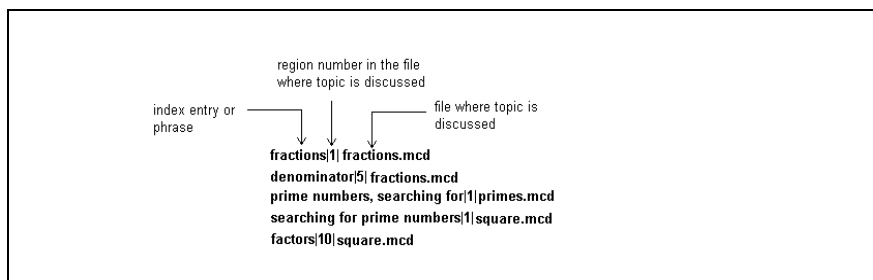


Figure 7-3: Sample index file.

To find the region number in a Mathcad file, open the file in Mathcad, click inside the region, and press **[Ctrl] [Shift] L**. You will see a message on the status bar at the bottom of the Mathcad window that says, for example, “Region Label is 3, Region Count is 2.” The region number, 2 in this example, is the Region Count number.

Tip If you are using unusual terms for a concept, you may want to include standard terms that readers might be looking for. The more entries you add to the index, the more helpful it will be for your users. It is also a good idea to list terms in multiple ways. For example, the topic “Prime Numbers” might be indexed under “numbers, prime” and “prime numbers.”

3. Before processing the index file, you need to sort it alphabetically. You can do this manually or use the sort routine from a number a variety of other applications such as Microsoft Word.
4. Place the following files in the HANDBOOK folder:
 - NEWDICTIONARY.EXE (index utility located in the main Mathcad folder)
 - the TXT file
 - the HBK file
5. In MSDOS or Command Prompt mode, change to the \HANDBOOK folder and type:
newdict index.txt book.hbk.

If there is an error in the index file, NEWDICTIONARY.EXE stops its execution and issues a line number on which the error occurs. If an error occurs, check and fix errors in the TXT or HBK file and run NEWDICTIONARY.EXE until it executes completely without issuing an error.

Note Errors will occur if there are any blank lines in either the index or the HBK file. Many errors also occur because of discrepancies between filenames in the index file and the HBK file or because filenames are missing.

6. This process will create two files, OUT.DCT and OUT.RFS. Rename them to correspond to the name of the HBK file and folder for your Electronic Book. For example, if you have a folder called ALGEBRA and an HBK file called ALGEBRA.HBK, rename the files to ALGEBRA.DCT and ALGEBRA.RFS.
7. Place the DCT and RFS files inside your Electronic Book folder. For example, put them in the ALGEBRA folder.

Note If you make any changes to the files in the Electronic Book that affect the position or number of regions or if you make any changes to the HBK file, you must re-create the index.

Additional information about creating Electronic Books is available in the Resource Center and on the MathSoft FTP site at **<ftp://ftp.mathsoft.com/pub/author>**.

Opening your Electronic Book

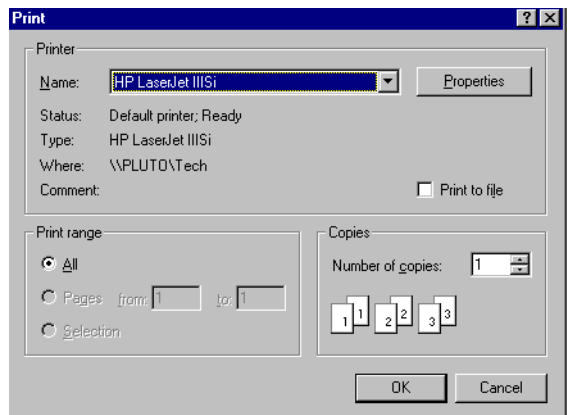
To open your Electronic Book:

1. Start Mathcad and choose **Open Book** from the **Help** menu.
2. Browse to the location of your .HBK file and click on it.
3. Click “Open.”

Your book opens in the Electronic Book window. Use the buttons on the toolbar at the top of the window to navigate. For example, click  to go to the Table of Contents or click  to open the searchable index. For more information on Electronic Books and the navigation tools, refer to Chapter 3, “On-Line Resources.”

Printing and Mailing

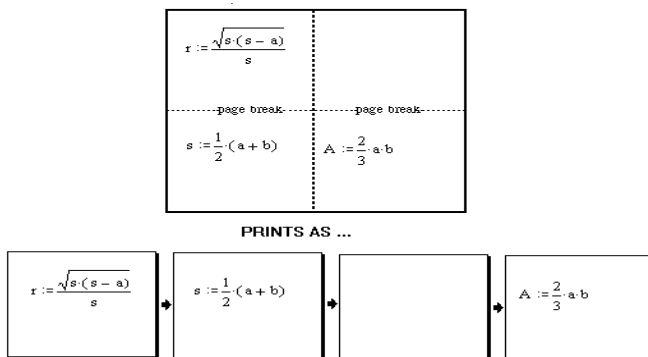
To print a Mathcad worksheet, choose **Print** from the **File** menu. The Print dialog box lets you control whether to print the entire worksheet, selected pages, or selected regions; what printer to print on; and the number of copies to print. The particular dialog box you see depends on the printer you’ve selected. A typical dialog box is shown at right.



Printing Wide Worksheets

Mathcad worksheets can be wider than a sheet of paper, since you can scroll as far to the right as you like in a Mathcad worksheet and place equations, text, and graphics wherever you like. As you scroll horizontally, however, you see dashed vertical lines appearing to indicate the right margins of successive “pages” corresponding to the settings for your printer. The sections of the worksheet separated by the dashed vertical lines print on separate sheets of paper, yet the page number at the bottom of the Mathcad window does not change as you scroll to the right.

You can think of the worksheet as being divided into vertical strips. Mathcad begins printing at the top of each strip and continues until it reaches the last region in this strip. It prints successive strips left to right. Note that certain layouts will produce one or more blank pages.



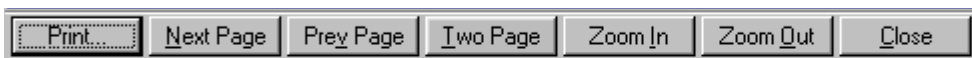
Tip You can control whether a wide worksheet is printed in its entirety or in a single page width. To do so, choose **Page Setup** from the **File** menu to open the Page Setup dialog box. Then, to suppress printing of anything to the right of the right margin, check “Print single page width.”

You can ask Mathcad to print a range of pages in the worksheet by using the Print dialog box. The page numbers in the dialog box refer only to horizontal divisions. For example, if your worksheet looks like that shown above, and you ask Mathcad to print page 2, you will see two sheets of paper corresponding to the lower-left and lower-right quadrants.

Tip Mathcad allows you to change the display of some operators including the **:=**, the bold equals, the derivative operator, and the multiplication operator. Before you print, you can choose **Options** from the **Math** menu and click on the Display tab to change the appearance of these operators. This can make your printout clearer to someone unfamiliar with Mathcad notation.

Print Preview

To check your worksheet’s layout before printing, choose **Print Preview** from the **File** menu or click  on the Standard toolbar. The Mathcad window shows the current section of your worksheet in miniature, as it will appear when printed, with a strip of buttons across the top of the window:



To print your worksheet from this screen, click “Print.” Click “Close” to go back to the main worksheet screen. The remaining buttons give you more control over the preview.

Tip Although you can use the “Zoom In” and “Zoom Out” buttons to magnify the worksheet, you can also magnify the worksheet by moving the cursor onto the previewed page so that the cursor changes to a magnifying glass. Then click the mouse. Click again to magnify your worksheet even more. Once you’re at the maximum magnification, clicking on the page de-magnifies it.

Note You cannot edit the current page or change its format in the Print Preview screen. To edit the page or change its format, return to the normal worksheet view by clicking “Close.”

Mailing

If you’re connected to a mail system that’s compatible with Microsoft’s Mail API (MAPI), you can use Mathcad to direct that system to send an electronic mail message and your current Mathcad worksheet. When you use Mathcad to send a worksheet by electronic mail, the recipient receives the worksheet as a file attached to an ordinary e-mail message, provided that the recipient’s mail system uses the same encoding technique as yours.

Tip The settings in your mail system determine how Mathcad worksheets are attached to or encoded in the mail message. We recommend that you use an encoding method such as MIME or UUENCODE, if available, to attach Mathcad worksheets to mail messages.

To send a Mathcad worksheet by electronic mail:

1. Open the worksheet you want to send.
2. Choose **Send** from the **File** menu.

Once you do so, your mail system launches and creates a new message with your worksheet as an attachment. You should then enter the text of your mail message, the address of the recipient, and any other information allowed by your mail system.

Chapter 8

Calculating in Mathcad

- τ Defining and Evaluating Variables
- τ Defining and Evaluating Functions
- τ Units and Dimensions
- τ Working with Results
- τ Controlling Calculation
- τ Animation
- τ Error Messages

Defining and Evaluating Variables

When you type an expression into a worksheet, you are usually doing one of two things:

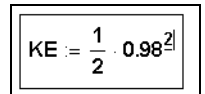
- You could be typing a variable or function name and assigning some value to it.
- You could be typing an equation and asking Mathcad to give you the answer.

We introduce these topics in this and the following section. See “Evaluating Expressions Numerically” on page 99 for details on numerical evaluation.

Defining a Variable

A variable definition defines the value of a variable everywhere below and to the right of the definition. To define a variable, follow these three steps:

1. Type the variable name to be defined. Chapter 4, “Working with Math,” contains a description of valid variable names.
2. Press the colon (:) key, or click  on the Calculator toolbar. The definition symbol (:=) appears with a blank placeholder to the right.
3. Type an expression to complete the definition. This expression can include numbers and any previously defined variables and functions.



The left-hand side of a “:=” can contain any of the following:

- A simple variable name like x .
- A subscripted variable name like v_i .

- A matrix whose elements are either of the above. For example, $\begin{bmatrix} x \\ y_1 \end{bmatrix}$. This technique allows you to define several variables at once: each element on the right-hand side is assigned simultaneously to the corresponding element on the left-hand side.
- A function name with an argument list of simple variable names. For example, $f(x, y, z)$. This is described further in the next section.
- A superscripted variable name like $\mathbf{M}^{(1)}$.

Built-in Variables

Mathcad includes *predefined* or *built-in variables*. Predefined variables can have a conventional value, like π and e , or be used as system variables to control how Mathcad works. See “Predefined Variables” on page 311 in the Appendices for a list of built-in variables in Mathcad.

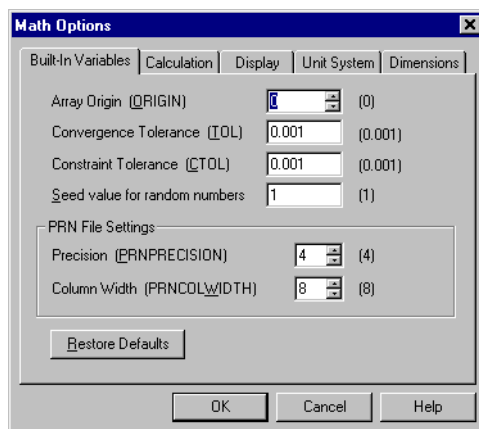
Note In addition to the built-in variables described here, Mathcad treats the names of all built-in *units* as predefined variables. See “Units and Dimensions” on page 106.

Although Mathcad’s predefined variables already have values when you start Mathcad, you can still redefine them. For example, if you want to use a variable called e with a value other than the one Mathcad provides, enter a new definition, like $e := 2$. The variable e takes on the new value everywhere in the worksheet below and to the right of the new definition. Alternatively, create a global definition for the variable as described in “Global Definitions” on page 100.

Note Mathcad’s predefined variables are defined for all fonts, sizes, and styles. This means that if you redefine e as described above, you can still use **e**, for example, as the base for natural logarithms. Note, however, that Greek letters are not included.

You can modify many of Mathcad’s built-in variables without having to explicitly define them in your worksheet. To do so, choose **Options** from the **Math** menu, and click the Built-In Variables tab on the Math Options dialog box.

To set new starting values for any of these variables, enter a new value in the appropriate text box and click “OK.” Then choose **Calculate Worksheet** from the **Math** menu to ensure that all existing equations take the new values into account.

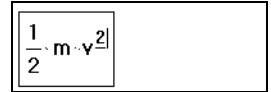


The numbers in brackets to the right of the variable names represent the default values for those variables. To restore these default values for the built-in variables listed in the dialog box, click “Restore Defaults” and then click “OK.”

Evaluating Expressions Numerically

To evaluate an expression numerically, follow these steps:

1. Type an expression containing any valid combination of numbers, variables, and functions. Any variables or functions in this expression should be defined earlier in the worksheet.
2. Press the “=” key, or click  on the Calculator toolbar. Mathcad computes the value of the expression and shows it after the equal sign.


$$\frac{1}{2} \cdot m \cdot v^2$$

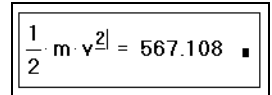
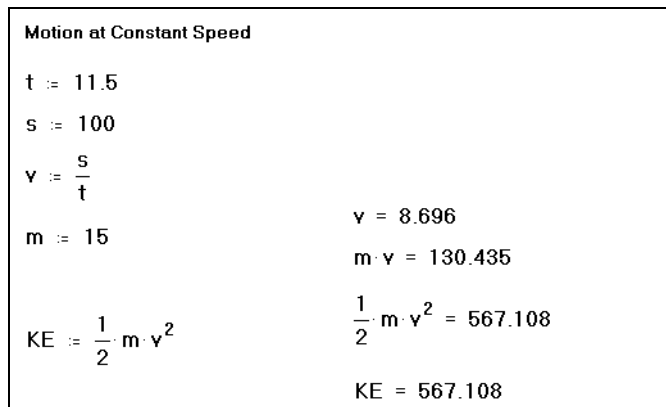

$$\frac{1}{2} \cdot m \cdot v^2 = 567.108 \blacksquare$$

Figure 8-1 shows some results calculated from preceding variable definitions.



Motion at Constant Speed

$$t := 11.5$$
$$s := 100$$
$$v := \frac{s}{t}$$
$$m := 15$$
$$KE := \frac{1}{2} \cdot m \cdot v^2$$
$$v = 8.696$$
$$m \cdot v = 130.435$$
$$\frac{1}{2} \cdot m \cdot v^2 = 567.108$$
$$KE = 567.108$$

Figure 8-1: Calculations based on simple variable definitions.

Tip Whenever you evaluate an expression, Mathcad shows a final placeholder at the end of the equation. You can use this placeholder for unit conversions, as explained in “Working with Results” on page 110. As soon as you click outside the region, Mathcad hides the placeholder.

How Mathcad Scans a Worksheet

Mathcad scans a worksheet the same way you read it: left to right and top to bottom. This means that a variable or function definition involving a “:=” affects everything below and to the right of it.

To see the placement of regions more clearly in your worksheet, choose **Regions** from the **View** menu. Mathcad displays blank space in gray and leaves regions in your background color.

Figure 8-2 shows examples of how placement of equations in a worksheet affects the evaluation of results. In the first evaluation, both x and y are highlighted (Mathcad shows them in red on screen) to indicate that they are undefined. This is because the definitions for x and y lie below where they are used. Because Mathcad scans from top to bottom, when it gets to the first equation, it doesn’t know the values of x and y .

The second evaluation, on the other hand, is below the definitions of x and y . By the time Mathcad gets to this equation, it has already assigned values to both x and y .

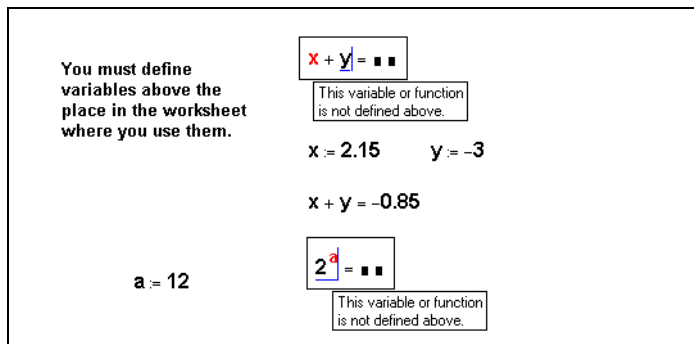


Figure 8-2: Mathcad evaluates equations from top to bottom in a worksheet. Undefined variables are highlighted.

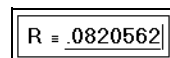
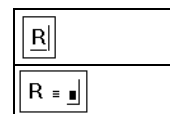
Note You can define a variable more than once in the same worksheet. Mathcad simply uses the first definition for all expressions below the first definition and above the second. For expressions below the second definition and above the third, Mathcad uses the second definition, and so on.

Global Definitions

Global definitions are exactly like local definitions except that they are evaluated before any local definitions. If you define a variable or function with a global definition, that variable or function is available to all local definitions in your worksheet, regardless of whether the local definition appears above or below the global definition.

To type a global definition, follow these steps:

1. Type a variable name or function to be defined.
2. Press the tilde (~) key, or click  on the Evaluation toolbar. The global definition symbol appears.
3. Type an expression. The expression can involve numbers or other globally defined variables and functions.



You can use global definitions for functions, subscripted variables, and anything else that normally uses the definition symbol “:=”.

This is the algorithm that Mathcad uses to evaluate all definitions, global and otherwise:

- First, Mathcad takes one pass through the entire worksheet from top to bottom. During this first pass, Mathcad evaluates global definitions only.
- Mathcad then makes a second pass through the worksheet from top to bottom. This time, Mathcad evaluates all definitions made with “:=” as well as all equations containing “=” and “≡”. Note that during this pass, global definitions do not use any local definitions.

Note A global definition of a variable can be overridden by a local definition of the same variable name with the definition symbol “:=”.

Figure 8-3 shows the results of a global definition for the variable R which appears at the bottom of the figure.

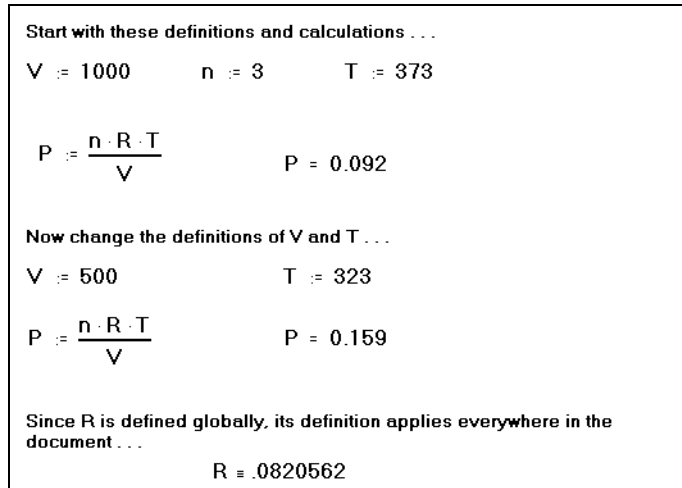


Figure 8-3: Using the global definition symbol.

Although global definitions are evaluated before any local definitions, Mathcad evaluates global definitions the same way it evaluates local definitions: top to bottom and left to right. This means that whenever you use a variable to the right of a “≡”:

- that variable must also have been defined with a “≡,” and
- the variable must have been defined *above* the place where you are trying to use it.

Otherwise, the variable is marked in red to indicate that it is undefined.

Tip It is good practice to allow only one definition for each global variable. Although you can do things like define a variable with two different global definitions or with one global and one local definition, this may make your worksheet difficult for others to understand.

Range Variables

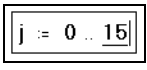
Iterative processes in Mathcad worksheets depend on *range variables*. Except for the way it is defined, a range variable looks just like a conventional variable. The difference is that a conventional variable takes on only one value. A range variable, on the other hand, takes on a range of values separated by uniform steps. For example, you could define a range variable to go from -4 through 4 in steps of 2 . If you now use this range variable in an expression, Mathcad evaluates that expression five times, once for each value taken by the range variable.

Range variables are crucial to exploiting Mathcad’s capabilities to their fullest. This section shows how to define and use range variables to perform iteration. For a

description of more advanced iterative operations made possible by the programming operators in Mathcad Professional, turn to Chapter 15, “Programming.”

Defining and using range variables

To define a range variable, type the variable name followed by a colon and a range of values. For example, here’s how to define the variable j ranging from 0 to 15 in steps of 1:

1. Type **j** and then press the colon key (:), or click  on the Calculator toolbar. The empty placeholder indicates that Mathcad expects a definition for j . At this point, Mathcad does not know whether j is to be a conventional variable or a range variable. 
2. Type **0**. Then press the semicolon key (;), or click  on the Calculator toolbar. This tells Mathcad that you are defining a range variable. Mathcad displays the semicolon as two periods “..” to indicate a range. Complete the range variable definition by typing **15** in the remaining placeholder. 

This definition indicates that j now takes on the values 0, 1, 2...15. To define a range variable that changes in steps other than 1, see the section “Types of ranges” on page 103.

Once you define a range variable, it takes on its complete range of values *every time you use it*. If you use a range variable in an equation, for example, Mathcad evaluates that equation once for each value of the range variable.

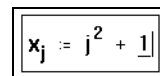
You must define a range variable exactly as shown above. There must be:

- a variable name on the left,
- either a “:=” or a “≡” in the middle, and
- a valid range on the right.

Note You *cannot* define a variable in terms of a range variable. For example, if after having defined j as shown you now define $i := j + 1$, Mathcad assumes you are trying to set a scalar variable equal to a range variable and marks the equation with an appropriate error message.

One application of range variables is to fill up the elements of a vector or matrix. You can define vector elements by using a range variable as a subscript. For example, to define x_j for each value of j :

- Type **$x[j : j^2[Space]+1$** .



$$x_j := j^2 + 1$$

Figure 8-4 shows the vector of values computed by this equation. Since j is a range variable, the entire equation is evaluated once for each value of j . This defines x_j for each value of j from 0 to 15.

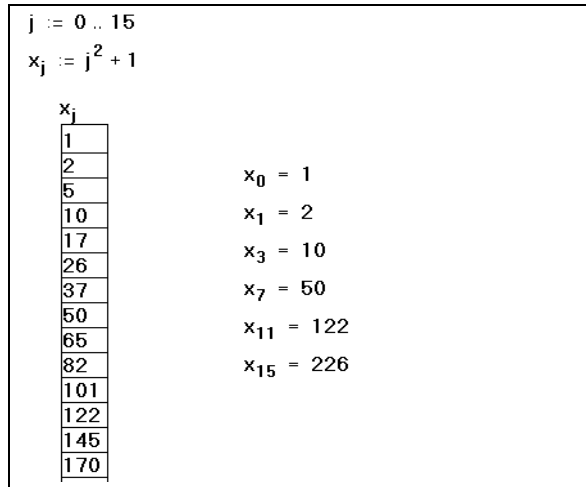


Figure 8-4: Using a range variable to define the values of a vector.

To understand how Mathcad computes with range variables, keep in mind this fundamental principle:

If you use a range variable in an expression, Mathcad evaluates the expression once for each value of the range variable.

If you use two or more range variables in an equation, Mathcad evaluates the equation once for each value of each range variable.

Tip Mathcad takes longer to compute equations with ranged expressions since there may be many computations for each equation. While Mathcad is computing, the mouse pointer changes its appearance. To learn how to interrupt a calculation in progress, see “Interrupting Calculations” on page 118.

Types of ranges

The definition of j in the previous section, ranging from 0 to 15, is an example of the simplest type of range definition. But Mathcad permits range variables with values ranging from any value to any other value, using any constant increment or decrement.

To define a range variable with a step size other than 1, type an equation of this form:

$$k:1, 1.1;2$$

This appears in your worksheet window as:

$$k := 1, 1.1 \dots 2$$

In this range definition:

- The variable k is the name of the range variable itself.
- The number 1 is the first value taken by the range variable k .

- The number 1.1 is the second value in the range. *Note that this is not the step size.* The step size in this example is 0.1, the difference between 1.1 and 1. If you omit the comma and the 1.1, Mathcad assumes a step size of one in whatever direction (up or down) is appropriate.
- The number 2 is the last value in the range. In this example, the range values are constantly increasing. If instead you had defined $k := 10 \dots 1$, then k would count down from 10 to 1. If the third number in the range definition is not an even number of increments from the starting value, the range will not go beyond it. For example, if you define $k := 10, 20 \dots 65$ then k takes values 10, 20, 30, . . . , 60.

Note You can use arbitrary scalar expressions in range definitions. However, these values must always be *real* numbers. Also note that if you use a fractional increment for a range variable, you will not be able to use that range variable as a subscript because subscripts must be integers.

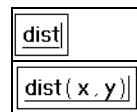
Defining and Evaluating Functions

As described in Chapter 10, “Built-in Functions,” Mathcad has an extensive built-in function set. You can augment Mathcad’s built-in function set by defining your own functions.

You define a function in much the same way you define a variable. The name goes on the left, a definition symbol goes in the middle, and an expression goes on the right. The main difference is that the name includes an *argument list*. The example below shows how to define a function called *dist*(x, y) that returns the distance between the point (x, y) and the origin.

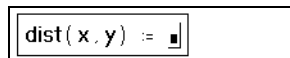
To type such a function definition:

1. Type the function name.
2. Type a left parenthesis followed by one or more names separated by commas. Complete this argument list by typing a right parenthesis.

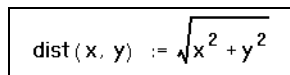


Note It makes no difference whether or not the names in the argument list have been defined or used elsewhere in the worksheet. What is important is that these arguments *must be names*. They cannot be more complicated expressions.

- Press the colon (:) key, or click  on the Calculator toolbar. You see the definition symbol (:=).



- Type an expression to define the function. In this example, the expression involves only the names in the argument list. In general though, the expression can contain any previously defined functions and variables as well.



Once you have defined a function, you can use it anywhere below and to the right of the definition, just as you would use a variable.

When you evaluate an expression containing a function, as shown in Figure 8-5, Mathcad:

1. evaluates the arguments you place between the parentheses,
2. replaces the dummy arguments in the function definition with the actual arguments you place between the parentheses,
3. performs whatever arithmetic is specified by the function definition,
4. returns the result as the value of the function.

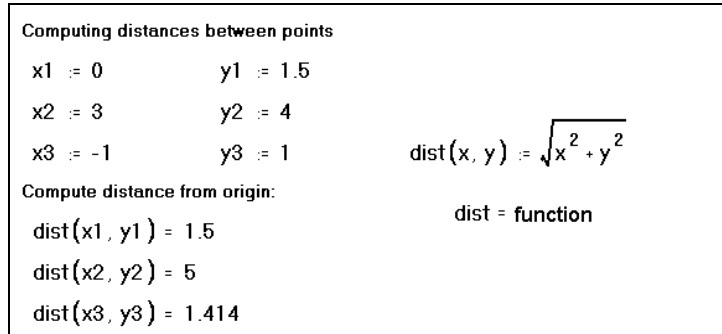


Figure 8-5: A user-defined function to compute the distance to the origin.

Note As shown in Figure 8-5, if you type only the name of a function without its arguments, Mathcad returns the word “function.”

The arguments of a user-defined function can represent scalars, vectors, or matrices.

For example, you could define the distance function as $dist(v) := \sqrt{v_0^2 + v_1^2}$. This is an example of a function that accepts a vector as an argument and returns a scalar result. See Chapter 11, “Vectors, Matrices, and Data Arrays,” for more information.

Note User-defined function names are font and case sensitive. The function **f**(x) is different from the function f(x) and SIN(x) is different from sin(x). Mathcad’s built-in functions, however, are defined for all fonts (except the Symbol font), sizes, and styles. This means that **sin**(x), sin(x), and **sin**(x) all refer to the same function.

Variables in User-Defined Functions

When you define a function, you don’t have to define any of the names in the argument list since you are telling Mathcad *what to do* with the arguments, not what they are. When you define a function, Mathcad doesn’t even have to know the types of the arguments—whether the arguments are scalars, vectors, matrices, and so on. It is only when Mathcad *evaluates* a function that it needs to know the argument types.

However, if in defining a function you use a variable name that *is not* in the argument list, you must define that variable name above the function definition. The value of that variable at the time you make the function definition then becomes a permanent part of the function. This is illustrated in Figure 8-6.

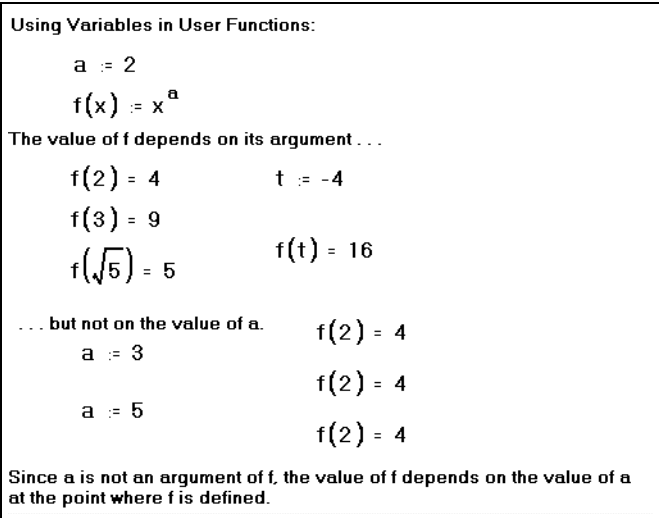


Figure 8-6: The value of a user function depends on its arguments.

If you want a function to depend on the value of a variable, you must include that variable as an argument. If not, Mathcad just uses that variable's fixed value at the point in the worksheet where the function is defined.

Recursive Function Definitions

Mathcad supports *recursive* function definitions—you may define the value of a function in terms of a previous value of the function. As shown in Figure 8-7, recursive functions are useful for defining arbitrary periodic functions, as well as elegantly implementing numerical functions like the factorial function

Note that a recursive function definition should always have at least two parts:

- An initial condition that prevents the recursion from going forever.
- A definition of the function in terms of some previous value(s) of the function.

Note If you do not specify an initial condition that stops the recursion, Mathcad generates a “stack overflow” error message when you try to evaluate the function.

Pro The programming operators in Mathcad Professional also support recursion. See the section “Programs Within Programs” in Chapter 15 for examples.

Units and Dimensions

When you first start Mathcad, a complete set of units is available for your calculations. You can treat these units just like built-in variables. To assign units to a number or expression, just multiply it by the name of the unit.

Mathcad recognizes most units by their common abbreviations. Lists of all of Mathcad's built-in units in several systems of units are in the Appendices. By default Mathcad uses units from the SI unit system (also known as the International System of

A simple recursive function definition.

$\text{factorial}(n) := \text{if}(n \neq 0, 1, n \cdot \text{factorial}(n - 1))$

$\text{factorial}(5) = 120$

Using recursion to define a periodic sawtooth wave.

$F(x) := 2 \cdot x \quad \text{period} := 2$

$G(x) := \text{if}(x < \text{period}, F(x), G(x - \text{period}))$

$x := 0, .01 \dots 10$

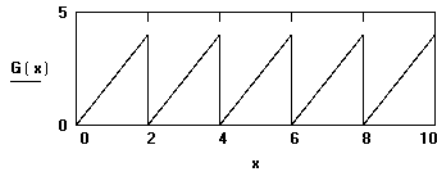


Figure 8-7: Mathcad allows recursive function definitions.

Units) in the *results* of any calculations, but you may use any supported units you wish in creating your expressions. See “Displaying Units of Results” on page 113 for more information about selecting a unit system for results.

For example, type expressions like the following:

mass:75*kg

acc:100*m/s^2

acc_g:9.8*m/s^2

F:mass*(acc + acc_g)

Figure 8-8 shows how these equations appear in a worksheet.

mass := 75·kg

acc := 100· $\frac{\text{m}}{\text{s}^2}$

acc_g := 9.8· $\frac{\text{m}}{\text{s}^2}$

F := mass·(acc + acc_g)

F = $8.235 \times 10^3 \text{ N}$

mass := 75kg <--Mathcad treats the multiplication as implied when you type an expression like **mass:75kg**

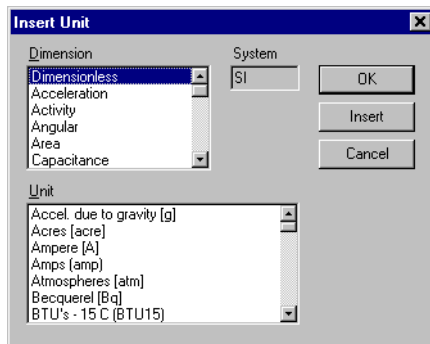
mass = 75 kg

Figure 8-8: Equations using units.

Tip If you define a variable which consists of a number followed immediately by a unit name, you can omit the multiplication symbol; Mathcad inserts a very small space and treats the multiplication as implied. See the definition of mass at the bottom of Figure 8-8.

You can also use the Insert Unit dialog box to insert one of Mathcad's built-in units into any placeholder. To use the Insert Unit dialog box:

1. Click in the empty placeholder and choose **Unit** from the **Insert** menu, or click  on the Standard toolbar. Mathcad opens the Insert Unit dialog box.
2. The list at the bottom shows built-in units, along with their Mathcad names, corresponding to whatever physical quantity is selected in the top scrolling list. When "Dimensionless" is selected at the top, a list of all available built-in units shows on the bottom.
3. If necessary, use the top scrolling list to display only those units corresponding to a particular physical quantity. This makes it easier to find a particular unit or to see what choices are appropriate.
4. In the bottom list, double-click the unit you want to insert, or click the unit you want and then click "Insert." Mathcad inserts that unit into the empty placeholder.



Note Mathcad performs some dimensional analysis by trying to match the dimensions of your selected result with one of the common physical quantities in the top scrolling list. If it finds a match, you'll see all the built-in units corresponding to the highlighted physical quantity in the bottom scrolling list. If nothing matches, Mathcad simply lists all available built-in units on the bottom.

Dimensional Checking

Whenever you enter an expression involving units, Mathcad checks it for dimensional consistency. If you add or subtract values with incompatible units, or violate other principles of dimensional analysis, Mathcad displays an appropriate error message.

For example, suppose you had defined acc as $100 \cdot m/s$ instead of $100 \cdot m/s^2$ as shown at right. Since acc is in units of velocity and acc_g is in units of acceleration, it is inappropriate to add them together. When you attempt to do so, Mathcad displays an error message.

Other unit errors are usually caused by one of the following:

- An incorrect unit conversion.

$mass := 75 \cdot kg$

$acc := 100 \cdot \frac{m}{s}$

$acc_g := 9.8 \cdot \frac{m}{s^2}$

$F := mass \cdot (acc + acc_g)$

The units in this expression do not match.

- A variable with the wrong units.
- Units in exponents or subscripts (for example $v_3 \cdot acre$ or $2^3 \cdot ft$).
- Units as arguments to inappropriate functions (for example, $\sin(0 \cdot henry)$).

Tip If you want to temporarily remove units from an argument, x , divide x by $UnitsOf(x)$. For example, if p is defined as 2 ft then $\sin(p)$ gives an error but $\sin\left(\frac{p}{UnitsOf(p)}\right) = 0.573$.

Defining Your Own Units

Although Mathcad recognizes many common units, you may need to define your own unit if that unit isn't one of Mathcad's built-in units or if you prefer to use your own abbreviation instead of Mathcad's abbreviation.

Note Although absolute temperature units are built into Mathcad, the Fahrenheit and Celsius temperature units are not. See the QuickSheet "Temperature Conversions" in the on-line Resource Center for examples of how to define these temperature scales and to convert between them.

You define your own units in terms of existing units in exactly the same way you would define a variable in terms of an existing variable. Figure 8-9 shows how to define new units as well as how to redefine existing units.

$\text{\AA} := 10^{-10} \cdot \text{m}$ $\mu\text{s} := 10^{-6} \cdot \text{s}$
 $\text{week} := 7 \cdot \text{day}$ $\text{kilo} := 1 \cdot \text{kg}$

Results are displayed by default in terms of the base units of the current unit system, but you can type other units into the units placeholder to see the result in different units.

$\text{\AA} = 1 \times 10^{-10} \text{ m}$ $\text{week} = 6.048 \times 10^5 \text{ s}$
 $\text{week} = 0.019 \text{ yr}$ $\mu\text{s} = 1.667 \times 10^{-8} \text{ min}$
 $\text{kilo} = 1 \text{ kg}$

Figure 8-9: Defining your own units.

Note Since units behave just like variables, you may run into unexpected conflicts. For example, if you define the variable m in your worksheet, you won't be able to use the built-in unit m for meters anywhere below that definition. However, Mathcad automatically displays the unit m in any results involving meters, as described in "Displaying Units of Results" on page 113.

Working with Results

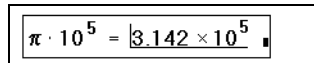
Formatting Results

The way that Mathcad displays numbers (the number of decimal places, whether to use *i* or *j* for imaginary numbers, and so on) is called the *result format*. You can set the result format for a single calculated result or for an entire worksheet.

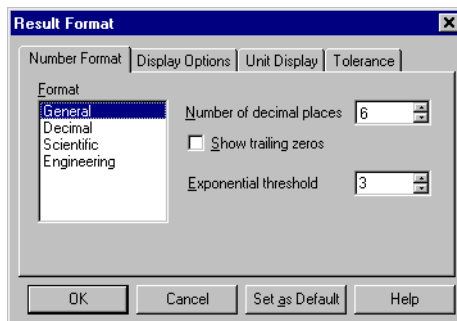
Setting the format of a single result

When you evaluate expressions numerically in Mathcad, results are formatted in the worksheet according to the worksheet default result format. You can modify the format for a single result as follows:

1. Click anywhere in the equation whose result you want to format.

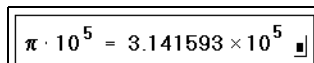

$$\pi \cdot 10^5 = 3.142 \times 10^5$$

2. Choose **Result** from the **Format** menu. Alternatively, double-click the equation itself. The Result Format dialog box appears.



3. Change the desired settings. See below to learn about the various settings in the dialog box. To display a result with six decimal places, you would increase “Number of decimal places” from 3 to 6.

4. Click “OK.” Mathcad redisplay the result using the new format.


$$\pi \cdot 10^5 = 3.141593 \times 10^5$$

To redisplay a result using the worksheet default result format settings, click on the result to enclose the result between the editing lines, delete the equal sign, and press **=** to replace the equal sign. The result is now restored to the default worksheet settings.

Note When the format of a result is changed, only the *appearance* of the result changes in the worksheet. Mathcad continues to maintain full precision internally for that result. To see a number as it is stored internally, click on the result, press **[Ctrl][Shift]N**, and look at the message line at the bottom of the Mathcad window. If you copy a result, however, Mathcad copies the number only to the precision displayed.

Setting worksheet default format

To change the default display of numerical results in your worksheet:

1. Click in a blank part of your worksheet.
2. Choose **Result** from the **Format** menu.
3. Change the desired settings in the Result Format dialog box.
4. Click “OK.”

Mathcad changes the display of all results whose formats have not been explicitly specified.

Alternatively, you can change the worksheet default by clicking on a particular result, choosing Result from the Format menu, changing the settings in the Result Format dialog box, and clicking “Set as Default.”

Tip Changing the worksheet default result format affects only the worksheet you are working in when you make the change. Any other worksheets open at the time retain their own default result formats. If you want to re-use your default result formats in other Mathcad worksheets, save your worksheet as a template as described in Chapter 7, “Worksheet Management.”

The Result Format dialog box

The tabs in the Result Format dialog box lead to pages containing options for formatting various aspects of a result.

The **Number Format** page lets you control the number of decimal places, trailing zeros, and whether a result is in exponential notation. Depending on the format scheme you choose under the Format section, you see different options.

- Choosing **General** lets you control the number of digits to the right of the decimal point, trailing zeros, and exponential threshold. A result is displayed in exponential notation when the exponential threshold is exceeded. You can display trailing zeros to the right of the decimal until you exceed 15 digits total.
- Choosing **Decimal** lets you control the number of digits to the right of the decimal point and never display the results in exponential notation. You can display trailing zeros to the right of the decimal point beyond 15 digits total, but only the first 15 digits are accurate.
- Choosing **Scientific** or **Engineering** lets you control the number of digits to the right of the decimal point and always display results in exponential notation. For Engineering, the exponents are displayed in multiples of three. You can use E-notation for the exponents by choosing “Show exponents as $\pm E\ 000$.” You can display trailing zeros to the right of the decimal point beyond 15 digits total, but only the first 15 digits are accurate.

Note Settings that are grayed can only be changed for the entire worksheet, as described in “Setting worksheet default format” on page 110.

The **Display Options** page lets you control whether arrays are displayed as tables or matrices, whether nested arrays are expanded, and whether i or j is used to indicated imaginary. You can also specify another radix such as Binary or Octal.

The **Unit Display** page gives you options to format units (as fractions) or simplify the units to derived units.

The **Tolerance** page allows you to specify when to hide a real or imaginary part of a result and how small a number has to be for it to display as zero.

On-line Help For more details and examples of the options available on a particular page in the Result Format dialog box, click the Help button at the bottom of the dialog box.

Figure 8-10 shows some examples of formatting options.

$x := 5.2574$ $y := \pi \cdot 10^4$	Definitions
$x = 5.26$	General format, Exponential threshold = 15, Number of decimal places = 2
$x = 5.2574$	Decimal format, Number of decimal places = 4
$x = 5.25740$	Decimal format, Number of decimal places = 5 Show trailing zeros ☒
$y = 3.142 \times 10^4$	Scientific format
$y = 31.416 \times 10^3$	Engineering format
$x = 5.257 \text{ E}+000$	Engineering format, Show exponents as E+000 ☒

Figure 8-10: Several ways to format the same number.

Complex Results

Complex numbers can arise in results if you enter an expression that contains a complex number. Even a Mathcad expression that involves only real numbers can have a complex value. For example, if you evaluate $\sqrt[3]{-1}$, Mathcad returns i . See Figure 8-11 for examples.

$$\sqrt[3]{-1} = -1$$

← nth root operator always returns a real valued root.

$n := 1, 3 \dots 5$

$\exp(n \cdot i \cdot 60 \cdot \text{deg})$

0.5 + 0.866i
-1
0.5 - 0.866i

$\exp(n \cdot i \cdot 60 \cdot \text{deg})^3$

-1
-1
-1

← Just as every number has two square roots, every number also has three cube roots.

Here, "i" is the imaginary unit. Type "i" to enter it.

$$(-1)^{\frac{1}{3}} = 0.5 + 0.866i$$

← Returns "principal value" of cube root, the one corresponding to n=1 in the above list.

Figure 8-11: Examples of complex results.

Note When complex numbers are available, many functions and operators we think of as returning unique results become multivalued. In general, when a function or operator is multivalued, Mathcad returns the *principal value*: the value making the smallest positive angle relative to the positive real axis in the complex plane. For example, when it evaluates $(-1)^{1/3}$, Mathcad returns $.5 + .866i$ despite the fact that we commonly think of the cube root of -1 as being -1 . This is because the number $.5 + .866i$ makes an angle of only 60 degrees from the positive real axis. The number -1 , on the other hand, is 180 degrees from the positive real axis. Mathcad's n th root operator returns -1 in this case, however.

Displaying Units of Results

Mathcad by default displays results in terms of the fundamental units of the unit system you're working with. Mathcad offers the following unit system choices: SI, CGS, MKS, U.S. customary units, or no unit system (see below).

Tip Check “Simplify units when possible” in the Result Format dialog box (see page 110) to see units in a result expressed in terms of derived units rather than in base units. Check “Format units” to see units in a result displayed as a built-up fraction containing terms with positive exponents only rather than as a product of units with positive and negative exponents.

You can have Mathcad redisplay a particular result in terms of any of Mathcad's built-in units. To do so:

1. Click in the result. You'll see an empty placeholder to its right. This is the *units placeholder*.
2. Click the units placeholder and choose **Unit** from the **Insert** menu, or click  on the Standard toolbar. Mathcad opens the Insert Unit dialog box. This is described in “Units and Dimensions” on page 106.
3. Double-click the unit in which you want to display the result. Mathcad inserts this unit in the units placeholder.

Note For some engineering units—such as *hp*, *cal*, *BTU*, and *Hz*—Mathcad adopts one common definition for the unit name but allows you to insert one of several alternative unit names, corresponding to alternative definitions of that unit, in your results. In the case of horsepower, for example, Mathcad uses the U.K. definition of the unit *hp* but gives you several variants, such as water horsepower, metric horsepower, boiler horsepower, and electric horsepower.

Another way to insert a unit is to type its name directly into the units placeholder. This method is more general since it works not only for built-in units but also for units you've defined yourself and for combinations of units.

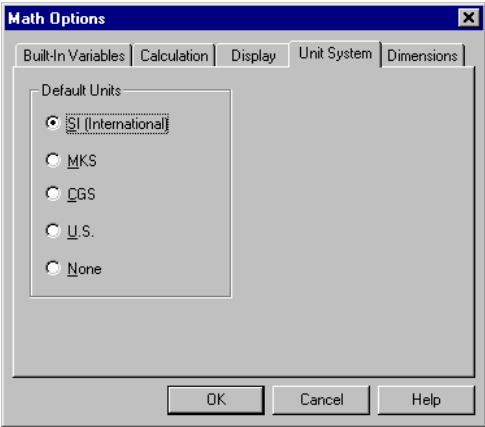
Unit systems

When you start Mathcad, the SI system of units is loaded by default. This means that when you use the equal sign to display a result having units, Mathcad automatically displays the units in the result in terms of base or derived SI units.

You can have Mathcad display results in terms of the units of any of the other built-in unit systems in Mathcad: CGS, US customary, MKS, or no unit system at all. To do so, choose **Options** from the **Math** menu and click the Unit System tab.

Select the default unit system in which you want to display results. The SI unit system, widely used by scientists and engineers in many countries, provides two additional base units over the other systems, one for luminosity (*candela*) and one for substance (*mole*), and the base SI electrical unit (*ampere*) differs from the base electrical unit in the other systems (*coulomb*).

The following table summarizes the base units available in Mathcad’s unit systems:



Unit System	Base Units
SI	<i>m, kg, s, A, K, cd, and mole</i>
MKS	<i>m, kg, sec, coul, and K</i>
CGS	<i>cm, gm, sec, coul, and K</i>
U.S.	<i>ft, lb, sec, coul, and K</i>
None	Displays results in terms of fundamental dimensions of length, mass, time, charge, and absolute temperature. All built-in units are disabled.

The standard SI unit names—such as *A* for *ampere*, *L* for *liter*, *s* for *second*, and *S* for *siemens*—are generally available only in the SI unit system. Many other unit names are available in all the available systems of units. For a listing of which units are available in each system, see the Appendices. Mathcad includes most units common to scientific and engineering practice. Where conventional unit prefixes such as *m-* for *milli-*, *n-* for *nano-*, etc. are not understood by Mathcad, you can easily define custom units such as μm as described in “Defining Your Own Units” on page 109.

Tip For examples of units with prefixes not already built into Mathcad, see the QuickSheets in the on-line Resource Center.

If you click “None” in the Unit System tab of the Math Options dialog box, Mathcad doesn’t understand any built-in units and displays answers in terms of the fundamental dimensions of *length, mass, time, charge, and temperature*. However, even if you are working in one of Mathcad’s built-in unit systems, you can always choose to see results in your worksheet displayed in terms of fundamental dimension names rather than the base units of the unit system. To do so:

1. Choose **Options** from the **Math** menu.
2. Click the Dimensions tab.
3. Check “Display dimensions.”
4. Click “OK.”

Unit conversions

There are two ways to convert from one set of units to another:

- By using the Insert Unit dialog box, or
- By typing the new units in the units placeholder itself.

To convert units using the Insert Unit dialog box:

1. Click the unit you want to replace.
2. Choose **Unit** from the **Insert** menu, or click  on the Standard toolbar.
3. In the scrolling list of units, double-click the unit in which you want to display the result.

As a quick shortcut, or if you want to display the result in terms of a unit not available through the Insert Unit dialog box—for example, a unit you defined yourself or an algebraic combination of units—you can edit the units placeholder directly.

Figure 8-12 shows F displayed both in terms of fundamental SI units and in terms of several combinations of units.

When you enter an inappropriate unit in the units placeholder, Mathcad inserts a combination of base units that generate the correct units for the displayed result. For example, in the last equation in Figure 8-12, $\text{kW} \cdot \text{s}$ is not a unit of force. Mathcad therefore inserts m^{-1} to cancel the extra length dimension.

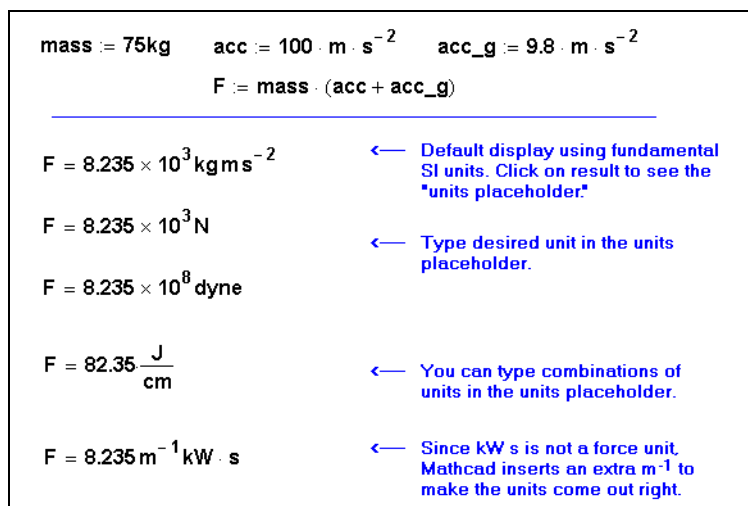


Figure 8-12: A calculated result displayed with different units

Whenever you enter units in the units placeholder, Mathcad divides the value to be displayed by whatever you enter in the units placeholder. This ensures that the complete displayed result—the number *times* the expression you entered for the placeholder—is a correct value for the equation.

Note Conversions involving an offset in addition to a multiplication, for example gauge pressure to absolute pressure, or degrees Fahrenheit to Celsius, cannot be performed directly with Mathcad's unit conversion mechanism. You can, however, perform conversions of this type by defining suitable functions. See the QuickSheet "Temperature Conversions" in the on-line Resource Center for examples of temperature conversion functions.

You can enter *any* variable, constant, or expression in a units placeholder. Mathcad then redisplayes the result in terms of the value contained in the units placeholder. For example, you can use the units placeholder to display a result as a multiple of π or in engineering notation (as a multiple of 10^3 , 10^6 , etc.).

Tip You can also use the units placeholder for dimensionless units like degrees and radians. Mathcad treats the unit *rad* as a constant equal to 1, so if you have a number or an expression in radians, you can type *deg* into the units placeholder to convert the result from radians to degrees.

Copying and Pasting Numerical Results

You can copy a numerical result and paste it either elsewhere in your worksheet or into a new application.

To copy a single number appearing to the right of an equal sign:

4. Click on the result to the right of the equal sign. This puts the result between the editing lines.
5. Choose **Copy** from the **Edit** menu, or click  on the Standard toolbar to place the result on the Clipboard.
6. Click wherever you want to paste the result. If you're pasting into another application, choose **Paste** from that application's **Edit** menu. If you're pasting into a Mathcad worksheet, choose **Paste** from Mathcad's **Edit** menu or click  on the Standard toolbar.

When you paste a numerical result into a Mathcad worksheet, it appears as:

- A math region consisting of a number if you paste it into empty space.
- A number if you paste it into a placeholder in a math region.
- A number if you paste it directly into text or into a placeholder in text created using the **Math Region** command on the **Insert** menu.

To copy more than one number, follow the steps for copying from an array. See "Displaying Arrays" on page 198 for information on copying and pasting arrays.

Note The **Copy** command copies the numerical result only to the precision displayed. To copy the result in greater precision, double-click it and increase "Displayed Precision" on the Result Format dialog box. **Copy** does not copy units and dimensions from a numerical result.

Controlling Calculation

When you start Mathcad, you are in *automatic mode*. This means that Mathcad updates results in the worksheet window automatically. You can tell you're in automatic mode because the word "Auto" appears in the message line at the bottom of the window.

If you don't want to wait for Mathcad to make computations as you edit, you can disable automatic mode by choosing **Automatic Calculation** from the **Math** menu. The word "Auto" disappears from the message line and the check beside **Automatic Calculation** disappears to indicate that automatic mode is now off. You are now in *manual mode*.

Tip The calculation mode—either manual or automatic—is a property saved in your Mathcad worksheet. As described in Chapter 7, "Worksheet Management," the calculation mode is also a property saved in Mathcad template (MCT) files.

Calculating in Automatic Mode

Here is how Mathcad works in automatic mode:

- As soon as you press the equal sign, Mathcad displays a result.
- As soon as you click outside of an equation having a "⋮=" or a "≡," Mathcad performs all calculations necessary to make the assignment statement.

When you process a definition in automatic mode by clicking outside the equation region, this is what happens:

- Mathcad evaluates the expression on the right side of the definition and assigns it to the name on the left.
- Mathcad then takes note of all other equations in the worksheet that are in any way affected by the definition you just made.
- Finally, Mathcad updates any of the affected equations that are currently visible in the worksheet window.

Note Although the equation you altered may affect equations throughout your worksheet, Mathcad performs only those calculations necessary to guarantee that whatever you can see in the window is up-to-date. This optimization ensures you don't have to wait for Mathcad to evaluate expressions that are not visible. If you print or move to the end of the worksheet, however, Mathcad automatically updates the whole worksheet.

Whenever Mathcad needs time to complete computations, the mouse pointer changes its appearance and the word "WAIT" appears on the message line. This can occur when you enter or calculate an equation, when you scroll, during printing, or when you enlarge a window to reveal additional equations. In all these cases, Mathcad evaluates pending calculations from earlier changes.

As Mathcad evaluates an expression, it surrounds it with a green rectangle. This makes it easy to follow the progress of a calculation.

To force Mathcad to recalculate all equations throughout the worksheet, choose **Calculate Worksheet** from the **Math** menu.

Calculating in Manual Mode

In manual mode, Mathcad does not compute equations or display results until you specifically request it to recalculate. This means that you don't have to wait for Mathcad to calculate as you enter equations or scroll around a worksheet.

Mathcad keeps track of pending computations while you're in manual mode. As soon as you make a change that requires computation, the word "Calc" appears on the message line. This is to remind you that the results you see in the window are not up-to-date and that you must recalculate them before you can be sure they are updated.

You can update the screen by choosing **Calculate** from the **Math** menu or clicking  on the Standard toolbar. Mathcad performs whatever computations are necessary to update all results visible in the worksheet window. When you move down to see more of the worksheet, the word "Calc" reappears on the message line to indicate that you must recalculate to see up-to-date results.

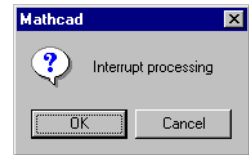
To process the whole worksheet, including those portions not visible in the worksheet window, choose **Calculate Worksheet** from the **Math** menu.

Note When you print a worksheet in manual calculation mode, the results on the printout are not necessarily up-to-date. In this case, make sure to choose **Calculate Worksheet** from the **Math** menu before you print.

Interrupting Calculations

To interrupt a computation in progress:

1. Press **[Esc]**. The dialog box shown at right appears.
2. Click "OK" to stop the calculations or "Cancel" to resume calculations.



If you click "OK," the equation that was being processed when you pressed **[Esc]** is marked with an error message (see "Error Messages" on page 121) indicating that calculation has been interrupted. To resume an interrupted calculation, first click in the equation having the error message, then choose **Calculate** from the **Math** menu or click  on the Standard toolbar.

Tip If you find yourself frequently interrupting calculations to avoid having to wait for Mathcad to recalculate as you edit your worksheet, you can switch to manual mode as described above.

Disabling Equations

You can *disable* a single equation so that it no longer calculates along with other regions in your worksheet. Disabling an equation does not affect Mathcad's equation editing, formatting, and display capabilities.

To disable calculation for a single equation in your worksheet, follow these steps:

1. Click on the equation you want to disable.
2. Choose **Properties** from the **Format** menu, and click the Calculation tab.
3. Under "Calculation Options" check "Disable Evaluation."

4. Mathcad shows a small rectangle after the equation to indicate that it is disabled. An example is shown at right.

$$KE := \frac{1}{2} \cdot m \cdot v^2$$

Tip An easy shortcut for disabling evaluation is to click with the right mouse button on an equation and select **Disable Evaluation** from the pop-up menu.

To re-enable calculation for a disabled equation:

1. Click on the equation to select it.
2. Choose **Properties** from the **Format** menu, and click the Calculation tab.
3. Remove the check from “Disable Evaluation.”

Mathcad removes the small rectangle beside the equation, and calculation is re-enabled.

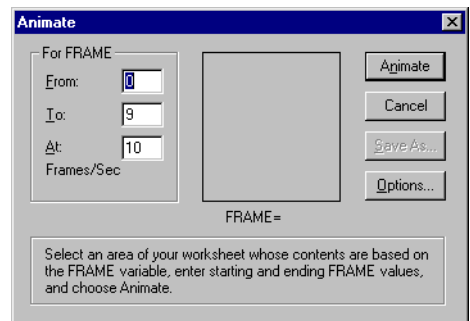
Animation

This section describes how to use Mathcad to create and play short animation clips by using the built-in variable FRAME. Anything that can be made to depend on this variable can be animated. This includes not only plots but numerical results as well. You can play back the animation clips at different speeds or save them for use by other applications.

Creating an Animation Clip

Mathcad comes with a predefined constant called FRAME whose sole purpose is to drive animations. The steps in creating any animation are as follows:

1. Create an expression or plot, or a group of expressions, whose appearance ultimately depends on the value of FRAME. This expression need not be a graph. It can be anything at all.
2. Choose **Animate** from the **View** menu to bring up the Animate dialog box.
3. Drag-select the portion of your worksheet you want to animate as shown in Figure 8-13. Draw a rectangle around as many regions as you want to appear in the animation.
4. Set the upper and lower limits for FRAME in the dialog box. When you record the animation, the FRAME variable increments by one as it proceeds from the lower limit to the upper limit.
5. Enter the playback speed in the Frames/Sec. box.
6. Click “Animate.” You’ll see a miniature rendition of your selection inside the dialog box. Mathcad redraws this once for each value of FRAME. This won’t necessarily match the playback speed since at this point you’re just *creating* the animation.
7. To save your animation clip as a Windows AVI file, suitable for viewing in other Windows applications, click “Save As” in the dialog box.



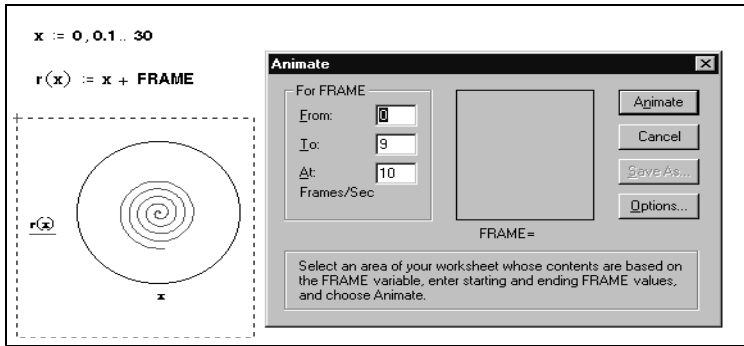


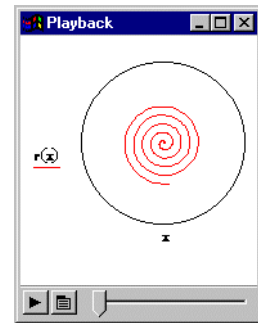
Figure 8-13: Selecting an area of a worksheet for animation.

Tip Since animation clips can take considerable disk space, Mathcad saves them in compressed format. Before creating the animation, you can choose what compression method to use or whether to compress at all. To do so, click “Options” in the Animate dialog box.

Playing an Animation Clip

As soon as you’ve created an animation clip as described in the previous section, Mathcad opens a Playback window:

The first frame of the animation clip you just created is already in the window. To play back the animation clip, click the arrow at the lower left corner of the window. You can also play back the animation clip on a frame by frame basis, either forward or backward. To do so, drag the slider below the animated picture to the left or right.



Tip You can control the playback speed by clicking the button to the right of the play button, which then opens a pop-up menu. Choose **Speed** from the menu and adjust the slider control.

Playing a Previously Saved Animation

If you have an existing Windows AVI file on your disk, you can play it within Mathcad. To do so:

1. Choose **Playback** from the **View** menu to bring up the Playback dialog box. The window is collapsed since no animation clip has been opened.
2. Click on the button to the right of the play button and choose **Open** from the menu. Use the Open File dialog box to locate and open the AVI file you want to play.

Once you’ve loaded a Windows AVI file, proceed as described in the previous section.



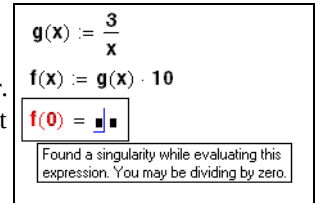
Tip To launch an animation directly from your worksheet, you can insert a hyperlink to an AVI file by choosing **Hyperlink** from the **Insert** menu. You can also embed a shortcut to the AVI file in your worksheet by dragging the icon for the AVI file from the Windows Explorer and dropping it into your worksheet. Finally, you can embed or link an OLE animation object in your worksheet (see “Inserting Objects” on page 71).

Error Messages

If Mathcad encounters an error when evaluating an expression, it marks the expression with an error message and highlights the offending name or operator in red.

An error message is visible only when you *click* on the associated expression, as shown to the right.

Mathcad cannot process an expression containing an error. If the expression is a definition, the variable or function it is supposed to define remains undefined. This can cause any expressions that reference that variable to be undefined as well.



Tip You can get on-line help about some error messages by clicking on them and pressing **[F1]**.

Finding the Source of an Error

When a Mathcad worksheet contains an expression that is dependent on one or more definitions made earlier in the worksheet, an error on that expression may originate in an earlier definition.

For example, in the figure above, the error appears on the third region, $f(0)$. However, $f(x)$ is based on the definition of $g(x)$. When x is zero, $g(x)$ is the first region that exhibits the error.

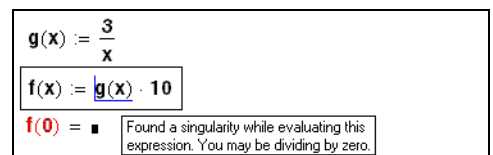
You can try to find the source of an error yourself simply by examining your worksheet to see where the error began, or you can use Mathcad to trace the error back through your worksheet. To find the source of an error using Mathcad:

1. Click on the region showing the error with the right mouse button and choose Trace Error from the pop-up menu. The Trace Error dialog box appears:

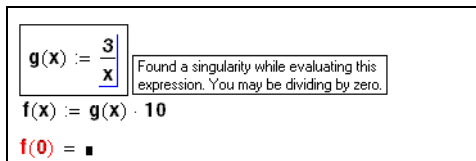


2. Use the buttons in the dialog box to navigate among the regions associated with the region showing the error.

For example, click Back to step back to the previous dependent region.



Or click First to jump to the first region causing the error.



$g(x) := \frac{3}{x}$

Found a singularity while evaluating this expression. You may be dividing by zero.

$f(x) := g(x) \cdot 10$

$f(0) =$

Tip If you anticipate time-consuming calculations, switch to manual mode as described in “Controlling Calculation” on page 117. When you are ready to recalculate, choose **Calculate** from the **Math** menu or click  on the Standard toolbar turn. Alternatively, turn on automatic mode again.

Fixing Errors

Once you have determined which expression caused the error, edit that expression to fix the error or change the variable definitions that led to the error. When you click in the expression and begin editing, Mathcad removes the error message. When you click outside the equation (or in manual calculation mode, when you recalculate), Mathcad recomputes the expression. Once you have fixed the error, Mathcad then recomputes the other expressions affected by the expression you changed.

Note When you define a function, Mathcad does not try to evaluate it until you subsequently use it in the worksheet. If there is an error, the use of the function is marked in error, even though the real problem may lie in the definition of the function itself, possibly much earlier in the worksheet.

Chapter 9

Operators

- τ Working with Operators
- τ Arithmetic and Boolean Operators
- τ Vector and Matrix Operators
- τ Summations and Products
- τ Derivatives
- τ Integrals
- Pro** τ Customizing Operators

Working with Operators

Inserting an Operator

You insert the common arithmetic operators into math expressions in Mathcad using the standard keystrokes, like * and +, that you use in spreadsheet and other applications. Additionally, all of Mathcad's operators can be inserted into math expressions by clicking buttons in the math toolbars. For example, you insert Mathcad's derivative

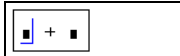
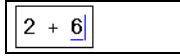
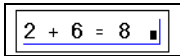
operator by clicking  on the Calculus toolbar, or by typing ?. Choose **Toolbars** from the **View** menu to see any of the math toolbars. See “Operators” on page 298 in the Appendices for a complete list of operators, their keystrokes, and descriptions.

Note In general, you only insert operators into blank space in your worksheet or when you have already clicked in a math region. To use operators in text, first click in the text and choose **Math Region** from the **Insert** menu. This creates a math placeholder in the text into which you can insert operators.

Tip You can find out the keyboard shortcut for inserting an operator by hovering the mouse pointer over an operator button in one of the Math toolbars and reading the tooltip that appears.

As introduced in Chapter 4, “Working with Math,” when you insert a Mathcad operator into a blank space in your worksheet, a mathematical symbol with empty *placeholders* appears in the worksheet. The placeholders are for you to enter expressions that are the *operands* of the operator. The number of empty placeholders varies with the operator: some operators like the factorial operator have only a single placeholder, while others such as the definite integral have several. You must enter a valid math expression in each placeholder of an operator in order to calculate a result.

Here is a very simple example involving Mathcad's addition operator:

1. Click in a blank space in your worksheet and click  on the Calculator toolbar, or simply type **+**. The addition operator with two placeholders appears. 
2. Enter **2** in the first placeholder. 
3. Click in the second placeholder, or press **[Tab]** to move the cursor, and enter **6**. 
4. Press **=**, or click  on the Evaluation toolbar, to see the numerical result. 

Tip See Chapter 4, “Working with Math,” for a discussion of how to build and edit more complex math expressions, including how to use the *editing lines* to specify what becomes the operand of the next operator you insert or delete.

Additional Operators

This chapter focuses on those Mathcad operators you can use to calculate numerical answers. Additional operators in Mathcad include:

- *Symbolic operators*, which can only be used to generate other math expressions or exact numerical answers. As described in Chapter 14, “Symbolic Calculation,” Mathcad’s symbolic processor understands virtually any Mathcad expression, but expressions that include the following operators on the Calculus toolbar can *only* be evaluated symbolically: indefinite integral , two-sided limit , limit from above , and limit from below . To evaluate an expression symbolically, click  on the Evaluation toolbar.

- Pro**
- *Programming operators*, which you use to link multiple Mathcad expressions via conditional branching, looping constructs, local scoping of variables, and other attributes of traditional programming languages. These operators, available only in Mathcad Professional (click  on the Math toolbar), are introduced in Chapter 15, “Programming.”

Changing the Display of an Operator

When you insert an operator into a worksheet, it has a certain default appearance. For example, when you type a colon **:** or click  on the Calculator toolbar, Mathcad shows the colon as the definition symbol **:=**. This is a special symbol used by Mathcad to indicate a variable or function definition.

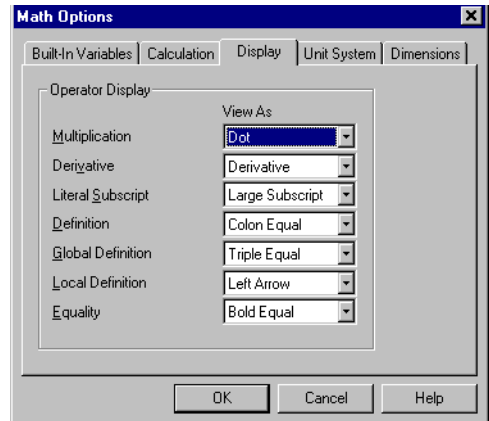
There may be times when you want to control the appearance of a special symbol such as the definition symbol. For example you may want the definition symbol to look like an ordinary equal sign, but you still want to use it to define variables and functions in your worksheet. Mathcad therefore allows you to change the appearance of some

operators, such as the definition symbol, so that they appear different but behave the same way.

To change the way an operator is displayed throughout a worksheet:

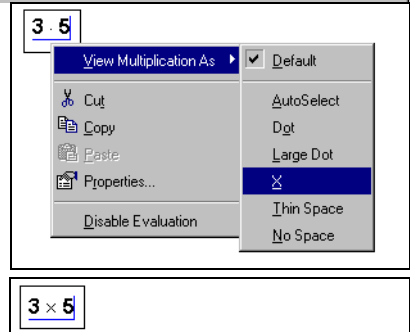
1. Choose **Options** from the **Math** menu.
2. Click the Display tab.
3. Use the drop-down options next to each operator to select a display option.
4. Click “OK.”

For information on the options available for each operator, click the Help button at the bottom of the Display tab in the Math Options dialog box.



To change the appearance of an operator in one or more individual expressions, click with the right mouse button and use the pop-up menu. For example, to change the multiplication in an expression from a dot to an X:

1. Click on the multiplication with the right mouse button.
2. Choose **View Multiplication As...** ⇒ **X** from the pop-up menu.



Arithmetic and Boolean Operators

Arithmetic Operators

You can freely combine all types of numbers with arithmetic operators you access on the Calculator toolbar. Figure 9-1 shows examples.

Boolean Operators

Mathcad includes logical or *Boolean* operators on the Boolean toolbar. Unlike other operators, the Boolean operators can return only a zero or a one. Despite this, they can be very useful to perform tests on your expressions.

$a := \pi$	Predefined variable	$a = 3.142$
$b := 123456789012$	Large floating point number	$b = 1.235 \times 10^{11}$
$c := 5 - 7i$	Complex number (could use 5-7j as well)	$c = 5 - 7i$
$e := 3.5\text{m}$	Dimensional value (SI unit system)	$e = 3.5\text{m}$
$a + 4 \cdot 10^{-5} = 3.142$		
$a \cdot \frac{d}{e} = 4.81 \times 10^3 \frac{1}{\text{m}}$	$a \cdot d + e = 4.81 \times 10^3 \frac{1}{\text{m}}$	
$b \cdot c = 6.173 \times 10^{11} - 8.642i \times 10^{11}$		

Figure 9-1: Combining different types of numbers with arithmetic operators.

The following table lists the Boolean operators available on the Boolean toolbar and their meaning. Note that the “Equal to” operator (bold equal sign) is different from the evaluation equal sign you insert by typing =.

Appearance	Button	Description	Keystroke
$w = z$		Equal to; displays as bold equal sign.	[Ctrl] =
$x < y$		Less than	<
$x > y$		Greater than	>
$x \leq y$		Less than or equal to	[Ctrl] 9
$x \geq y$		Greater than or equal to	[Ctrl] 0
$w \neq z$		Not equal to	[Ctrl] 3
$\neg z$		Not	[Ctrl] [Shift] 1
$w \wedge z$		And	[Ctrl] [Shift] 7
$w \vee z$		Or	[Ctrl] [Shift] 6
$w \oplus z$		Xor (Exclusive Or)	[Ctrl] [Shift] 5

Note The Boolean operators return 1 if the expression is true, 0 otherwise. The four operators $>$, $<$, $\leq$, and $\geq$ cannot take complex numbers because the concepts of greater than and less than lose their meaning in the complex plane.

$10 > 0 = 1$	$10 < 0 = 0$	$3 + 5 = 7 = 0$	Evaluation equals
$.5 = \frac{1}{2} = 1$	$14 \neq 10 = 1$	$12345 < 12345 = 0$	Boolean equals
$\frac{1}{3} < \frac{1}{2} = 1$	$19^2 \geq 360 = 1$	$2000 \neq 2000 = 0$	
$1 \vee 1 = 1$	$1 \wedge 0 = 0$	$\neg 1 = 0$	
$1 \oplus 1 = 0$	$2 \wedge 0 = 0$	$\neg(1 - 1) = 1$	

Figure 9-2: Using boolean operators.

Tip The comparative boolean operators such as $<$ and $>$ can also be used to compare *strings*. Mathcad compares two strings character by character by determining the ASCII codes of the characters. For example, the string “Euler” precedes the string “Mach” in ASCII order and so the expression (“Euler” $<$ “Mach”) evaluates to 1. To determine the character ordering Mathcad uses in comparing strings, see “ASCII codes” on page 316 in the Appendices.

Complex Operators

Mathcad has the following arithmetic operators for working with complex numbers:

Appearance	Button	Description
$\bar{z}$		Complex conjugate of z . To apply the conjugate operator to an expression, select the expression, then press the double-quote key ("). The conjugate of the complex number $a + b \cdot i$ is $a - b \cdot i$.
$ z $		The magnitude of the number z .

Figure 9-3 shows some examples of how to use complex numbers in Mathcad.

Vector and Matrix Operators

Most of the operators on the Calculator toolbar also have meaning for vectors and matrices. For example, when you use the addition operator to add two arrays of the same size, Mathcad performs the standard element-by-element addition. Mathcad also uses the conventional arithmetic operators for matrix subtraction, matrix multiplication, integer powers, and determinants, among others.

Some of Mathcad’s operators have special meanings for vectors and matrices, and many of these are grouped on the Matrix toolbar (click  on the Math toolbar). For example, the multiplication symbol means multiplication when applied to two numbers, but it means dot product when applied to vectors, and matrix multiplication when applied to matrices.

$$r := 2 \quad \theta := \frac{3 \cdot \pi}{4} \quad \text{Type q [Ctrl] G for } \theta$$

Define complex variables z1 and z2

$$z1 := \sqrt{-1} \quad z2 := r \cdot e^{(i \cdot \theta)} \quad \text{Type ii for the complex variable i}$$

$$z1 = i \quad z2 = -1.414 + 1.414i$$

Now compute with them

$$z1 + z2 = -1.414 + 2.414i \quad \text{Re}(z2) = -1.414$$

$$z1 \cdot z2 = -1.414 - 1.414i \quad \text{Im}(z2) = 1.414$$

$$\overline{z2} = -1.414 - 1.414i \quad \sin(z2) = -2.152 + 0.302i$$

$$\frac{z2}{z1} = 1.414 + 1.414i \quad \sinh(z2) = -0.302 + 2.152i$$

$$\ln(z2) = 0.693 + 2.356i$$

Decompose z2 in polar form

$$|z2| = 2 \quad \arg(z2) = 2.356$$

Figure 9-3: Complex numbers in Mathcad.

The table below describes Mathcad's vector and matrix operations. Operators not listed in this table do not work for vectors and matrices. You can, however, use the vectorize

operator (click  on the Matrix toolbar) to perform any scalar operation or function element by element on a vector or matrix. See "Doing Calculations in Parallel" on page 201. Figure 9-4 shows some ways to use vector and matrix operators.

Matrix M ...	Vectors v and w ...	
$M := \begin{pmatrix} 0 & 1 & 2 \\ 3 & 0 & 2 \\ 5 & 3 & 1 \end{pmatrix}$	$v := \begin{pmatrix} 3 + 10i \\ 1 - 4i \\ 5 \cdot 10 \end{pmatrix}$	$v = \begin{pmatrix} 13 \\ -3 \\ 50 \end{pmatrix} \quad w := 2 \cdot v \quad w = \begin{pmatrix} 26 \\ -6 \\ 100 \end{pmatrix}$
Sum ...	Determinant ...	Dot and Cross Product ...
$\sum v = 60$	$ M = 25$	$v \cdot w = 5.356 \cdot 10^3 \quad v \times w = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$
Inverse ...	Transpose ...	
$M^{-1} = \begin{pmatrix} -0.24 & 0.2 & 0.08 \\ 0.28 & -0.4 & 0.24 \\ 0.36 & 0.2 & -0.12 \end{pmatrix}$	$w^T = (26 \ -6 \ 100)$	
$M \cdot M^{-1} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	Solve linear system $Mx=v$ with inverse ...	
	$x := M^{-1} \cdot v$	
	$x = \begin{pmatrix} 0.28 \\ 16.84 \\ -1.92 \end{pmatrix}$	$M \cdot x = \begin{pmatrix} 13 \\ -3 \\ 50 \end{pmatrix}$

Figure 9-4: Vector and matrix operations.

In the following table,

- **A** and **B** represent arrays, either vector or matrix.
- **u** and **v** represent vectors.
- **M** represents a square matrix.
- u_i and v_i represent the individual elements of vectors **u** and **v**.
- z represents a scalar.
- m and n represent integers.

Appearance	Button	Description
$\mathbf{A} \cdot z$		Scalar multiplication. Multiplies each element of A by the scalar z .
$\mathbf{u} \cdot \mathbf{v}$		Dot product. Returns a scalar: $\sum \langle u_i \cdot v_i \rangle$. The vectors must have the same number of elements.
$\mathbf{A} \cdot \mathbf{B}$		Matrix multiplication. Returns the matrix product of A and B . The number of columns in A must match the number of rows in B .
$\mathbf{A} \cdot \mathbf{v}$		Vector/matrix multiplication. Returns the product of A and v . The number of columns in A must match the number of rows in v .
$\frac{\mathbf{A}}{z}$		Scalar division. Divides each element of the array A by the scalar z .
$\mathbf{A} \div z$		Scalar division. Divides each element of the array A by the scalar z . Type [Ctrl] / to insert.
$\mathbf{A} + \mathbf{B}$		Vector and matrix addition. Adds corresponding elements of A and B . The arrays A and B must have the same number of rows and columns.
$\mathbf{A} + z$		Scalar addition. Adds z to each element of A .
$\mathbf{A} - \mathbf{B}$		Vector and matrix subtraction. Subtracts corresponding elements of A and B . The arrays A and B must have the same number of rows and columns.
$\mathbf{A} - z$		Scalar subtraction. Subtracts z from each element of A .
$-\mathbf{A}$		Negative of vector or matrix. Returns an array whose elements are the negatives of the elements of A .

$\mathbf{M}^n$		n th power of square matrix $\mathbf{M}$ (using matrix multiplication). n must be an integer. $\mathbf{M}^{-1}$ represents the inverse of $\mathbf{M}$. Other negative powers are powers of the inverse. Returns a matrix.
$ \mathbf{v} $		Magnitude of vector. Returns $\sqrt{\mathbf{v} \cdot \bar{\mathbf{v}}}$ where $\bar{\mathbf{v}}$ is the complex conjugate of $\mathbf{v}$.
$ \mathbf{M} $		Determinant. $\mathbf{M}$ must be a square matrix.
$\mathbf{A}^T$		Transpose. Interchanges row and columns of $\mathbf{A}$.
$\mathbf{u} \times \mathbf{v}$		Cross product. $\mathbf{u}$ and $\mathbf{v}$ must be three-element vectors; result is another three-element vector.
$\bar{\mathbf{A}}$		Complex conjugate. Takes complex conjugate of each element of $\mathbf{A}$. Insert in math with the double-quote key (").
$\Sigma \mathbf{v}$		Vector sum. Sum elements in $\mathbf{v}$.
$\vec{\mathbf{A}}$		Vectorize. Treat all operations in $\mathbf{A}$ element by element. See “Doing Calculations in Parallel” on page 201 for details.
$\mathbf{A}^{(n)}$		Array superscript. n th column of array $\mathbf{A}$. Returns a vector.
v_n		Vector subscript. n th element of a vector.
$A_{m,n}$		Matrix subscript. m,n th element of a matrix.

Tip Operators and functions that expect vectors always expect column vectors. They do not apply to row vectors. To change a row vector into a column vector, use the transpose operator by clicking



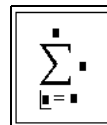
on the Matrix toolbar.

Summations and Products

The summation operator sums an expression over all values of an index. The iterated product operator works much the same way. It takes the product of an expression over all values of an index.

To create a summation operator in your worksheet:

1. Click in a blank space. Then click  on the Calculus toolbar. A summation sign with four placeholders appears.



2. Type a variable name in the placeholder to the left of the equal sign. This variable is the index of summation. It is defined only within the summation operator and therefore has no effect on, and is not influenced by, variable definitions outside the summation operator.
3. Type integers, or any expressions that evaluate to integers, in the placeholders to the right of the equal sign and above the sigma.
4. Type the expression you want to sum in the remaining placeholder. Usually, this expression involves the index of summation. If this expression has several terms, first type an apostrophe (') to create parentheses around the placeholder.

$$\sum_{n=1}^{\cdot} \cdot$$

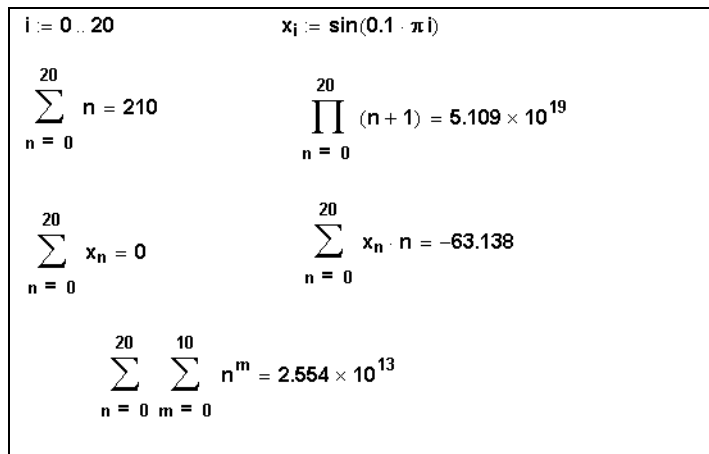
$$\sum_{n=1}^{10} \cdot$$

$$\sum_{n=1}^{10} n^2$$

Iterated products are similar to summations. Just click  on the Calculus toolbar and fill in the placeholders as described earlier.

Tip Use the keyboard shortcut **[Ctrl][Shift]4** to enter the iterated sum and the shortcut **[Ctrl][Shift]3** to enter the iterated product operator.

Figure 9-5 shows some examples of how to use the summation and product operators. To evaluate multiple summations, place another summation in the final placeholder of the first summation. An example appears at the bottom of Figure 9-5.



$i := 0..20$ $x_i := \sin(0.1 \cdot \pi i)$

$\sum_{n=0}^{20} n = 210$ $\prod_{n=0}^{20} (n+1) = 5.109 \times 10^{19}$

$\sum_{n=0}^{20} x_n = 0$ $\sum_{n=0}^{20} x_n \cdot n = -63.138$

$\sum_{n=0}^{20} \sum_{m=0}^{10} n^m = 2.554 \times 10^{13}$

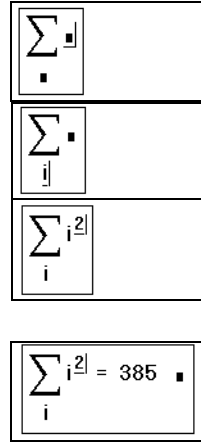
Figure 9-5: Summations and products.

When you use the summation operator shown in Figure 9-5, the summation must be carried out over subsequent integers and in steps of one. Mathcad provides more general versions of these operators that can use any range variable you define as an index of summation. To use these operators:

1. Define a range variable. For example, type **i:1,2..10**.

$$i := 1, 2..10$$

- Click in a blank space. Then click  on the Calculus toolbar. A summation sign with two placeholders appears.
- Click on the bottom placeholder and type the name of a range variable.
- Click on the placeholder to the right of the summation sign and type an expression involving the range variable. If this expression has several terms, first type an apostrophe (') to create parentheses around the placeholder.
- Press =, or click  on the Evaluation toolbar, to get a result.



$$\sum_i i^2 = 385$$

Tip To enter the expression in the example above using fewer keystrokes and mouse clicks, type **i\$i^2**.

A generalized version of the iterated product also exists. To use it, click  on the Calculus toolbar. Then fill in the two placeholders.

Tip The operation of summing the elements of a vector is so common that Mathcad provides a special operator for it. The vector sum operator (click  on the Matrix toolbar) sums the elements of a vector without needing a range variable.

Variable Upper Limit of Summation

Mathcad's range summation operator runs through each value of the range variable you place in the bottom placeholder. It is possible, by judicious use of Boolean expressions, to sum only up to a particular value. In Figure 9-6, the term $i \leq x$ returns the value 1 whenever it is true and 0 whenever it is false. Although the summation operator still sums over each value of the index of summation, those terms for which $i > x$ are multiplied by 0 and hence do not contribute to the summation.

You can also use the four-placeholder summation and product operators to compute sums and products with a variable upper limit, but note that the upper limit in these operators must be an integer.

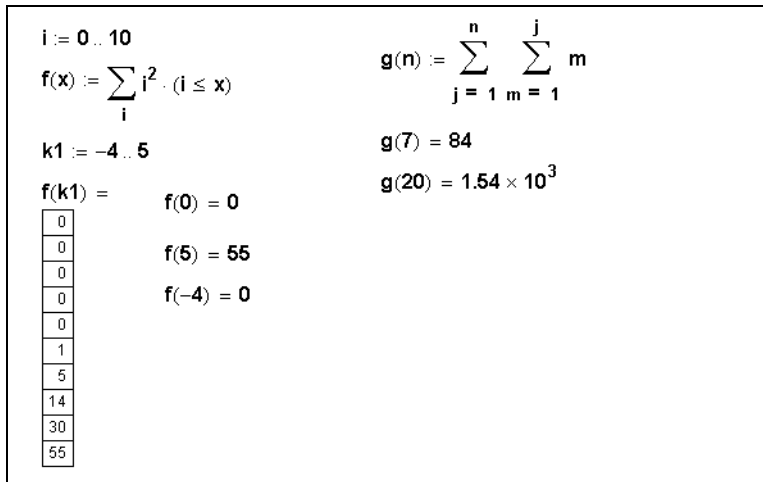


Figure 9-6: A variable upper limit of summation

Derivatives

You can use Mathcad's derivative operators to evaluate the first or higher order derivatives of a function at a particular point.

As an example, here's how to evaluate the first derivative of x^3 with respect to x at the point $x = 2$:

1. First define the point at which you want to evaluate the derivative. As a shortcut, type **x:2**.
2. Click below the definition of x . Then click  on the Calculus toolbar. A derivative operator appears with two placeholders.
3. Type **x** in the bottom placeholder. You are differentiating with respect to this variable. In the placeholder to the right of the $\frac{d}{dx}$, enter **x^3**. This is the expression to be differentiated.
4. Press =, or click  on the Evaluation toolbar, to get the result.

$$x := 2$$

$$\frac{d}{dx} \square$$

$$\frac{d}{dx} x^3$$

$$\frac{d}{dx} x^3 = 12$$

Figure 9-7 shows examples of differentiation in Mathcad.

With Mathcad's derivative algorithm, you can expect the first derivative to be accurate within 7 or 8 significant digits, provided that the value at which you evaluate the derivative is not too close to a singularity of the function. The accuracy of this algorithm tends to decrease by one significant digit for each increase in the order of the derivative (see "Derivatives of Higher Order" on page 135).

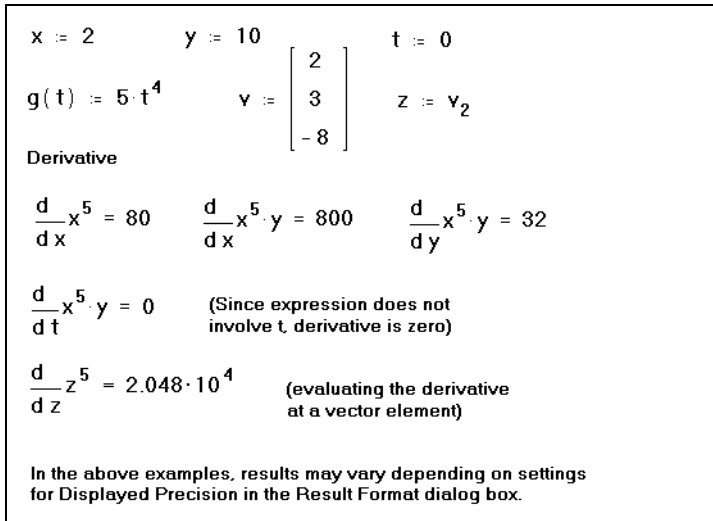


Figure 9-7: Examples of Mathcad differentiation.

Note Keep in mind that the result of numerical differentiation is not a function, but a single number: the computed derivative at the indicated value of the differentiation variable. In the previous example, the derivative of x^3 is not the expression $3x^2$ but $3x^2$ evaluated at $x = 2$. To evaluate derivatives symbolically, see Chapter 14, “Symbolic Calculation.”

Although differentiation returns just one number, you can still define one function as the derivative of another. For example:

$$f(x) := \frac{d}{dx} g(x)$$

Evaluating $f(x)$ returns the numerically computed derivative of $g(x)$ at x .

You can use this technique to evaluate the derivative of a function at many points. An example of this is shown in Figure 9-8.

There are some important things to remember about differentiation in Mathcad:

- The expression to be differentiated can be either real or complex.
- The differentiation variable must be a single variable name. If you want to evaluate the derivative at several different values stored in a vector, you must evaluate the derivative at each individual vector element (see Figure 9-8).

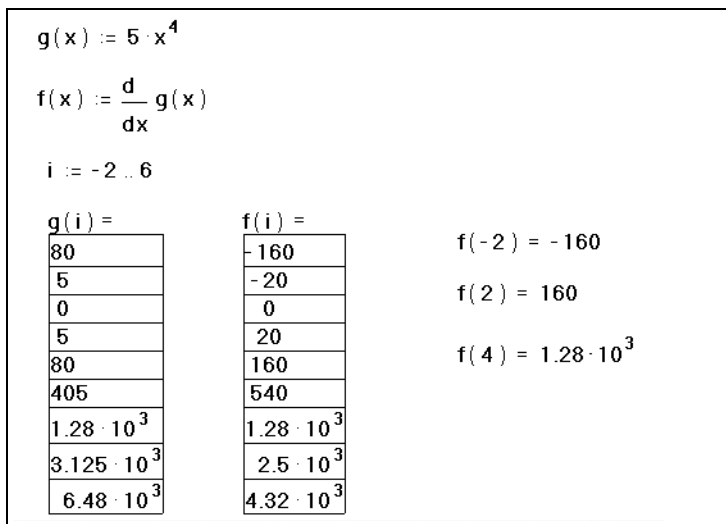


Figure 9-8: Evaluating the derivative of a function at several points

Tip You can change the display of the derivative operator to partial derivative symbols. For example you can make $\frac{d}{dx}$ look like $\frac{\partial}{\partial x}$. To change the display of a derivative operator to partial derivative symbols, click on it with the mouse button and choose **View Derivative As... $\Rightarrow$ Partial**. Or to change the display of all the derivative operators in a worksheet, choose **Options** from the **Math** menu, click on the Display tab, and select “Partial Derivative” next to Derivative. See “Changing the Display of an Operator” on page 124 for additional information.

Derivatives of Higher Order

To evaluate a higher order derivative, insert the n th derivative operator using steps similar to those for inserting the derivative operator described above.

As an example, here’s how to evaluate the third derivative of x^9 with respect to x at the point $x = 2$: After defining x as 2:

1. Click below the definition of x . Then click  on the Calculus toolbar. A derivative operator appears with four placeholders.
2. Click on the bottom-most placeholder and type **x**.
3. Click on the expression above and to the right of the previous placeholder and type **3**. This must be an integer between 0 and 5 inclusive. Note that the placeholder in the numerator automatically mirrors whatever you’ve typed.

4. Click on the placeholder to the right of the $\frac{d}{dx}$ and type

$$\frac{d^3}{dx^3} x^9$$

x^9. This is the expression to be differentiated.

5. Press =, or click  on the Evaluation toolbar, to see the result.

$$\frac{d^3}{dx^3} x^9 = 3.226 \cdot 10^4$$

Note For $n = 1$, the n th derivative operator gives the same answer as the first-derivative operator discussed on page 133.

Integrals

You can use Mathcad's integral operator to numerically evaluate the definite integral of a function over some interval.

As an example, here's how to evaluate the definite integral of $\sin^2(x)$ from 0 to $\pi/4$. (In Mathcad you enter $\sin^2(x)$ as $\sin(x)^2$.) Follow these steps:

1. Click in a blank space and click  on the Calculus toolbar. An integral symbol appears, with placeholders for the integrand, limits of integration, and variable of integration.

$$\int \quad d$$

2. Click on the bottom placeholder and type **0**. Click on the top placeholder and type **p[Ctrl]G/4**. These are the upper and lower limits of integration.

$$\int_0^{\pi/4} \quad d$$

3. Click on the placeholder between the integral sign and the "d." Then type **sin(x)^2**. This is the expression to be integrated.

$$\int_0^{\pi/4} \sin(x)^2 \quad d$$

4. Click on the remaining placeholder and type **x**. This is the variable of integration. Then press =, or click  on the Evaluation toolbar, to see the result.

$$\int_0^{\pi/4} \sin(x)^2 \quad dx = 0.143$$

Note Some points to keep in mind when you evaluate integrals in Mathcad: 1) The limits of integration must be real. The expression to be integrated can, however, be either real or complex. 2) Except for the integrating variable, all variables in the integrand must have been defined previously in the worksheet. 3) The integrating variable must be a single variable name. 4) If the integrating variable involves units, the upper and lower limits of integration must have the same units.

Integration Algorithms and AutoSelect

Mathcad has a number of numerical integration methods at its disposal to calculate the numerical approximation of an integral. When you evaluate an integral, by default Mathcad uses an *AutoSelect* procedure to choose the most accurate integration method.

If you have Mathcad Professional, you can override AutoSelect and choose from among the available integration algorithms yourself.

Here are the methods from which Mathcad chooses when you evaluate an integral numerically:

Romberg

Applies a Romberg integration method that divides the interval of integration into equally spaced subintervals.

Adaptive

Applies an adaptive quadrature algorithm in cases where the integrand varies considerably in magnitude over the interval of integration.

Infinite Limit

Applies an algorithm designed for improper integral evaluation in cases where either limit of integration is ∞ or $-\infty$.

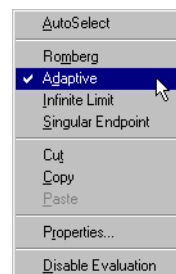
Singular Endpoint

Applies a routine that avoids use of the interval endpoints in cases where the integrand is undefined at either limit of integration.

Note Although designed to handle a wide range of problems, Mathcad's integration algorithms—like all numerical methods—can have difficulty with ill-behaved integrands. For example, if the expression to be integrated has singularities or discontinuities the solution may still be inaccurate.

Pro With Mathcad Professional, you can override Mathcad's integration AutoSelect as follows:

1. Evaluate the value of the integral as described on page 136, allowing Mathcad to AutoSelect an integration algorithm.
2. Click with the right mouse button on the integral.
3. Click one of the listed integration methods on the pop-up menu. Mathcad recalculates the integral using the method you selected.



Tip In some cases, you may be able to find an exact numerical value for your integral by using Mathcad's symbolic integration capability. You can also use this capability to evaluate *indefinite* integrals. See Chapter 14, "Symbolic Calculation."

Variable Limits of Integration

Although the result of an integration is a single number, you can always use an integral with a range variable to obtain results for many numbers at once. You might do this, for example, when you set up a variable limit of integration. Figure 9-9 shows how to do this.

Keep in mind that calculations such as those shown in Figure 9-9 require repeatedly evaluating an integral. This may take considerable time depending on the complexity

$i := 0 \dots 5$ $g_i := \int_0^i f(x) \, dx$ $g = \begin{bmatrix} 0 \\ 3.833333 \\ 12.666667 \\ 28.5 \\ 53.333333 \\ 89.166667 \end{bmatrix}$	$f(x) := x^2 + 3 \cdot x + 2$ $f_i := \int_{-i}^0 f(x) \, dx$ $f = \begin{bmatrix} 0 \\ 0.833333 \\ 0.666667 \\ 1.5 \\ 5.333333 \\ 14.166667 \end{bmatrix}$
--	---

Figure 9-9: Variable limits of integration.

of the integrals, the length of the interval, and the value of the tolerance parameter TOL (see below).

Tolerance for Integrals

Mathcad's numerical integration algorithms make successive estimates of the value of the integral and return a value when the two most recent estimates differ by less than the value of the built-in variable TOL.

As described in “Built-in Variables” on page 98, you can change the value of the tolerance by including definitions for TOL directly in your worksheet. You can also change the tolerance by using the Built-In Variables tab when you choose **Options** from the **Math** menu. To see the effect of changing the tolerance, choose **Calculate Document** from the **Math** menu to recalculate all the equations in the worksheet.

If Mathcad's approximation to an integral fails to converge to an answer, Mathcad marks the integral with an error message. Failure to converge can occur when the function has singularities or “spikes” in the interval or when the interval is extremely long.

Note When you change the tolerance, keep in mind the trade-off between accuracy and computation time. If you decrease (tighten) the tolerance, Mathcad computes integrals more accurately, but takes longer to return a result. Conversely, if you increase (loosen) the tolerance, Mathcad computes more quickly, but the answers are less accurate.

Contour Integrals

You can use Mathcad to evaluate complex contour integrals. To do so, first parametrize the contour and then integrate over the parameter. If the parameter is something other than arc length, you must also include the derivative of the parametrization as a correction factor. Figure 9-10 shows an example. Note that the imaginary unit i used in specifying the path must be typed as **1i**.

$$\begin{aligned}
 x(t) &:= 2 \cdot \cos(t) & y(t) &:= 2 \cdot \sin(t) \\
 \text{Path :} & & z(t) &:= x(t) + i \cdot y(t) \\
 \text{Function to integrate :} & & f(z) &:= \frac{1}{z} \\
 \int_0^\pi f(z(t)) \cdot \frac{d}{dt} z(t) dt &= 3.142i
 \end{aligned}$$

Figure 9-10: A complex contour integral in Mathcad.

Multiple integrals

You can also use Mathcad to evaluate double or multiple integrals. To set up a double integral, for example, click  on the Calculus toolbar twice. Fill in the integrand, the limits, and the integrating variable for each integral. Figure 9-11 shows an example.

Center of mass of triangle described
by: $0 < x < 1$ and $0 < y < x$
and mass density proportional to the
distance from the origin.

$$\begin{aligned}
 \delta(x, y) &:= \sqrt{x^2 + y^2} \\
 \text{mass} &:= \int_0^1 \int_0^x \delta(x, y) dy dx & \text{mass} &= 0.383 \\
 \text{xctr} &:= \frac{1}{\text{mass}} \cdot \int_0^1 \int_0^x x \cdot \delta(x, y) dy dx & \text{xctr} &= 0.75 \\
 \text{yctr} &:= \frac{1}{\text{mass}} \cdot \int_0^1 \int_0^x y \cdot \delta(x, y) dy dx & \text{yctr} &= 0.398
 \end{aligned}$$

Figure 9-11: Double integrals.

Note Multiple integrals generally take much longer to converge to an answer than single integrals. Wherever possible, use an equivalent single integral in place of a multiple integral.

Customizing Operators

Pro This section describes how you can use Mathcad Professional to define and use your own customized operators.

You can think of operators and functions as being fundamentally very similar. A function takes “arguments” and returns a result. An operator, likewise, takes “operands” and returns a result. The differences are largely notational:

- Functions usually have names you can spell, like *tan* or *spline*; operators are generally math symbols like “+” or “×.”
- Arguments to a function are enclosed by parentheses, they come after the function’s name, and they’re separated by commas. Operands, on the other hand, can appear elsewhere. For example, you’ll often see $f(x, y)$ but you’ll rarely see $x f y$. Similarly, you’ll often find “ $x + y$ ” but you rarely find “ $+(x, y)$.”

Defining a Custom Operator

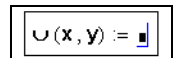
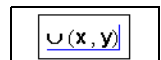
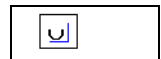
You define a custom operator just as if you were defining a function that happens to have an unusual looking name:

1. Type the operator name followed by a pair of parentheses. Enter the operands (two at the most) between the parentheses.
2. Enter the definition symbol $:=$.
3. Type an expression describing what you want the operator to do with its operands on the other side of the definition symbol.

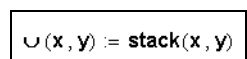
Tip Mathcad provides a collection of math symbols to define custom operators. To access these symbols, open the QuickSheets from the Resource Center (choose **Resource Center** on the **Help** menu) and then click on “Extra Math Symbols.” You can drag any of these symbols to your worksheet for use in creating a new operator name.

For example, suppose you want to define a new union operator using the symbol “ $\cup$ ”.

1. Drag the symbol into your worksheet from the “Extra Math Symbols” QuickSheet.
2. Type a left parenthesis followed by two names separated by a comma. Complete this argument list by typing a right parenthesis.
3. Press the colon (:) key, or click  on the Calculator toolbar. You see the definition symbol followed by a placeholder.



4. Type the function definition in the placeholder.



At this point, you’ve defined a function which behaves in every way like the user-defined functions described in Chapter 8, “Calculating in Mathcad.” You could, if you wanted to, type “ $\cup(1, 2) =$ ” in your worksheet and see the result, a vector with the elements 1 and 2, on the other side of the equal sign.

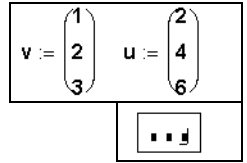
Tip Once you’ve defined the new operator, click on “Personal QuickSheets” in the QuickSheets of the Mathcad Resource Center. Then choose **Annotate Book** from the **Book** menu and drag or type the definition into the QuickSheet. Then choose **Save Section** from the **Book** menu. When you need to use this operator again, just open your Personal QuickSheet and drag it into a new worksheet.

Using a Custom Operator

Once you’ve defined a new operator, you can use it in your calculations just as you would use any of Mathcad’s built-in operators. The procedure for using a custom operator depends on whether the operator has one operand (like “–1 ” or “5!”) or two (like “1 ÷ 2 ”).

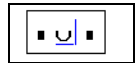
To use an operator having two operands:

1. Define any variables you want to use as arguments.

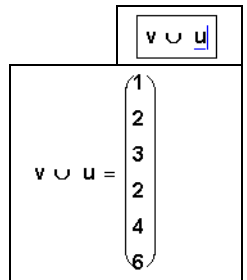


The diagram shows two variable definitions: $v := \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$ and $u := \begin{pmatrix} 2 \\ 4 \\ 6 \end{pmatrix}$. Below these is a small box containing three empty placeholders, representing the state of the Evaluation toolbar after clicking the 'x f y' button.

2. Click **x f y** on the Evaluation toolbar. You’ll see three empty placeholders.
3. In the middle placeholder, insert the name of the operator. Alternatively, copy the name from the operator definition and paste it into the placeholder.



4. In the remaining two placeholders, enter the two operands.
5. Press =, or click **=** on the Evaluation toolbar, to get the result.



The diagram shows the final result: $v \cup u = \begin{pmatrix} 1 \\ 2 \\ 3 \\ 2 \\ 4 \\ 6 \end{pmatrix}$. The operator '∪' is applied to the vectors v and u, resulting in a concatenated vector.

Tip An alternative way to display an operator having two operands is to click **x f y** on the Evaluation toolbar. If you follow the preceding steps using this operator, you’ll see a tree-shaped display.

To insert an operator having only one operand, decide first whether you want the operator to appear before the operand, as in “–1 ,” or after the operand as in “5!.” The former is called a *prefix* operator; the latter is a *postfix* operator. The example below shows how to use a postfix operator. The steps for creating a prefix operator are almost identical.

The following example shows how to define and use a new logical Not operator. First define an operator “ $\neg(x)$ ”. To do so, follow the steps for defining $\cup(x, y)$ in the previous section, substituting the symbol “ $\neg$ ” for “ $\cup$ ” and using only one argument instead of two.

$$\neg(x) := \neg x$$

Then, to evaluate with the new operator:

1. Click  on the Evaluation toolbar to make a *postfix* operator.

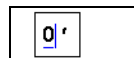


Otherwise, click . In either case, you see two empty placeholders.

2. If you clicked , put the operator name in the second placeholder. Otherwise put it in the first placeholder. In either case, you may find it more convenient to copy the name from the operator definition and paste it into the placeholder.



3. In the remaining placeholder, place the operand.



4. Press =, or click  on the Evaluation toolbar, to see the result.

$$0 \neg = 1$$

Tip Just as Mathcad can display a custom operator as if it were a function, you can conversely display a function as if it were an operator. For example, many publishers prefer to omit parentheses around the arguments to certain functions such as trigonometric functions, ie. $\sin x$ rather than $\sin(x)$. To create this notation, you can treat the $\sin$ function as an operator with one operand.

Chapter 10

Built-in Functions

- τ Inserting Built-in Functions
- τ Core Mathematical Functions
- τ Discrete Transform Functions
- τ Vector and Matrix Functions
- τ Solving and Optimization Functions
- τ Statistics, Probability, and Data Analysis Functions
- τ Finance Functions
- Pro** τ Differential Equation Functions
- τ Miscellaneous Functions

Inserting Built-in Functions

Mathcad's set of built-in functions can change depending on whether you've installed additional Extension Packs or whether you've written your own built-in functions. These functions can come from the following sources:

Built-in Mathcad functions

This is the core set of functions that come with Mathcad. These functions are introduced in this chapter.

Pro Mathcad Extension Packs

An Extension Pack consists of a collection of advanced functions geared to a particular area of application. Documentation for these functions comes with an Electronic Book accompanying the Extension Pack itself. The list of available Extension Packs currently includes collections for signal processing, image processing, steam tables, numerical analysis, and wavelets. To find out more about these and other Extension Packs, contact MathSoft or your local distributor, or visit MathSoft's Web site at:

<http://www.mathsoft.com/>

or refer to Mathcad Add-on Products in the on-line Resource Center (choose **Resource Center** from the **Help** menu). After you purchase and install an Extension Pack, the additional functions appear in the Insert Function dialog box.

Pro Built-In functions you write yourself in C

If you have Mathcad Professional and a supported 32-bit C/C++ compiler, you can write your own built-in functions. For details see the Adobe Acrobat file CREATING A USER DLL.PDF included on your installation CD.

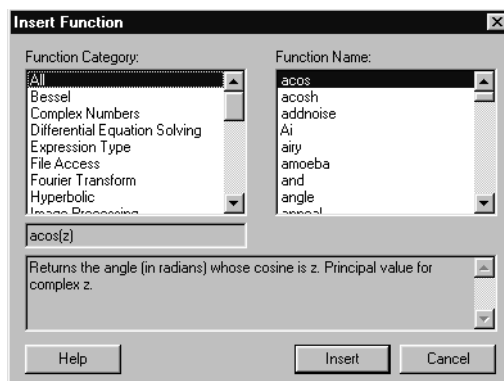
Insert Function Feature

To see a list of built-in functions available with your copy of Mathcad, arranged alphabetically or by category, or to insert a function together with placeholders for its arguments, use the Insert Function dialog box:

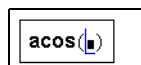
1. Click in a blank area of your worksheet or on a placeholder.
2. Choose **Function** from the

Insert menu or click  on the Standard toolbar. Mathcad opens the Insert Function dialog box.

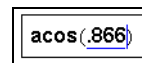
3. Click a Function Category or click “All” to see all available functions sorted alphabetically.



4. Double-click the name of the function you want to insert from the right-hand scrolling list, or click “Insert.” The function and placeholders for its arguments are inserted into the worksheet.



5. Fill in the placeholders.



To apply a function to an expression you have already entered, place the expression between the two editing lines and follow the steps given above. See Chapter 4, “Working with Math,” for information about using the editing lines.

You can also simply type the name of a built-in function directly into a math placeholder or in a math region.

Tip Although built-in function names are not font sensitive, they are case sensitive. If you do not use the Insert Function dialog box to insert a function name, you must enter the name of a built-in function in a math region exactly as it appears in the tables throughout this chapter: uppercase, lowercase, or mixed, as indicated.

Note Throughout this chapter and in the Insert Function dialog box, brackets, [], around an argument indicate that the argument is optional.

Assistance for Using Built-in Functions

Mathcad includes several sources of assistance for using built-in functions:

- The Insert Function dialog box gives you a convenient way to look up a function by category, to see the arguments required, and to see a brief function synopsis. When you click “Help” in the Insert Function dialog box, you immediately open the Help topic associated with the currently selected function.

- The on-line Help system (choose **Mathcad Help** from the **Help** menu, or click  on the Standard toolbar) provides both overview and detailed help topics on functions and function categories.
- The Resource Center (choose **Resource Center** from the **Help** menu, or click  on the Standard toolbar) includes a range of Mathcad files to help you use built-in functions to solve problems including Tutorials and QuickSheet examples.
- The *Mathcad Reference Manual*, which provides details on the syntax, arguments, algorithms, and behavior of all of Mathcad's built-in functions, operators, and keywords. The *Reference Manual* is provided as an Adobe Acrobat file on all Mathcad installation media. Mathcad Professional includes a printed copy.

Core Mathematical Functions

Trigonometric Functions

- $\text{angle}(x, y)$ Returns the angle (in radians) from the positive x -axis to point (x, y) in the x - y plane. The result is between 0 and 2π .
- $\cos(z)$ Returns the cosine of z . In a right triangle, this is the ratio of the length of the side *adjacent* to the angle over the length of the hypotenuse.
- $\cot(z)$ Returns $1/\tan(z)$, the cotangent of z . z should not be a multiple of π .
- $\csc(z)$ Returns $1/\sin(z)$, the cosecant of z . z should not be a multiple of π .
- $\sec(z)$ Returns $1/\cos(z)$, the secant of z . z should not be an odd multiple of $\pi/2$.
- $\sin(z)$ Returns the sine of z . In a right triangle, this is the ratio of the length of the side *opposite* the angle over the length of the hypotenuse.
- $\tan(z)$ Returns $\sin(z)/\cos(z)$, the tangent of z . In a right triangle, this is the ratio of the length of the side *opposite* the angle over the length of the side *adjacent* to the angle. z should not be an odd multiple of $\pi/2$.

Mathcad's trig functions and their inverses accept any scalar argument: real, complex, or imaginary. They also return complex numbers wherever appropriate.

Note Trigonometric functions expect their arguments in *radians*. To pass an argument in degrees, use the built-in unit *deg*. For example, to evaluate the sine of 45 degrees, type **sin(45*deg)**.

Tip In Mathcad you enter powers of trig functions such $\sin^2(x)$ as $\sin(x)^2$. Alternatively, you can use the prefix operator in Mathcad Professional, described on "Customizing Operators" on page 140. For example, to type $\sin^2(x)$. Click  on the Evaluation toolbar, enter **sin²** in the left-hand placeholder, enter **(x)** in the right-hand placeholder.

Inverse Trigonometric Functions

<code>acos(z)</code>	Returns the angle (in radians) whose cosine is z .
<code>acot(z)</code>	Returns the angle (in radians) whose cotangent is z .
<code>acsc(z)</code>	Returns the angle (in radians) whose cosecant is z .
<code>asec(z)</code>	Returns the angle (in radians) whose secant is z .
<code>asin(z)</code>	Returns the angle (in radians) whose sine is z .
<code>atan(z)</code>	Returns the angle (in radians) whose tangent is z .
<code>atan2(x, y)</code>	Returns the angle (in radians) from the positive x -axis to point (x, y) in the x - y plane.

With the exception of `atan2` and `acot`, the inverse trigonometric functions can take either a real or complex argument and return an angle in radians between $-\pi/2$ and $\pi/2$, or the principal value in the case of a complex argument. `atan2` takes only real arguments and returns a result between $-\pi$ and π , `acot` returns an angle in radians between 0 and π for a real argument or the principal value in the case of a complex argument.

To convert a result into degrees, either divide the result by the built-in unit *deg* or type **deg** in the units placeholder as described in “Displaying Units of Results” on page 113.

Hyperbolic Functions

<code>acosh(z)</code>	Returns the number whose hyperbolic cosine is z .
<code>acoth(z)</code>	Returns the number whose hyperbolic cotangent is z .
<code>acsch(z)</code>	Returns the number whose hyperbolic cosecant is z .
<code>asech(z)</code>	Returns the number whose hyperbolic secant is z .
<code>asinh(z)</code>	Returns the number whose hyperbolic sine is z .
<code>atanh(z)</code>	Returns the number whose hyperbolic tangent is z .
<code>cosh(z)</code>	Returns the hyperbolic cosine of z .
<code>coth(z)</code>	Returns $1/\tanh(z)$, the hyperbolic cotangent of z .
<code>csch(z)</code>	Returns $1/\sinh(z)$, the hyperbolic cosecant of z .
<code>sech(z)</code>	Returns $1/\cosh(z)$, the hyperbolic secant of z .
<code>sinh(z)</code>	Returns the hyperbolic sine of z .
<code>tanh(z)</code>	Returns $\sinh(z)/\cosh(z)$, the hyperbolic tangent of z .

Log and Exponential Functions

$\exp(z)$	Returns e raised to the power z .
$\ln(z)$	Returns the natural log of z . ($z \neq 0$).
$\log(z, b)$	Returns the base b logarithm of z . ($z \neq 0$, $b \neq 0$). If b is omitted, returns the base 10 logarithm.

Mathcad's exponential and logarithmic functions can accept and return complex arguments. $\ln$ returns the *principal branch* of the natural log function.

Bessel Functions

Pro	$\text{Ai}(x)$	Returns the value of the Airy function of the first kind. x must be real.
Pro	$\text{bei}(n, x)$	Returns the value of the imaginary Bessel Kelvin function of order n .
Pro	$\text{ber}(n, x)$	Returns the value of the real Bessel Kelvin function of order n .
Pro	$\text{Bi}(x)$	Returns the value of the Airy function of the second kind. x must be real.
	$\text{I0}(x)$	Returns the value of the zeroth order modified Bessel function of the first kind. x must be real.
	$\text{I1}(x)$	Returns the value of the first order modified Bessel function of the first kind. x must be real.
	$\text{In}(m, x)$	Returns the value of the m th order modified Bessel function of the first kind. x must be real, m is an integer, $0 \leq m \leq 100$.
	$\text{J0}(x)$	Returns the value of the zeroth order Bessel function of the first kind. x must be real.
	$\text{J1}(x)$	Returns the value of the first order Bessel function of the first kind. x must be real.
	$\text{Jn}(m, x)$	Returns the value of the m th order Bessel function of the first kind. x real, $0 \leq m \leq 100$.
Pro	$\text{js}(n, x)$	Returns the value of the spherical Bessel function of the first kind, of integer order n . $x > 0$, $n \geq -200$.
	$\text{K0}(x)$	Returns the value of the zeroth order modified Bessel function of the second kind. x real, $x > 0$.
	$\text{K1}(x)$	Returns the value of the first order modified Bessel function of the second kind. x real, $x > 0$.
	$\text{Kn}(m, x)$	Returns the value of the m th order modified Bessel function of the second kind. $x > 0$, m is an integer, $0 \leq m \leq 100$.

$Y0(x)$	Returns the value of the zeroth order Bessel function of the second kind. x real, $x > 0$.
$Y1(x)$	Returns the value of the first order Bessel function of the second kind. x real, $x > 0$.
$Yn(m, x)$	Returns the value of the m th order Bessel function of the second kind. $x > 0$, m is an integer, $0 \leq m \leq 100$.
Pro $ys(n, x)$	Returns the value of the spherical Bessel function of the second kind, of integer order n . x must be real. $x > 0$, $n \geq -200$.

Complex Numbers

$\arg(z)$	Returns the angle in complex plane from real axis to z . The result is between $-\pi$ and π radians.
$\text{csgn}(z)$	Returns 0 if $z = 0$, 1 if $\text{Re}(z) > 0$ or ($\text{Re}(z) = 0$ and $\text{Im}(z) > 0$), -1 otherwise.
$\text{Im}(z)$	Returns the imaginary part of a number z .
$\text{Re}(z)$	Returns the real part of a number z .
$\text{signum}(z)$	Returns 1 if $z = 0$, $z/ z $ otherwise.

Piecewise Continuous Functions

$\text{if}(cond, tvl, fvl)$	Returns tvl if $cond$ is nonzero (true), fvl if $cond$ is zero (false). $cond$ is usually a Boolean expression.
$\delta(m, n)$	Kronecker's delta function. Returns 1 if $m = n$, 0 otherwise. Both arguments must be integers.
$\varepsilon(i, j, k)$	Completely antisymmetric tensor of rank 3. i, j , and k must be integers between 0 and 2 inclusive (or between ORIGIN and ORIGIN + 2 inclusive if ORIGIN $\neq$ 0). Result is 0 if any two are the same, 1 for even permutations, -1 for odd permutations.
$\Phi(x)$	Heaviside step function. Returns 1 if $x \geq 0$, 0 otherwise. x must be real.
$\text{sign}(x)$	Returns 0 if $x = 0$, 1 if $x > 0$, and -1 otherwise. x must be real.

Note The *if* function is generally useful for branching in calculation: choosing one of two values based on a condition. Although the first argument of the *if* function, *cond*, can be any expression at all, it is usually a Boolean expression that relates two math expressions with a Boolean operator. See “Arithmetic and Boolean Operators” on page 125.

Number Theory/ Combinatorics

$\text{combin}(n, k)$	Returns the number of combinations: the number of subsets of size k that can be formed from n objects. n and k integers, $0 \leq k \leq n$.
$\text{gcd}(\mathbf{A}, \mathbf{B}, \mathbf{C}, \dots)$	Returns the greatest common divisor: the largest integer that evenly divides all the elements in arrays or scalars $\mathbf{A}, \mathbf{B}, \mathbf{C}, \dots$. The elements of $\mathbf{A}, \mathbf{B}, \mathbf{C}, \dots$ must be non-negative integers.
$\text{lcm}(\mathbf{A}, \mathbf{B}, \mathbf{C}, \dots)$	Returns the least common multiple: the smallest positive integer that is a multiple of all the elements in the arrays or scalars $\mathbf{A}, \mathbf{B}, \mathbf{C}, \dots$. The elements of $\mathbf{A}, \mathbf{B}, \mathbf{C}, \dots$ must be non-negative integers.
$\text{mod}(x, y)$	Remainder on dividing real number x by y ($y \neq 0$). Result has same sign as x .
$\text{permut}(n, k)$	Returns the number of permutations: the number of ways of ordering n distinct objects taken k at a time. n and k integers, $0 \leq k \leq n$.

Truncation and Round-Off Functions

$\text{ceil}(x)$	Least integer $\geq x$ (x real).
$\text{floor}(x)$	Greatest integer $\leq x$ (x real).
$\text{round}(x, n)$	Rounds real number x to n decimal places. If $n < 0$, x is rounded to the left of the decimal point. If n is omitted, returns x rounded to the nearest integer.
$\text{trunc}(x)$	Returns the integer part of a real number x by removing the fractional part.

Special Functions

	$\text{erf}(x)$	Returns the value of the error function at x . x must be real.
	$\text{erfc}(x)$	Returns the value of the complementary error function at x : $1 - \text{erf}(x)$. x real.
Pro	$\text{fhyper}(a, b, c, x)$	Returns the value of the Gauss hypergeometric function at the point x given parameters a, b, c . $-1 < x < 1$.
	$\Gamma(z)$	Returns the value of the classical Euler gamma function at z , a real or complex number. Undefined for $z = 0, -1, -2, \dots$
	$\Gamma(x, y)$	Returns the value of the extended Euler gamma function for real numbers $x > 0, y \geq 0$.
Pro	$\text{Her}(n, x)$	Returns the value of the Hermite polynomial of degree n at x .
Pro	$\text{ibeta}(a, x, y)$	Returns the value of the incomplete beta function of x and y with parameter a . $0 \leq a \leq 1$

Pro	Jac(n, a, b, x)	Returns the value of the Jacobi polynomial of degree n at x with parameters a and b .
Pro	Lag(n, x)	Returns the value of the Laguerre polynomial of degree n at x .
Pro	Leg(n, x)	Returns the value of the Legendre polynomial of degree n at x .
Pro	mhyper(a, b, x)	Returns the value of the confluent hypergeometric function at the point x given parameters a and b .
Pro	Tcheb(n, x)	Returns the value of the Chebyshev polynomial of degree n , of the first kind, at x .
Pro	Ucheb(n, x)	Returns the value of the Chebyshev polynomial of degree n , of the second kind, at x .

Discrete Transform Functions

Mathcad contains a variety of functions for performing discrete transforms. All of these functions require vectors as arguments.

Note When you define a vector **v** for use with Fourier or wavelet transforms, Mathcad indexes the vector beginning at 0, by default, unless you have set the value of the built-in variable **ORIGIN** to a value other than 0 (see page 98). If you do not define v_0 , Mathcad automatically sets it to zero. This can distort the results of the transform functions.

Fourier Transforms on Real and Complex Data

cfft(A)	Returns the fast Fourier transform of a vector or matrix of complex data representing equally spaced measurements in the time domain. The array returned is the same size as its argument.
icfft(A)	Returns the inverse Fourier transform of a vector or matrix of data corresponding to <i>cfft</i> . Returns an array of the same size as its argument.
fft(v)	Returns the fast discrete Fourier transform of a 2^m element vector of real data representing measurements at regular intervals in the time domain. $m > 2$.
ifft(v)	Returns the inverse Fourier transform of a vector of data corresponding to <i>fft</i> . Takes a vector of size $1 + 2^{n-1}$, and returns a real vector of size 2^n . $n > 2$.
CFFT(A)	Returns a transform identical to <i>cfft</i> , except using a different normalizing factor and sign convention.
ICFFT(A)	Returns the inverse Fourier transform of a vector or matrix of data corresponding to <i>CFFT</i> . Returns an array of the same size as its argument.

FFT(v)	Returns a transform identical to <i>fft</i> , except using a different normalizing factor and sign convention.
IFFT(v)	Returns the inverse Fourier transform of a vector of data corresponding to <i>FFT</i> . Takes a vector of size $1 + 2^{n-1}$, and returns a real vector of size 2^n .

Mathcad comes with two types of Fourier transform pairs: *fft* / *ifft* (or the alternative *FFT* / *IFFT*) and *cfft* / *icfft* (or the alternative *CFFT* / *ICFFT*). These functions are discrete: they apply to and return vectors and matrices only. You cannot use them with other functions.

Use the *fft* and *ifft* (or *FFT* / *IFFT*) functions if:

- the data values in the time domain are real, and
- the data vector has 2^m elements.

Use the *cfft* and *icfft* (or *CFFT* / *ICFFT*) functions in all other cases.

Be sure to use these functions in pairs. For example, if you used *CFFT* to go from the time domain to the frequency domain, you must use *ICFFT* to transform back to the time domain. See Figure 10-1 for an example.

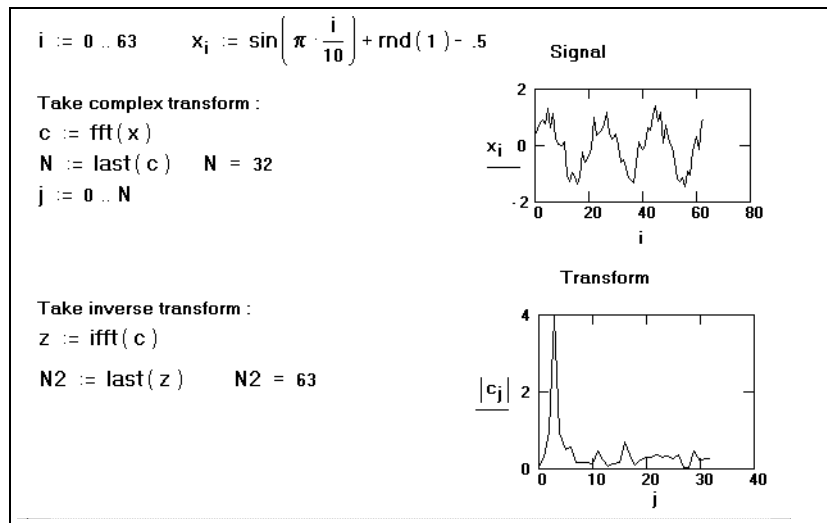


Figure 10-1: Use of fast Fourier transforms in Mathcad. Since the random number generator gives different numbers every time, you may not be able to recreate this example exactly as you see it.

Note Different sources use different conventions concerning the initial factor of the Fourier transform and whether to conjugate the results of either the transform or the inverse transform. The functions *fft*, *ifft*, *cfft*, and *icfft* use $1/\sqrt{N}$ as a normalizing factor and a positive exponent in going from the time to the frequency domain. The functions *FFT*, *IFFT*, *CFFT*, and *ICFFT* use $1/N$ as a normalizing factor and a negative exponent in going from the time to the frequency domain.

Wavelet Transforms

- Pro** `wave(v)` Returns the discrete wavelet transform of **v**, a 2^m element vector containing real data, using the Daubechies four-coefficient wavelet filter. The vector returned is the same size as **v**.
- Pro** `iwave(v)` Returns the inverse discrete wavelet transform of **v**, a 2^m element vector containing real data. The vector returned is the same size as **v**.

Vector and Matrix Functions

Note that functions that expect vectors always expect column vectors rather than row vectors. To change a row vector into a column vector, use the transpose operator (click



on the Matrix toolbar).

Size and Scope of an Array

- `cols(A)` Returns the number of columns in array **A**. If **A** is a scalar, returns 0.
- `last(v)` Returns the index of the last element in vector **v**.
- `length(v)` Returns the number of elements in vector **v**.
- `max(A, B, C, ...)` Returns the largest of the strings, arrays, or scalars **A, B, C, ...** If any value is complex, returns the largest real part plus i times the largest imaginary part.
- `min(A, B, C, ...)` Returns the smallest of the strings, arrays, or scalars **A, B, C, ...** If any value is complex, returns the smallest real part plus i times the smallest imaginary part.
- `rows(A)` Returns the number of rows in array **A**. If **A** is a scalar, returns 0.

Special Types of Matrices

- Pro** `diag(v)` Returns a diagonal matrix containing on its diagonal the elements of **v**.
- Pro** `geninv(A)` Returns the left inverse matrix **L** of **A**, such that $\mathbf{L} \cdot \mathbf{A} = \mathbf{I}$, where **I** is the identity matrix having the same number of columns as **A**. Matrix **A** is an $m \times n$ real-valued matrix, where $m \geq n$.
- `identity(n)` Returns an $n \times n$ matrix of 0's with 1's on the diagonal.
- `rref(A)` Returns the reduced-row echelon form of **A**.

$$\begin{array}{l}
 \mathbf{v} := \begin{bmatrix} 2 \\ 8 \\ 9 \\ 7 \end{bmatrix} \quad \text{diag}(\mathbf{v}) = \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 8 & 0 & 0 \\ 0 & 0 & 9 & 0 \\ 0 & 0 & 0 & 7 \end{bmatrix} \quad \leftarrow \text{A diagonal matrix formed from a vector. (Mathcad Professional)} \\
 \\
 \mathbf{A} := \begin{pmatrix} 2 & 4 & 6 \\ 4 & 5 & 6 \\ 2 & 7 & 12 \end{pmatrix} \quad \text{rref}(\mathbf{A}) = \begin{pmatrix} 1 & 0 & -1 \\ 0 & 1 & 2 \\ 0 & 0 & 0 \end{pmatrix} \quad \leftarrow \text{The reduced-row echelon form of a matrix.} \\
 \\
 \mathbf{B} := \begin{bmatrix} 5 + 2i \\ 2.54 - 3i \\ 3 + (4 + .8) \cdot i \end{bmatrix} \quad \text{Im}(\mathbf{B}) = \begin{pmatrix} 2 \\ -3 \\ 4.8 \end{pmatrix} \quad \leftarrow \text{The imaginary part of a matrix.}
 \end{array}$$

Figure 10-2: Functions for transforming arrays.

Special Characteristics of a Matrix

- Pro** $\text{cond1}(\mathbf{M})$ Returns the condition number of the matrix $\mathbf{M}$ based on the L_1 norm.
- Pro** $\text{cond2}(\mathbf{M})$ Returns the condition number of the matrix $\mathbf{M}$ based on the L_2 norm.
- Pro** $\text{conde}(\mathbf{M})$ Returns the condition number of the matrix $\mathbf{M}$ based on the Euclidean norm.
- Pro** $\text{condi}(\mathbf{M})$ Returns the condition number of the matrix $\mathbf{M}$ based on the infinity norm.
- Pro** $\text{norm1}(\mathbf{M})$ Returns the L_1 norm of the matrix $\mathbf{M}$.
- Pro** $\text{norm2}(\mathbf{M})$ Returns the L_2 norm of the matrix $\mathbf{M}$.
- Pro** $\text{norme}(\mathbf{M})$ Returns the Euclidean norm of the matrix $\mathbf{M}$.
- Pro** $\text{normi}(\mathbf{M})$ Returns the infinity norm of the matrix $\mathbf{M}$.
- $\text{rank}(\mathbf{A})$ Returns the rank of the real-valued matrix $\mathbf{A}$.
- $\text{tr}(\mathbf{M})$ Returns the sum of the diagonal elements, known as the *trace*, of $\mathbf{M}$.

Forming New Matrices

- $\text{augment}(\mathbf{A}, \mathbf{B}, \mathbf{C}, \dots)$ Returns an array formed by placing $\mathbf{A}, \mathbf{B}, \mathbf{C}, \dots$ left to right. $\mathbf{A}, \mathbf{B}, \mathbf{C}, \dots$ are arrays having the same number of rows or they are scalars and single-row arrays.

<code>CreateMesh(F, [[s0], [s1], [t0], [t1], [sgrid], [tgrid], [fmap]])</code>	Returns a nested array of three matrices representing the x-, y-, and z-coordinates of a parametric surface defined by the function, F . F is a three-element vector-valued function of two variables. <i>s0</i> , <i>s1</i> , <i>t0</i> , and <i>t1</i> are the variable limits, and <i>sgrid</i> and <i>tgrid</i> are the number of gridpoints. All must be real scalars. fmap is a three-element vector-valued mapping function. All arguments but the function argument are optional.
<code>CreateSpace(F, [[t0], [t1], [tgrid], [fmap]])</code>	Returns a nested array of three vectors representing the x-, y-, and z-coordinates of a space curve defined by the function, F . F is a three-element vector-valued function of one variable. <i>t0</i> and <i>t1</i> are the variable limits, and <i>tgrid</i> is the number of gridpoints. All must be real scalars. fmap is a three-element vector-valued mapping function. All arguments but the function argument are optional.
<code>matrix(m, n, f)</code>	Creates a matrix in which the <i>i,j</i> th element contains <i>f(i, j)</i> where <i>i</i> = 0, 1, ..., <i>m</i> - 1 and <i>j</i> = 0, 1, ..., <i>n</i> - 1. Function <i>f</i> must have been defined previously in the worksheet.
<code>stack(A, B, C, ...)</code>	Returns an array formed by placing A, B, C, ... top to bottom. A, B, C, ... are arrays having the same number of columns or they are scalars and vectors.
<code>submatrix(A, ir, jr, ic, jc)</code>	Returns a submatrix of A consisting of all elements contained in rows <i>ir</i> through <i>jr</i> and columns <i>ic</i> through <i>jc</i> . Make sure <i>ir</i> ≤ <i>jr</i> and <i>ic</i> ≤ <i>jc</i> or the order of rows or columns will be reversed.

Note For the functions *CreateMesh* and *CreateSpace*, instead of using a vector-valued function, **F**, you can use three functions, **f1**, **f2**, and **f3**, representing the x-, y-, and z-coordinates of the parametric surface or space curve. Your call to one of these functions might look something like this: *CreateMesh(f1, f2, f3)*.

Alternatively, for **CreateMesh**, you can use a single function of two variables such as

$$F(x, y) = \frac{\sin(x) + \cos(y)}{2}.$$

Figure 10-3 shows examples of using *stack* and *augment*.

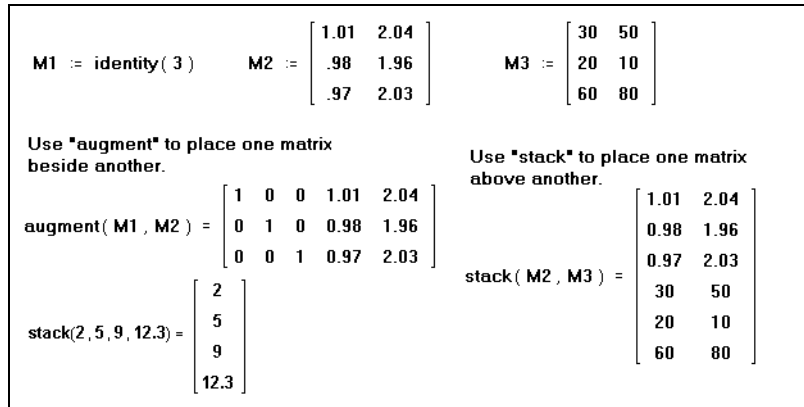


Figure 10-3: Joining matrices with the augment and stack functions.

Eigenvalues and Eigenvectors

- $\text{eigenvals}(\mathbf{M})$ Returns a vector containing the eigenvalues of the square matrix $\mathbf{M}$.
- $\text{eigenvec}(\mathbf{M}, z)$ Returns a vector containing the normalized eigenvector corresponding to the eigenvalue z of the square matrix $\mathbf{M}$.
- Pro** $\text{eigenvecs}(\mathbf{M})$ Returns a matrix containing normalized eigenvectors corresponding to the eigenvalues of the square matrix $\mathbf{M}$. The n th column of the matrix returned is an eigenvector corresponding to the n th eigenvalue returned by *eigenvals*.
- Pro** $\text{genvals}(\mathbf{M}, \mathbf{N})$ Returns a vector $\mathbf{v}$ of computed eigenvalues each of which satisfies the generalized eigenvalue problem $\mathbf{M} \cdot \mathbf{x} = v_i \cdot \mathbf{N} \cdot \mathbf{x}$. Vector $\mathbf{x}$ is the corresponding eigenvector. $\mathbf{M}$ and $\mathbf{N}$ are real square matrices having the same number of columns.
- Pro** $\text{genvecs}(\mathbf{M}, \mathbf{N})$ Returns a matrix containing the normalized eigenvectors corresponding to the eigenvalues in $\mathbf{v}$, the vector returned by *genvals*. The n th column of this matrix is the eigenvector $\mathbf{x}$ satisfying the generalized eigenvalue problem $\mathbf{M} \cdot \mathbf{x} = v_n \cdot \mathbf{N} \cdot \mathbf{x}$. Matrices $\mathbf{M}$ and $\mathbf{N}$ are real-valued square matrices having the same number of columns.

Figure 10-4 shows how some of these functions are used.

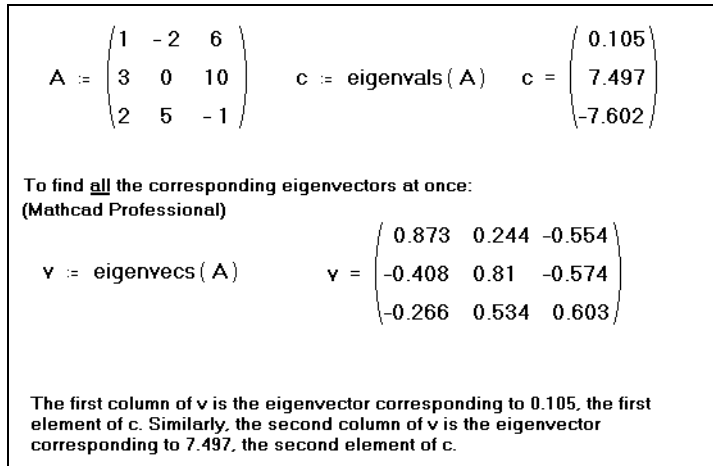


Figure 10-4: Eigenvalues and eigenvectors in Mathcad.

Solving a Linear System of Equations

Pro `lsolve(M, v)` Returns a solution vector x such that $M \cdot x = v$. v is a vector having the same number of rows as the matrix M .

With Mathcad Professional, you can use the *lsolve* function to solve a linear system of equations whose coefficients are arranged in a matrix M .

Note The argument M for *lsolve* must be a matrix that is neither singular nor nearly singular. An alternative to *lsolve* is to solve a linear system by using matrix inversion. See “Solving and Optimization Functions” on page 157 for additional solving functions.

Decomposition

Pro `cholesky(M)` Returns a lower triangular matrix L such that $L \cdot L^T = M$. This uses only the upper triangular part of M . The upper triangular of M , when reflected about the diagonal, must form a positive definite matrix.

Pro `lu(M)` Returns a single matrix containing the three square matrices P , L , and U , all having the same size as M and joined together side by side, in that order. These three matrices satisfy the equation $P \cdot M = L \cdot U$, where L and U are lower and upper triangular respectively.

Pro `qr(A)` Returns a matrix whose first n columns contain the square, orthonormal matrix Q , and whose remaining columns contain the upper triangular matrix, R . Matrices Q and R satisfy the equation $A = Q \cdot R$, where A is a real-valued array.

Pro	svd(A)	Returns a single matrix containing two stacked matrices U and V , where U is the upper $m \times n$ submatrix and V is the lower $n \times n$ submatrix. Matrices U and V satisfy the equation $\mathbf{A} = \mathbf{U} \cdot \text{diag}(\mathbf{s}) \cdot \mathbf{V}^T$, where s is a vector returned by svds(A). A is an $m \times n$ array of real values, where $m \geq n$.
Pro	svds(A)	Returns a vector containing the singular values of the $m \times n$ real-valued array A , where $m \geq n$.

Sorting Functions

	csort(A, <i>n</i>)	Returns an array formed by rearranging rows of the matrix A such that the elements in column <i>n</i> are in ascending order. The result has the same size as A .
	reverse(A)	Returns an array in which the elements of a vector, or the rows of a matrix, are in reverse order.
	rsort(A, <i>n</i>)	Returns an array formed by rearranging the columns of the matrix A such that the elements in row <i>n</i> are in ascending order. The result has the same size as A .
	sort(v)	Returns the elements of the vector v sorted in ascending order.

Tip Unless you change the value of ORIGIN, matrices are numbered by default starting with row zero and column zero. To sort on the first column of a matrix, for example, use **csort(**A**, 0)**.

Solving and Optimization Functions

This section describes how to solve equations ranging from a single equation in one unknown to large systems with multiple unknowns. The techniques described here generate numeric solutions. Chapter 14, “Symbolic Calculation,” describes a variety of techniques for solving equations symbolically.

Finding Roots

	polyroots(v)	Returns the roots of an <i>n</i> th degree polynomial whose coefficients are in v , a vector of length $n + 1$. Returns a vector of length <i>n</i> .
	root(<i>f</i>(z), z)	Returns the value of <i>z</i> which the expression <i>f</i> (<i>z</i>) is equal to 0. The arguments are a real- or complex-valued expression <i>f</i> (<i>z</i>) and a real or complex scalar, <i>z</i> . Must be preceded in the worksheet by a guess value for <i>z</i> . Returns a scalar.
	root(<i>f</i>(z), z, <i>a</i>, <i>b</i>)	Returns the value of <i>z</i> lying between <i>a</i> and <i>b</i> at which the expression <i>f</i> (<i>z</i>) is equal to 0. The arguments to this function are a real-valued expression <i>f</i> (<i>z</i>), a real scalar, <i>z</i> , and real endpoints $a < b$. No guess value for <i>z</i> is required. Returns a scalar.

Note When you specify the arguments a and b for the root function, Mathcad will only find a root for the function f if $f(a)$ is positive and $f(b)$ is negative or vice versa.

The *root* function solves a single equation in a single unknown. This function takes an arbitrary expression or function and one of the variables from the expression. *root* can also take a range in which the solution lies. It then varies that variable until the expression is equal to zero and lies in the specified range. Once this is done, the function returns the value that makes the expression equal zero and lies in the specified range.

Tip *root* makes successive estimates of the value of the root and returns a value when the two most recent estimates differ by less than the value of the tolerance parameter, TOL. As described in “Built-in Variables” on page 98, you can change the value of the tolerance, and hence the accuracy of the solution, by including definitions for TOL directly in your worksheet. You can also change the tolerance by using the Built-In Variables tab when you choose **Options** from the **Math** menu.

To find the roots of a polynomial or an expression having the form:

$$v_n x^n + \dots + v_2 x^2 + v_1 x + v_0$$

you can use *polyroots* rather than *root*. *polyroots* does not require a guess value, and *polyroots* returns all roots at once, whether real or complex. Figure 10-5 shows examples.

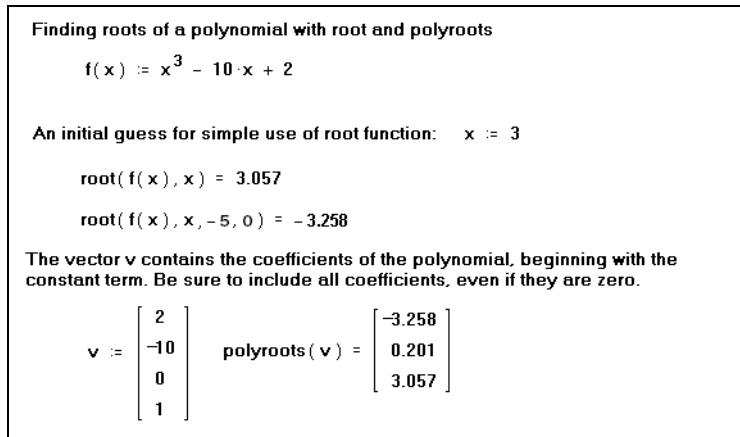


Figure 10-5: Finding roots with root and polyroots.

By default, *polyroots* uses a LaGuerre method of finding roots. If you want to use the companion matrix method instead, click on the *polyroots* function with the right mouse button and choose **Companion Matrix** from the pop-up menu.

Note *root* and *polyroots* can solve only one equation in one unknown, and they always return numerical answers. To solve several equations simultaneously, use the techniques described in the next section, “Linear/Nonlinear System Solving and Optimization.” To solve an equation symbolically, or to find an exact numerical answer in terms of elementary functions, choose

Linear/ Nonlinear System Solving and Optimization

Mathcad includes powerful numerical solving functions that readily solve problems of the following types:

- Linear systems of equations with constraints (equalities or inequalities).
- Nonlinear systems of equations with constraints.
- Optimization (maximization or minimization) of an objective function.
- Optimization (maximization or minimization) of an objective function with constraints.
- Linear programming, in which all constraints are either equalities or inequalities that compare linear functions to constants and the objective function is of the form:

$$c_0x_0 + c_1x_1 + \dots + c_nx_n$$

- Pro**
- Quadratic programming, in which all constraints are linear but the objective function contains not only linear terms but also quadratic terms. Quadratic programming features are available in the *Solving and Optimization Extension Pack (Expert Solver)* for Mathcad Professional, available for separate sale from MathSoft or your local distributor or reseller.

Note The size of the problem you can solve depends on your version of Mathcad. Mathcad Standard solves systems of up to 50 variables, while Mathcad Professional solves nonlinear systems of up to 200 variables and linear systems of up to 500 variables. The *Solving and Optimization Extension Pack (Expert Solver)* for Mathcad Professional solves nonlinear systems of up to 250 variables, linear systems of up to 1000 variables, and quadratic systems of up to 1000 variables.

Solve Blocks

The general form for using system solving functions in Mathcad is within the body of a *solve block*. There are four general steps to creating a solve block. These are:

1. Provide an initial guess (definition) for each of the unknowns you intend to solve for. Mathcad solves equations by making iterative calculations that ultimately converge on a valid solution. The initial guesses you provide give Mathcad a place to start searching for solutions. Guess values are usually required for all systems.
2. Type the word *Given* in a separate math region below the guess definitions. This tells Mathcad that what follows is a system of constraint equations. Be sure you don't type “Given” in a text region.
3. Now enter the constraints (equalities and inequalities) in any order below the word *Given*. Make sure you use the bold equal symbol (click  on the Boolean toolbar or press [Ctrl]=) for any equality. You can separate the left and right sides of an inequality with any of the symbols <, >, ≤, and ≥.
4. Enter any equation that involves one of the functions *Find*, *Maximize*, *Minimize*, or *Minerr* below the constraints.

Note Solve blocks cannot be nested inside each other—each solve block can have only one *Given* and one *Find* (or *Maximize*, *Minimize*, or *Minerr*). You can, however, define a function like $f(x) := \text{Find}(x)$ at the end of one solve block and refer to this function in another solve block.

Solve Blocks

- Find**($z0, z1, \dots$) Returns values of $z0, z1, \dots$ that satisfy the constraints in a solve block. $z0, z1, \dots$ are real or complex scalars, vectors, arrays, or individual elements of vectors equal in number to the number of unknowns in the system. Returns a scalar for a single unknown; otherwise returns a vector of solutions.
- Maximize**($f, z0, z1, \dots$) Returns values of $z0, z1, \dots$ that make the function f take on its largest value. $z0, z1, \dots$ are real or complex scalars, vectors, arrays, or individual elements of vectors equal in number to the number of unknowns in the system. Returns a scalar for a single unknown; otherwise returns a vector of solutions. Solve block constraints are optional.
- Minerr**($z0, z1, \dots$) Returns values of $z0, z1, \dots$ that come closest to satisfying the constraints in a solve block. $z0, z1, \dots$ are real or complex scalars, vectors, arrays, or individual elements of vectors equal in number to the number of unknowns in the system. Returns a scalar for a single unknown; otherwise returns a vector of solutions.
- Minimize**($f, z0, z1, \dots$) Returns values of $z0, z1, \dots$ that make the function f take on its smallest value. $z0, z1, \dots$ are real or complex scalars, vectors, arrays, or individual elements of vectors equal in number to the number of unknowns in the system. Returns a scalar for a single unknown; otherwise returns a vector of solutions. Solve block constraints are optional.

Tip Unlike most Mathcad functions, the solving functions *Find*, *Maximize*, *Minerr*, and *Minimize* can be entered in math regions with either an initial lowercase or an initial capital letter.

Figure 10-6 shows a solve block with several kinds of constraints and ending with a call to the *Find* function. There are two unknowns. As a result, the *Find* function here takes two arguments, x and y , and returns a vector with two elements.

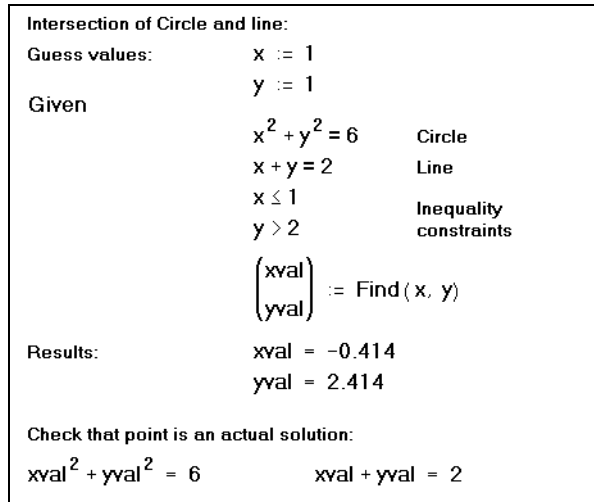


Figure 10-6: A solve block with both equalities and inequalities.

Constraints

The table below lists the kinds of constraints that can appear in a solve block between the keyword *Given* and one of the functions *Find*, *Maximize*, *Minerr*, and *Minimize*. In the table, x and y represent real-valued expressions, and z and w represent arbitrary expressions. The Boolean constraints are inserted using buttons on the Boolean toolbar. While constraints are often scalar expressions, they may also be vector or array expressions.

Condition	Button	Description
$w = z$		Constrained to be equal.
$x > y$		Greater than.
$x < y$		Less than.
$x \geq y$		Greater than or equal to.
$x \leq y$		Less than or equal to.
$x \wedge y$		And
$x \vee y$		Or
$x \otimes y$		Xor (Exclusive Or)
$\neg x$		Not

Mathcad does not allow the following inside a solve block:

- Constraints with “ $\neq$.”
- Assignment statements (statements like $x := 1$).

You can, however, include compound statements such as $1 \leq x \leq 3$.

Note Mathcad returns only one solution for a solve block. There may, however, be multiple solutions to a set of equations. To find a different solution, try different guess values or enter an additional inequality constraint that the current solution does not satisfy.

Tolerances for solving

Mathcad's numerical solvers make use of two tolerance parameters in calculating solutions in solve blocks:

- **Convergence tolerance.** The solvers calculate successive estimates of the values of the solutions and return values when the two most recent estimates differ by less than the value of the built-in variable TOL. A smaller value of TOL often results in a more accurate solution, but the solution may take longer to calculate.
- **Constraint tolerance.** This parameter, determined by the value of the built-in variable CTOL, controls how closely a constraint must be met for a solution to be acceptable. For example, if the constraint tolerance were 0.0001, a constraint such as $x < 2$ would be considered satisfied if, in fact, the value of x satisfied $x < 2.0001$.

Procedures for modifying the values of these tolerances are described in “Built-in Variables” on page 98.

Tip If you use *Minerr* in a solve block, you should always include additional checks on the reasonableness of the results. The built-in variable ERR returns the size of the error vector for the approximate solution returned by *Minerr*. There is no built-in variable for determining the size of the error for individual solutions to the unknowns.

Solving algorithms and AutoSelect

When you solve an equation, by default Mathcad uses an *AutoSelect* procedure to choose an appropriate solving algorithm. You can override Mathcad's choice of algorithm and select another available algorithm yourself.

Here are the available solving methods:

Linear

Applies a linear programming algorithm to the problem. Guess values for the unknowns are not required.

Nonlinear

Applies either a conjugate gradient, Levenberg-Marquardt, or quasi-Newton solving routine to the problem. Guess values for all unknowns must precede the solve block. If you use Mathcad Professional, choose **Nonlinear**⇒**Advanced Options** from the pop-up menu to control settings for the conjugate gradient and quasi-Newton solvers.

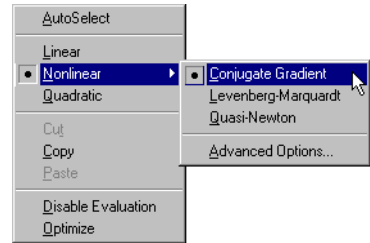
Note The Levenberg-Marquardt method is not available for the *Maximize* and *Minimize* functions.

Pro Quadratic

Applies a quadratic programming algorithm to the problem. The option is available only if the *Solving and Optimization Extension Pack (Expert Solver)* for Mathcad Professional is installed. Guess values for the unknowns are not required.

You can override Mathcad's default choice of solving algorithm as follows:

5. Create and evaluate a solve block, allowing Mathcad to AutoSelect an algorithm.
6. Click with the right mouse button on the name of the function that terminates the solve block, and remove the check from **AutoSelect** on the pop-up menu.
7. Check one of the available solving methods on the pop-up menu. Mathcad recalculates the solution using the method you selected.



Pro Reports

If you have the *Solving and Optimization Extension Pack (Expert Solver)* for Mathcad Professional installed, you can generate reports for a linear optimization problems. To generate a report, click on a solving function with the right mouse button and choose **Report** from the pop-up menu. For more information on reports, refer to the on-line Help.

Pro Mixed integer programming

If you have *Solving and Optimization Extension Pack (Expert Solver)* for Mathcad Professional installed, you can perform mixed integer programming. Using mixed integer programming you can force the solution for an unknown variable to be a binary number (1 or 0) or an integer. For more information mixed integer programming, refer to the on-line Help.

Statistics, Probability, and Data Analysis Functions

Statistics

- | | |
|---|---|
| $\text{corr}(\mathbf{A}, \mathbf{B})$ | Returns the Pearson's r correlation coefficient for the $m \times n$ arrays $\mathbf{A}$ and $\mathbf{B}$. |
| $\text{cvar}(\mathbf{A}, \mathbf{B})$ | Returns the covariance of the elements in $m \times n$ arrays $\mathbf{A}$ and $\mathbf{B}$. |
| $\text{gmean}(\mathbf{A}, \mathbf{B}, \mathbf{C}, \dots)$ | Returns the geometric mean of the elements of the arrays or scalars $\mathbf{A}, \mathbf{B}, \mathbf{C}, \dots$. All elements must be real and greater than 0. |

<code>hist(int, A)</code>	Returns a vector representing the frequencies with which values in A fall in the intervals represented by int . When int is a vector of intervals in ascending order, the <i>i</i> th element of the returned vector is the number of points in data falling between the <i>i</i> th and (<i>i</i> +1)th element of int . When int is an integer, it is the number of subintervals of equal length. Both int and A must be real.
<code>hmean(A, B, C, ...)</code>	Returns the harmonic mean of the elements of the arrays or scalars A, B, C, ... All elements must be nonzero.
<code>kurt(A, B, C, ...)</code>	Returns the kurtosis of the elements of the arrays or scalars A, B, C, ...
<code>mean(A, B, C, ...)</code>	Returns the arithmetic mean of the elements of the arrays or scalars A, B, C, ...
<code>median(A, B, C, ...)</code>	Returns the median of the elements of the arrays or scalars A, B, C, ... , the value above and below which there are an equal number of values. If there are an even number of elements, this is the arithmetic mean of the two central values.
<code>mode(A, B, C, ...)</code>	Returns the element from the arrays or scalars A, B, C, ... that occurs most often.
<code>skew(A, B, C, ...)</code>	Returns the skewness of the elements of the arrays or scalars A, B, C, ...
<code>stdev(A, B, C, ...)</code>	Returns the population standard deviation (square root of the variance) of the elements of the arrays or scalars A, B, C, ...
<code>Stdev(A, B, C, ...)</code>	Returns the sample standard deviation (square root of the sample variance) of the elements of the arrays or scalars A, B, C, ...
<code>var(A, B, C, ...)</code>	Returns the population variance of the elements of the arrays or scalars A, B, C, ...
<code>Var(A, B, C, ...)</code>	Returns the sample variance of the elements of the arrays or scalars A, B, C, ...

Probability Distributions

Mathcad includes functions for working with several common probability densities. These functions fall into four classes:

- **Probability densities.** These functions, beginning with the letter “d,” give the likelihood that a random variable will take on a particular value.
- **Cumulative probability distributions.** These functions, beginning with the letter “p,” give the probability that a random variable will take on a value *less than or equal to* a specified value. These are obtained by simply integrating (or summing when appropriate) the corresponding probability density from $-\infty$ to a specified value.
- **Inverse cumulative probability distributions.** These functions, beginning with the letter “q,” take a probability *p* between 0 and 1 as an argument and return a

value such that the probability that a random variable will be *less than or equal to* that value is p .

- **Random number generators.** These functions, beginning with the letter “r,” return a vector of m elements drawn from the corresponding probability distribution. Each time you recalculate an equation containing one of these functions, Mathcad generates new random numbers.

Tip Mathcad’s random number generators have a “seed value” associated with them. A given seed value always generates the same sequence of random numbers, and choosing **Calculate** from the **Math** menu advances Mathcad along this random number sequence. Changing the seed value, however, advances Mathcad along a different random number sequence. To change the seed value, choose **Options** from the **Math** menu and enter a value on the Built-In Variables tab.

Note See comments above about the nomenclature, arguments, and returned values for the probability distribution functions in the following table.

Probability Distributions

$\text{dbeta}(x, s_1, s_2)$ $\text{pbeta}(x, s_1, s_2)$ $\text{qbeta}(p, s_1, s_2)$ $\text{rbeta}(m, s_1, s_2)$	Probability distribution functions for the beta distribution in which $(s_1, s_2 > 0)$ are the shape parameters. $0 < x < 1$. $0 \leq p \leq 1$.
$\text{dbinom}(k, n, p)$ $\text{pbinom}(k, n, p)$ $\text{qbinom}(p, n, r)$ $\text{rbinom}(m, n, p)$	Probability distribution functions for the binomial distribution in which n and k are integers satisfying $0 \leq k \leq n$. $0 \leq p \leq 1$. r is the probability of success on a single trial.
$\text{dcauchy}(x, l, s)$ $\text{pcauchy}(x, l, s)$ $\text{qcauchy}(p, l, s)$ $\text{rcauchy}(m, l, s)$	Probability distribution functions for the Cauchy distribution in which l is a location parameter and $s > 0$ is a scale parameter. $0 < p < 1$.
$\text{dchisq}(x, d)$ $\text{pchisq}(x, d)$ $\text{qchisq}(p, d)$ $\text{rchisq}(m, d)$	Probability distribution functions for the chi-squared distribution in which $d > 0$ are the degrees of freedom and $x \geq 0$. $0 \leq p < 1$.
$\text{dexp}(x, r)$ $\text{pexp}(x, r)$ $\text{qexp}(p, r)$ $\text{rexp}(m, r)$	Probability distribution functions for the exponential distribution in which $r > 0$ is the rate and $x \geq 0$. $0 \leq p < 1$.
$\text{dF}(x, d_1, d_2)$ $\text{pF}(x, d_1, d_2)$ $\text{qF}(p, d_1, d_2)$ $\text{rF}(m, d_1, d_2)$	Probability distribution functions for the F distribution in which $(d_1, d_2 > 0)$ are the degrees of freedom and $x \geq 0$. $0 \leq p < 1$.
$\text{dgamma}(x, s)$ $\text{pgamma}(x, s)$ $\text{qgamma}(p, s)$ $\text{rgamma}(m, s)$	Probability distribution functions for the gamma distribution in which $s > 0$ is the shape parameter and $x \geq 0$. $0 \leq p < 1$.

dgeom(k, p) pgeom(k, p) qgeom(p, r) rgeom(m, p)	Probability distribution functions for the geometric distribution in which $0 < p \leq 1$ is the probability of success and k is a nonnegative integer. r is the probability of success on a single trial.
dhypergeom(M, a, b, n) phypergeom(M, a, b, n) qhypergeom(p, a, b, n) rhypergeom(M, a, b, n)	Probability distribution functions for the hypergeometric distribution in which M, a, b , and n are integers with $0 \leq M \leq a$, $0 \leq n - M \leq b$, and $0 \leq n \leq a + b$. $0 \leq p < 1$.
dlnorm(x, μ, σ) plnorm(x, μ, σ) qlnorm(p, μ, σ) rlnorm(m, μ, σ)	Probability distribution functions for the lognormal distribution in which μ is the logmean and $\sigma > 0$ is the logdeviation. $x \geq 0$. $0 \leq p < 1$.
dlogis(x, l, s) plogis(x, l, s) qlogis(p, l, s) rlogis(m, l, s)	Probability distribution functions for the logistic distribution in which l is the location parameter and $s > 0$ is the scale parameter. $0 < p < 1$.
dnbinom(k, n, p) pnbinom(k, n, p) qnbinom(p, n, r) rnbinom(m, n, p)	Probability distribution functions for the negative binomial distribution in which $0 < p \leq 1$ and n and k are integers, $n > 0$ and $k \geq 0$.
dnorm(x, μ, σ) pnorm(x, μ, σ) qnorm(p, μ, σ) rnorm(m, μ, σ)	Probability distribution functions for the normal distribution in which μ and σ are the mean and standard deviation. $\sigma > 0$.
dpois(k, λ) ppois(k, λ) qpois(p, λ) rpois(m, λ)	Probability distribution functions for the Poisson distribution in which $\lambda > 0$ and k is a nonnegative integer. $0 \leq p \leq 1$.
dt(x, d) pt(x, d) qt(p, d) rt(m, d)	Probability distribution functions for Student's t distribution in which $d > 0$ are the degrees of freedom. $0 < p < 1$.
dunif(x, a, b) punif(x, a, b) qunif(p, a, b) runif(m, a, b)	Probability distribution functions for the uniform distribution in which b and a are the endpoints of the interval with $a \leq x \leq b$. $0 \leq p \leq 1$.
dweibull(x, s) pweibull(x, s) qweibull(p, s) rweibull(m, s)	Probability distribution functions for the Weibull distribution in which $s > 0$ is the shape parameter and $x \geq 0$. $0 < p < 1$.

Tip Two additional functions that are useful for common probability calculations are *rnd*(x), which is equivalent to *runif*(1, 0, x), and *cnorm*(x), which is equivalent to *pnorm*(x , 0, 1).

Interpolation and Prediction Functions

Note Whenever you use arrays in any of the functions described in this section, be sure that every element in the array contains a data value. Mathcad assigns 0 to any elements you have not explicitly assigned.

- Pro** **bspline(vx, vy, u, n)** Returns a vector of coefficients of a B-spline of degree n , which is used in the *interp* function. The knot locations of the spline are specified in vector **u**. **vx** and **vy** must be real vectors of the same length. The values in **vx** must be in ascending order.
- cspline(vx, vy)** Returns a vector of coefficients of a cubic spline with cubic endpoints, which is used in the *interp* function. **vx** and **vy** must be real vectors of the same length. The values in **vx** must be in ascending order.
- interp(vs, vx, vy, x)** Returns the interpolated y value corresponding to the argument x . The vector **vs** is a vector of intermediate results obtained by evaluating *bspline*, *cspline*, *lspline*, or *pspline* or the regression routine *regress* or *loess* using the data vectors **vx** and **vy**. **vx** and **vy** must be real vectors of the same length. The vector **vx** must be in ascending order.
- linterp(vx, vy, x)** Uses the data vectors **vx** and **vy** to return a linearly interpolated y value corresponding to the argument x . **vx** and **vy** must be real vectors of the same length. The vector **vx** must be in ascending order.
- lspline(vx, vy)** Returns a vector of coefficients of a cubic spline with linear endpoints, which is used in the *interp* function. **vx** and **vy** must be real vectors of the same length. The values in **vx** must be in ascending order.
- predict(v, m, n)** Returns n predicted values based on m consecutive values from the data vector **v**. Elements in **v** should represent samples taken at equal intervals. m and n are integers
- pspline(vx, vy)** Returns a vector of coefficients of a cubic spline with parabolic endpoints, which is used in the *interp* function. **vx** and **vy** must be real vectors of the same length. The values in **vx** must be in ascending order.

Interpolation involves using existing data points to predict values between these data points. Mathcad allows you to connect the data points either with straight lines (linear interpolation) or with sections of a cubic polynomial (cubic spline interpolation). Unlike the regression functions discussed in the next section, these interpolation functions return a curve which must pass through the points you specify. If your data is noisy, you should consider using regression functions instead (see page 169).

Cubic spline interpolation passes a curve through a set of points in such a way that the first and second derivatives of the curve are continuous across each point. This curve is assembled by taking three adjacent points and constructing a cubic polynomial

passing through those points. These cubic polynomials are then strung together to form the completed curve. In the case of “traditional” cubic splines, the data points to be interpolated define the “knots” where the polynomials are joined, but B-splines (implemented in the function *bspline*) allow you to join the polynomials at arbitrary points.

Linear prediction involves using existing data values to predict values beyond the existing ones.

The coefficients returned by the spline interpolation functions *bspline*, *cspline*, *lspline*, and *pspline* and the regression functions *regress* and *loess* described in the next section are designed to be passed to Mathcad’s *interp* function. *interp* returns a single interpolated *y* value for a given *x* value, but as a practical matter you’ll probably be evaluating *interp* for many different points, as shown in Figure 10-7. It usually makes sense to store the coefficients returned by the spline or regression functions in a vector (such as **vs** in Figure 10-7) that can be passed to *interp* for evaluation, plotting, or further calculation.

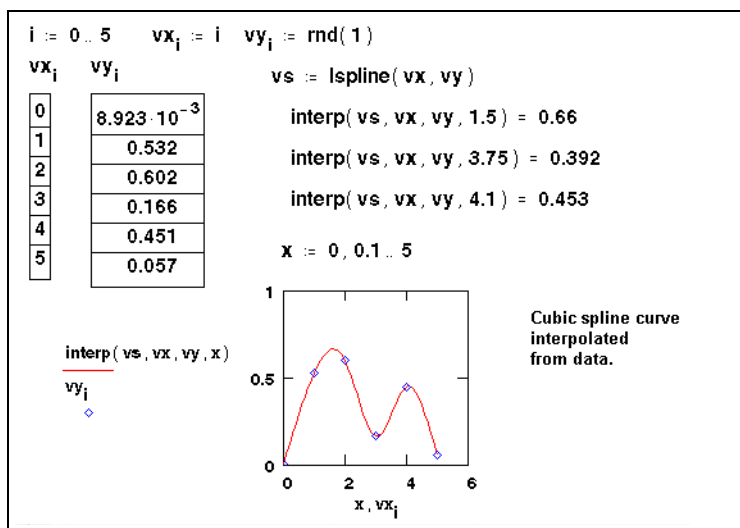


Figure 10-7: Spline curve for the points stored in *vx* and *vy*. Since the random number generator gives different numbers every time, you may not be able to recreate this example exactly as you see it.

Tip For best results with spline interpolation, do not use the *interp* function on values of *x* far from the fitted points. Splines are intended for interpolation, not extrapolation.

Note Mathcad handles *two-dimensional* cubic spline interpolation in much the same way as the one-dimensional case illustrated: in this case the spline function takes two matrix arguments, **Mxy** and **Mz**. The first is an $n \times 2$ matrix specifying the points along the diagonal of a rectangular grid, and the second is an $n \times n$ matrix of *z*-values representing the surface to be interpolated. Mathcad passes a *surface* through the grid of points. This surface corresponds to a cubic polynomial in *x* and *y* in which the first and second partial derivatives are continuous in the

Regression and Smoothing Functions

Mathcad includes a number of functions for performing *regression*. Typically, these functions generate a curve or surface of a specified type in some sense minimizes the error between itself and the data you supply. The functions differ primarily in the type of curve or surface they use to fit the data. Unlike interpolation functions, these functions do not require that the fitted curve or surface pass through points you supply, and they are therefore less sensitive to spurious data.

Smoothing involves taking a set of y (and possibly x) values and returning a new set of y values that is smoother than the original set. Unlike the regression and interpolation functions, smoothing results in a new set of y values, not a function that can be evaluated between the data points you specify. Thus, if you are interested in y values *between* the y values you specify, you should use a regression or interpolation function.

Linear regression

intercept(vx**, **vy**)** Returns a scalar: the y -intercept of the least-squares regression line for the data points in **vx** and **vy**.

slope(vx**, **vy**)** Returns a scalar: the slope of the least-squares regression line for the data points in **vx** and **vy**.

stderr(vx**, **vy**)** Returns the standard error associated with linear regression of the elements of **vy** on the elements of **vx**.

line(vx**, **vy**)** Returns the y -intercept and slope of the line that best approximates the data in **vx** and **vy**.

medfit(vx**, **vy**)** Returns the y -intercept and slope of the line that best approximates the data in **vx** and **vy** using median-median regression.

Polynomial regression

regress(vx**, **vy**, n)** Returns a vector of coefficients for the n th degree least-squares polynomial fit for the data points specified in vectors **vx** and **vy**. This vector becomes the first argument of the *interp* function.

Pro **loess(**vx**, **vy**, $span$)** Returns a vector specifying a set of second order polynomials that best fit particular neighborhoods of data points specified in vectors **vx** and **vy**. This vector becomes the first argument of the *interp* function. The argument $span$, $span > 0$, specifies how large a neighborhood *loess* considers in performing this local regression.

These functions are useful when you have a set of measured y values corresponding to x values (or possibly multiple x values) and you want to fit a polynomial through those y values.

Use *regress* when you want to use a *single* polynomial to fit all your data values. The *regress* function lets you fit a polynomial of any order. However as a practical matter, you rarely should go beyond $n = 4$.

The *loess* function, available in Mathcad Professional, performs a more localized regression. Instead of generating a single polynomial the way *regress* does, *loess* generates a different second order polynomial depending on where you are on the curve. It does this by examining the data in a small neighborhood of a point you're interested in.

As in the case of Mathcad's spline interpolation functions, the coefficients returned by *regress* and *loess* are designed to be passed to Mathcad's *interp* function. *interp* returns a single interpolated *y* value for a given *x* value, but as a practical matter you'll probably be evaluating *interp* for many different points.

Note Mathcad also allows *multivariate* polynomial regression with *regress* or *loess* to fit *y* values corresponding to two or more independent variables. In this case, the regression function's first two arguments are **Mx** and **vy**: the first is an $n \times m$ matrix specifying the *m* values of *n* predictor variables, and the second is a vector of response data corresponding to the factors in **Mx**. For an example see the "Data Analysis" QuickSheets in the Resource Center (choose **Resource Center** from the **Help** menu). You can add independent variables by simply adding columns to the **Mx** array and a corresponding number of rows to the vector you pass to the *interp* function.

Specialized regression

expfit(**vx**, **vy**, **vg**) Returns the parameter values for the exponential curve

$a \cdot e^{(b \cdot x)} + c$ that best approximates the data in **vx** and **vy**. Vector **vg** specifies guess values for the three unknown parameters *a*, *b*, and *c*. See Figure 10-8 for an example.

lgsfit(**vx**, **vy**, **vg**) Returns the parameter values for the logistic curve

$a / (1 + b \cdot e^{(-c \cdot x)})$ that best approximates the data in **vx** and **vy**. Vector **vg** specifies guess values for the three unknown parameters *a*, *b*, and *c*.

logfit(**vx**, **vy**, **vg**) Returns the parameter values for the logarithmic curve

$a \cdot \ln(x + b) + c$ that best approximates the data in **vx** and **vy**. Vector **vg** specifies guess values for the three unknown parameters *a*, *b*, and *c*.

pwrfit(**vx**, **vy**, **vg**) Returns the parameter values for the power curve

$a \cdot x^b + c$ that best approximates the data in **vx** and **vy**. Vector **vg** specifies guess values for the three unknown parameters *a*, *b*, and *c*.

sinfit(**vx**, **vy**, **vg**) Returns the parameter values for the sine curve

$a \cdot \sin(x + b) + c$ that best approximates the data in **vx** and **vy**. Vector **vg** specifies guess values for the four unknown parameters *a*, *b*, and *c*.

Use these functions when you have a set of measured *y* values corresponding to *x* values and you want to fit a special type of curve through those *y* values. Although you can use the *genfit* function described on page 171 to perform a curve fit on any function, the functions outlined above are simpler. Use them if they address the particular function curve to which you are fitting your data.

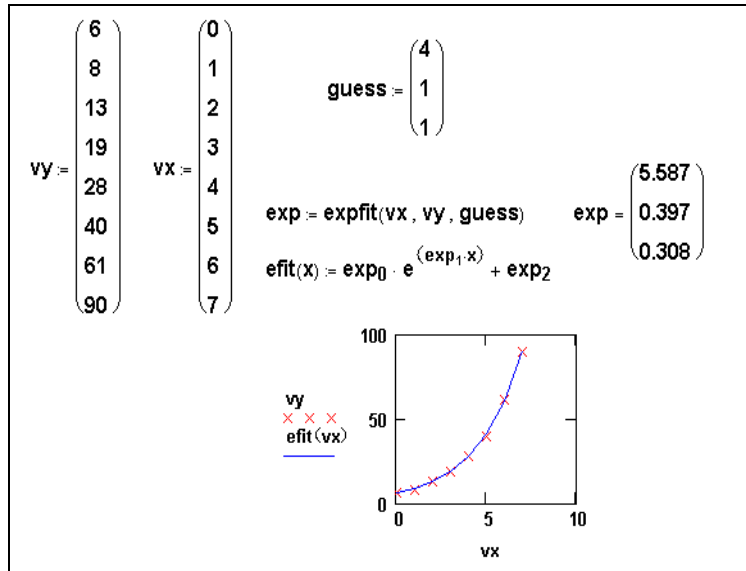


Figure 10-8: Using the specialized regression function `expfit`.

Generalized regression

linfit($\mathbf{vx}$, $\mathbf{vy}$, $\mathbf{F}$) Returns a vector containing the coefficients used to create a linear combination of the functions in $\mathbf{F}$ that best approximates the data points in vectors $\mathbf{vx}$ and $\mathbf{vy}$. $\mathbf{F}$ is a function which returns a vector consisting of the functions to be linearly combined. Elements of $\mathbf{vx}$ should be in ascending order.

genfit($\mathbf{vx}$, $\mathbf{vy}$, $\mathbf{vg}$, $\mathbf{F}$) Returns the parameters for the best fit by the (possibly nonlinear) function defined by f to the data points in the vectors $\mathbf{vx}$ and $\mathbf{vy}$. $\mathbf{F}$ is a function that returns an $n + 1$ element vector containing f and its partial derivatives with respect to each of its n parameters. $\mathbf{vg}$ is an n -element vector of guess values for the n parameters.

linfit is designed to model your data by a linear combination of arbitrary functions:

$$y = a_0 \cdot f_0(x) + a_1 \cdot f_1(x) + \dots + a_n \cdot f_n(x)$$

genfit is designed to model your data by some arbitrary (possibly nonlinear) function whose parameters must be chosen. For example, if your data is to be modeled by the sum

$$f(x) = 2 \cdot \sin(a_1 x) + 3 \cdot \tanh(a_2 x)$$

and you wish to solve for the unknown parameters a_1 and a_2 , you would use *genfit*. An example of using *genfit* is given in Figure 10-9.

Anything you can do with *linfit* you can also do, albeit less conveniently, with *genfit*. The difference between these two functions is the difference between solving a system of linear equations and solving a system of nonlinear equations. The latter generally

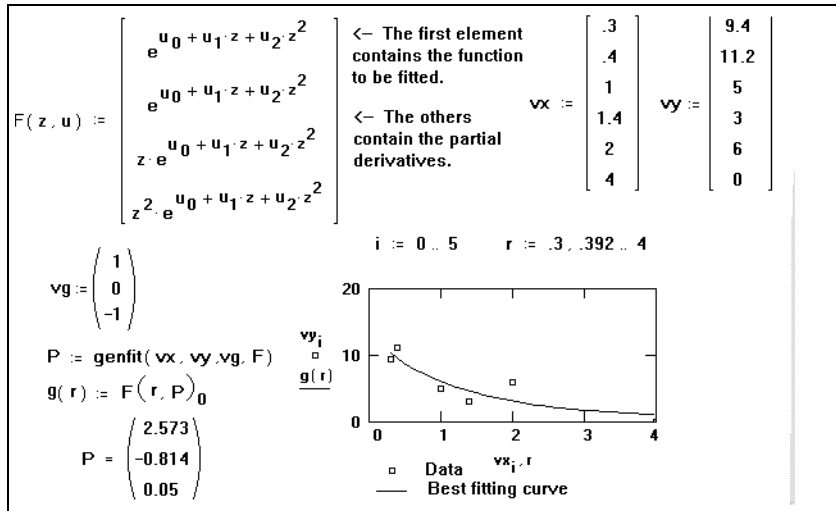


Figure 10-9: Using *genfit* for finding the parameters of a function so that it best fits the data.

must be solved by iteration, which explains why *genfit* needs a vector of guess values as an argument and *linfit* does not.

Smoothing functions

medsmooth(vy, n) Returns an m -element vector created by smoothing **vy** with running medians. **vy** is an m -element vector of real numbers. n is the width of the window over which smoothing occurs. n must be an odd number less than the number of elements in **vy**.

Pro ksmooth(vx,vy, b) Returns an n -element vector created by using a Gaussian kernel to return weighted averages of **vy**. **vy** and **vx** are n -element vectors of real numbers. The bandwidth b controls the smoothing window and should be set to a few times the spacing between your x data points. Elements in **vx** must be in ascending order.

Pro supsmooth(vx,vy) Returns an n -element vector created by the piecewise use of a symmetric k -nearest neighbor linear least-squares fitting procedure in which k is adaptively chosen. **vy** and **vx** are n -element vectors of real numbers. The elements of **vx** must be in increasing order.

medsmooth is the most robust of the three smoothing functions since it is least likely to be affected by spurious data points. This function uses a running median smoother, computes the residuals, smooths the residuals the same way, and adds these two smoothed vectors together. Note that *medsmooth* leaves the first and last $(n-1)/2$ points unchanged. In practice, the length of the smoothing window, n , should be small compared to the length of the data set.

Pro ksmooth, available in Mathcad Professional, uses a Gaussian kernel to compute local weighted averages of the input vector **vy**. This smoother is most useful when your data lies along a band of relatively constant width. If your data lies scattered along a band

whose width fluctuates considerably, you should use an adaptive smoother like *supsmooth*, also available in Mathcad Professional. *supsmooth* uses a symmetric *k* nearest neighbor linear least-squares fitting procedure to make a series of line segments through your data. Unlike *ksmooth* which uses a fixed bandwidth for all your data, *supsmooth* adaptively chooses different bandwidths for different portions of your data.

Finance Functions

These personal finance functions perform a variety of calculations for making credit and investment decisions. All finance functions take only real values. Payments you make, such as deposits in a savings account or payments toward a loan, must be entered as negative numbers. Cash you receive, such as dividend checks, must be entered as positive numbers. If you want to specify the timing of a payment, use the optional timing variable, *type*, which can be equal to 0 for the end of the period and 1 for the beginning. If omitted, *type* is 0.

Rate and period

<code>cnper(rate, pv, fv)</code>	Returns the number of compounding periods required for an investment to yield a future value, <i>fv</i> , given a present value, <i>pv</i> , and an interest rate period, <i>rate</i> . <i>rate</i> > -1. <i>pv</i> > 0. <i>fv</i> > 0.
<code>crate(nper, pv, fv)</code>	Returns the fixed interest rate required for an investment at present value, <i>pv</i> , to yield a future value, <i>fv</i> , over a given number of compounding periods, <i>nper</i> . <i>nper</i> is a positive integer. <i>pv</i> > 0. <i>fv</i> > 0.
<code>nper(rate, pmt, pv, [[fv], [type]])</code>	Returns the number of compounding periods for an investment or loan based on periodic, constant payments, <i>pmt</i> , using a fixed interest rate, <i>rate</i> , and a present value, <i>pv</i> . If omitted, <i>fv</i> = 0 and <i>type</i> = 0. If <i>pmt</i> > 0, <i>rate</i> and <i>pv</i> must be opposite signs.
<code>rate(nper, pmt, pv, [[fv], [type], [guess]])</code>	Returns the interest rate per period of an investment or loan over a number of compounding periods, <i>nper</i> , given a periodic, constant payment, <i>pmt</i> , and a present value, <i>pv</i> . <i>nper</i> is a positive integer. If omitted, <i>fv</i> = 0, <i>type</i> = 0, and <i>guess</i> = 0.1 (10%).

Tip If *rate* does not converge to within 1×10^{-7} after 20 iterations, *rate* returns an error. In such a case, try different values for *guess*. In most cases, *rate* converges if *guess* is between 0 and 1.

Cumulative interest and principal

<code>cumint(rate, nper, pv, start, end, [type])</code>	Returns the cumulative interest paid on a loan between a starting period, <i>start</i> , and an ending period, <i>end</i> , given a fixed interest rate, <i>rate</i> , the total number of compounding periods, <i>nper</i> , and the present value of the loan, <i>p</i> <i>v</i> . <i>rate</i> ≥ 0. <i>nper</i> , <i>start</i> , and <i>end</i> are positive integers. If omitted, <i>type</i> = 0.
<code>cumprn(rate, nper, pv, start, end, [type])</code>	Returns the cumulative principal paid on a loan between a starting period, <i>start</i> , and an ending period, <i>end</i> , given a fixed interest rate, <i>rate</i> , the total number of compounding periods, <i>nper</i> , and the present value of the loan, <i>p</i> <i>v</i> . <i>rate</i> ≥ 0. <i>nper</i> , <i>start</i> , and <i>end</i> are positive integers. If omitted, <i>type</i> = 0.

A home mortgage has the following terms:			
Interest rate:	9% (annually)	Term:	30yrs
Present value:	\$125,000		
Calculate the total (cumulative) interest paid in the 10th year of payments (payments 121 through 132):			
$rate := \frac{9\%}{12}$	$nper := 30 \cdot 12$	p <i>v</i> := 125000	
$start := 120$	$end := 132$		
$cumint(rate, nper, pv, start, end) = -10815.54$			

Figure 10-10: Using the *cumint* function.

Interest rate

<code>eff(rate, nper)</code>	Returns the effective annual interest rate (APR) given the nominal interest rate, <i>rate</i> , and the number of compounding periods per year, <i>nper</i> . <i>nper</i> is positive.
<code>nom(rate, nper)</code>	Returns the nominal interest rate given the effective annual interest rate (APR), <i>rate</i> , and the number of compounding periods per year, <i>nper</i> . <i>rate</i> > -1. <i>nper</i> is positive.

Future value

<code>f_v(rate, nper, pmt, [[p_v], [type]])</code>	Returns the future value of an investment or loan over a number of compounding periods, <i>nper</i> , given a periodic, constant payment, <i>p</i> <i>m</i> <i>t</i> , and a fixed interest rate, <i>rate</i> . <i>nper</i> is a positive integer. If omitted, present value <i>p</i> <i>v</i> = 0 and <i>type</i> = 0.
<code>f_vadj(<i>prin</i>, v)</code>	Returns the future value of an initial principal, <i>prin</i> , after applying a series of compound interest rates stored in v . v is a vector.

`fvc(rate, v)`

Returns the future value of a list of cash flows occurring at regular intervals, **v**, earning an interest rate, *rate*. **v** is a vector.

Note When using functions that require information about rates and periods, use the same unit of time for each. For example, if you make monthly payments on a four-year loan at an annual interest rate of 12%, use 1% as the interest rate per period (one month) and 48 months as the number of periods.

Payment

`pmt(rate, nper, pv, [[fv],
[type]])`

Returns the payment for an investment or loan based on periodic constant payments over a given number of compounding periods, *nper*, using a fixed interest rate, *rate*, and a present value, *pv*. *nper* is a positive integer. If omitted, future value *fv* = 0 and *type* = 0.

`ipmt(rate, per, nper, pv, [[fv],
[type]])`

Returns the interest payment of an investment or loan for a given period, *per*, based on periodic constant payments over a given number of compounding periods, *nper*, using a fixed interest rate, *rate*, and a present value, *pv*. *per* and *nper* are positive integers, $per \leq nper$. If omitted, future value *fv* = 0 and *type* = 0.

`ppmt(rate, per, nper, pv, [[fv],
[type]])`

Returns the payment on the principal, of a investment or loan, for a given period, *per*, based on periodic constant payments over a given number of compounding periods, *nper*, using a fixed interest rate, *rate*, and a present value, *pv*. *per* and *nper* are positive integers, $per \leq nper$. If omitted, future value *fv* = 0 and *type* = 0.

The monthly payment for a \$10,000 loan that accrues interest at an annual rate of 8% and that you must pay off in 10 months:

$rate := \frac{8\%}{12} \quad nper := 10 \quad pv := 10000 \quad per := 5$

$pmt(rate, nper, pv) = -1037.03$

The portion of the 5th payment that is interest:

$ipmt(rate, per, nper, pv) = -40.53$

The portion of the 5th payment that is principal:

$ppmt(rate, per, nper, pv) = -996.50$

Figure 10-11: Using the *pmt*, *ipmt*, and *ppmt* functions.

Internal rate of return

<code>irr(v, [<i>guess</i>])</code>	Returns the internal rate of return for a series of cash flows, v , occurring at regular intervals. v is a vector that must contain at least one positive value and one negative value. If omitted, <i>guess</i> = 0.1 (10%).
<code>mirr(v, <i>fin_rate</i>, <i>rein_rate</i>)</code>	Returns the modified internal rate of return for a series of cash flows occurring at regular intervals, v , given a finance rate payable on the cash flows you borrow, <i>fin_rate</i> , and a reinvestment rate earned on the cash flows as you reinvest them, <i>rein_rate</i> . v is a vector that must contain at least one positive value and one negative value.

Note If *irr* does not converge to within 1×10^{-5} of a percentage after 20 iterations, *irr* returns an error. In such a case, try different values for *guess*. In most cases, *irr* converges if *guess* is between 0 and 1.

Present value

<code>pv(<i>rate</i>, <i>nper</i>, <i>pmt</i>, [[<i>fv</i>] [<i>type</i>]])</code>	Returns the present value of an investment or loan based on periodic constant payments over a given number of compounding periods, <i>nper</i> , using a fixed interest rate, <i>rate</i> , and a payment, <i>pmt</i> . <i>nper</i> is a positive integer. If omitted, <i>fv</i> = 0 and <i>type</i> = 0.
<code>npv(<i>rate</i>, v)</code>	Returns the net present value of an investment given a discount rate, <i>rate</i> , and a series of cash flows occurring at regular intervals, v . v is a vector.

Note *irr* and *npv* are related functions. The internal rate of return (*irr*) is the rate for which the net present value (*npv*) is zero.

Differential Equation Functions

Pro This section describes how you can use Mathcad Professional to solve differential equations.

In a differential equation, you solve for an unknown function rather than just a variable. For ordinary differential equations, the unknown function is a function of one variable. Partial differential equations are differential equations in which the unknown is a function of two or more variables.

The easiest way to solve a single differential equation of any order in Mathcad Professional is to use a solve block and the function *Odesolve*. To solve systems of equations or to have more control over the solving process, you can use the general-purpose differential equation solver *rkfixed*. Alternatively you can choose from additional, more specialized functions for solving differential equations.

Solving a Differential Equation Using a Solve Block

Pro `Odesolve(x, b, [step])` Returns a function of x which is the solution to a single ordinary differential equation. b is the terminal point of the integration interval. $step$ (optional) is the number of steps.

To solve a single differential equation of any order, use a *solve block* and the function *Odesolve*. A solve block for solving a differential equation is similar to a solve block for solving a system of equations as described on page 159. There are three steps to creating a differential equation solve block:

1. Type the word *Given*. You can type *Given* or *given* in any style, but be sure not to type it in a text region.
2. Type the differential equation and constraints in any order below the word *Given*.

Use the bold equal sign (click  on the Boolean toolbar or press **[Ctrl]=**) for an equality. The independent variable x must be explicitly indicated throughout. A typical initial value constraint might be $y(a)=c$ or $y'(a)=d$; Mathcad does not allow more complicated constraints like $y(a)+y'(a)=e$. The differential equation can be written using the derivative operators d/dx , d^2/dx^2 , d^3/dx^3 , ... (press **?** or **[Ctrl]?** to insert the derivative or n th derivative operators), or using prime notation $y'(x)$, $y''(x)$, $y'''(x)$, ... (Press **[Ctrl]** **[F7]** for the prime symbol.)

3. Finally, type the *Odesolve* function. The terminal point b must be larger than the initial point a .

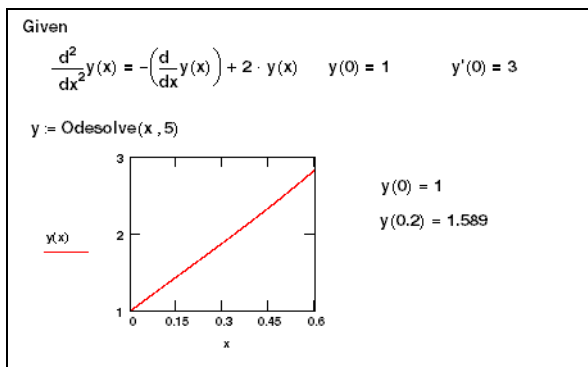


Figure 10-12: Solving a single differential equation.

Tip Prime notation is only allowed inside a solve block. If you use it outside of a solve block, you see an error.

The output of *Odesolve* is a function of x , interpolated from a table of values computed using the fixed step method employed by the function *rkfixed*, described below. If you prefer to use an adaptive step method employed by the function *Rkadapt*:

1. Click on *Odesolve* with the right mouse button.
2. Choose **Adaptive** from the pop-up menu.

Note Mathcad is very specific about the types of expressions that can appear between *Given* and *Odesolve*. The lower derivative terms can appear nonlinearly in the differential equation (e.g., they can be multiplied together or raised to powers), but the highest derivative term must appear linearly. Inequality constraints are not allowed. There must be n independent equality constraints for an n th order differential equation. For an initial value problem, the values for $y(x)$ and its first $n-1$ derivatives at a single initial point a are required. For a boundary value problem, the n equality constraints should prescribe values for $y(x)$ and certain derivatives at exactly two points a and b .

General Purpose Differential Equation Solver: *rkfixed*

To solve systems of equations or to specify a starting point for the interval and the number of points at which to approximate a solution, you can use the differential equation solver *rkfixed*. Alternatively you can choose from additional, more specialized functions for solving differential equations, described in the following section “Specialized Differential Equation Solvers.”

First order differential equations

A first order differential equation is one in which the highest order derivative of the unknown function is the first derivative. To solve a first order differential equation in Mathcad, you can use *Odesolve* as described in “Solving a Differential Equation Using a Solve Block” or you can use *rkfixed*. *rkfixed* uses the fourth order Runge-Kutta method to solve a first order differential equation and return a two-column matrix in which:

- The left-hand column contains the points at which the solution to the differential equation is evaluated.
- The right-hand column contains the corresponding values of the solution.

rkfixed(**y**, $x1$, $x2$, $npoints$, **D**)

y = A vector of n initial values where n is the order of the differential equation or the size of the system of equations you’re solving. For a first order differential equation, the vector degenerates to one point, $y(0) = y(x1)$.

$x1$, $x2$ = The endpoints of the interval on which the solution to the differential equations will be evaluated. The initial values in **y** are the values at $x1$.

$npoints$ = The number of points beyond the initial point at which the solution is to be approximated. This controls the number of rows ($1 + npoints$) in the matrix returned by *rkfixed*.

D(x , **y**) = An n -element vector-valued function containing the first n derivatives of the unknown functions.

Figure 10-13 shows how to solve the differential equation $\frac{dy}{dx} + 3 \cdot y = 0$ subject to the initial condition $y(0) = 4$.

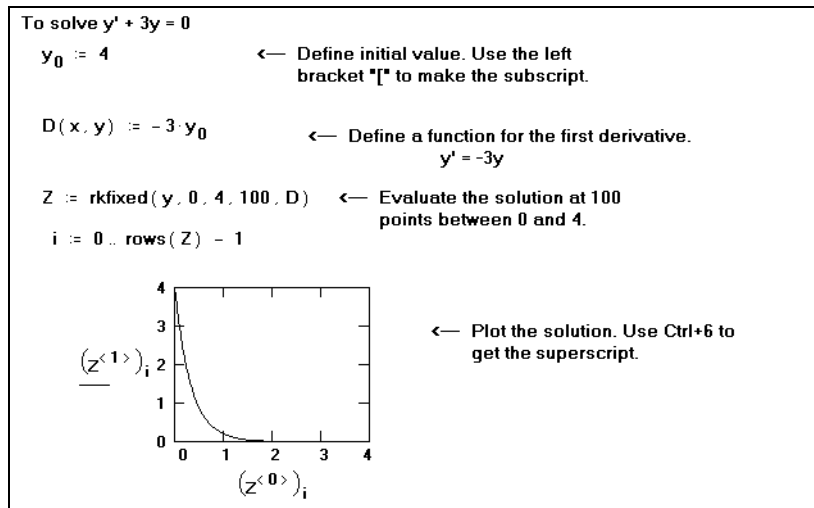


Figure 10-13: Solving a first order differential equation.

Note The most difficult part of solving a differential equation using *rkfixed*, particularly with nonlinear differential equations, is solving for the first derivative so you can define the function $D(x, y)$. In such cases, you can sometimes solve for $y'(x)$ symbolically and paste it into the definition for $D(x, y)$. To do so, use the symbolic solving techniques discussed in the section “Examples of Symbolic Calculation” in Chapter 14.

Second and higher order differential equations

You can use *Odesolve* and a solve block, as described on page 177, or you can use *rkfixed* to solve higher order differential equations. Using *rkfixed* to solve a second order differential equation is similar to solving a first order differential equation. The key differences are:

- The vector of initial values y now has two elements: the value of the function and its first derivative at the starting value, x_1 .
- The function $D(t, y)$ is now a vector with two elements:

$$D(t, y) = \begin{bmatrix} y'(t) \\ y''(t) \end{bmatrix}$$

- The solution matrix contains three columns: the left-hand one for the t values; the middle one for $y(t)$; and the right-hand one for $y'(t)$.

See Figure 10-14 for an example using *rkfixed*. See Figure 10-12 for an example using *Odesolve*.

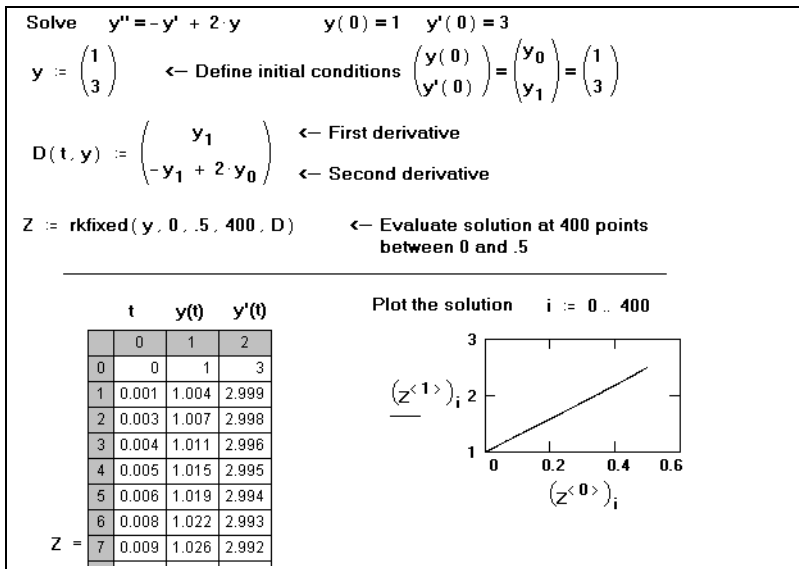


Figure 10-14: Solving a second order differential equation.

Tip The procedure for solving higher order differential equations is an extension of that used for second order differential equations. The main differences are that the vector of initial values y now has n elements for specifying initial conditions of $y, y', y'', \dots, y^{(n-1)}$, the function D is now a vector with n elements corresponding to the first n derivatives of the unknown functions, and the solution matrix contains n columns: the left-hand one for the t values and the remaining columns for values of $y(t), y'(t), y''(t), \dots, y^{(n-1)}(t)$.

Systems of differential equations

Solving a system of differential equations is similar to solving a higher order differential equation using *rkfixed*. To solve a system of first order differential equations:

1. Define a vector containing the initial values of each unknown function.
2. Define a vector-valued function containing the first derivatives of each of the unknown functions.
3. Decide at which points you want to evaluate the solutions.
4. Pass all this information into *rkfixed*.

rkfixed returns a matrix whose first column contains the points at which the solutions are evaluated and whose remaining columns contain the solution functions evaluated at the corresponding point. Figure 10-15 shows an example solving the equations:

$$x'_0(t) = \mu \cdot x_0(t) - x_1(t) - (x_0(t)^2 + x_1(t)^2) \cdot x_0(t)$$

$$x'_1(t) = \mu \cdot x_1(t) + x_0(t) - (x_0(t)^2 + x_1(t)^2) \cdot x_1(t)$$

with initial conditions $x_0(0) = 0$ and $x_1(0) = 1$.

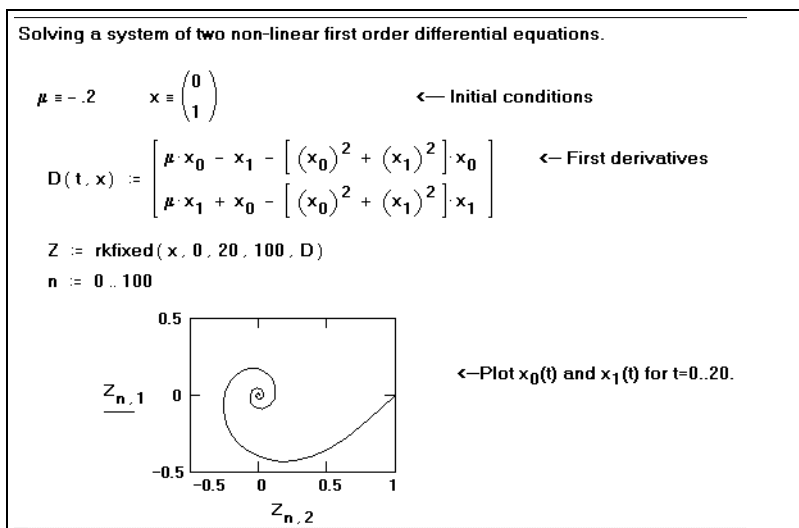


Figure 10-15: A system of first order linear equations.

Solving a system of n th order differential equations is similar to solving a system of first order differential equations. The main differences are:

- The vector of initial conditions must contain initial values for the $n - 1$ derivatives of each unknown function in addition to initial values for the functions themselves.
- The vector-valued function must contain expressions for the $n - 1$ derivatives of each unknown function in addition to the n th derivative.

rkfixed returns a matrix in which the first column contains the values at which the solutions and their derivatives are to be evaluated, and the remaining columns contain the solutions and their derivatives evaluated at the corresponding point in the first column. The order in which the solution and its derivatives appear matches the order in which you put them into the vector of initial conditions.

Specialized Differential Equation Solvers

Mathcad Professional includes several, more specialized functions for solving differential equations, and there are cases in which you may want to use these rather than the general-purpose *rkfixed*. These cases fall into the three categories given below. Each of these functions solves differential equations numerically: you always get back a matrix containing the values of the function evaluated over a set of points.

Tip When solving a differential equation it is a good idea to try more than one differential equation solver because one method might suit your differential equation better than another method.

Smooth systems

When you know the solution is smooth, use *Bulstoer*, which uses the Bulirsch-Stoer method rather than the Runge-Kutta method used by *rkfixed*.

Pro *Bulstoer*(**y**, *x1*, *x2*, *npoints*, **D**)

The argument list and the matrix returned by *Bulstoer* are identical to that for *rkfixed*.

Slowly varying solutions

Given a fixed number of points, you can approximate a function more accurately if you evaluate it frequently wherever it's changing fast and infrequently wherever it's changing more slowly.

Pro *Rkadapt*(**y**, *x1*, *x2*, *npoints*, **D**)

The argument list and the matrix returned by *Rkadapt* are identical in form to that for *rkfixed*.

If you know that the solution has this property, you may be better off using *Rkadapt*. Unlike *rkfixed*, *Rkadapt* examines how fast the solution is changing and adapts its step size accordingly.

Note Although *Rkadapt* uses nonuniform step sizes internally when it solves the differential equation, it nevertheless returns the solution at equally spaced points.

Stiff systems

A system of differential equations expressed in the form $\mathbf{y}' = \mathbf{A} \cdot \mathbf{x}$ is a *stiff system* if the matrix **A** is nearly singular. Under these conditions, the solution returned by *rkfixed* may oscillate or be unstable. When solving a stiff system, you should use one of the two differential equation solvers specifically designed for stiff systems, *Stiffb* and *Stiffc*, which use the Bulirsch-Stoer method and the Rosenbrock method, respectively. They take the same arguments as *rkfixed* as well as one additional argument.

Pro *Stiffb*(**y**, *x1*, *x2*, *npoints*, **D**, **J**)
Stiffc(**y**, *x1*, *x2*, *npoints*, **D**, **J**)

J(**x**, **y**) = A function you define that returns the $n \times (n + 1)$ matrix whose first column contains the derivatives $\partial \mathbf{D} / \partial x$ and whose remaining rows and columns form the Jacobian matrix $(\partial \mathbf{D} / \partial y_k)$ for the system of differential equations. For example, if:

$$\mathbf{D}(\mathbf{x}, \mathbf{y}) = \begin{bmatrix} x \cdot y_1 \\ -2 \cdot y_1 \cdot y_0 \end{bmatrix} \quad \text{then} \quad \mathbf{J}(\mathbf{x}, \mathbf{y}) = \begin{bmatrix} y_1 & 0 & x \\ 0 & -2 \cdot y_1 & -2 \cdot y_0 \end{bmatrix}$$

See *rkfixed* for a description of other parameters.

Evaluating Only the Final Value

If you care about only the value of the solution at the endpoint, $y(x_2)$, rather than over a number of uniformly spaced x values in the integration interval bounded by x_1 and x_2 , use the functions listed below. Each function corresponds to the capitalized versions already discussed. The properties of each of these functions are identical to those of the corresponding function in the previous sections, except for the arguments given below:

Pro `bulstoer(y, x1, x2, acc, D, kmax, save)`
 `rkadapt(y, x1, x2, acc, D, kmax, save)`
 `stiffb(y, x1, x2, acc, D, J, kmax, save)`
 `stiffr(y, x1, x2, acc, D, J, kmax, save)`

acc = Controls the accuracy of the solution. A small value of *acc* forces the algorithm to take smaller steps along the trajectory, thereby increasing the accuracy of the solution. Values of *acc* around 0.001 generally yield accurate solutions.

kmax = The maximum number of intermediate points at which the solution will be approximated. The value of *kmax* places an upper bound on the number of rows of the matrix returned by these functions.

save = The smallest allowable spacing between the values at which the solutions are to be approximated. This places a lower bound on the difference between any two numbers in the first column of the matrix returned by the function.

Boundary Value Problems

The specialized differential equation solvers discussed above are useful for solving *initial value problems*. In some cases, however, you may know the value taken by the solution at the *endpoints* of the interval of integration, which is a *boundary value problem*.

To solve boundary value problems in Mathcad, you can use *Odesolve*, described in “Solving a Differential Equation Using a Solve Block”, or *sbval* or *bvalfit* as described here.

Two-point boundary value problems

Two-point boundary value problems are one-dimensional systems of differential equations in which the solution is a function of a single variable and the value of the solution is known at two points. You can use *sbval* in the following case:

- You have an n th order differential equation.
- You know some, but not all, of the values of the solution and its first $n - 1$ derivatives at the beginning of the interval of integration, x_1 , and at the end of the interval of integration, x_2 .
- Between what you know about the solution at x_1 and at x_2 , you have n known values.

sbval returns a vector containing those initial values left unspecified at the first endpoint of the interval. Once you know the missing initial values at $x1$, you have an initial value problem that can be solved using any of the functions discussed earlier in this section.

Pro

sbval(**v**, $x1$, $x2$, **D**, **load**, **score**)

v = Vector of guesses for initial values left unspecified at $x1$.

$x1$, $x2$ = The endpoints of the interval on which the solution to the differential equations will be evaluated.

D(x , **y**) = An n -element vector-valued function containing the first derivatives of the unknown functions.

load($x1$, **v**) = A vector-valued function whose n elements correspond to the values of the n unknown functions at $x1$. Some of these values will be constants specified by your initial conditions. Others will be unknown at the outset but will be found by *sbval*. If a value is unknown, you should use the corresponding guess value from **v**.

score($x2$, **y**) = A vector-valued function having the same number of elements as **v**. Each element is the difference between an initial condition at $x2$, as originally specified, and the corresponding estimate from the solution. The *score* vector measures how closely the proposed solution matches the initial conditions at $x2$. A value of 0 for any element indicates a perfect match between the corresponding initial condition and that returned by *sbval*.

Note As shown in Figure 10-16, *sbval* does not actually return a solution to a differential equation. It merely computes the initial values the solution must have in order for the solution to match the final values you specify. You must then take the initial values returned by *sbval* and solve the resulting initial value problem using a function such as *rkfixed*.

It's also possible that you don't have all the information you need to use *sbval* but you do know something about the solution and its first $n - 1$ derivatives at some intermediate value, xf . *bvalfit* solves a two-point boundary value problem of this type by shooting from the endpoints and matching the trajectories of the solution and its derivatives at the intermediate point. This method becomes especially useful when the derivative has a discontinuity somewhere in the integration interval.

Convert to initial value problem: $y^{(5)} + y = 0$ with $y(0) = 0$ $y'(0) = 7$
 $y(1) = 1$ $y'(1) = 10$ $y''(1) = 5$

$$v := \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \leftarrow \begin{matrix} \text{guess value for } y''(0) \\ y'''(0) \\ y^{iv}(0) \end{matrix}$$

$$D(x, y) := \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ -y_0 \end{bmatrix} \leftarrow \begin{matrix} \text{D vector for the differential} \\ \text{equation: } y^{(5)} + y = 0 \end{matrix}$$

$$load(x1, v) := \begin{bmatrix} 0 \\ 7 \\ v_0 \\ v_1 \\ v_2 \end{bmatrix} \leftarrow \begin{matrix} \text{known } y(0) \\ \text{known } y'(0) \\ \text{Unknown initial} \\ \text{conditions. To be} \\ \text{solved for by sbval.} \end{matrix}$$

$$score(x2, y) := \begin{pmatrix} y_0 - 1 \\ y_1 - 10 \\ y_2 - 5 \end{pmatrix} \leftarrow \begin{matrix} \text{Difference between} \\ \text{computed and given} \\ \text{values of } y \end{matrix}$$

$$S := sbval(v, 0, 1, D, load, score)$$

$$S = \begin{pmatrix} -85.014 \\ 348.107 \\ -516.257 \end{pmatrix} \leftarrow \begin{matrix} y''(0) \\ y'''(0) \\ y^{iv}(0) \end{matrix} \quad \begin{matrix} \text{Missing initial conditions,} \\ \text{to be used with rkfixed.} \end{matrix}$$

Figure 10-16: Using sbval to obtain initial values corresponding to given final values of a solution to a differential equation.

Pro $bvalfit(v1, v2, x1, x2, xf, D, load1, load2, score)$

v1, v2 = Vector **v1** contains guesses for initial values left unspecified at $x1$.
 Vector **v2** contains guesses for initial values left unspecified at $x2$.

$x1, x2$ = The endpoints of the interval on which the solution to the differential equations will be evaluated.

xf = A point between $x1$ and $x2$ at which the trajectories of the solutions beginning at $x1$ and those beginning at $x2$ are constrained to be equal.

D(x, y) = An n -element vector-valued function containing the first derivatives of the unknown functions.

load1(x1, v1) = A vector-valued function whose n elements correspond to the values of the n unknown functions at $x1$. Some of these values will be constants specified by your initial conditions. If a value is unknown, you should use the corresponding guess value from **v1**.

load2(x2, v2) = Analogous to **load1** but for values taken by the n unknown functions at $x2$.

score(xf, y) = An n element vector valued function used to specify how you want the solutions to match at xf . You'll usually want to define $score(xf, y) := y$ to make the solutions to all unknown functions match up at xf .

Partial differential equations

A second type of boundary value problem arises when you are solving a partial differential equation. Rather than being fixed at two points, the solution is fixed at a whole continuum of points representing some boundary.

Two partial differential equations that arise often in the analysis of physical systems are Poisson's equation:

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = \rho(x, y)$$

and its homogeneous form, Laplace's equation.

Tip To type a partial differential equation symbol such as $\frac{\partial}{\partial x}$, insert the derivative operator $\frac{d}{dx}$ by typing **?**, click on the derivative operator with the right mouse button, and choose **View Derivative As** $\Rightarrow$ **Partial Derivative** from the pop-up menu.

Mathcad has two functions for solving these equations over a square boundary. You should use *relax* if you know the value taken by the unknown function $u(x, y)$ on all four sides of a square region.

If $u(x, y)$ is zero on all four sides of the square, you can use *multigrid*, which often solves the problem faster than *relax*. Note that if the boundary condition is the same on all four sides, you can simply transform the equation to an equivalent one in which the value is zero on all four sides.

relax returns a square matrix in which:

- An element's location in the matrix corresponds to its location within the square region, and
- Its value approximates the value of the solution at that point.

This function uses the relaxation method to converge to the solution. Poisson's equation on a square domain is represented by:

$$a_{j,k} u_{j+1,k} + b_{j,k} u_{j-1,k} + c_{j,k} u_{j,k+1} + d_{j,k} u_{j,k-1} + e_{j,k} u_{j,k} = f_{j,k}$$

Pro `relax(a, b, c, d, e, f, u, rjac)`

a . . . e = Square matrices all of the same size containing coefficients of the above equation.

f = Square matrix containing the source term at each point in the region in which the solution is sought.

u = Square matrix containing boundary values along the edges of the region and initial guesses for the solution inside the region.

r_{jac} = Spectral radius of the Jacobi iteration. This number between 0 and 1 controls the convergence of the relaxation algorithm. Its optimal value depends on the details of your problem.

Pro `multigrid(M, ncycle)`

$\mathbf{M} = (1 + 2^n)$ row square matrix whose elements correspond to the source term at the corresponding point in the square domain.

$ncycle$ = The number of cycles at each level of the *multigrid* iteration. A value of 2 generally gives a good approximation of the solution.

Miscellaneous Functions

Expression Type

Pro `IsArray(x)` Returns 1 if x is a matrix or vector, 0 otherwise.

Pro `IsScalar(x)` Returns 1 if x is a real or complex number, 0 otherwise.

Pro `IsString(x)` Returns 1 if x is a string, 0 otherwise.

Pro `UnitsOf(x)` Returns the units of x , 1 otherwise.

String Functions

Pro `concat(S1, S2, S3, ...)` Returns a string formed by appending string $S2$ to the end of string $S1$, $S3$ to the end of $S2$, and so on.

Pro `error(S)` Returns the string S as an error tip. When Mathcad evaluates the *error* function, the expression is highlighted in red and further numerical evaluation is suspended. When you click on the expression, the string appears in an error tip.

Pro `num2str(z)` Returns a string formed by converting the real or complex number z into a decimal-valued string.

Pro `search(S, S1, m)` Returns the starting position of the substring $S1$ in string S beginning from position m , or -1 if no substring is found. m must be a nonnegative integer.

Pro `str2num(S)` Returns a constant formed by converting the characters in string S to a number. S must contain only characters which constitute an integer, a floating-point or complex number, or an e-format number such as $4.51e-3$ (for $4.51 \cdot 10^{-3}$). Spaces are ignored.

Pro `str2vec(S)` Returns a vector of ASCII codes corresponding to the characters in string S .

Pro `strlen(S)` Returns the number of characters in string S .

Pro	<code>substr(S, m, n)</code>	Returns a substring of <i>S</i> beginning with the character in the <i>m</i> th position and having at most <i>n</i> characters. <i>m</i> and <i>n</i> must be nonnegative integers.
Pro	<code>vec2str(v)</code>	Returns a string formed by converting a vector v of ASCII codes to characters. The elements of v must be integers between 0 and 255.

The strings used and returned by most of these functions are typed in a math placeholder by pressing the double-quote key (") and entering any combination of letters, numbers, or other ASCII characters. Mathcad automatically places double quotes around the string expression and displays quotes around a string returned as a result.

Note When evaluating the functions *search* and *substr*, Mathcad assumes that the first character in a string is at position 0.

File Access Functions

The file argument you supply to a Mathcad file access function is a *string*—or a variable to which a string is assigned—that corresponds either to:

- The name of a data or image file in the folder of the Mathcad worksheet you’re currently working on.
- The name of a colormap file (see page 190) in the CMAP subfolder of your Mathcad installation folder.
- A full or relative path to a data, image, or colormap file located elsewhere on a local or network file system.

Reading and writing ASCII data files

`READPRN(file)` Reads a structured data file. Returns a matrix. Each line in the data file becomes a row in the matrix. The number of elements in each row must be the same. Usually used as follows:
A := `READPRN(file)`

`WRITEPRN(file)` Writes a matrix into a data file. Each row becomes a line in the file. Must be used in a definition of the form `WRITEPRN(file) := A`

`APPENDPRN(file)` Appends a matrix to an existing file. Each row in the matrix becomes a new line in the data file. Must be used in a definition of the form `APPENDPRN(file) := A`. Existing data must have as many columns as **A**.

Files in plain ASCII format consist only of numbers separated by commas, spaces, or carriage returns. The numbers in the data files can be integers like **3** or **-1**, floating-point numbers like **2.54**, or E-format numbers like **4.51E-4** (for $4.51 \cdot 10^{-4}$).

Tip These ASCII data file access functions are provided mainly for compatibility with worksheets created in earlier versions of Mathcad. The Input Table and File Read/Write component provide more general methods of importing and exporting data in a variety of formats. See Chapter 11, “Vectors, Matrices, and Data Arrays.”

Reading and writing image files

	<code>READBMP(<i>file</i>)</code>	Creates a matrix containing a grayscale representation of the image in BMP format <i>file</i> . Each element in the matrix corresponds to a pixel. The value of a matrix element determines the shade of gray associated with the corresponding pixel. Each element is an integer between 0 (black) and 255 (white).
	<code>READRGB(<i>file</i>)</code>	Creates a matrix in which the color information in BMP format <i>file</i> is represented by the appropriate values of red, green, and blue. This matrix consists of three submatrices, each with the same number of columns and rows. Three matrix elements, rather than one, correspond to each pixel. Each element is an integer between 0 and 255. The three corresponding elements, when taken together, establish the color of the pixel.
	<code>WRITEBMP(<i>file</i>)</code>	Creates a grayscale BMP file from the matrix. Must be used in a definition of the form <code>WRITEBMP(<i>file</i>) := A</code> .
	<code>WRITERGB(<i>file</i>)</code>	Creates a color BMP file from a matrix in which the image is stored in RGB format. Must be used in a definition of the form <code>WRITERGB(<i>file</i>) := A</code> .
Pro	<code>READ_IMAGE(<i>file</i>)</code>	Creates a matrix containing a grayscale representation of the image in BMP, GIF, JPG, PCX, or TGA format <i>file</i> .
Pro	<code>READ_HLS(<i>file</i>)</code> <code>READ_HSV(<i>file</i>)</code>	Creates a matrix in which the color information in BMP, GIF, JPG, PCX, or TGA format <i>file</i> is represented by the appropriate values of hue, lightness, and saturation (HLS) or hue, saturation, and value (HSV).
Pro	<code>READ_RED(<i>file</i>)</code> <code>READ_GREEN(<i>file</i>)</code> <code>READ_BLUE(<i>file</i>)</code>	Extracts only the red, green, or blue component from a color image in BMP, GIF, JPG, PCX, or TGA format <i>file</i> . The result has one-third the number of columns that the matrix returned by <code>READRGB</code> would have had.
Pro	<code>READ_HLS_HUE(<i>file</i>)</code> <code>READ_HLS_LIGHT(<i>file</i>)</code> <code>READ_HLS_SAT(<i>file</i>)</code>	Extracts only the hue, lightness, or saturation component from a color image in BMP, GIF, JPG, PCX, or TGA format <i>file</i> . The result has one-third the number of columns that the matrix returned by <code>READ_HLS</code> would have had.
Pro	<code>READ_HSV_HUE(<i>file</i>)</code> <code>READ_HSV_SAT(<i>file</i>)</code> <code>READ_HSV_VALUE(<i>file</i>)</code>	Extracts only the hue, saturation, or value component from a color image in BMP, GIF, JPG, PCX, or TGA format <i>file</i> . The result has one-third the number of columns that the matrix returned by <code>READ_HSV</code> would have had.
Pro	<code>WRITE_HLS(<i>file</i>)</code>	Creates a color BMP file out of a matrix in which the image is stored in HLS format. Must be used in a definition of the form <code>WRITE_HLS(<i>file</i>) := A</code> .

Pro `WRITE_HSV(file)` Creates a color BMP file out of a matrix in which the image is stored in HSV format. Must be used in a definition of the form `WRITE_HSV(file) := A`.

Functions Related to 3D Graphs

Loading and saving colormaps

Pro `LoadColormap(file)` Returns an array containing the values in the colormap *file*.

Pro `SaveColormap(file, M)` Creates a colormap *file* containing the values in the three-column array **M**. Returns the number of rows written to *file*.

A colormap is a .CMP file containing three columns of values that represent levels of red, green, and blue. You can apply a colormap to a 3D plot as described in “Fill Color” on page 236. Each value in a colormap should be an integer between 0 and 255, inclusive. By default Mathcad saves and loads colormaps from the CMAPS subfolder of the location where you installed Mathcad.

Graphing 3D polyhedra

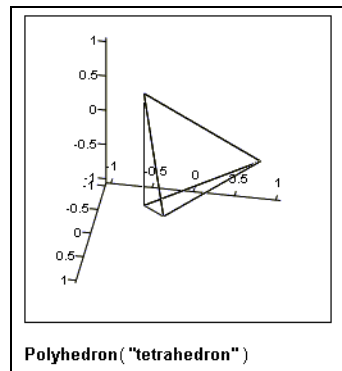
Pro `PolyLookup(n)` Returns a vector containing the name, the dual name, and the Wythoff symbol for the uniform polyhedron whose number code is *n*. *n* is a positive integer less than 81, a name typed as a string, or a Wythoff symbol typed as a string.

Pro `Polyhedron(S)` Generates the uniform polyhedron whose name, number code, or Wythoff symbol is string *S*.

The uniform polyhedra are regular polyhedra whose vertices are congruent. Each has a name, a number, a dual (the name of another polyhedron), and a Wythoff symbol associated with it. To look up the name, Wythoff symbol, and dual name of a polyhedron, use *PolyLookup*.

To graph a uniform polyhedron:

1. Click in a blank spot of your worksheet. Choose **Graph**⇒**Surface Plot** from the **Insert** menu.
2. In the placeholder, enter the *Polyhedron* function with an appropriate string argument.
3. Click outside the plot or press **[Enter]**.



Chapter 11

Vectors, Matrices, and Data Arrays

- τ Creating Arrays
- τ Accessing Array Elements
- τ Displaying Arrays
- τ Working with Arrays
- τ Nested Arrays

Creating Arrays

As introduced in “Inserting Math” on page 33, one technique of creating an array is to use the **Matrix** command on the **Insert** menu to create an array of empty placeholders and then to enter expressions directly into the placeholders. This technique can only be used for small arrays, but it can be used to create arrays of any kind of Mathcad expression, not just numbers. This section describes this technique and other approaches for creating arrays of arbitrary size:

- Using range variables to fill in the elements. This technique is useful when you have some explicit formula for the array elements in terms of their indices.
- Using the File Read/Write component to import data from external files in a variety of formats.
- Entering numbers manually in a spreadsheet-like input table.

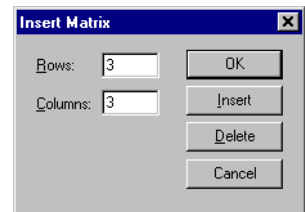
Unlike the Insert Matrix command, however, these procedures can be used *only* for creating arrays of numbers, as opposed to arbitrary math expressions.

Note The effective array size limit depends on the memory available on your system—usually at least 1 million elements. In no system is it higher than 8 million elements.

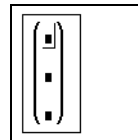
Insert Matrix Command

To insert a vector or matrix in Mathcad, follow these steps:

1. Click in either a blank space or on a math placeholder.
2. Choose **Matrix** from the **Insert** menu, or click  on the Matrix toolbar. A dialog box appears, as shown at right.



3. Enter the appropriate number of elements in the text boxes for “Rows” and “Columns.” For example, to create a three-element vector, enter **3** and **1**.



4. An array with blank placeholders appears in your worksheet.

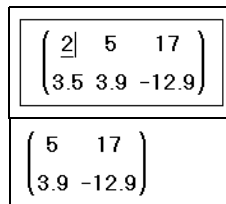
Next, fill in the array elements. You can enter any Mathcad expression into the placeholders of an array created in this way. Simply click in a placeholder and type a number or Mathcad expression. Use the **[Tab]** key to move from placeholder to placeholder.

Note Arrays created using the **Matrix** command on the **Insert** menu are limited to 100 elements.

Changing the size of a vector or matrix

You can change the size of a matrix by inserting and deleting rows and columns:

1. Click on one of the matrix elements to place it between the editing lines. Mathcad begins inserting or deleting with this element.
2. Choose **Matrix** from the **Insert** menu. Type the number of rows and/or columns you want to insert or delete. Then press either “Insert” or “Delete.” For example, to delete the column that holds the selected element, type **1** in the box next to “Columns,” **0** in the box next to “Rows,” and press “Delete.”



Note If you insert rows or columns, Mathcad inserts rows *below* the selected element and inserts columns to the *right* of the selected element. If you delete rows or columns, Mathcad begins with the row or column occupied by the selected element and deletes rows from that element downward and columns from that element rightward. To insert a row above the top row or a column to the left of the first column, first place the entire matrix between the editing lines.

Creating Arrays with Range Variables

As introduced in “Range Variables” on page 101, you can use one or more range variables to fill up the elements of an array. If you use two range variables in an equation, for example, Mathcad runs through each value of each range variable. This is useful for defining matrices. For example, to define a 5×5 matrix whose i,j th element is $i + j$, enter the equations shown in Figure 11-1.

Recall that you enter the range variable operator by pressing the semicolon key (;) or

clicking  on the Calculator toolbar. You enter the subscript operator by clicking  on the Matrix toolbar.

The $x_{i,j}$ equation is evaluated for each value of each range variable, for a total of 25 evaluations. The result is the matrix shown at the bottom of Figure 11-1, with 5 rows and 5 columns. The element in the i th row and j th column of this matrix is $i + j$.

Note To be used to define an array element, a range variable can take on only whole-number values.

$$\begin{array}{l}
 i := 0 \dots 4 \quad j := 0 \dots 4 \\
 x_{i,j} := i + j \\
 x = \begin{pmatrix} 0 & 1 & 2 & 3 & 4 \\ 1 & 2 & 3 & 4 & 5 \\ 2 & 3 & 4 & 5 & 6 \\ 3 & 4 & 5 & 6 & 7 \\ 4 & 5 & 6 & 7 & 8 \end{pmatrix}
 \end{array}$$

Figure 11-1: Defining a matrix using range variables.

Tip You can also define individual array elements using the subscript operator, as described in “Accessing Array Elements” on page 196.

Reading a Data File

Mathcad provides the *File Read/Write component* to read a data file and store the data in a Mathcad array variable.

Note A component is a specialized OLE object that you insert into a Mathcad worksheet to create a link between the worksheet and either a data source or another application containing data. For more information on components, including specialized components for linking other computational applications dynamically to arrays in a Mathcad worksheet, see Chapter 16, “Advanced Computational Features.”

You can read data from files in a variety of formats, including:

- Excel (*.XLS)
- MATLAB (*.MAT)
- Lotus 1-2-3 (*.WK*)
- ASCII editors (*.DAT, *.CSV, *.PRN, *.TXT)

Tip Mathcad also provides a number of built-in functions for importing ASCII data files and image files. See “File Access Functions” on page 188.

To read in data using the File Read/Write component:

1. Click in a blank spot of your worksheet.
2. Choose **Component** from the **Insert** menu.
3. Choose File Read or Write from the list and click “Next.” This launches the File Read or Write Wizard.
4. Choose “Read from a file” and press “Next” to continue through the Wizard.
5. Specify the type of data file you want to read. Enter the path to the data file or use the “Browse” button to locate it.

- Press “Finish.” You’ll see the File Read/Write component icon and the path to the data file. For example, if you specify a data file called data.txt in the C:\WINDOWS folder, you’ll see the component at right.



In the placeholder that appears, enter the name of the Mathcad variable to which the data from the file will be assigned. When you click outside the component, the data file is read in and the data is assigned to the Mathcad array variable you entered into the placeholder.

Each time you calculate the worksheet, Mathcad re-reads the data from the file you have specified. Figure 11-2 shows an example of reading in data using the File Read/Write component. If you want to import data from a file just once into Mathcad, refer to “Importing Once from a Data File” on page 196.

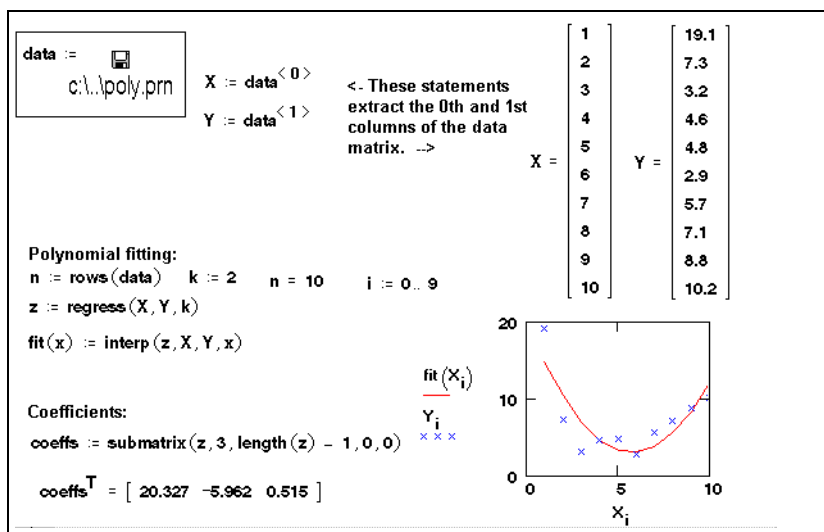


Figure 11-2: Reading in data from a data file. Whenever you calculate the worksheet, the data file is read in.

To read in a different data file or a different type of data file:

- Click with the right mouse button on the component and select **Choose File** from the component pop-up menu.
- In the “Files of type” text box, choose the type of file you’d like to import. Use the dialog box to browse to the data file, select the data file, and click “Open.”

Tip By default, Mathcad reads in the entire data file and creates an array with the variable name you provide. To read in only certain rows or columns of a data file, click once on the component to select it, then click with the right mouse button on the component and choose **Properties** from the pop-up menu. Use the Properties dialog box to specify the row and columns at which to start and stop reading.

Entering Data into a Table

To get the convenience of a spreadsheet-like interface for entering data, you can create an array using the Input Table component:

1. Click in a blank spot in your worksheet and choose **Component** from the **Insert** menu.
2. Select **Input Table** from the list and click “Next.” The Input Table component is inserted into your worksheet.
3. Enter the name of the Mathcad variable to which the data will be assigned in the placeholder that appears.
4. Click in the component and enter data into the cells. Each row must have the same number of data values. If you do not enter a number into a cell, Mathcad inserts 0 into the cell.

Figure 11-3 shows two input tables. Notice that when you create an input table, you’re actually assigning elements to an array that has the name of the variable you entered into the placeholder.

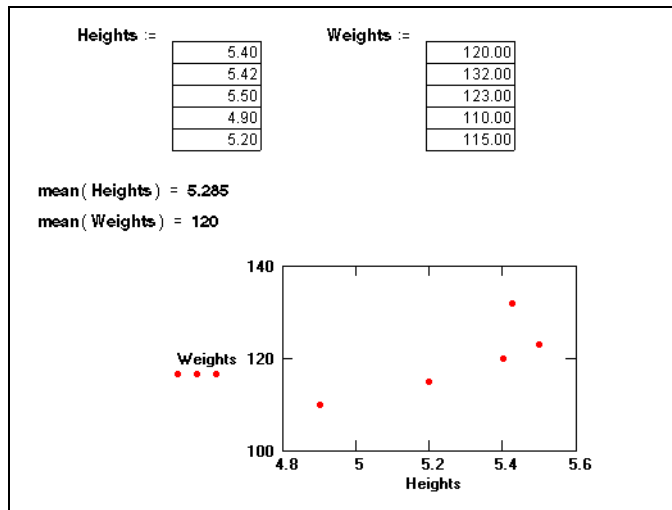


Figure 11-3: Using input tables to create arrays of data.

When you click the table, you can edit the values in it. The scroll bars let you scroll through the table. To resize the table, move the cursor to one of these handles along the sides of the region so that it changes to a double-headed arrow. Then press and hold down the mouse button and drag the cursor to change the table’s dimensions.

Tip You can copy data from an input table as follows: first select some data, then click with the right mouse button on the component and choose **Copy** from the pop-up menu. You can paste a single number from the Clipboard into the table by selecting a cell and choosing **Paste** from the pop-up menu. Choosing **Paste Table** from the pop-up menu overwrites the entire table with values in the Clipboard.

Importing Once from a Data File

You can use an input table to import data a single time from a data file. To do so:

1. Insert an input table by following the instructions given above.
2. In the placeholder that appears to the left, enter the name of the Mathcad variable to which this data will be assigned.
3. Click on the table to select it. Then click on it with the right mouse button on the input table so that you see the pop-up menu.
4. Choose **Import**.
5. The Read from File dialog box appears. In the “Files of type” text box, choose the type of file you’d like to import. Use the dialog box to browse to the data file and click “Open.”

The data from the data file appears in your worksheet in the input table.

Note Unlike the File Read/Write component, the Import feature of an input table reads the data only when you choose **Import**, not each time you calculate the worksheet. If you want the data to be imported each time you calculate, use the File Read/Write component as described in “Reading a Data File” on page 193.

Accessing Array Elements

You can access all the elements of an array simply by using its variable name, or you can access the elements individually or in groups.

Subscripts

You access individual elements of a vector or matrix by using the subscript operator described in “Vector and Matrix Operators” on page 127. Insert the subscript operator

by clicking  on the Matrix toolbar or by typing `[`. To access an element of a vector, enter one number in the subscript. To access a matrix element, enter two numbers separated by a comma. To refer to the i th element of a vector, type $\mathbf{v}[\mathbf{i}]$. In general, to refer to the element in the i th row, j th column of matrix $\mathbf{M}$, type $\mathbf{M}[\mathbf{i}, \mathbf{j}]$.

Figure 11-4 shows examples of how to define individual matrix elements and how to view them.

Note When you define vector or matrix elements, you may leave gaps in the vector or matrix. For example, if $\mathbf{v}$ is undefined and you define $\mathbf{v}_3$ as 10, then $\mathbf{v}_0$, $\mathbf{v}_1$, and $\mathbf{v}_2$ are all undefined. Mathcad fills these gaps with zeros until you enter specific values for them, as shown in Figure 11-4. Be careful of inadvertently creating very large vectors and matrices by doing this. Also note that vector and matrix elements by default are numbered starting with row zero and column zero unless the built-in variable ORIGIN has a value other than zero (see page 197).

You can use this kind of subscript notation in Mathcad to perform parallel calculations on the elements of an array. See “Doing Calculations in Parallel” on page 201.

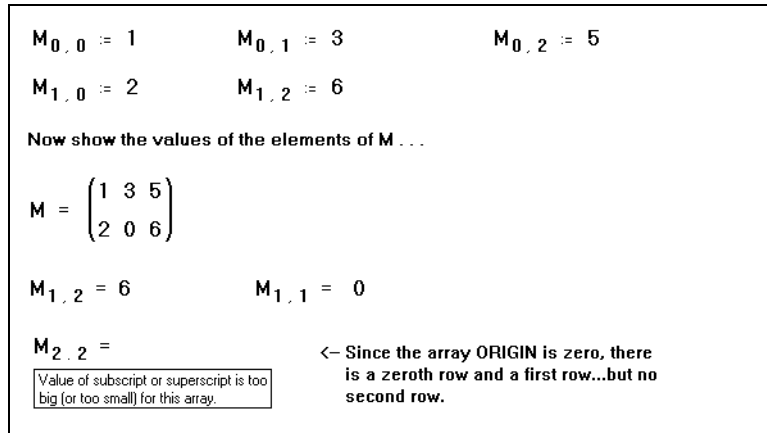


Figure 11-4: Defining and viewing matrix elements.

Tip If you want to define or access a group of array elements at once, you can use a range variable in a subscript.

Accessing Rows and Columns

Although you can use a range variable to access all the elements in a row or column of an array, Mathcad provides a column operator for quickly accessing all the elements

in a column. Click  on the Matrix toolbar for the column operator. Figure 11-5 shows how to extract the third column of the matrix **M**.

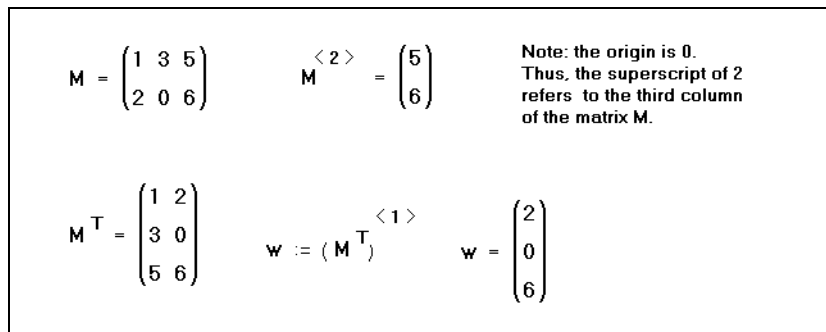


Figure 11-5: Extracting a column from a matrix.

To extract a single row from a matrix, transpose the matrix using the transpose operator (click  on the Matrix toolbar) and then extract a column using the column operator. This is shown on the right-hand side of Figure 11-5.

Changing the Array Origin

When you use subscripts to refer to array elements, Mathcad assumes the array begins at the current value of the built-in variable **ORIGIN**. By default, **ORIGIN** is 0, but you can change its value. See “Built-in Variables” on page 98 for details.

Figure 11-6 shows a worksheet with the ORIGIN set to 1. If you try to refer to the zeroth element of an array in this case, Mathcad displays an error message.

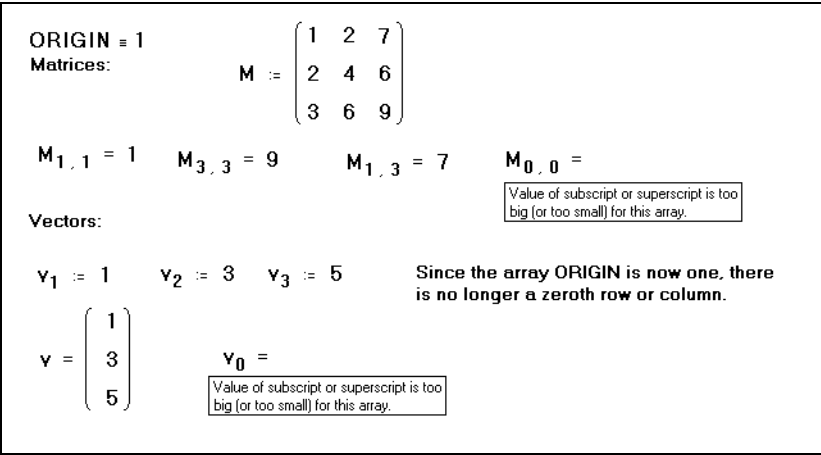


Figure 11-6: Arrays beginning at element one instead of at element zero.

Displaying Arrays

As described in “Formatting Results” on page 110, Mathcad automatically displays matrices and vectors having more than nine rows or columns as output tables rather than as matrices or vectors. Smaller arrays are displayed by default in traditional matrix notation. Figure 11-7 shows an example.

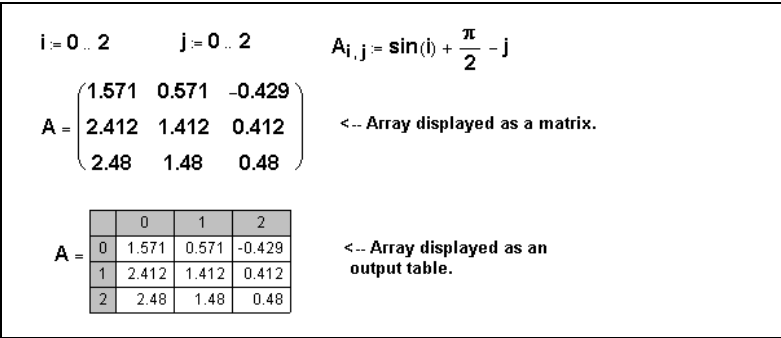


Figure 11-7: Display results as a matrix or in an output table.

Note An output table displays a portion of an array. To the left of each row and at the top of each column, there is a number indicating the index of the row or column. Click with the right mouse button on the output table and select **Properties** from the pop-up menu to control whether row and column numbers appear and the font used for values in the table. If your results extend beyond the table, a scroll bar appears along the appropriate edge of the table. You can scroll through the table using these scroll bars just as you would scroll through any window.

To resize an output table:

1. Click the output table. You'll see handles along the sides of the table.
2. Move the cursor to one of these handles so that it changes to a double-headed arrow.
3. Press and hold down the mouse button and drag the cursor in the direction you want the table's dimensions to change.

Tip You can change the alignment of the table with respect to the expression on the left-hand side of the equal sign. Click with the right mouse button on the table, then choose one of the **Alignment** options from the pop-up menu.

Changing the Display of Arrays — Table versus Matrix

Although matrices and vectors having more than nine rows or columns are automatically displayed as scrolling output tables, you can have Mathcad display them as matrices. You can also change matrices to output tables. To do so:

1. Click on the scrolling output table.
2. Choose **Result** from the **Format** menu.
3. Click on the Display Options tab.
4. Choose Matrix or Table in the “Matrix display style” drop-down box.
5. Click “OK.”

To display all the results in your worksheet as matrices or as tables regardless of their size, click “Set as Default” in the Result Format dialog box rather than “OK.”

Note Mathcad cannot display extremely large arrays in matrix form. You should display a large array as a scrolling output table.

Changing the Format of Displayed Elements

You format the numbers in the array the same way you format other numerical results, as described in “Formatting Results” on page 110. Just click on the displayed array and choose **Result** from the **Format** menu, and modify the settings there. When you click “OK,” Mathcad applies the selected format to all the numbers in the table, vector, or matrix. It is not possible to format the numbers individually.

Copying and Pasting Arrays

You can copy an array of numbers directly from a spreadsheet or database into Mathcad where you can take advantage of its free-form interface and its advanced mathematical tools. Once you've performed the necessary computations, you can paste the resulting array of numbers back to its source or even into another application.

To copy just one number from a result array, simply click the number and choose **Copy**

from the **Edit** menu, or click  on the Standard toolbar. Copying multiple numbers from a vector or matrix result differs depending on whether the array is displayed as a matrix or as an output table. See “Formatting Results” on page 110 for more information on how vector and matrix results are displayed.

To copy a result array displayed as a matrix:

1. Drag-select the array to the right of the equal sign to place the entire array between the editing lines.
2. Choose **Copy** from the **Edit** menu. This places the entire array on the Clipboard.
3. Click wherever you want to paste the result. If you're pasting into another application, choose **Paste** from that application's **Edit** menu. If you're pasting into

a Mathcad worksheet, choose **Paste** from Mathcad's **Edit** menu, or click  on the Standard toolbar.

Note You may only paste an array into a math placeholder or into a blank space in a Mathcad worksheet.

When you display array results as a table, you can copy some or all of the numbers from the table and use them elsewhere:

1. Click on the first number you want to copy.
2. Drag the mouse in the direction of the other values you want to copy while holding the mouse button down.
3. Click on the selected values with the right mouse button and choose **Copy Selection** from the pop-up menu.

To copy all the values in a row or column, click on the column or row number shown to the left of the row or at the top of the column. All the values in the row or column are selected. Then choose **Copy** from the **Edit** menu.

After you have copied one or more numbers from an output table, you can paste them into another part of your worksheet or into another application. Figure 11-8 shows an example of a new matrix created by copying and pasting numbers from an output table.

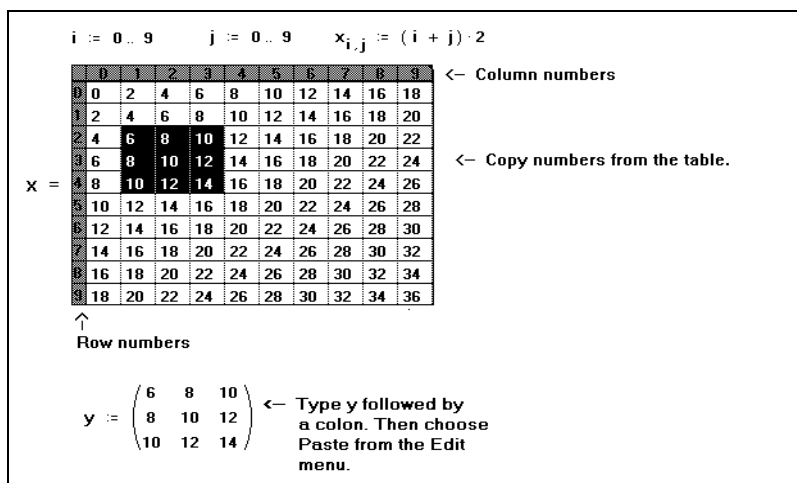


Figure 11-8: Copying and pasting results from an output table.

Working with Arrays

Once you create an array, you can use it in calculations. There are many operators and functions designed for use with vectors and matrices. See “Vector and Matrix Operators” on page 127 and “Vector and Matrix Functions” on page 152 for an overview. This section highlights the vectorize operator, which permits efficient parallel calculations on the elements of arrays. You can also display the values of an array graphically or export them to a data file or another application.

Doing Calculations in Parallel

Any calculation Mathcad can perform with single values, it can also perform with vectors or matrices of values. There are two ways to do this:

- Iterate over each element using range variables. See for example “Creating Arrays with Range Variables” on page 192.
- Use the *vectorize operator*, which allows Mathcad to perform the same operation efficiently on each *element* of a vector or matrix.

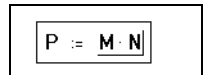
Mathematical notation often shows repeated operations with subscripts. For example, to define a matrix **P** by multiplying corresponding elements of the matrices **M** and **N**, you would write:

$$P_{i,j} = M_{i,j} \cdot N_{i,j}$$

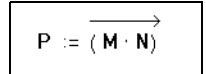
Note that this is not matrix multiplication, but multiplication element by element. It is possible to perform this operation in Mathcad using subscripts, but it is much faster to perform exactly the same operation with a vectorized equation.

Here’s how to apply the vectorize operator to an expression like **M · N**:

1. Select the whole expression by clicking inside it and pressing [**Space**] until the right-hand side is surrounded by the editing lines.



2. Click  on the Matrix toolbar to apply the vectorize operator. Mathcad puts an arrow over the top of the selected expression.



Properties of the vectorize operator

- The vectorize operator changes the meaning of the other *operators* and *functions* to which it applies. The vectorize operator tells Mathcad to apply the operators and functions with their scalar meanings, element by element. It does not change the meaning of the actual names and numbers. If you apply the vectorize operator to a single name, it simply draws an arrow over the name. You can use this arrow for cosmetic purposes only if you like.
- Since operations between two arrays are performed element by element, all arrays under a vectorize operator must be the same size. Operations between an array and a scalar are performed by applying the scalar to each element of the array.
- You can use any of the following matrix operations under a vectorize operator: dot product, matrix multiplication, matrix powers, matrix inverse, determinant, or

magnitude of a vector. The vectorize operator transforms these operations into element-by-element scalar multiplication, exponentiation, or absolute value, as appropriate.

Tip A number of Mathcad's built-in functions and operators ordinarily take scalar arguments but *implicitly* vectorize arguments that are vectors (one-column arrays): they automatically compute a result element by element, whether you apply the vectorize operator or not. Functions that implicitly vectorize vector arguments include the trigonometric, logarithmic, Bessel, and probability distribution functions. Operators that implicitly vectorize vector arguments include the factorial, square and n th root, and relational operators. You must continue to use the vectorize operator on arrays of other sizes with these functions and operators.

For example, suppose you want to apply the quadratic formula to three vectors containing coefficients a , b , and c . Figure 11-9 shows how to do this with the vectorize operator.

Coefficients as follows ...

$$a := \begin{pmatrix} 1 \\ 1 \\ 2 \\ 2 \end{pmatrix} \quad b := \begin{pmatrix} 3 \\ 2 \\ 1 \\ 0 \end{pmatrix} \quad c := \begin{pmatrix} 2 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

Compute a root ...

$$x := \frac{-b + \sqrt{b^2 - 4 \cdot a \cdot c}}{2 \cdot a}$$

$$x = \begin{pmatrix} -1 \\ -1 \\ -0.25 + 0.661i \\ 0.707i \end{pmatrix}$$

$$\frac{\rightarrow}{(a \cdot x^2 + b \cdot x + c)} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad \dots \text{ should be zero}$$

Figure 11-9: Quadratic formula with vectors and the vectorize operator.

The vectorize operator, appearing as an arrow above the quadratic formula in Figure 11-9, is essential in this calculation. Without it, Mathcad would interpret $a \cdot c$ as a vector dot product and also flag the square root of a vector as illegal. But with the vectorize operator, both $a \cdot c$ and the square root are performed element by element.

Graphical Display of Arrays

In addition to looking at the actual numbers making up an array, you can also see a graphical representation of those same numbers. There are several ways to do this:

- For an arbitrary array, you can use the various three-dimensional plot types discussed in Chapter 13, “3D Plots.”
- For an array of integers between 0 and 255, you can look at a grayscale image by choosing **Picture** from the **Insert** menu and entering the array's name in the placeholder.

- For three arrays of integers between 0 and 255 representing the red, green, and blue components of an image, choose **Picture** from the **Insert** menu and enter the arrays' names, separated by commas, in the placeholder.

See Chapter 6, “Working with Graphics and Other Objects,” for more on viewing a matrix (or three matrices, in the case of a color image) in the picture operator.

Writing to a Data File

The File Read/Write component allows you to write the values stored in a Mathcad variable to a data file in a variety of formats, including the following:

- Excel (*.XLS)
- MATLAB (*.MAT)
- Lotus 1-2-3 (*.WK*)
- ASCII editors (*.DAT, *.CSV, *.PRN, *.TXT)

Tip Mathcad also provides a number of built-in functions to export arrays as ASCII data files or image files. See “File Access Functions” on page 188.

To export data using the File Read/Write component:

1. Click in a blank spot in your worksheet.
2. Choose **Component** from the **Insert** menu.
3. Select File Read or Write from the list and click “Next.” This launches the File Read or Write Wizard.
4. Choose “Write to a file” and press “Next” to continue through the Wizard.
5. Specify the type of data file you want to create. Also enter the path to the data file you want to write or click the “Browse” button to locate it.
6. Press “Finish.” You’ll see the File Read/Write component icon and the path to the data file. For example, if you specify a data file called data.txt, you’ll see the component at right.



In the placeholder, enter the name of the Mathcad variable containing the data to be written to the data file. When you click outside the component, all the values in the array are written to the file you specified. Each time you calculate the worksheet, the data file is rewritten. See Figure 11-10 for an example.

To change the name of the data file being created to or to change the type of file being created:

1. Click once on the component to select it.
2. Click with the right mouse button on the component and select **Choose File** from the pop-up menu to open the Write to File dialog box.
3. Choose the type of file you’d like to create in the “Files of type” text box. Use the dialog box to browse to the folder in which the data file will be created and click “Open.”

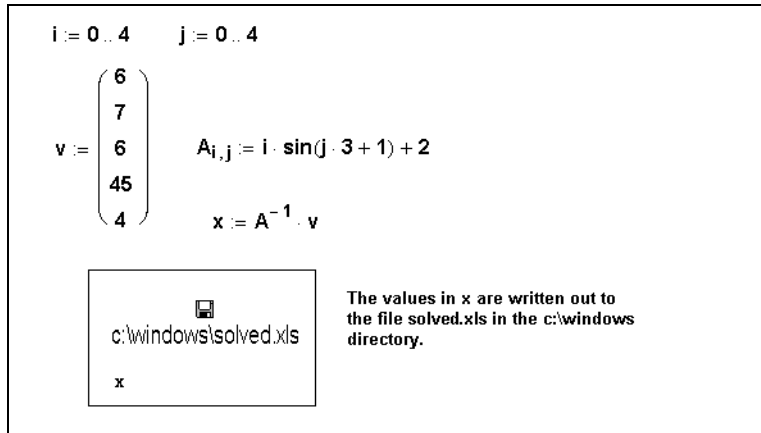


Figure 11-10: Exporting data with the File Read/Write component

Tip When you display an array as an output table, as described in “Displaying Arrays” on page 198, you can export data directly from the table. Click with the right mouse button on the output table, choose **Export** from the pop-up menu, and enter the name of the file that will receive the data. Unlike the File Read/Write component, the output table writes the data only when you choose **Export**, not each time you calculate the worksheet.

Nested Arrays

An array element need not be a scalar. It’s possible to make an array element itself be another array. This allows you to create arrays within arrays.

These arrays behave very much like arrays whose elements are all scalars. However, there are some distinctions, as described below.

Note Most of Mathcad’s operators and functions do not work with nested arrays, since there is no universally accepted definition of what constitutes correct behavior in this context. Certain operators and functions are nevertheless useful and appropriate for nested arrays. Functions that enumerate rows or columns, or that partition, augment, and stack matrices, can be applied to nested arrays. The transpose, subscript, and column array operators and the Boolean equal sign likewise support nested arrays.

Defining a Nested Array

You define a nested array in much the same way you would define any array. The only difference is that you cannot use the **Matrix** command from the **Insert** menu when you’ve selected a placeholder within an existing array. You can, however, click on a placeholder in an array and type the *name* of another array. Figure 11-11 shows several ways to define a nested array. Additional methods include using a file access function such as READPRN in the array of placeholders created using the Insert Matrix command, and using the programming operators in Mathcad Professional to build up an array whose elements are themselves arrays.

Three ways to define nested arrays...		
Using range variables	Using the Matrices command	Defining element by element
$m := 0..3$	$u := \begin{pmatrix} 1 \\ 2 \end{pmatrix}$	$B_0 := 1$
$n := 0..3$	$v := (2 \ 4)$	$B_1 := \text{identity}(2)$
$M_{m,n} := \text{identity}(m+1)$	$V := \begin{pmatrix} u \\ v \end{pmatrix}$	$B_2 := \begin{pmatrix} B_0 & 2 & v \end{pmatrix}$
----- Displaying the elements -----		
$M_{0,0} = (1)$	$V_0 = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$	$B_0 = 1$
$M_{1,1} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$	$V_1 = (2 \ 4)$	$B_1 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$
$M_{2,2} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$	$V = \begin{pmatrix} \{2,1\} \\ \{1,2\} \end{pmatrix}$	$B = \begin{pmatrix} 1 \\ \{2,2\} \\ \{1,3\} \end{pmatrix}$

Figure 11-11: Defining nested arrays.

Note The display of a nested array is controlled by Display Styles settings in the Result Format dialog box (see page 110). You can expand a nested array when the array is displayed in matrix form; otherwise, whenever an array element is itself an array, you see bracket notation showing the number of rows and columns rather than the array itself. If the nested array is displayed as an output table, you can see the underlying array temporarily. Click on the array element, then click with the right mouse button and choose **Down One Level** from the pop-up menu. Choose **Up One Level** from the pop-up menu to restore the array element to non-expanded form.

Chapter 12

2D Plots

- τ Overview of 2D Plotting
- τ Graphing Functions and Expressions
- τ Plotting Vectors of Data
- τ Formatting a 2D Plot
- τ Modifying Your 2D Plot's Perspective

Overview of 2D Plotting

To visually represent a function or expression of a single variable or X-Y data in Mathcad, you can create a 2D plot. You can create either a Cartesian X-Y plot or a polar plot. A typical X-Y plot shows horizontal x-values versus vertical y-values, while a typical polar plot shows angular values, θ , versus radial values, r . Figure 12-1 shows several examples of 2D plots.

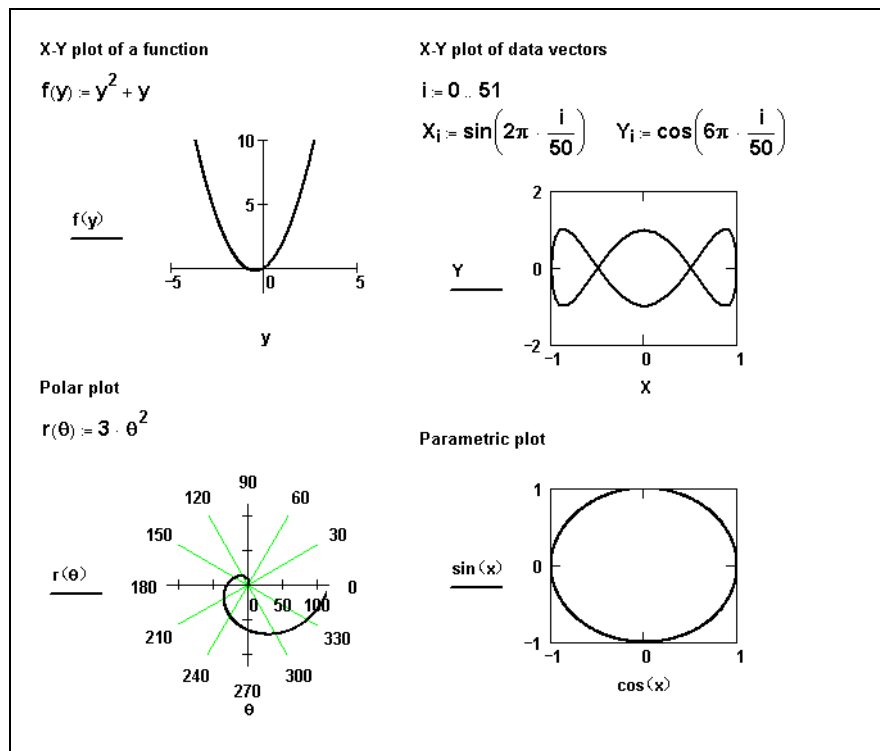
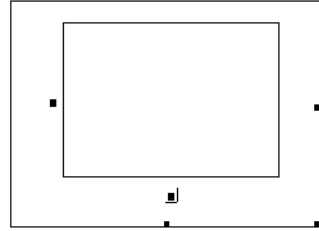


Figure 12-1: Examples of 2D plots.

Creating an X-Y Plot

In general, to create an X-Y plot:

1. Click in your worksheet where you want the graph to appear.
2. Choose **Graph**⇒**X-Y Plot** from the **Insert** menu
or click  on the Graph toolbar. Alternatively, type **[Shift] 2** or **@**. Mathcad inserts a blank X-Y plot.
3. Fill in both the x-axis placeholder (bottom center) and the y-axis placeholder (left center) with a function, expression, or variable.
4. Click outside the plot or press **[Enter]**.



Mathcad automatically chooses axis limits for you. If you want to specify the axis limits yourself, click in the plot and type over the numbers in the placeholders at the ends of the axes.

Mathcad creates the plot over a default range using default limits. See “Formatting a 2D Plot” on page 216 for information on modifying these defaults.

Note If a point is complex, Mathcad does not graph it. To graph the real or imaginary part of a point or expression, use the *Re* and *Im* functions to extract the real and imaginary parts, respectively.

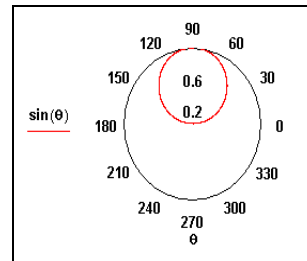
To resize a plot, click in the plot to select it. Then move the cursor to a handle along the right or bottom edge of the plot until the cursor changes to a double-headed arrow. Hold the mouse button down and drag the mouse in the direction that you want the plot’s dimension to change.

Note If some points in a function or expression are valid and others are not, Mathcad plots only the valid ones. If the points are not contiguous, Mathcad does not connect them with a line. You may therefore see a blank plot if none of the points are contiguous. To see the points, format the trace to have symbols. See “Formatting a 2D Plot” on page 216 for information on formatting traces.

Creating a polar plot

In general, to create a polar plot:

1. Click in your worksheet where you want the graph to appear.
2. Choose **Graph**⇒**Polar Plot** from the **Insert** menu or
click  on the Graph toolbar.
3. Fill in both the angular-axis placeholder (bottom center) and the radial-axis placeholder (left center) with a function, expression, or variable.
4. Click outside the plot or press **[Enter]**.



Mathcad creates the plot over a default range using default limits. See “Formatting a 2D Plot” on page 216 for information on modifying these defaults.

The remaining sections in this chapter focus on plotting functions, expressions, and data. Although the instructions and figures typically show X-Y plots, the instructions apply to polar plots as well.

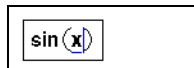
Graphing Functions and Expressions

2D QuickPlots

A 2D *QuickPlot* is a plot created from an expression or function which represents the y-coordinates of the plot. With a *QuickPlot*, there is no need to define the independent or x-axis variable.

To create an X-Y plot of a single expression or function:

1. Click in your worksheet where you want the graph to appear.
2. Enter the expression or function of a single variable you want to plot. Make sure the editing lines remain in the expression.

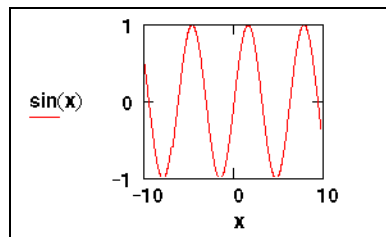


3. Choose **Graph**⇒**X-Y Plot** from the **Insert**

menu or click  on the Graph toolbar.

4. Click outside the graph or press **[Enter]**.

Mathcad automatically produces a plot over a default domain for the independent variable, from -10 to 10 .



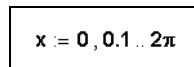
To change the default domain for the independent variable in a 2D QuickPlot, change the axis limits on the plot:

1. Click on the plot, and then click on one of the four axis limit placeholders located at the ends of the axes.
2. Type the value of the axis limit you want. There is no restriction on the values you can enter in these placeholders.
3. Click outside the graph or press **[Enter]** to see the updated graph.

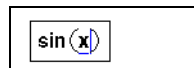
Defining an independent variable

If you don't want Mathcad to use a default range for the independent variable, you can define the independent variable as a range variable before creating the plot. For example:

1. Define a range variable, such as x , that takes on the values you want to graph. The range variable need not be called x ; you can use any valid name. See "Range Variables" on page 101.



2. Enter an expression or function you want to plot using that variable. Make sure the editing lines remain in the expression.

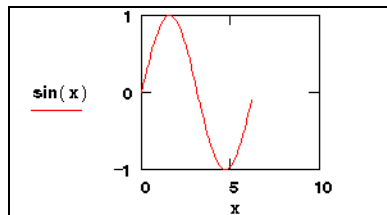


3. Choose **Graph**⇒**X-Y Plot** from the **Insert** menu or click  on the Graph toolbar.

4. Type the name of the variable into the x-axis placeholder.

- Click outside the graph or press **[Enter]**.

Mathcad graphs one point for every value of the range variable, and, unless you specify otherwise, connects each pair of points with a straight line.



Tip To override Mathcad's choices for the axis limits on a plot, click in the plot and type over the limits in the placeholders at the ends of the axes. See "Setting Axis Limits" on page 217 for more information.

Plotting Multiple 2D Curves

You can graph several traces on the same X-Y or polar plot. A graph can show several y-axis (or radial) expressions against the same x-axis (or angular) expression. See Figure 12-3. Or it can match up several y-axis (or radial) expressions with the corresponding number of x-axis (or angular) expressions. See Figure 12-2.

To create a *QuickPlot* containing more than one trace:

- Enter the expressions or functions of a single variable you want to plot, separated by commas. Make sure the editing lines remain in the expression.

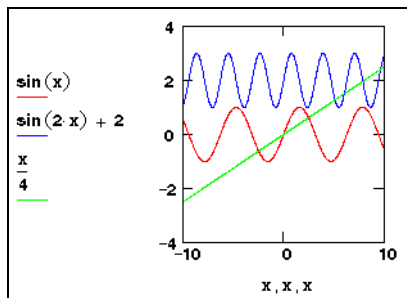
$$\sin(x), \sin(2x) + 2, \frac{x}{4}$$

- Choose **Graph**⇒**X-Y Plot** from the **Insert**

menu or click  on the Graph toolbar.

- Click outside the graph or press **[Enter]**.

Mathcad produces a single graph containing plots of all the expressions or functions, over a default range for the independent variable(s), from -10 to 10 . You can change the axis range by editing the upper and lower limits on the x-axis as described in "Setting Axis Limits" on page 217.



Note In a *QuickPlot* with multiple traces, you need not use the same independent variable in every y-axis (or radial-axis) expression. Mathcad will provide the appropriate corresponding variable in the x-axis (or angular-axis) placeholder.

In general, to create a graph containing several independent curves:

- Choose **Graph**⇒**X-Y Plot** from the **Insert** menu or click  on the Graph toolbar.
- Enter two or more expressions separated by commas in the y-axis placeholder.
- Enter the same number of expressions separated by commas in the x-axis placeholder.

Mathcad matches up the expressions in pairs—the first x -axis expression with first y -axis expression, the second with the second, and so on. It then draws a trace for each pair. Figure 12-2 shows an example.

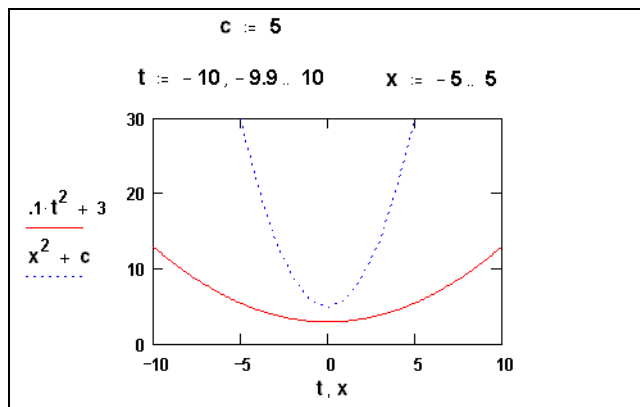


Figure 12-2: Graph with multiple expressions on both axes.

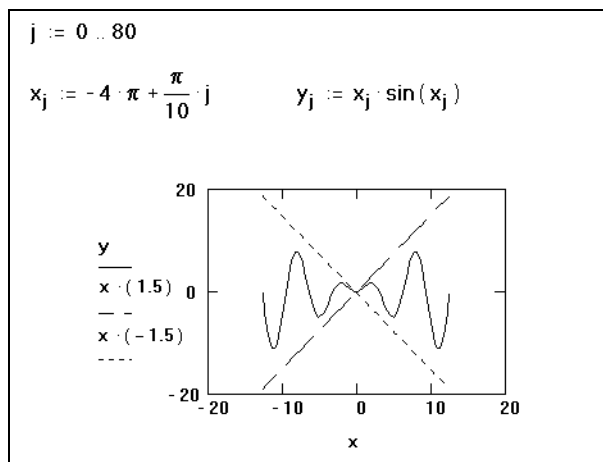


Figure 12-3: Graph with multiple y -axis expressions.

Note All traces on a graph share the same axis limits. For each axis, all expressions and limits on that axis must have compatible units.

Creating a parametric plot

A parametric plot is one in which a function or expression is plotted against another function or expression that uses the same independent variable. You can create either an X-Y or polar parametric plot.

To create an X-Y parametric plot:

1. Click in your worksheet where you want the graph to appear.

2. Choose **Graph**⇒**X-Y Plot** from the **Insert** menu or click  on the Graph toolbar. Mathcad inserts a blank X-Y plot with empty placeholders.
3. In both the x -axis and y -axis placeholders, enter a function or expression.
4. Click outside the plot or press **[Enter]**.

Mathcad produces a *QuickPlot* over a default range for the independent variable. Figure 12-1 shows an example of a parametric plot.

If you don't want Mathcad to use a default range for the plot, define the independent variable as a range variable before creating the plot. Mathcad graphs one point for each value of the independent variable and connects each pair of points with a straight line. Figure 12-4 shows two functions of θ plotted against each other. The range variable θ was previously defined. For more information, see "Range Variables" on page 101.

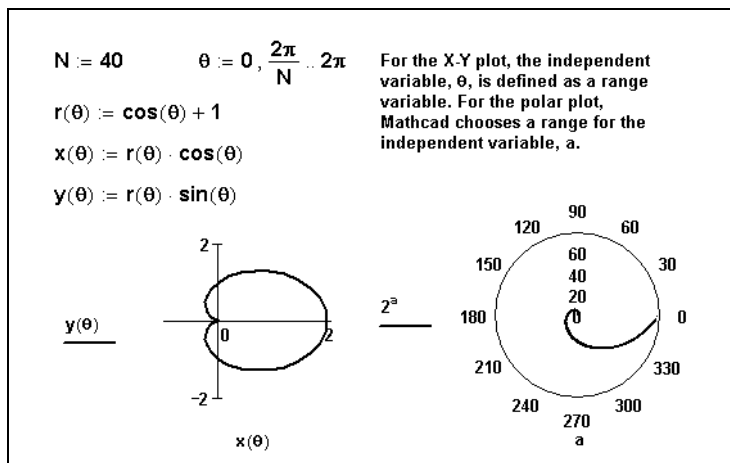


Figure 12-4: Graphing one function against another.

Plotting Vectors of Data

To graph a vector of data, you can create either an X-Y plot or a polar plot. When creating either type of plot, you need to use the vector subscript (see "Vector and Matrix Operators" on page 127) to specify which elements to plot. Additionally, you can use Axum LE (see "Using Axum to Plot Data" on page 214) to create a 2D plot of your vectors or data. Some graphs of data vectors are shown in Figure 12-5.

Plotting a single vector of data

To create an X-Y plot of a single vector of data:

1. Define a range variable, such as i , that references the subscript of each element of the vector you want to plot. For example, for a vector with 10 elements, your subscript range variable would be $i := 0 .. 9$.
2. Click in your worksheet where you want the graph to appear.

3. Choose **Graph**⇒**X-Y Plot** from the **Insert** menu or click  on the Graph toolbar. Mathcad inserts a blank X-Y plot.
4. Enter i in the bottom placeholder and the vector name with the subscript (y_i for example) in the placeholder on the left. Type **[** as a shortcut to create the subscript.
5. Click outside the graph or press **[Enter]**.

Note Subscripts must be integers greater than or equal to ORIGIN. This means that the x-axis or angular variable used in the graphs in Figure 12-5 can run through whole-number values only. If you want to graph fractional or negative values on the x-axis, graph a function or graph one vector against another, as described in the next section.

Tip If you have a handful of data points, you can use an input table to create a vector as shown in the second graph in Figure 12-5 or in Figure 12-7. For more information, see “Entering Data into a Table” on page 195.

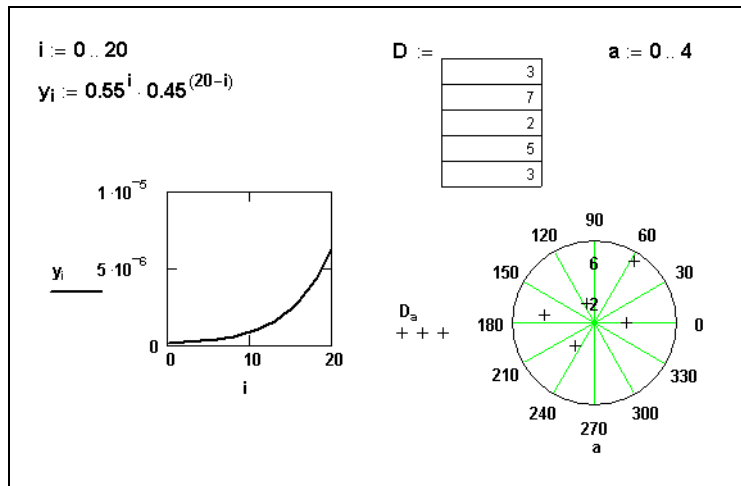


Figure 12-5: Graphing a vector.

Plotting one data vector against another

To graph all the elements of one data vector against all the elements in another, enter the names of the vectors in the axis placeholders of an X-Y plot or polar plot.

For example, to create an X-Y plot of two data vectors x and y :

1. Define the vectors x and y .
2. Click in your worksheet where you want the graph to appear.
3. Choose **Graph**⇒**X-Y Plot** from the **Insert** menu, or click  on the Graph toolbar.
4. Enter y in the y-axis placeholder and x in the x-axis placeholder.

5. Click outside the graph or press **[Enter]**.

Mathcad plots the elements in the vector x against the elements in the vector y .

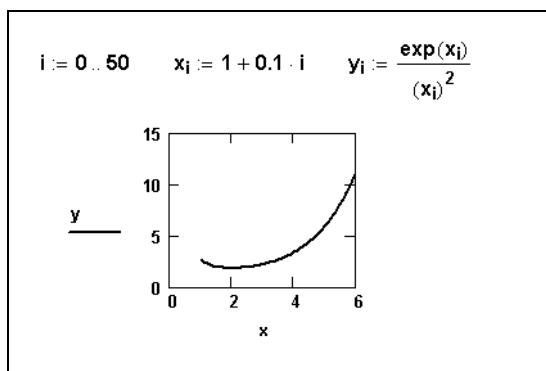


Figure 12-6: Graphing two vectors.

Note If the vectors being plotted are not the same length, Mathcad plots the number of elements in the shorter vector.

If you want to plot only certain vector elements, define a range variable and use it as a subscript on the vector names. In the example above, to plot the fifth through tenth elements of x and y against each other:

1. Define a range variable, such as k , going from 4 to 9 in increments of 1. (Note that the first elements of the vectors x and y are x_0 and y_0 by default.)
2. Enter y_k and x_k in the axis placeholders.

Note If you have a set of data values to graph, create a vector by reading in data from a data file, by pasting in the data from the Clipboard, or by typing data directly into an input table. See Chapter 11, “Vectors, Matrices, and Data Arrays.” See Figure 12-7 for an example showing the use of an input table.

Using Axum to Plot Data

Included on your Mathcad CD is the application Axum LE. Axum LE is a version of Axum that features numerous 2D plot types, complete control over graph formatting, and a full set of annotation tools. Axum LE gives you fine control over every aspect of your 2D plots, and you can integrate these plots into your Mathcad worksheets.

There are two basic methods for integrating plots from Axum into your Mathcad worksheet. You can create a graph in Axum and insert it into your Mathcad worksheet as an object. Or, you can define data in your Mathcad worksheet, send it to Axum, and create a dynamic Axum graph directly in your Mathcad worksheet. Here are brief instructions for each method.

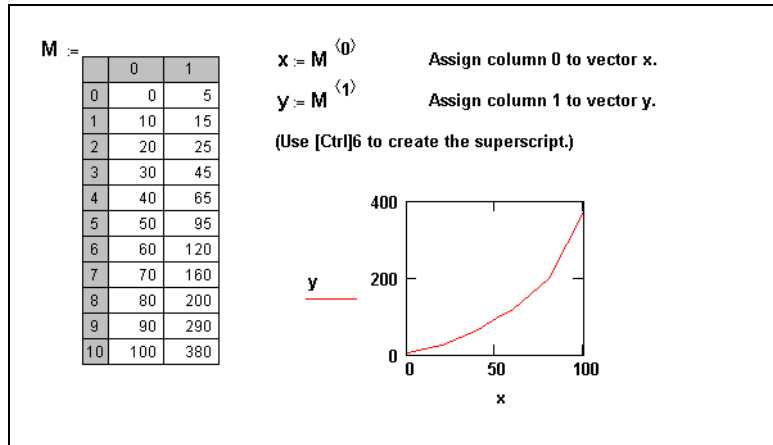


Figure 12-7: Plotting vectors from input table data.

Inserting an Axum graph object

To insert an Axum graph object:

1. Create and save a graph in Axum. For more information about using Axum, see Axum's on line Help.
2. In Mathcad, choose **Object** from the **Insert** menu. In the Insert Object dialog box, click Create from File, and browse for your saved Axum graph sheet. Once you have selected your graph sheet, click OK.
3. The Axum graph appears into your Mathcad worksheet. If you check Link in the Insert Object dialog, you may activate the plot from within your Mathcad worksheet, by double-clicking, and make changes to the original Axum graph.

For more information about inserting objects, refer to "Chapter 6: Working with Graphics and Other Objects."

Inserting a Dynamic Axum graph

To insert an Axum graph linked to data in your Mathcad worksheet:

1. In Mathcad, define the vector(s) of data you wish to plot.
2. Click in a blank spot in your worksheet. Be sure to click below or to the right of your vector(s) of data.
3. Choose **Component** from the **Insert** menu. Select Axum Graph from the list and click Next. Choose a plot type and specify as many input variables as you have data vectors. Click Finish.
4. A blank Axum graph appears in your Mathcad worksheet. Enter the name(s) of your data vector(s) in the placeholders in the bottom left corner of the graph. Click outside the graph or press **[Enter]**.

If you change the vectors of data upon which your Axum graph component is dependent, your graph updates automatically. For more information about using components, refer to "Chapter 16: Advanced Computational Features." Figure 12-8

shows an Axum graph that has been customized with axes labels, a title, and text and graphic annotations.

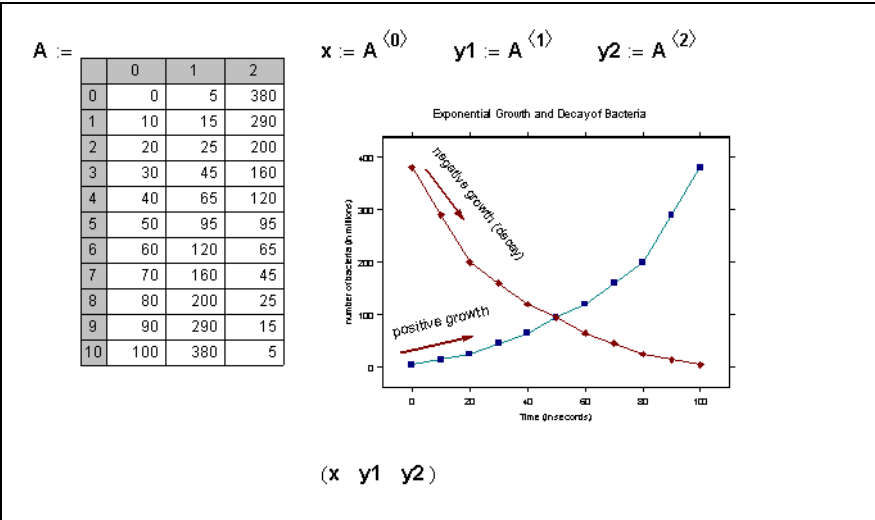


Figure 12-8: An Axum graph in a Mathcad worksheet.

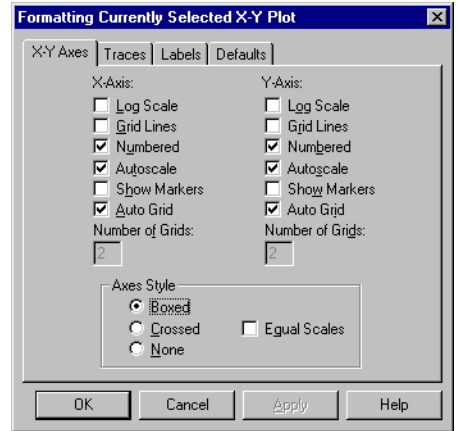
Note If you want to create an Axum graph component with multiple independent traces, define x - and y -vectors for each plot. Then, choose the plot type “Scatter Plots of XY Pairs” from the Axum Graph dialog, and specify as many input variables as you have vectors of data. Enter the vector names in the placeholders in xy -pairs, i.e., ($x1$ $y1$ $x2$ $y2$ etc.)

Formatting a 2D Plot

When you create an X-Y plot or a polar plot, Mathcad uses the default settings to format the axes and traces. You can, however, reformat the axes and traces. You can also add titles and labels and control the default settings of the graph.

To format a 2D graph:

1. Double-click the graph. Alternatively, click once on the graph and choose **Graph⇒X-Y Plot** or **Graph⇒Polar Plot** from the **Format** menu. You'll see the dialog box for formatting a selected graph.
2. Click the tab for the page you want to work with. Use the **Axes** tab to determine the appearance of the axes and grid lines. Use the **Traces** tab to set the color, type, and width of the traces. Use the **Labels** tab to insert labels on the axes. Use the **Defaults** tab to specify the default appearance of your graphs.
3. Make the appropriate changes in the dialog box.
4. Click **Apply** to see the effect of your changes *without* closing the dialog box.
5. Close the dialog by clicking **OK**.



Note In the **Axes** page, make sure you turn options on and off in the appropriate axis column. In the **Traces** page, click on a trace's name in the **Legend Label** column and change characteristics by clicking on the arrow beside each of the drop-down options.

Tip If you double-click an axis on a graph, you'll see a formatting dialog box for that axis alone.

On-line Help Click **Help** in the dialog box for details on particular formatting options.

Setting Axis Limits

When you create a 2D graph, the **Autoscale** option is turned on. Use the **Axes** page of the plot formatting dialog box to turn **Autoscale** on or off:

- With **Autoscale** on, Mathcad automatically sets each axis limit to the first major tick mark beyond the end of the data. This is a reasonably round number large enough to display every point being graphed.
- With **Autoscale** off, Mathcad automatically sets the axis limits exactly at the data limits.

Specifying Other Limits

You can override Mathcad's automatic limits by entering limits directly on the graph. To do so:

1. Click the graph to select it. Mathcad displays four additional numbers, one by each axis limit. These numbers are enclosed within corner symbols, as illustrated in the selected plot in Figure 12-9.

- Click on one of these numbers and type a number to replace it. Do the same for the other numbers if you want to change more than one limit.
- Click outside the graph. Mathcad redraws it using the new axis limits you specified. The corner symbols below the limits you changed disappear. Figure 12-9 shows the effect of manually setting limits on a graph.

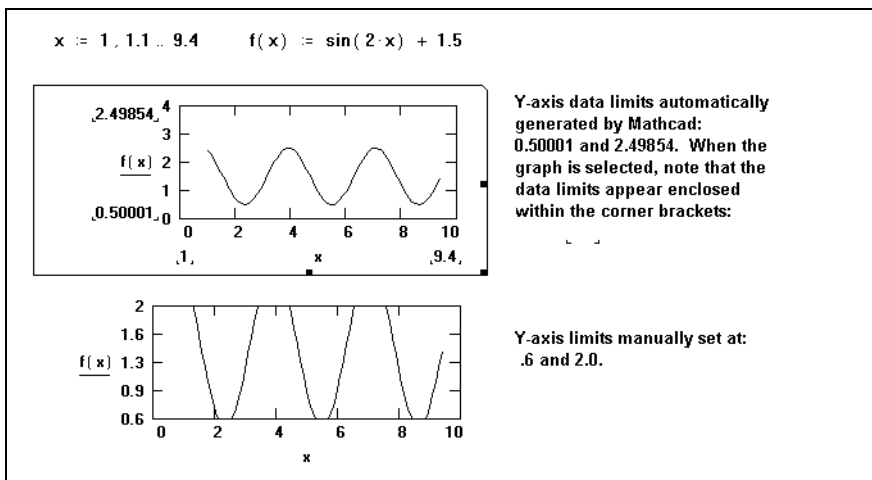


Figure 12-9: Data limits set automatically and manually.

Setting Default Formats

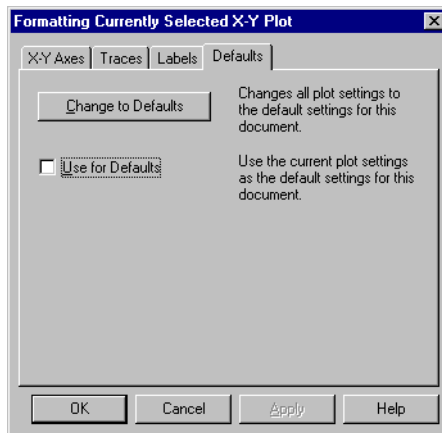
Mathcad uses default settings to format the axes and traces of new graphs you create.

Copying Defaults from an Existing Graph

One way to create a new set of defaults is to use the format settings of an existing graph. The advantage of this method is that you can actually see how the format settings look as you define them.

To use the format of a particular graph as the default graph format:

- Double-click the graph, or click in the graph and choose **Graph⇒X-Y Plot** (or **Graph⇒Polar Plot**) from the **Format** menu. Mathcad displays the dialog box for formatting a selected graph.
- Click the Defaults tab to see the Defaults page.
- Check Use for Defaults. When you click OK, to close the dialog box, Mathcad saves these settings as your default settings.



Setting Defaults Without Using a Graph

You can use the Setting Default Formats dialog box to change default plot settings. To set defaults this way:

1. Make sure that you don't have any graphs selected.
2. Choose **Graph**⇒**X-Y Plot** (or **Graph**⇒**Polar Plot**) from the **Format** menu. You'll see the Setting Default Formats dialog box.
3. Change the appropriate settings on the Axes and Traces pages.
4. Click OK to accept your changes and close the dialog box.

Adding Custom Titles, Labels, and Other Annotations

One way to add titles and labels to your 2D graph is to use the options on the Labels tab of the 2D Plot Format dialog box. A second way to add titles and labels, as well as annotations, is to create text or some other object in your worksheet and then move it on top of the graph.

To create an annotation for your 2D graph:

1. Create a text region, or insert a graphic object in your worksheet by pasting it in or by choosing **Object** from the **Insert** menu.
2. Drag the text or object onto your 2D graph and position it appropriately.

Figure 12-10 shows a graph containing both a text region (“inflection pt”) and a graphic object (an arrow).

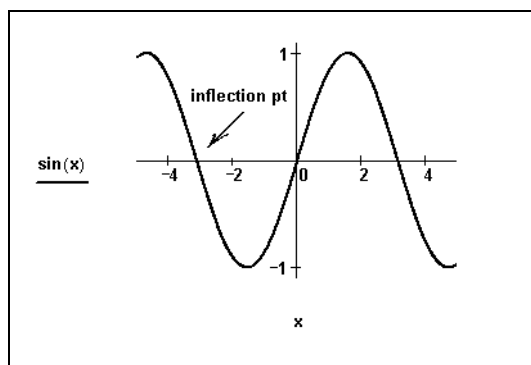


Figure 12-10: Mathcad graph with annotations.

Note If you choose **Separate Regions** from the **Format** menu, all overlapping regions in your worksheet will separate. In the case of annotated graph, such as the one shown above, all annotations move below the graph when you separate regions.

Modifying Your 2D Plot's Perspective

Mathcad provides the following options for manipulating your 2D graph:

- You can zoom in on a portion of the graph.
- You can get the x - and y -coordinates for any point that was plotted to construct an individual plot.
- You can get the x - and y -coordinates for any location within the graph.

Zooming in on a Plot

Mathcad allows you to select a region of a graph and magnify it. To zoom in on a portion of a graph, follow these steps:

1. Click in the graph and choose **Graph**⇒**Zoom** from

the **Format** menu, or click  on the Graph toolbar. The Zoom dialog box appears. The X-Y Zoom dialog box is shown to the right.



2. If necessary, reposition the Zoom dialog box so that you can see the entire region of the graph you want to zoom.
3. Click the mouse at one corner of the region in the graph you want to magnify.
4. Press and hold down the mouse button and drag the mouse. A dashed selection outline emerges from the anchor point. The coordinates of the selected region are listed in the Min and Max text boxes (or the Radius text box of the Polar Zoom dialog box).
5. When the selection outline just encloses the region you want to magnify, let go of the mouse button. If necessary, click on the selection outline, hold the mouse button down, and move the outline to another part of the graph.
6. Click Zoom to redraw the graph. The axis limits are temporarily set to the coordinates specified in the Zoom dialog box. To make these axis limits permanent, click OK.

Tip If you're working with a graph that has already been zoomed, you can restore the default appearance of the graph. To do so, click Full View in the Zoom dialog box.

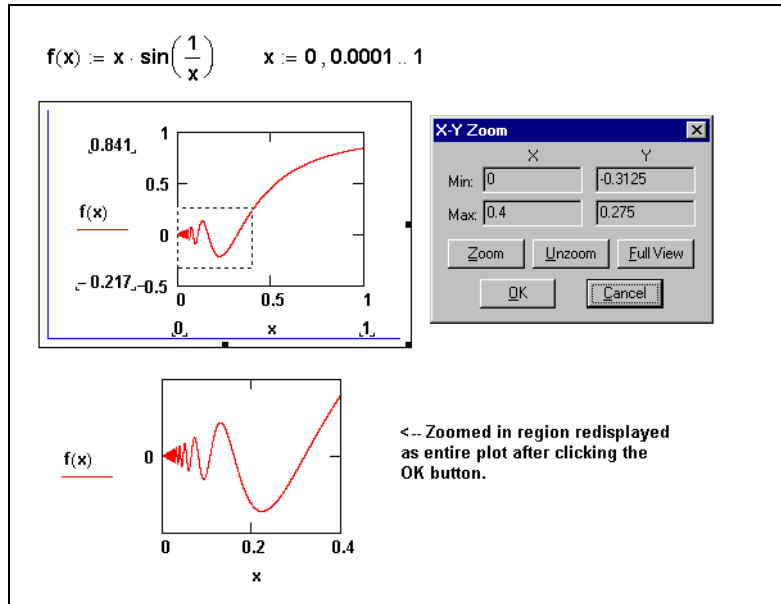
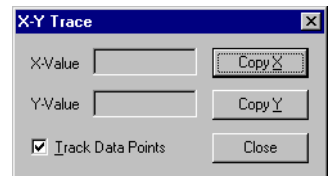


Figure 12-11: A zoomed-in region of an X-Y plot.

Getting a Readout of Plot Coordinates

To see a readout of coordinates of the specific points that make up a trace, follow these steps:

1. Click in the graph and choose **Graph**⇒**Trace** from the **Format** menu, or click  on the Graph toolbar. The X-Y Trace dialog box appears as in the example at right. Check Track Data Points if it isn't already checked. If necessary, reposition the Trace dialog box so that you can see the entire graph.
2. Click and drag the mouse along the trace whose coordinates you want to see. A dotted crosshair jumps from one point to the next as you move the pointer along the trace.
3. If you release the mouse button, you can use the left and right arrows to move to the previous and next data points. Use the up and down arrows to select other traces.
4. As the pointer reaches each point on the trace, Mathcad displays the values of that point in the X-Value and Y-Value boxes (or the Radius and Angle boxes in the Polar Trace dialog box).
5. The values of the last point selected are shown in the boxes. The crosshair remains until you click outside the plot.



Tip When Track Data Points is unchecked in the Trace dialog box, you can see a readout of coordinates for any location in a graph, not just the data points that created an individual plot.

Figure 12-12 shows an example of a plot whose coordinates are being read.

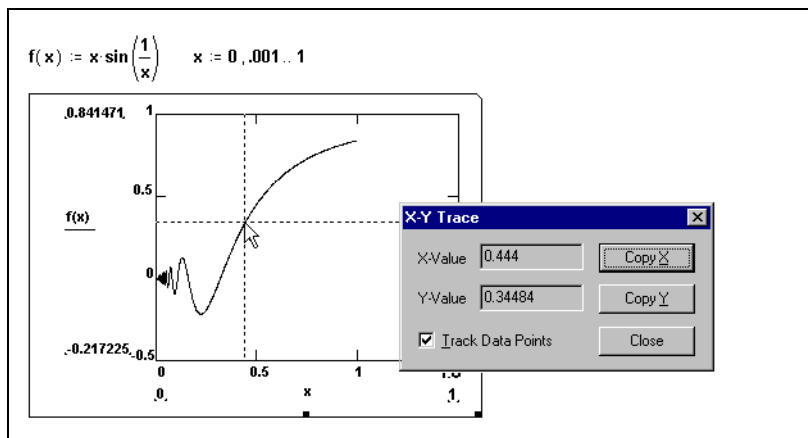


Figure 12-12: Reading coordinates from a graph.

To copy and paste a coordinate using the Clipboard:

1. Click Copy X or Copy Y (or Copy Radius or Copy Angle in the case of a polar plot).
2. You can then paste that value into a math or text region in your Mathcad worksheet, into a spreadsheet, or into any other application that allows pasting from the Clipboard.

Chapter 13

3D Plots

- τ Overview of 3D Plotting
- τ Creating 3D Plots of Functions
- τ Creating 3D Plots of Data
- τ Formatting a 3D Plot
- τ Rotating and Zooming on 3D Plots

Overview of 3D Plotting

To visually represent in three dimensions a function of one or two variables or to plot data in the form of x -, y -, and z -coordinates, you can create a surface plot, a contour plot, a 3D bar plot, a 3D scatter plot, or a vector field plot. Create these different plot types using commands from the **Insert** menu or the 3D Plot Wizard. You can also place more than one 3D plot on the same graph. Mathcad renders 3D plots with sophisticated, high performance OpenGL graphics.

Inserting a 3D Plot

In general, to create a three-dimensional plot:

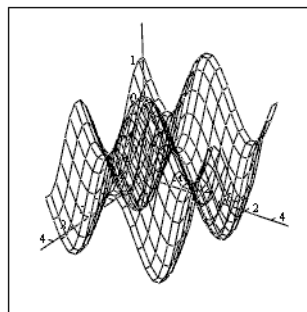
1. Define a function of two variables or a matrix of data.
2. Click in the worksheet where you want the plot to appear. Then choose **Graph** from the **Insert** menu and select a 3D plot. Alternatively, click one of the 3D graph buttons on the Graph toolbar. Mathcad inserts a blank 3D plot with axes and an empty placeholder.
3. Enter the name of the function or matrix in the placeholder.
4. Click outside the plot or press [**Enter**]. Mathcad creates the plot according to the function or matrix of data.

For example, the surface plot shown below was created in Mathcad from the function:

$$F(x, y) := \sin(x) + \cos(y)$$

When you create a 3D plot from a function, it's called a *QuickPlot*. A *QuickPlot* uses default ranges and grids for the independent variables. To change these settings, double-click on the graph and use the QuickPlot Data page of the 3D Plot Format dialog. For more information on modifying these and other plot characteristics, see “Formatting a 3D Plot” on page 234.

To learn how to create a plot from a matrix of values, see the shown in Figure 13-2 on page 229.



F

3D Plot Wizard

The *3D Plot Wizard* gives you more control over the format settings of the plot as you insert it. To use the Wizard:

1. Click in your worksheet wherever you want the graph to appear.
2. Choose **Graph**⇒**3D Plot Wizard** from the **Insert** menu. The first page of the 3D Plot Wizard appears.
3. Select the type of three-dimensional graph you want to see and click “Next.”
4. Make your selections for the appearance and coloring of the plot on subsequent pages of the Wizard. Click “Finish” and a graph region with a blank placeholder appears.
5. Enter appropriate arguments (a function name, data vectors, etc.) for the 3D plot into the placeholder.
6. Click outside the plot or press [**Enter**].

The plot is created using the settings you specified in the Wizard. For information on modifying the appearance of your plot, see “Formatting a 3D Plot” on page 234.

Creating 3D Plots of Functions

This section describes how to create various 3D plots from functions in Mathcad, also known as *QuickPlots*. Although the instructions focus on using commands on the **Insert** menu and changing settings through the 3D Plot Format dialog box, you can also use the 3D Plot Wizard, as described on page 224.

Tip To see a variety of two- and three-dimensional functions and data sets visualized in plots, open the “Practical Curves and Surfaces” section of QuickSheets in the Mathcad Resource Center (choose **Resource Center** from the **Help** menu and click on “QuickSheets”).

Creating a Surface, Bar, Contour, or Scatter Plot

You can visualize any function of two variables as a surface, bar, contour, or scatter plot in three dimensions.

Step 1: Define a function or set of functions

First, define the function in your worksheet in any one of the following forms:

$F(x, y) := \sin(x) + \cos(y)$	$G(u, v) := \begin{pmatrix} 2 \cdot u \\ 2 \cdot u \cdot \cos(v) \\ 2 \cdot \cos(v) \end{pmatrix}$	$\begin{aligned} X(u, v) &:= v \\ Y(u, v) &:= v \cdot \cos(u) \\ Z(u, v) &:= \sin(u) \end{aligned}$
--------------------------------	--	---

$F(x,y)$ is a function of two variables. In this type of function, the x - and y -coordinates of the plot vary, by default, from -5 to 5 with a step size of 0.5 . Each z -coordinate is determined by the function using these x - and y -values.

$G(u,v)$ is a vector-valued function of two variables. In this type of function, the independent variables u and v vary, by default, from -5 to 5 with a step size of 0.5 . The x -, y -, and z -coordinates are plotted parametrically according to the definitions in the three elements of the vector using these u - and v -values.

$X(u,v)$, $Y(u,v)$, and $Z(u,v)$ are functions of two variables. In this type of function triple, the independent variables u and v vary, by default, from -5 to 5 with a step size of 0.5 . The x -, y -, and z -coordinates are plotted parametrically according to the three function definitions using these u - and v -values.

Note The function descriptions above assume that you are working in Cartesian coordinates. If your function represents spherical or cylindrical, rather than Cartesian, coordinates, you can automatically convert the function to Cartesian coordinates. Double-click on the plot, go to the QuickPlot Data page of the 3D Plot Format dialog box, and click “Spherical” or “Cylindrical” under Coordinate System.

Step 2: Insert a 3D plot

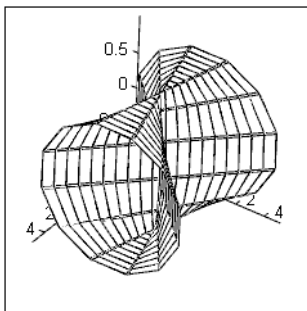
After you define a function or set of functions to plot, choose **Graph** from the **Insert** menu and select a 3D plot type.

For example, to create a surface plot from the functions X , Y , and Z , defined above:

1. Choose **Graph**⇒**Surface Plot** from the **Insert** menu or click  on the Graph toolbar. Mathcad inserts a blank 3D plot.
2. Enter the name of the functions in the placeholder. When you have more than one function definition for a single surface, separate the function names by commas and enclose the function names in parentheses. For this example, type:

(X, Y, Z)

3. Press **[Enter]** or click outside the plot.



(X, Y, Z)

To change your plot to a different plot type:

1. Double-click on the graph to bring up the 3D Plot Format dialog box.
2. In the Display As section on the General tab, select Bar Plot, Contour Plot, or Data Points from the array of plot types.
3. Click “OK.”

Figure 13-1 shows a 3D scatter plot created from the function G , and a contour plot created from the function F , both defined above:

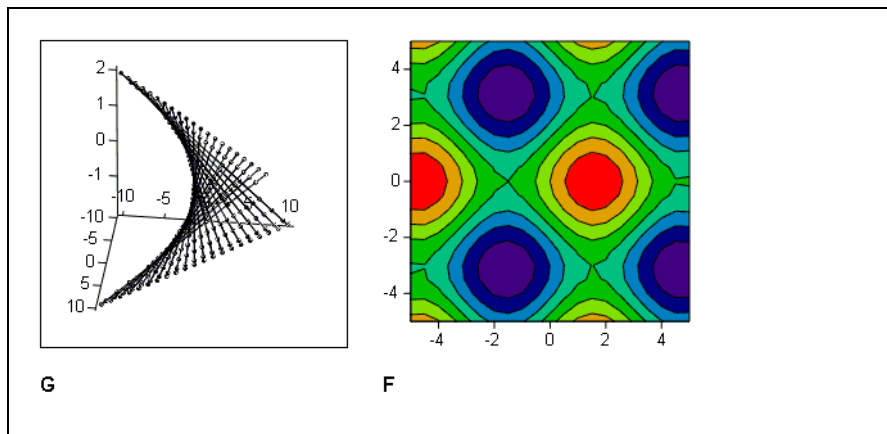


Figure 13-1: A scatter plot and a contour plot created from functions of two variables.

Note All 3D *QuickPlots* are parametric curves or surfaces. In other words, all *QuickPlots* are created from three vectors or matrices of data representing the x -, y -, and z -coordinates of the plot. In the case of a single function of two variables, Mathcad internally creates two matrices of x - and y -data over the default range -5 to 5 with a step size of 0.5, and then generates z -data using these x - and y -coordinates.

To change the default ranges and grids for the independent variables, double-click on the graph and use the QuickPlot Data page of the 3D Plot Format dialog. For more information on modifying these and other plot characteristics, see “Formatting a 3D Plot” on page 234.

Creating a Space Curve

You can visualize any parametrically-defined function of one variable as a scatter plot in three dimensions.

Step 1: Define a function or set of functions

First, define the function in your worksheet in one of the following forms:

$$H(u) := \begin{pmatrix} \sin(u) \\ \cos(u) \\ \sin(u) \cdot \cos(u) \end{pmatrix} \quad \begin{array}{l} R(u) := 2 \cdot u \\ S(u) := u^2 \\ T(u) := \cos(u) \end{array}$$

$H(u)$ is a vector-valued function of one variable. In this type of function, the independent variable u varies, by default, from -5 to 5 with a step size of 0.5 . The x -, y -, and z -coordinates of the plot are determined by the functions in each element of the vector using these u -values.

$R(u)$, $S(u)$, and $T(u)$ are functions of one variable. In this type of function triple, the independent variable u varies, by default, from -5 to 5 with a step size of 0.5 . The x -, y -, and z -coordinates are plotted according to the function definitions using these u -values.

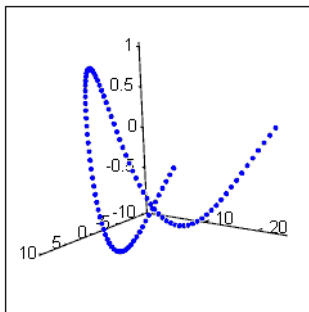
Note A space curve often represents the path of a particle in motion through space where u is a time parameter.

Step 2: Insert a 3D scatter plot

To create a space curve from a single function or set of functions:

1. Choose **Graph**⇒**3D Scatter Plot** from the **Insert** menu or click  on the Graph toolbar. Mathcad inserts a blank 3D plot.
2. Enter the name of function or functions in the placeholder. When you have more than one function definition, separate the function names by commas and enclose the function names in parentheses. To create a space curve from the functions R , S , and T , defined above, type: (R, S, T)

3. Press **[Enter]** or click outside the plot.



(R, S, T)

For general information on formatting 3D plots, refer to “Formatting a 3D Plot” on page 234. For specific information on formatting a scatter plot, refer to the topic “Scatter Plots” in the on-line Help.

Creating 3D Plots of Data

This section describes how to create various 3D plots from data in Mathcad. Although the instructions focus on using commands on the **Insert** menu and changing settings through the 3D Plot Format dialog, you can also use the 3D Plot Wizard, as described on page 224.

Creating a Surface, Bar, or Scatter Plot

Surface, bar, and scatter plots are useful for visualizing two-dimensional data contained in an array as either a connected surface, bars above and below the zero plane, or points in space.

For example, to create a surface plot from data:

1. Create or import a matrix of values to plot. The row and column numbers represent the x- and y-coordinate values. The matrix elements themselves are the z-coordinate values plotted as heights above and below the xy-plane (at $z = 0$).
2. Choose **Graph⇒Surface Plot** from the **Insert** menu or click  on the Graph toolbar. Mathcad inserts a blank 3D plot.
3. Enter the name of the matrix in the placeholder.
4. Press **[Enter]** or click outside the plot. Figure 13-2 shows a 3D bar plot created from a matrix, M:

In the default perspective, the first row of the matrix extends from the back left corner of the grid to the right, while the first column extends from the back left corner out toward the viewer. See “Formatting a 3D Plot” on page 234 to learn how to change this default view.

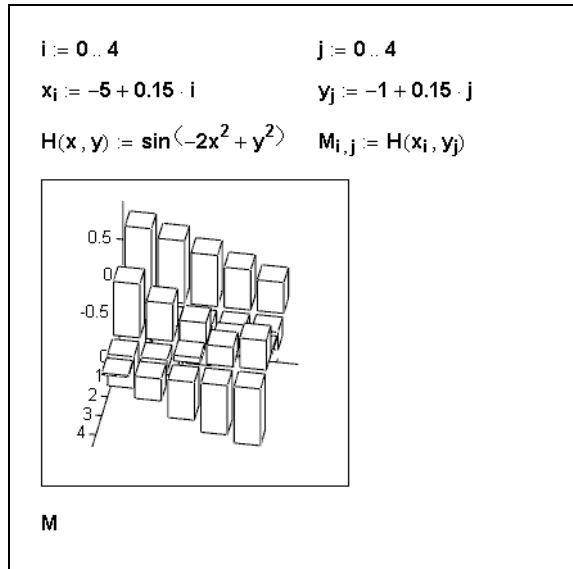


Figure 13-2: Defining a matrix of data and plotting it as a 3D bar plot.

Creating a Parametric Surface Plot

A parametric surface plot is created by passing three matrices representing the x -, y -, and z - coordinates of your points in space to the surface plot.

To create a parametric surface plot:

1. Create or import three matrices having the same number of rows and columns.
2. Choose **Graph**⇒**Surface Plot** from the **Insert** menu or click  on the Graph toolbar. Mathcad inserts a blank 3D plot.
3. Type the names of the three matrices separated by commas and enclosed in parentheses in the placeholder. For example:
 (X, Y, Z)
4. Press [**Enter**] or click outside the plot.

Figure 13-3 shows a parametric surface plot created from the matrices, X , Y , and Z , defined above the plot.

For general information on formatting 3D plots, refer to “Formatting a 3D Plot” on page 234. For specific information on formatting a parametric surface plot, refer to the topic “Surface Plots” in the on-line Help.

Note The underlying parameter space is a rectangular sheet covered by a uniform mesh. In effect, the three matrices map this sheet into three-dimensional space. For example, the matrices X , Y , and Z defined in Figure 13-3 carry out a mapping that rolls the sheet into a tube and then joins the ends of the tube to form a torus.

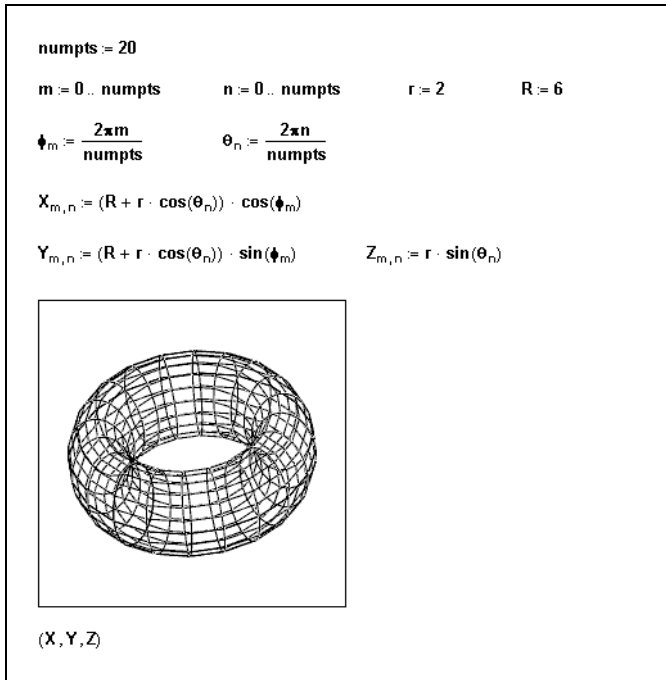


Figure 13-3: Defining data for a parametric surface plot.

Creating a Three-dimensional Parametric Curve

A three-dimensional parametric curve is created by passing three vectors representing the x-, y-, and z-coordinates of your points in space to the surface plot.

To create a three-dimensional parametric curve:

1. Create or import three vectors having the same number of rows.
2. Choose **Graph⇒Scatter Plot** from the **Insert** menu or click  on the Graph toolbar. Mathcad inserts a blank 3D plot.
3. Type the names of the three vectors separated by commas and enclosed in parentheses in the placeholder. For example:
(X, Y, Z)
4. Press **[Enter]** or click outside the plot.

Figure 13-4 shows a three-dimensional parametric curve created from the vectors, P, Q, and R, defined above the plot:

For general information on formatting 3D plots, refer to “Formatting a 3D Plot” on page 234. For specific information on formatting a scatter plot, refer to the topic “Scatter Plots” in the on-line Help.

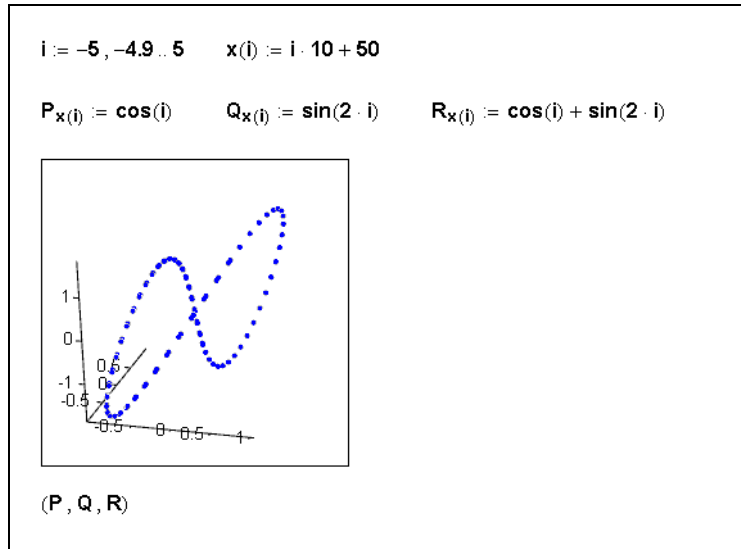


Figure 13-4: Defining data for a space curve.

Creating a Contour Plot

To view three-dimensional data as a two-dimensional contour map, you can create a contour plot:

1. Define or import a matrix of values to plot.
2. Choose **Graph**⇒**Contour Plot** from the **Insert** menu or click  on the Graph toolbar. Mathcad shows a blank plot with a single placeholder.
3. Type the name of the matrix in the placeholder.
4. Press [**Enter**] or click outside the plot.

Figure 13-5 shows a contour plot created from the matrix, C, defined above the plot:

The contour plot is a visual representation of the matrix's level curves. Mathcad assumes that the rows and columns represent equally spaced intervals on the axes, and then linearly interpolates the values of this matrix to form level curves or contours. Each level curve is formed such that no two cross. By default, the z-contours are shown on the x-y plane. Mathcad plots the matrix such that the element in row 0 and column 0 is in the lower left corner. Thus the rows of the matrix correspond to values on the x-axis, increasing to the right, and the columns correspond to values along the y-axis, increasing toward the top.

For general information on formatting 3D plots, refer to “Formatting a 3D Plot” on page 234. For specific information on formatting a contour plot, refer to the topic “Contour Plots” in the on-line Help.

Note If you create a contour plot of a function as described above, the positive x-axis of the plot extends to the right and the positive y-axis extends toward the top of the plot.

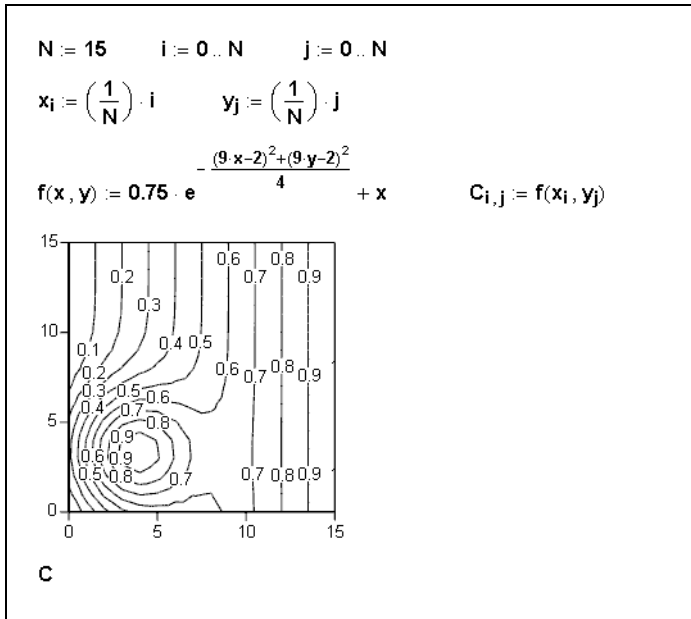


Figure 13-5: Defining data for a contour plot.

Creating a Vector Field Plot

In a vector field plot, each point in the x - y plane is assigned a two-dimensional vector. There are two ways to set up the data needed for a vector field plot:

1. Create a matrix of complex numbers in which the following conditions exist:
 - The row and column numbers represent the x - and y -coordinates
 - The real part of each matrix element is the x -component of the vector associated with that row and column
 - The imaginary part of each element is the y -component of the vector associated with that row and column.
2. Create two matrices having the same number of rows and columns. The first matrix should have the x -components of the vectors, the second the y -components.

Once you have defined your data, as described above, to create a vector field plot:

1. Choose **Graph⇒Vector Field Plot** from the **Insert** menu or click  on the Graph toolbar.
2. Type the name(s) of the matrix or matrices in the placeholder. If you have more than one matrix for a vector field plot, separate the matrix names by commas and enclose the matrix name set in parentheses. For example:

(X, Y)

3. Press **[Enter]** or click outside the plot.

Figure 13-6 shows a vector field plot created from the matrix, Q , defined above the plot:

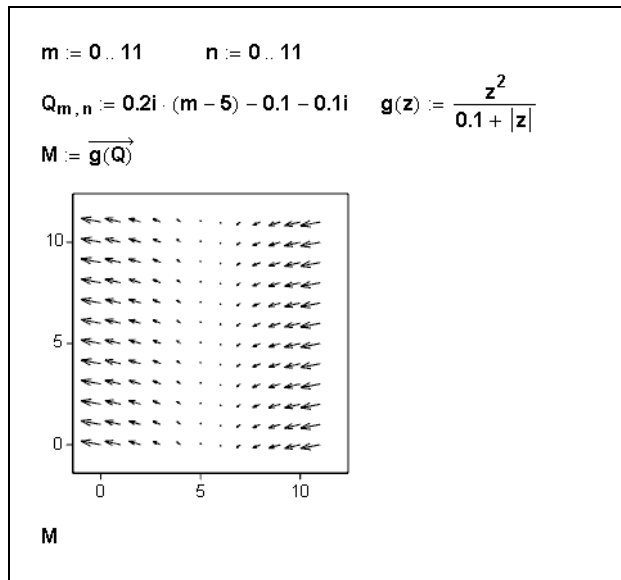


Figure 13-6: Defining data for a vector field plot.

For general information on formatting 3D plots, refer to “Formatting a 3D Plot.” For specific information on formatting a vector field plot, refer to the topic “Vector field plots” in the on-line Help.

Graphing Multiple 3D Plots

Pro Just as you can plot more than one trace on a two-dimensional graph, you can place more than one surface, curve, contour, bar, or scatter plot on a three-dimensional graph. For example, to create a 3D graph with a contour plot and a surface plot:

1. Define two functions of two variables or any combination of two acceptable argument sets for a 3D plot (two matrices, two sets of three vectors, etc.).
2. Choose **Graph**⇒**Contour Plot** from the **Insert** menu or click  on the Graph toolbar. Mathcad inserts a blank 3D plot.
3. Enter the name of the function or matrix for the contour plot into the placeholder. Then type , (a comma).
4. Enter the name of the function or matrix for the surface plot.
5. Press [**Enter**] or click outside the plot. You see two contour plots.
6. Double-click the graph to bring up the 3D Plot Format dialog box. In the Display As section of the General tab, click the tab labeled Plot 2 and select Surface from the array of plot types. Click “OK.”

Both the contour plot and the surface plot, with default format settings, appear in a single graph.

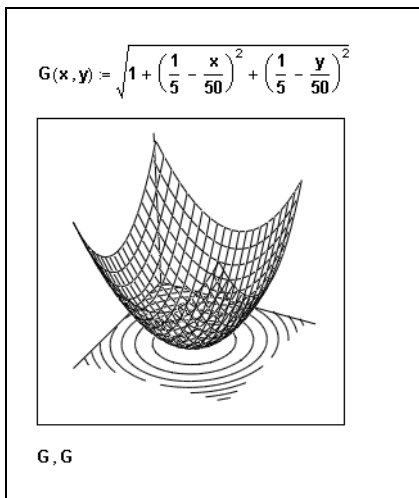


Figure 13-7: Two plots, one contour and one surface, shown on the same graph.

Tip As a general rule, you will not want to create a 3D graph with more than two or three plots together since they may obscure each other and make the graph difficult to interpret.

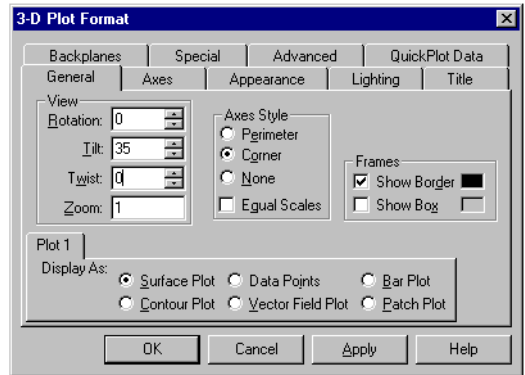
Formatting a 3D Plot

A three-dimensional plot's default appearance depends on how you insert it. When you choose **Graph**⇒**3D Plot Wizard** from the **Insert** menu, you make selections in the pages of the Wizard that determine a plot's appearance. When you insert a plot by choosing a plot type from the **Insert** menu, however, the plot automatically acquires default characteristics.

You can change the appearance of any 3D plot after it is inserted. To do so, you use the many options available in the 3D Plot Format dialog box. For example, you can use the options to change a plot's color, format the axes, add backplanes, and format the lines or points.

To bring up the 3D Plot Format dialog box:

1. Click once on the plot to select it and choose **Graph⇒3D Plot** from the **Format** menu. Alternatively, double-click the plot itself. Mathcad brings up the 3D Plot Format dialog box. The General page is shown at right. The remaining tabs take you to additional pages.
2. Click the tab for the page you want to work with.
3. Make the appropriate changes in the dialog box.
4. Click Apply to see the effect of your changes *without* closing the dialog box.
5. Close the dialog by clicking OK.



The 3D Plot Format Dialog Box

The tabs in the 3D Plot Format dialog box bring you to pages containing options for formatting various aspects of a three-dimensional plot. Some options available on certain pages in the dialog box depend on the kind of plot you are formatting. Options on other pages are available for any three-dimensional graph.

- The **General** page gives you access to basic options that control the overall appearance of the graph. Use these options to control the position of a plot, set the axis style, draw a border or a box, or convert a plot to another type.
- The options on the **Axes** page allow you to control exactly how each axis looks. You can specify the weight of each axis and whether it has numbers or tick marks. You can also specify the axis limits. Use the tabs at the top of the page to format the *x*-, *y*-, or *z*-axis.
- The **Backplanes** page has options for specifying whether a backplane is filled with a color, has a border, or has grid lines or tick marks. Use the tabs at the top of the page to format the *xy*-, *yz*-, or *xz*-backplane.

Note Both the Backplanes page and the Axes page have options for setting and formatting grid lines. When you set the grid lines for an axis on the Axes page, you set them for the two backplanes shared by the axis. When you set the grid lines on the Backplanes tab, you set them for one backplane only.

- Use the options on the **Appearance** page to format the surfaces, lines, and points that make up a plot. For example, you can apply color directly to a plot's surface, its contours, or its lines and points. The following sections discuss how to control the surfaces, lines, and points of a plot.
- The **Lighting** page options control both the overall lighting of the plot as well as individual lights directed onto it. See "Lighting" on page 240 for more information on lighting.

- The **Title** page provides a text box for entering a title for the graph and options for specifying the location of the title on the graph.
- The **Special** page allows you to control options related to specific kinds of plots. For example, the Bar Plot Layout options let you specify the way the bars are arranged in a 3D bar plot.
- The **Advanced** page has options used only when you need very fine control over the appearance of a plot, such as the vertical scale.
- The **QuickPlot Data** page contains the range and grid settings for the independent variables that control a 3D QuickPlot. Additionally, you can specify whether your function(s) are in Cartesian, spherical, or cylindrical coordinates.

On-line Help For details on the options available on a particular page in the 3D Plot Format dialog box, click the Help button at the bottom of the dialog box.

Some options in the 3D Plot Format dialog box work together to control the appearance of a plot. For example, the choices on the Appearance page, the Lighting page, and the Special and Advanced pages together control the color of a plot.

Note When you format a graph containing more than one plot (using Mathcad Professional), as described in “Graphing Multiple 3D Plots” on page 233, some options in the 3D Plot Format dialog box apply to an entire graph while others apply to individual plots. For example, all the options on the Axes, Backplanes, and Lighting pages are for the graph as a whole: each plot on the graph uses common axes, backplanes, and lighting. However, options on the Appearance tab are specific to each plot on the graph. That is, each plot can be filled with its own color, have its own lines drawn, etc. Use the tabs labeled Plot 1, Plot 2, etc. to control the settings for individual plots.

Fill Color

The color of a plot is primarily determined by its fill color. This section describes the ways to apply color to a plot by filling its surfaces or contours. A plot’s color and shading are also affected by *lighting*, as described in more detail in page 240.

Mathcad allows you to apply either a solid color or a colormap to the surface or contours of a plot. A solid color is useful when you don’t want to overcomplicate a plot with many colors or when you want to use lighting to shade a plot. A colormap applies an array of color to a plot according to its coordinates.

Note Mathcad comes with a variety of colormaps for applying rainbow colors and shades of gray, red, green, and blue. You can also create and load custom colormaps in Mathcad Professional by using the *SaveColormap* and *LoadColormap* functions, described on page 190. By default, a colormap is applied in the direction of the z-values, or according to the height of the plot. You can apply the colormap in the direction of the x-values or y-values by clicking the Advanced tab and choosing a direction in the Colormap section. For more information on colormaps, see on-line Help.

The options on the Appearance page of the 3D Plot Format dialog box allow you to fill the plot's surface with a solid color or a colormap. For example, to color the bars in a 3D bar plot according to a colormap:

-
- 3-D Plot Format**
- Backplanes | Special | Advanced | QuickPlot Data
- General | Axes | Appearance | Lighting | Title
- Plot 1
- Fill Options**
- ☒ Fill Surface
 - ☐ Fill Contours
 - ☐ No Fill
 - ☐ Alternate Mesh
 - ☒ Smooth Shading
- Line Options**
- ☐ W/eframe
 - ☐ Contour Lines
 - ☒ No Lines
 - ☐ Hide Lines
 - 1 Weight
- Point Options**
- ☐ Draw Points
 - Symbol: dots
 - 1 Size
- Color Options**
- ☐ Colormap
 - ☒ Solid Color
- OK Cancel Apply Help

Figure 13-8 shows an example.

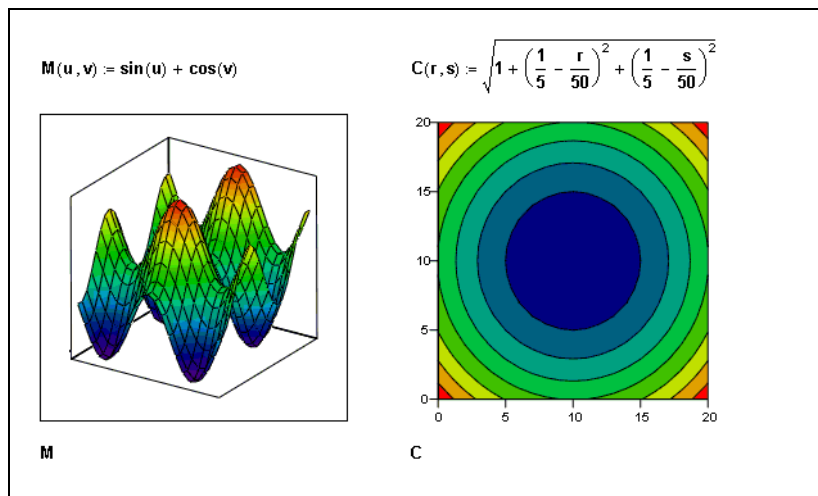


Figure 13-8: Filling the surface or contours of a plot.

The plot is shaded using the default colormap “Rainbow.” To choose a different colormap, click the Advanced tab of the 3D Plot Format dialog box and select a colormap from the Choose Colormap drop-down menu.

If you wanted to fill the bars of the plot with a solid color, choose Solid Color instead of Colormap and click the color box next to Solid Color to select a color.

When you format a surface plot, you can choose Fill Contours instead of Fill Surface in the Fill Options section of the Appearance page. If you fill the contours of a surface plot, the plot is filled according to its contours rather than directly by its data. You can

fill according to the x -, y -, or z -contours or two at the same time. For a contour plot, you must choose Fill Contours instead of Fill Surface to fill the contours of the plot.

For example, to fill a contour plot with color:

1. Double-click the graph to bring up the tabbed dialog box.
2. Click the Appearance tab.
3. In the Fill Options section, click Fill Contours.
4. Click Apply to preview the plot. Click OK to close the dialog box.

The plot is shaded using the default colormap Rainbow. To choose a different colormap, click the Advanced tab of the 3D Plot Format dialog box and select a colormap from the Choose Colormap drop-down menu.

Note If you have a contour plot projected on a plane other than the x - y plane, you can fill the contour using options on the Special page of the 3D Plot Format dialog box. To do so, click the Special tab, then choose a contour direction from the drop-down menu. Click Fill for each contour you want to color. For example, if you have Fill checked for the z -contours and x -contours, you will see contour color on both the x - y backplane and the y - z backplane.

Lines

Mathcad provides many ways to control the appearance of the lines on a three-dimensional plot. You can draw the lines so they form a wireframe, or you can draw only the contour lines. You can also control the weight and color of the lines on a plot.

Drawing a Wireframe

To control whether lines form a wireframe on a plot, use the options on the Appearance page of the 3D Plot Format dialog box. For example, to remove the wireframe on a surface plot as shown in Figure 13-9:

1. Double-click the graph to bring up the tabbed dialog box.
2. Click the Appearance tab.
3. In the Line Options section, click No Lines.
4. Click Apply to preview the plot. Click OK to close the dialog box.

To turn lines on again later, choose Wireframe on the Appearance page.

Drawing Contour Lines

When you format a surface plot, you can choose Contour instead of Wireframe in the Line Options section of the Appearance page. Contour lines are those drawn according to the contours of a surface. You can draw either the x -, y -, or z - contour lines, two of these contours lines, or all three.

Note For contour plots, Mathcad always chooses Contour instead of Wireframe to draw contour lines.

For example, to draw lines showing the x -contours of a surface plot:

1. Double-click the graph to bring up the tabbed dialog box.
2. Click the Appearance tab.
3. Click Contour in the Line Options section.

4. Click the Special tab.
5. Verify that Z-Contours is selected in the drop-down menu at the bottom of the Contour Options section. Click Draw Lines to remove the check mark. This turns lines off for the z-contours.
6. Choose Z-Contours from the drop-down menu on the Special page.
7. Check Draw Lines.
8. Click Apply to preview the plot. Click OK to close the dialog box.

The surface plot is drawn with contour lines perpendicular to the z-axis, as shown in Figure 13-9.

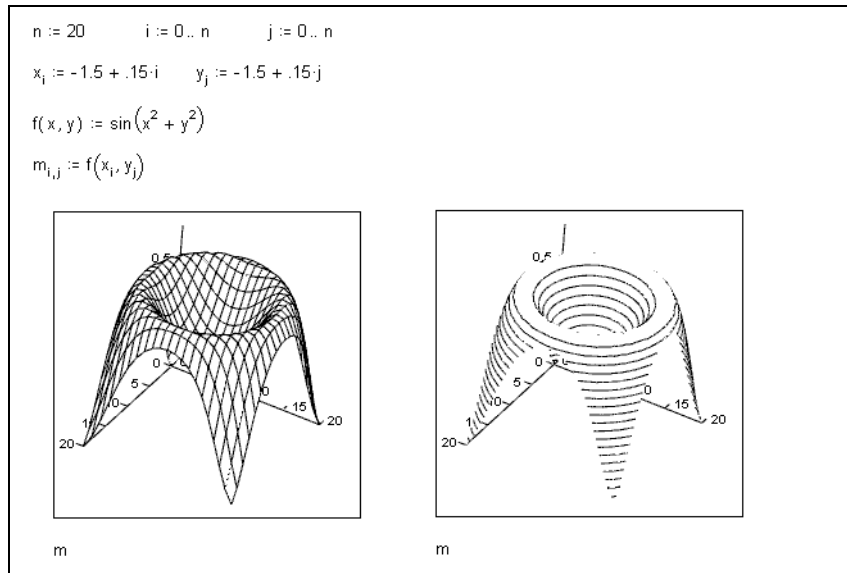


Figure 13-9: A wireframe vs. contour lines on a surface plot.

Note When you format a contour plot on a multi-plot graph (see page 224), the options in the drop-down menu on the Special tab determine on which backplane the contour lines are drawn. For example, if you have Draw Lines checked for the z-contours and x-contours, you will see contour lines on both the x-y backplane and the y-z backplane.

Line Color

You can control the color of the lines in a plot using the color options in the Line Options section of the Appearance page. Just as you can fill a plot's surface with a colormap or a solid color, described on page 237, you can also apply a colormap or solid color to the lines in a plot.

For example, to make the lines of a contour plot orange:

1. Double-click the graph to bring up the tabbed dialog box.
2. Click the Appearance tab.

3. In the Line Options section, click Contour to draw contour lines and Solid Color.
4. Click the color box next to Solid Color, click the orange box, and click OK.
5. Click Apply to preview the plot. Click OK to close the dialog box.

Points

You can draw and format points on most three-dimensional plots, since all 3D plots are constructed from discrete data points. (The exceptions are vector field plots, contour plots, bar plots, and patch plots.) Points are most useful, however, on a 3D scatter plot in which points are the main focus of the plot. Mathcad allows you to control the symbol used for the points in a plot as well as the color and size of the symbol.

To draw or remove points on a surface plot:

1. Double-click the graph to bring up the 3D Plot Format dialog box.
2. Click the Appearance tab.
3. In the Points Options section, check (or uncheck) Draw Points.
4. Click Apply to preview the plot. Click OK to close the dialog box.

To format the symbol, color, and size of the points on your 3D scatter plot using the Points Options section of the Appearance tab:

- Choose a Symbol from the drop-down list to change the symbol displayed.
- Use the arrows next to Size to increase or decrease the size of the symbol.
- Click the color box next to Solid Color and choose a hue from the color palette, or click Colormap to change the coloring of the symbols.

Lighting

The color of a three-dimensional plot is a result of color you use to fill its surface, lines, and points as well as the color of any ambient light or directed lights shining on it. This behavior is identical to the effect of light on object color in the real world. Objects reflect and absorb light depending on their color. For example, a yellow ball reflects mostly yellow light and absorbs others, and it can look grayish under dim lighting, green under blue lighting, and bright yellow in bright lighting.

You can fill a plot's surfaces, contours, lines, and points with either a solid color or a colormap using the options on the Appearance and Advanced pages of the 3D Plot Format dialog box.

Light is controlled using the options on the Lighting page of the 3D Plot Format dialog box. If you are content to fill a plot with a colormap, you may not need to use lighting at all. However, if you want to shade the plot differently, or if you fill the plot with a solid color and want to shade it, you can enable lighting.

Note If your 3D graph contains multiple plots, lighting affects all the plots in a graph, but you can fill individual plots with color independently.

Note If you want lighting to be the sole determinant of the color of a plot, use the Appearance page options in the 3D Plot Format dialog box to fill the plot with solid white.

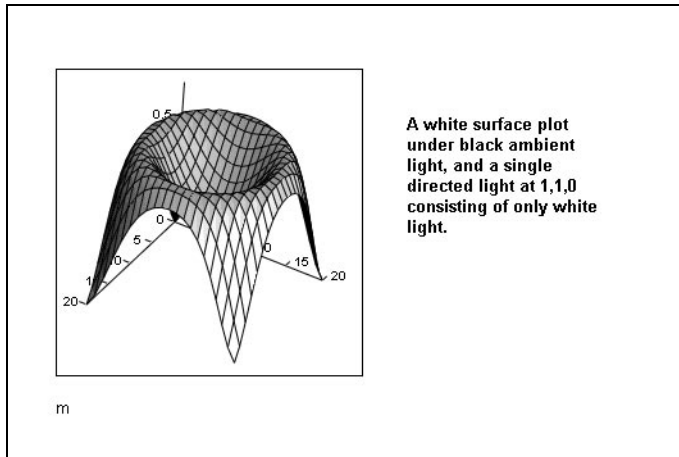


Figure 13-10: A white surface plot with lighting enabled.

To enable lighting:

1. Double-click the plot to open the tabbed dialog box.
2. Click the Lighting tab.
3. Check Enable Lighting in the Lighting section.
4. Click the options on tabs labeled Light 1, Light 2, etc. to enable a directed light and set its color and location. Mathcad lets you set up to eight directed lights.
5. Click the Ambient Light Color box to set the ambient light color. Note that black corresponds to no ambient light.
6. Click Apply to preview the plot. Click OK to close the dialog box.

On-line Help For details on the options available on the Lighting page, click the Help button at the bottom of the dialog box. For additional information on lighting, see “Advanced Topics” under Overview and Tutorial in the Mathcad Resource Center.

Changing One 3D Plot to Another

You can change almost any three-dimensional plot into another kind of three-dimensional plot by using the Display As options on the General tab in the 3D Plot Format dialog box. Simply select another available 3D plot type and click Apply or OK to change the plot instantaneously to another type. Figure 13-11 shows the same matrix displayed as three different plot types.

Note Some three-dimensional plots cannot be converted to other forms. For example, you cannot convert a vector field plot into any other kind of plot. If a plot cannot be converted to another kind of plot, that plot type is grayed in the 3D Plot Format dialog box.

Annotations

In addition to adding a title to your three-dimensional plot by using options on the Title page of the 3D Plot Format dialog box, you can annotate a three-dimensional plot by

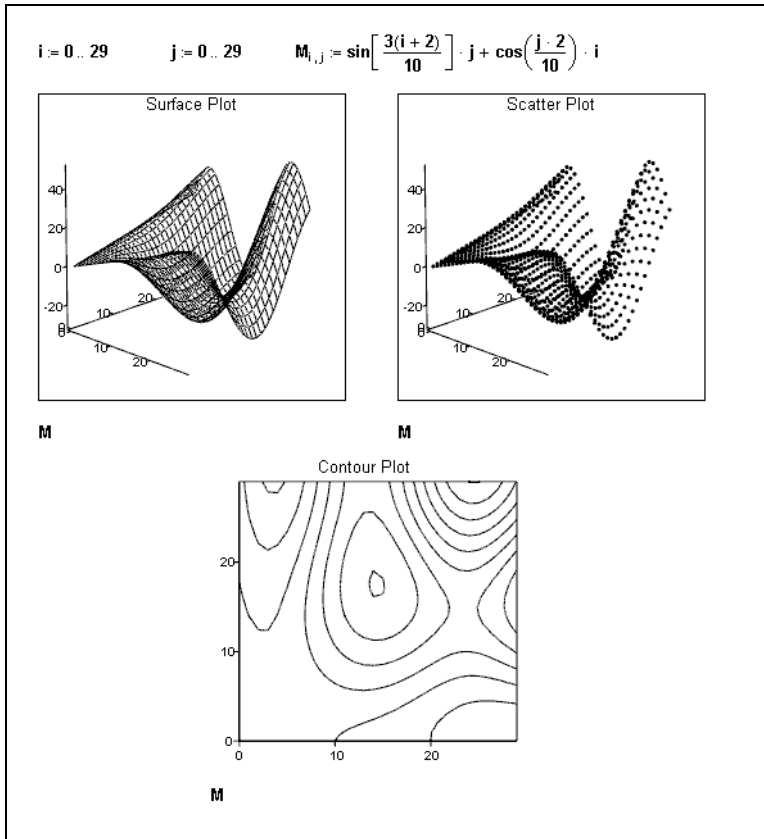


Figure 13-11: The same data displayed in several different 3D plots.

placing text or bitmaps anywhere on it. This allows you to label or highlight any part of the plot that you wish.

To add a text annotation to a three-dimensional plot:

1. Create a text region in your worksheet using the methods described in Chapter 5, “Working with Text.”
2. Drag the text region from its location in your worksheet and drop it directly onto the plot. See “Moving and Copying Regions” on page 10 for more on dragging and dropping regions.

You can select the text annotation on your plot to reposition it. To edit a text annotation on a plot, select the text and drag it off the plot to your worksheet. You can now edit the text region. Then drag the text region back onto the plot.

Tip You can drag a bitmap image from your Mathcad worksheet onto a three-dimensional plot just as you drag and drop text annotations. To place a bitmap you created in another application onto a three-dimensional plot, copy the bitmap from the other application to the Clipboard, click on the plot with the right mouse button, and choose **Paste Special** from the pop-up menu.

Modifying 3D QuickPlot Data

When you create a 3D QuickPlot, as described on page 224, you can change the range and step size of each independent variable by using the settings on the QuickPlot Data page of the 3D Plot Format dialog box.

To change the range of either independent variable:

1. Set the start and end values of either range using the text boxes for each range.
2. Click Apply to keep the dialog box open. Click OK to close the dialog box.

To change the step size, the number of grids generated along each variable's axis between the start and end values:

1. Use the arrows next to # of Grids for each range to increase or decrease the grid value. Alternatively, you can type in a value in the text box.
2. Click Apply to keep the dialog box open. Click OK to close the dialog box.

Note The ranges you set for the independent variables in the QuickPlot Data page do not necessarily control the axis limits of the plot, unless you are plotting a single function of two variables in Cartesian coordinates. In all other cases, the axis limits are determined by the x -, y -, and z -data generated for the QuickPlot by your function(s).

To perform automatic coordinate system conversions on your QuickPlot data:

1. Click the radio button under Coordinate System corresponding the coordinate system of the function you are plotting.
2. Click Apply to keep the dialog box open. Click OK to close the dialog box.

Rotating and Zooming on 3D Plots

You can resize a three-dimensional plot using the same methods you use to resize any graph region in Mathcad. Click on it and use the handles that appear along the edges to drag out the edges. Mathcad provides several additional options for manipulating the presentation of your 3D plot:

- You can rotate the plot to see it from a different perspective.
- You can set the plot in motion about an axis of rotation so that it spins continuously.
- You can zoom in or out on a portion of the plot.

Note When you rotate, spin, or zoom a three-dimensional plot, any visible axes move or resize themselves with the plot. Text or graphic annotations you add to the plot (see page 241) remain anchored at their original sizes and positions.

Rotating a Plot

You can rotate a plot interactively with the mouse or by specifying parameters in the 3D Plot Format dialog box.

To rotate a three-dimensional plot interactively by using the mouse:

1. Click in the plot, and hold the mouse button down.

2. Drag the mouse in the direction you want the plot to turn.
3. Release the mouse button when the plot is in the desired position.

To rotate a three-dimensional plot by using the 3D Plot Format dialog box:

1. Click once on the plot to select it and choose **Graph⇒3D Plot** from the **Format** menu. Alternatively, double-click the plot.
2. Click the General tab.
3. Edit the settings for Rotation, Tilt, and Twist in the View options.
4. Click Apply to preview the plot. Click OK to close the dialog box.

Spinning a Plot

You can set a plot in motion so that it spins continuously about an axis of rotation:

1. Click in the plot, and hold the **[Shift]** key and the mouse button down.
2. Drag the mouse in the direction you want the plot to spin.
3. Release the mouse button to set the plot in motion.

The plot spins continuously until you click again inside the plot.

Note If you make changes to equations that affect a plot, the plot recomputes even when it is spinning!

Tip To create an AVI file of a spinning plot, see the techniques in “Animation” on page 119.

Zooming a Plot

You can zoom in or out of a plot interactively or by specifying a zoom factor in the 3D Plot Format dialog box.

To zoom in on a three-dimensional plot by using the mouse:

1. Click in the plot, and hold the **[Ctrl]** key and the mouse button down.
2. Drag the mouse toward the top of the plot to zoom out, or drag the mouse toward the bottom to zoom in.
3. Release the mouse button when the plot is at the desired zoom factor.

Tip If you use an IntelliMouse-compatible mouse with a center wheel, you can rotate the wheel to zoom in or out of a three-dimensional plot.

To zoom in or out of a three-dimensional plot by using the 3D Plot Format dialog box:

1. Click once on the plot to select it and choose **Graph⇒3D Plot** from the **Format** menu. Alternatively, double-click the plot.
2. Click the General tab.
3. Edit the Zoom setting in the View options.
4. Click Apply to preview the plot. Click OK to close the dialog box.

Chapter 15

Programming

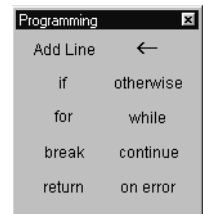
Pro

- τ Defining a Program
- τ Conditional Statements
- τ Looping
- τ Controlling Program Execution
- τ Error Handling
- τ Programs Within Programs

Defining a Program

A Mathcad program is a special kind of expression you can create in Mathcad Professional. It's an expression made up of a sequence of statements created using *programming operators*,

available on the Programming toolbar. Click  on the Math toolbar, or choose **Toolbars**⇒**Programming** from the **View** menu, to open the Programming toolbar.



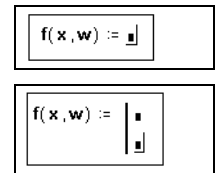
You can think of a program as a compound expression that involves potentially many programming operators. Like any expression, a program returns a value—a scalar, vector, array, nested array, or string—when followed by the equal sign or the symbolic equal sign. Just as you can define a variable or function in terms of an expression, you can also define them in terms of a program.

The following example shows how to make a simple program to define the function:

$$f(x, w) = \log\left(\frac{x}{w}\right)$$

Although the example chosen is simple enough not to require programming, it illustrates how to separate the statements that make up a program and how to use the local assignment operator, “←.”

1. Type the left side of the function definition, followed by a “:=”. Make sure the placeholder is selected.
2. Click  on the Programming toolbar. Alternatively, press **]**. You'll see a vertical bar with two placeholders, which will hold the statements that comprise your program.



- Click in the top placeholder. Type **z**, then click  on the Programming toolbar. Alternatively, press **{** to insert a “←,” which is also known as the local definition symbol.
- Type **x/w** in the placeholder to the right of the local definition symbol. Then press **[Tab]** to move to the bottom placeholder, or click on the bottom placeholder.
- Enter the value to be returned by the program in the remaining placeholder. Type **log(z)**.

$$f(x, w) := \left| \begin{array}{l} z \leftarrow \\ \end{array} \right|$$

$$f(x, w) := \left| \begin{array}{l} z \leftarrow \frac{x}{w} \\ \end{array} \right|$$

$$f(x, w) := \left| \begin{array}{l} z \leftarrow \frac{x}{w} \\ \log(z) \end{array} \right|$$

You can now use this function just as you would any other function in your worksheet.

Note You cannot use Mathcad’s usual assignment operator, “:=,” inside a program. You must use the local assignment operator, represented by “←,” instead. Variables defined inside a program with the local assignment operator, such as *z* in the example above, are local to the program and are undefined elsewhere in the worksheet. However, within a program, you can refer to Mathcad variables and functions defined previously in the worksheet.

Figure 15-1 shows a more complex example involving the quadratic formula. Although you can define the quadratic formula with a single statement as shown in the top half of the figure, you may find it easier to define it with a series of simple statements as shown in the bottom half.

$q(a, b, c) := \frac{-b + \sqrt{b^2 - 4 \cdot a \cdot c}}{2 \cdot a}$	Although you can define complicated functions all on one line...
$r(a, b, c) := \left \begin{array}{l} \text{discr} \leftarrow b^2 - 4 \cdot a \cdot c \\ \text{num} \leftarrow -b + \sqrt{\text{discr}} \\ \text{denom} \leftarrow 2 \cdot a \\ \frac{\text{num}}{\text{denom}} \end{array} \right $	...it's sometimes easier to break them up into simpler steps anyway.

Figure 15-1: A more complex function defined in terms of both an expression and a program.

Tip A program can have any number of statements. To add a statement, click  on the Programming toolbar or press **[]**. Mathcad inserts a placeholder below whatever statement you’ve selected. To delete the placeholder, click on it and press **[Bksp]**.

As with any expression, a Mathcad program must have a value. This value is simply the value of the last statement executed by the program. It can be a string expression, a single number, or an array of numbers. It can even be an array of arrays (see “Nested Arrays” on page 204).

You can also write a Mathcad program to return a *symbolic* expression. When you evaluate a program using the symbolic equal sign, “ $\rightarrow$,” described in Chapter 14, “Symbolic Calculation,” Mathcad passes the expression to its symbolic processor and, when possible, returns a simplified symbolic expression. You can use Mathcad’s ability to evaluate programs symbolically to generate complicated symbolic expressions, polynomials, and matrices. Figure 15-2 shows a function that, when evaluated symbolically, generates symbolic polynomials.

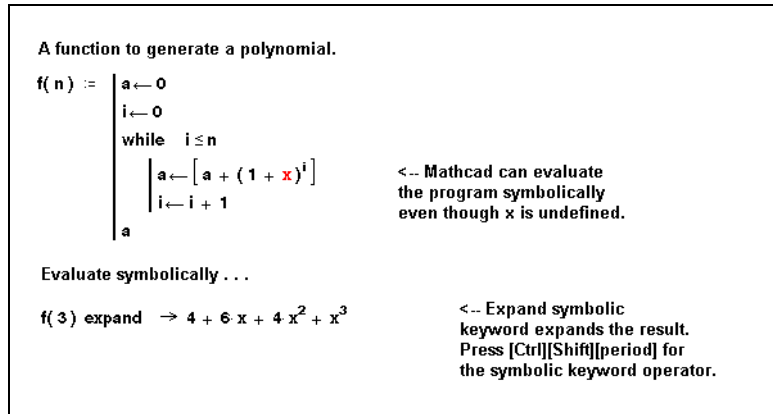


Figure 15-2: Using a Mathcad program to generate a symbolic expression.

Note Programs that include the **return** and **on error** statements, described on page 273 and page 275, cannot be evaluated symbolically since the symbolic processor does not recognize these operators.

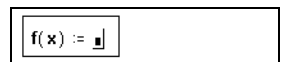
On-line Help For programming examples, see the “Programming” section in the Resource Center QuickSheets. The Resource Center also includes a special section, “The Treasury Guide to Programming,” which provides detailed examples and applications of Mathcad programs.

Conditional Statements

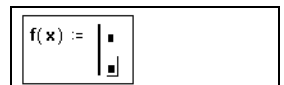
In general, Mathcad evaluates each statement in your program from the top down. There may be times, however, when you want Mathcad to evaluate a statement only when a particular condition is met. You can do this by including an **if** statement.

For example, suppose you want to define a function that forms a semicircle around the origin but is otherwise constant. To do this:

1. Type the left side of the function definition, followed by a “:=”. Make sure the placeholder is selected.



2. Click **Add Line** on the Programming toolbar. Alternatively, press **]**. You’ll see a vertical bar with two



placeholders. These placeholders will hold the statements making up your program.

- Click **if** on the Programming toolbar in the top placeholder. Alternatively, press **]**. Do not type “if.”
- Enter a Boolean expression in the right placeholder using one of the relational operators on the Boolean toolbar. In the left placeholder, type the value you want the program to return whenever the expression in the right placeholder is true. If necessary, add more placeholders by clicking **Add Line**.
- Select the remaining placeholder and click **otherwise** on the Programming toolbar or press **[Ctrl] 3**.
- Type the value you want the program to return if the condition in the first statement is false.

$$f(x) := \begin{cases} \text{if } \end{cases}$$

$$f(x) := \begin{cases} 0 & \text{if } |x| > 2 \\ \sqrt{4 - x^2} & \text{otherwise} \end{cases}$$

Figure 15-3 shows a plot of this function.

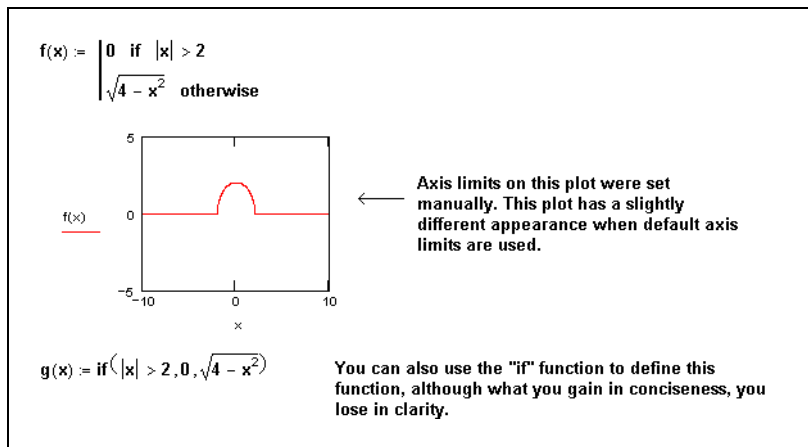


Figure 15-3: Using the **if** statement to define a piecewise continuous function.

Note The **if** statement in a Mathcad program is not the same as the *if* function (see “Piecewise Continuous Functions” on page 148). Although it is not hard to define a simple program using the *if* function, as shown in Figure 15-3, the *if* function can become unwieldy as the number of branches exceeds two.

Looping

One of the greatest strengths of programmability is the ability to execute a sequence of statements repeatedly in a loop. Mathcad provides two loop structures. The choice of which loop to use depends on how you plan to tell the loop to stop executing.

- If you know exactly how many times you want a loop to execute, use a **for** loop.

- If you want the loop to stop when a condition has been met, but you don't know how many loops will be required, use a **while** loop.

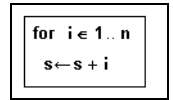
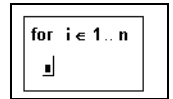
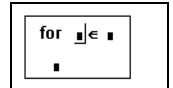
Tip See “Controlling Program Execution” on page 272 for methods to interrupt calculation within the body of a loop.

“For” Loops

A **for** loop terminates after a predetermined number of iterations. Iteration is controlled by an *iteration variable* defined at the top of the loop. The definition of the iteration variable is local to the program.

To create a **for** loop:

1. Click **for** on the Programming toolbar or press [Ctrl] “. Do not type the word “for.”
2. Type the name of the iteration variable in the placeholder to the left of the “∈.”
3. Enter the range of values the iteration variable should take in the placeholder to the right of the “∈.” You usually specify this range the same way you would for a range variable (see page 101).
4. Type the expression you want to evaluate in the remaining placeholder. This expression generally involves the iteration variable. If necessary, add placeholders by clicking **Add Line** on the Programming toolbar.



The upper half of Figure 15-4 shows this **for** loop being used to add a sequence of integers.

Note Although the expression to the right of the “∈” is usually a range, it can also be a vector or a list of scalars, ranges, and vectors separated by commas. The lower half of Figure 15-4 shows an example in which the iteration variable is defined as the elements of two vectors.

```

sum(n) := | s ← 0
          | for i ∈ 1..n
          | s ← s + i

sum(44) = 990

join(r, s) := | m ← 0
              | for x ∈ r, s
              |   | v_m ← x
              |   | m ← m + 1
              | v

```

Equivalent to... $n := 44 \quad \sum_{i=1}^n i = 990$

$$r := \begin{pmatrix} 100 \\ 101 \\ 102 \end{pmatrix} \quad s := \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

$$join(r, s) = \begin{pmatrix} 1 \\ 2 \\ 100 \\ 101 \\ 102 \end{pmatrix}$$

$i + 2 = \square$

← “i” is undefined anywhere outside the program.

This variable or function is not defined above.

Figure 15-4: Using a **for** loop with two different kinds of iteration variables.

“While” Loops

A **while** loop is driven by the truth of some condition. Because of this, you don’t need to know in advance how many times the loop will execute. It is important, however, to have a statement somewhere, either within the loop or elsewhere in the program, that eventually makes the condition false. Otherwise, the loop executes indefinitely.

To create a **while** loop:

1. Click  on the Programming toolbar or press **[Ctrl]]**. Do not type the word “while.”
2. Click in the top placeholder and type a condition. This is typically a Boolean expression like the one shown.
3. Type the expression you want evaluated in the remaining placeholder. If necessary, add placeholders by clicking  on the Programming toolbar.

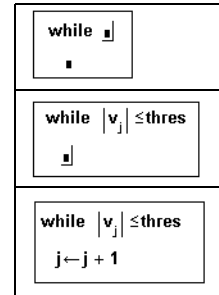


Figure 15-5 shows a larger program incorporating the above loop.

Upon encountering a **while** loop, Mathcad checks the condition. If the condition is true, Mathcad executes the body of the loop and checks the condition again. If the condition is false, Mathcad exits the loop.

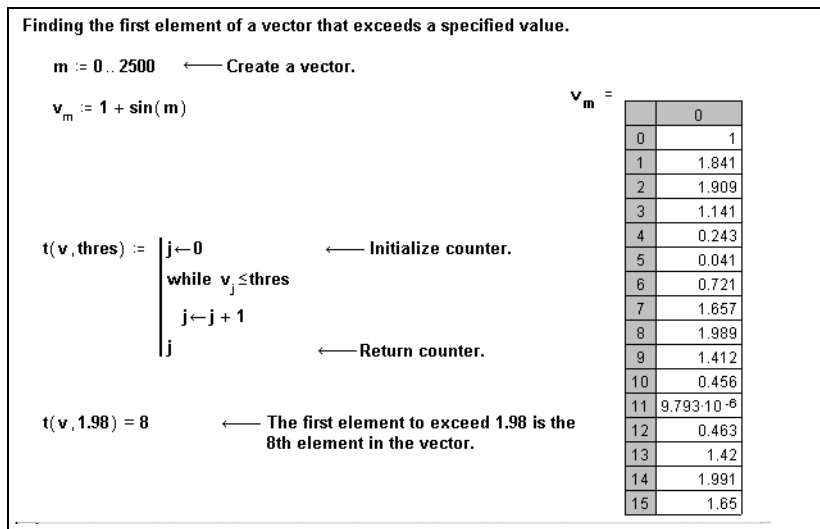


Figure 15-5: Using a **while** loop to find the first occurrence of a particular number in a matrix.

Controlling Program Execution

The Programming toolbar in Mathcad Professional includes three statements for controlling program execution:

- Use the **break** statement within a **for** or **while** loop to interrupt the loop when a condition occurs and move execution to the next statement outside the loop.
- Use the **continue** statement within a **for** or **while** loop to interrupt the current iteration and force program execution to continue with the next iteration of the loop.
- Use the **return** statement to stop a program and return a particular value from within the program rather than from the last statement evaluated.

The “Break” Statement

It is often useful to break out of a loop upon the occurrence of some condition. For example, in Figure 15-6 a **break** statement is used to stop a loop when a negative number is encountered in an input vector.

To insert a **break** statement, click on a placeholder inside a loop and click  on the Programming toolbar or press **[Ctrl] {**. Do not type the word “break.” You typically insert **break** into the left-hand placeholder of an **if** statement. The **break** is evaluated only when the right-hand side of the **if** is true.

Tip To create the program on the left in Figure 15-6, for example, you would click  first, then click .

The “Continue” Statement

To ignore an iteration of a loop, use **continue**. For example, in Figure 15-6 a **continue** statement is used to ignore nonpositive numbers in an input vector.

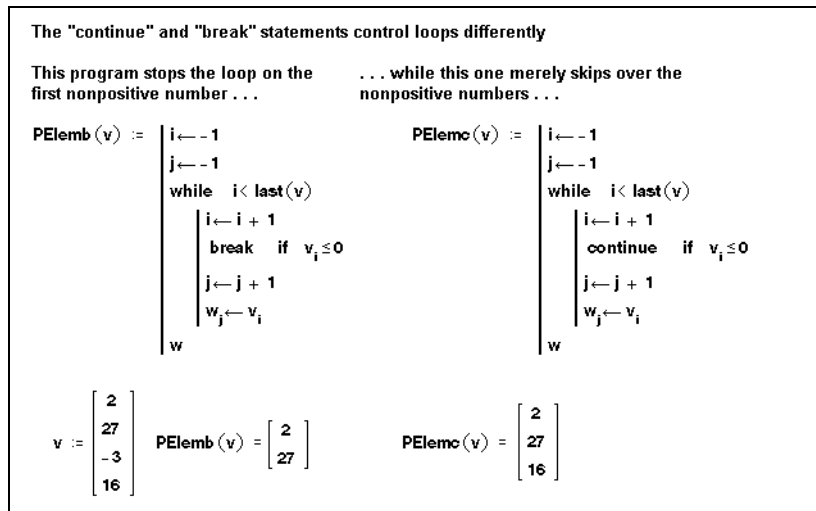


Figure 15-6: The **break** statement halts the loop. Program execution resumes on the next iteration when **continue** is used.

To insert the **continue** statement, click on a placeholder inside a loop and click  on the Programming toolbar or press **[Ctrl]]**. Do not type the word “continue.” As with **break**, you typically insert **continue** into the left-hand placeholder of an **if**

statement. The **continue** statement is evaluated only when the right-hand side of the **if** is true.

The “Return” Statement

A Mathcad program returns the value of the last expression evaluated in the program. In simple programs, the last expression evaluated is in the last line of the program. As you create more complicated programs, you may need more flexibility. The **return** statement allows you to interrupt the program and return particular values other than the default value.

A **return** statement can be used anywhere in a program, even within a deeply nested loop, to force program termination and the return of a scalar, vector, array, or string. As with **break** and **continue**, you typically use **return** on the left-hand side of an **if** statement, and the **return** statement is evaluated only when the right-hand side of the **if** statement is true.

The following program fragment shows how a **return** statement is used to return a string upon the occurrence of a particular condition:

1. Click  on the Programming toolbar.
2. Now click  on the Programming toolbar or press **[Ctrl]**. Do not type “return.”
3. Create a string by typing the double-quote key (") on the placeholder to the right of **return**. Then type the string to be returned by the program. Mathcad displays the string between a pair of quotes.
4. Type a condition in the placeholder to the right of **if**. This is typically a Boolean expression like the one shown. (Type **[Ctrl]=** for the bold equal sign.)

■ if ■
return ■ if ■

return "int" if ■

return "int" if floor(x) = x

In this example, the program returns the string “int” when the expression $\text{floor}(x) = x$ is true.

Tip You can add more lines to the expression to the right of **return** by clicking  on the Programming toolbar.

Error Handling

Errors may occur during program execution that cause Mathcad to stop calculating the program. For example, because of a particular input, a program may attempt to divide by 0 in an expression and therefore encounter a singularity error. In these cases Mathcad treats the program as it does any math expression: it marks the offending expression with an error message and highlights the offending name or operator in a different color, as described in Chapter 8, “Calculating in Mathcad.”

Mathcad Professional gives you two features to improve error handling in programs:

- The **on error** statement on the Programming toolbar allows you to trap a numerical error that would otherwise force Mathcad to stop calculating the program.
- The *error* string function gives you access to Mathcad's error tip mechanism and lets you customize error messages issued by your program.

“On Error” Statement

In some cases you may be able to anticipate program inputs that lead to a numerical error (such as a singularity, an overflow, or a failure to converge) that would force Mathcad to stop calculating the program. In more complicated cases, especially when your programs rely heavily on Mathcad's numerical operators or built-in function set, you may not be able to anticipate or enumerate all of the possible numerical errors that can occur in a program. The **on error** statement is designed as a general-purpose error trap to compute an alternative expression when a numerical error occurs that would otherwise force Mathcad to stop calculating the program.

To use the **on error** statement, click  on the Programming toolbar or type **[Ctrl] '**. Do not type “on error.” In the placeholder to the right of **on error**, create the program statement(s) you ordinarily expect to evaluate but in which you wish to trap any numerical errors. In the placeholder to the left of **on error**, create the program statement(s) you want to evaluate should the default expression on the right-hand side fail.

Figure 15-7 shows **on error** operating in a program to find a root of an expression.

A custom root-finding program that demonstrates the “on error” statement.

This program by default tries to find a real root of an expression, using a guess of 0, but will try to search for a complex root, using a guess of $0 + 2i$, if that fails:

```

RF(f, x) := root(f(x), x)          <-- Mathcad's usual numerical root finder.

RootFind(f) := | gr ← 0
                | gc ← 0 + 2i
                | RF(f, gc) on error RF(f, gr)          <-- Click the “on error”
                                                         button on the
                                                         programming palette to
                                                         insert the operator here.

f1(x) := x2 - 2x - 3              f2(x) := x2 - x + 3

RootFind(f1) = -1.000             RootFind(f2) = 0.500 + 1.658i

Without “on error” the second example cannot converge:

```

RF(f2, 0) =

Can't converge to a solution. Try a different guess value or check that a solution really exists.

Figure 15-7: The **on error** statement traps numerical errors in a program.

Issuing Error Messages

Just as Mathcad automatically stops further evaluation and produces an appropriate “error tip” on an expression that generates an error (see the bottom of Figure 15-7 for an example), you can cause evaluation to stop and make custom error tips appear when your programs or other expressions are used improperly or cannot return answers.

Mathcad Professional's *error* string function gives you this capability. This function, described in "String Functions" on page 187, suspends further numerical evaluation of an expression and produces an error tip whose text is simply the string it takes as an argument. Typically you use the *error* string function in the placeholder on the left-hand side of an **if** or **on error** programming statement so that an error and appropriate error tip are generated when a particular condition is encountered.

Figure 15-8 shows how custom errors can be used even in a small program.

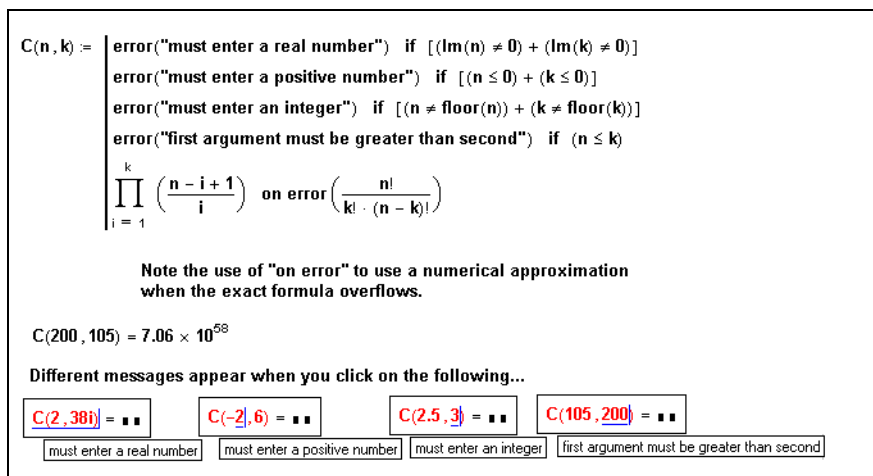


Figure 15-8: Generating custom errors via the error string function.

Note Some error strings are automatically translated to a Mathcad error message that is similar to the error string. For example "must be real" is translated to "This value must be real. Its imaginary part must be zero."

Programs Within Programs

The examples in previous sections have been chosen more for illustrative purposes rather than their power. This section shows examples of more sophisticated programs. Much of the flexibility inherent in programming arises from the ability to embed programming structures inside one another. In Mathcad, you can do this in the following ways:

- You can make one of the statements in a program be another program, or you can define a program elsewhere and call it from within another program as if it were a subroutine.
- You can define a function recursively.

Subroutines

Figure 15-9 shows two examples of programs containing a statement which is itself a program. In principle, there is no limit to how deeply nested a program can be.

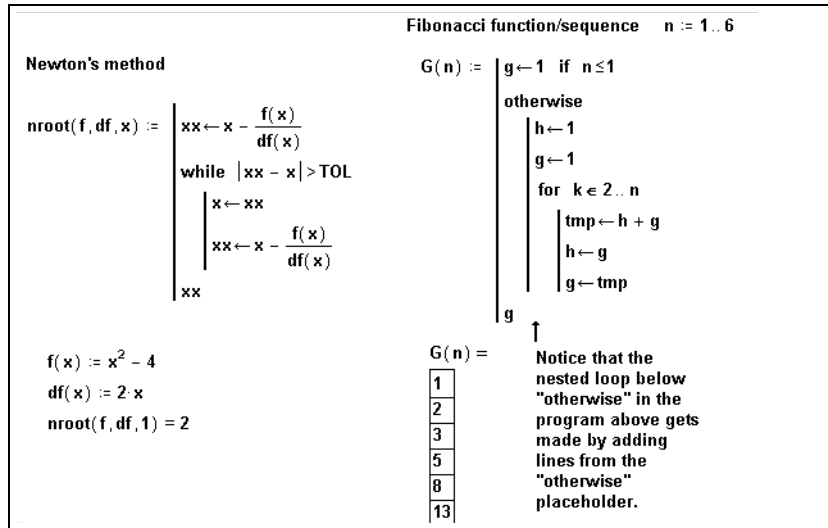


Figure 15-9: Programs in which statements are themselves programs.

One way many programmers avoid overly complicated programs is to bury the complexity in *subroutines*. Figure 15-10 shows an example of this technique.

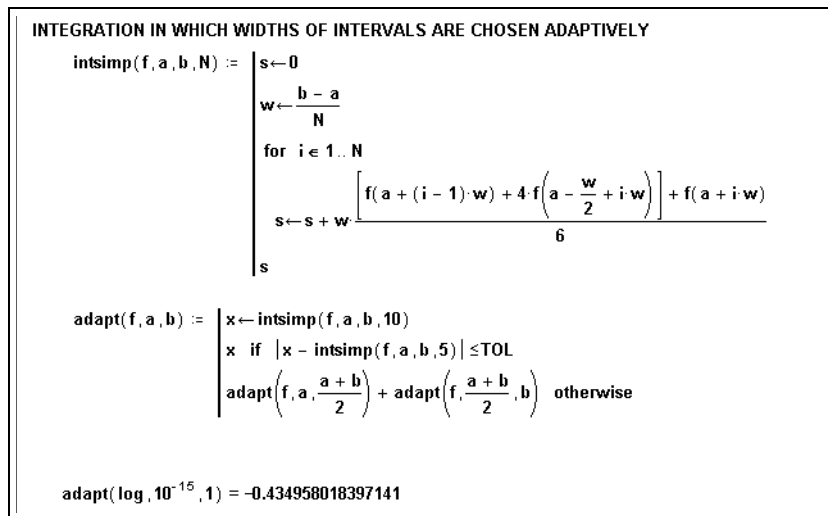


Figure 15-10: Using a subroutine to manage complexity.

Tip Breaking up long programs with subroutines is good programming practice. Long programs and those containing deeply nested statements can become difficult for other users to understand at a glance. They are also more cumbersome to edit and debug.

In Figure 15-10, the function *adapt* carries out an adaptive quadrature or integration routine by using *intsimp* to approximate the area in each subinterval. By defining

intsimp elsewhere and using it within *adapt*, the program used to define *adapt* becomes considerably simpler.

Recursion

Recursion is a powerful programming technique that involves defining a function in terms of itself, as shown in Figure 15-11. See also the definition of *adapt* in Figure 15-10. Recursive function definitions should always have at least two parts:

- A definition of the function in terms of a previous value of the function.
- An initial condition to prevent the recursion from continuing forever.

The idea is similar to mathematical induction: if you can determine $f(n + 1)$ from $f(n)$, and you know $f(0)$, then you know all there is to know about f .

Tip Recursive function definitions, despite their elegance and conciseness, are not always computationally efficient. You may find that an equivalent definition using one of the iterative loops described earlier will evaluate more quickly.

<u>Factorial function</u>	
$\text{factorial}(n) :=$	$\begin{cases} 1 & \text{if } n=1 \\ n \cdot \text{factorial}(n-1) & \text{otherwise} \end{cases}$
$\text{factorial}(5) =$	120
<u>Compound interest</u>	
$P(n, i, P_0) :=$	$\begin{cases} P_0 & \text{if } n=0 \\ P(n-1, i, P_0) \cdot (1+i\%) & \text{otherwise} \end{cases}$
$P(3, 12, 100) =$	140.493
Type Ctrl= to generate the boolean equals signs used inside the programs.	

Figure 15-11: Defining functions recursively.

Index

↵ (Enter key) 2
∫ (integral) 136, 299
→ (symbolic equal sign) 246
× (vector cross product) 130, 299
Σ (vector sum) 130, 299
→ (vectorize operator) 130, 201, 298
Σ and Π (summation and product) 130, 299
! (factorial) 123, 298
% 98, 298, 311
() (parentheses) 48
+ (with line break) 300
+, −, ·, and / 125, 300
:= (definition) 13, 97
<, >, ≤, ≥ (inequalities) 125, 300
= (Boolean equal) 126, 300
= (evaluating expression) 14, 99
| · | (determinant) 130, 299
| · | (magnitude/absolute value) 130, 299
∞ (infinity) 98, 311
≠ (not equal to) 126, 300
√ (square root) 125, 299
3D Plot Format dialog box 234
aborting calculations in progress 118
absolute value 127, 299
accessing Mathcad from other applications 294
acos function 146
acosh function 146
ActiveX 292, 294
adaptive smoothing of data 173
addition 125, 300
Ai and *Bi* Bessel functions 147
Airy functions 147
algorithms
 See numerical methods
aligning
 output tables 199
 regions 79
 text 60
and 125
angle function 145
Animate command 119
animation
 compressing AVI files 120
 creating 119
 playback 120
 saving 119
 saving with worksheet 121
 speed 119–120

antisymmetric tensor function 148
APPEND function 188
APPENDPRN function 188
approximations
 Minerr function 160
 root of expression 157
area
 collapsing 85
 deleting 87
 inserting 85
 locking and unlocking 85
 naming 85
 password protecting 85
arg function 148
arguments
 of functions 104
arrays
 calculations by element 201
 copying and pasting 199
 creating 35, 191
 defining with range variables 192
 displaying in results 198
 exporting data from 203
 extracting a row or column 197
 extracting a subarray 154
 functions for 152
 graphical display of 202
 importing data into 193, 196
 nested 111, 204
 operators for 127
 ORIGIN used with 197
 See also matrices and vectors
arrow keys, for editing 9, 314
ASCII codes
 entering in strings 36
 table 316
asin function 146
asinh function 146
assume keyword 249
assume keyword modifier 251
atan function 146
atan2 function 146
atanh function 146
augment function 153, 155
Auto (on status bar) 117
automatic calculation mode 117
autoscaling of axis limits 217
AutoSelect
 in numerical integration 136
 in solving 162
 overriding 137, 163

- AVI files
 - compression 120
 - creating 119
 - hyperlinking from worksheet 121
 - autoplay 120
- Axum component 283
- Axum graphs in Mathcad 214
- Axum LE 214
- background color 82
- bar plots (3D)
 - formatting 234
- base 10 logarithm 147
- base b logarithm 147
- base of results (decimal/hex/octal/binary) 111
- base units 113
- bei* and *ber* Bessel functions 147
- Bessel functions 147
- Bessel Kelvin functions 147
- binary numbers 34, 111
- bitmaps
 - color palettes 70
 - copying from the Clipboard 69
 - creating pictures from 68
 - functions for reading 189
- blank lines, inserting or deleting 80
- blank pages in printouts 94
- blank space between regions 10
- BMP files 68, 84, 189
- bookmarks 26
- Boolean operators 125, 127, 148, 161
- border around a region 10
- boundary value problems 183
- break statement 273
- breaking equations 255, 300
- bspline* function 167
- B-splines 167–168
- built-in functions
 - listed by type 143
 - symbolic only 301
- built-in variables 98
- bulleted paragraphs 60
- business functions 173
- bvalfit* function 185
- CAD drawings 286
- Calc on message line 118
- calculation 14
 - controlling 117
 - disabling for individual equation 118
 - equations 14, 99, 110
 - interrupting 118
 - locking area 85
 - order in worksheets 99
 - restarting after interrupting 118
 - units in 113
- calculator, using Mathcad as 12
- calling Mathcad from other applications 294
- ceil* function 149
- Celsius 109, 116
- CFFT* function 150
- cfft* function 150
- CGS units 114, 305
- characters, deleting or inserting in math 43
- Chi* function 301
- cholesky* function 156
- Ci* function 301
- Clipboard 49, 255
- closing a worksheet 20
- closing Mathcad
 - See exiting Mathcad
- CMP (colormap) files 190
- cnorm* function 166
- cnper* function 173
- coeffs keyword 247–248
- Collaboratory 22, 28
- collapsing an area 86
- collect keyword 248
- colon (:) as definition symbol 13, 97
- color
 - Electronic Book annotation 24
 - equation highlight 82
 - in equations 52
 - in text 59
 - of worksheet background 82
- color images
 - displaying 67
 - reading 189
- color palettes for bitmaps 70
- colormap files 190, 236
- cols* function 152
- column vectors
 - See vectors
- combin* function 149
- combinatorics functions 149
- common logarithm 147
- complementary error function 149
- complex conjugate 127, 298
- complex keyword 247–248
- complex numbers
 - conjugate 127
 - determining angle 127
 - display of 111
 - entering 34

- imaginary unit symbol 111
- magnitude of 127
- operators and functions for 127, 149
- real and imaginary parts 127
- vector field plots 232
- complex threshold 111
- Component Wizard 281
- components
 - inserting 281
 - overview of 280
- compounding periods 173
- computing results 14, 99, 104
- concat* function 187
- cond1* and *cond2* functions 153
- conde* and *condi* functions 153
- condition number of matrix 153
- conditional
 - functions 148, 188
 - statement 269
- conjugate (complex) 127, 298
- constants
 - changing the font style of 51
 - See also* numbers and predefined variables
- constraint
 - in solve blocks 161
 - tolerance 162
- context menu
 - See* pop-up menu
- continue statement 273
- contour integrals 138
- contour plots
 - creating 231
 - formatting 234
 - See also* plots, 3D
- convert keyword 249, 254
- Convert to Partial Fraction command 254
- converting to partial fractions 249
- copy and paste 10, 71, 116
- copying
 - expressions 49
 - from Electronic Book 25
 - regions 10
 - results 116
- corr* function 163
- cos* function 145
- cosh* function 146
- cosine (*cos*) function 187
- cosine integral 301
- cot* function 145
- coth* function 146
- covariance 163

- crate* function 173
- CreateMesh* function 154
- CreateSpace* function 154
- creating
 - 2D plots 207
 - 3D plots 223
 - contour plots 231
 - Electronic Books 89
 - hyperlinks 87
 - pop-up window 87
 - surface plots 225, 227–228
 - text regions 55
 - vector field plots 232
 - worksheet templates 75
- cross product 130, 299
- crosshair for insertion 10
- csc* function 145
- csch* function 146
- csign* function 148
- csort* function 157
- cspline* function 167
- CTOL variable 162, 311
- cube root 112, 299
- cubic spline interpolation 167
- cumint* function 174
- cumprn* function 174
- cumulative distribution functions 164
- cumulative interest paid 174
- cumulative principal paid 174
- curve fitting
 - functions for 169
 - polynomial 169–170
 - using cubic splines 167
- curves, finding area under 136
- custom operators 140
- cvar* function 163
- δ function 148
- d/dx
 - See* derivatives
- dashed selection rectangle 10
- data
 - entering into a table 195
 - graphing 214
- data files
 - exporting from an array 204
 - functions for reading and writing 188
 - importing data from 196
 - reading data from 193
 - reading into a matrix 193
 - writing from an array 203
- data input 196

- date in header or footer 84
- dbeta* function 165
- dbinom* function 165
- dcauchy* function 165
- dchisq* function 165
- debugging a worksheet 121
- decimal places
 - in displayed results 111
 - internal precision 110
- decimal points
 - numerical calculation 111
 - symbolic calculation 247
- decomposition
 - matrix 156
 - partial fraction 247
 - singular value 157
- default formats
 - 2D plots 218
 - 3D plots 234
 - numerical results 110
 - template 75
 - worksheet layout 75
- defining
 - complex numbers 34
 - functions 104
 - global variables 100
 - local variables in program 268
 - multiple definitions of variable 100
 - numbers 33
 - operators 140
 - programs 267
 - range variables 102
 - See also* creating
 - strings 35
 - units 109, 113
 - variables 13, 97
- definite integral 258
- definition symbol ($:=$) 13, 97
- degrees, converting to radians 116, 145
- deleting
 - blank lines 80
 - characters in math 43
 - equations 11
 - hard page breaks 83
 - hyperlinks 88
 - operators 46
 - parentheses 49
 - parts of an expression 50
 - regions 11
 - text 55
- delta function 148, 301
- derivatives 133, 299
 - higher order 135
 - symbolic 256
- determinant 130, 262, 299
- device-independent bitmap 69
- dexp* function 165
- dF* function 165
- dgamma* function 165
- dgeom* function 166
- dhypergeom* function 166
- diagonal matrix (*diag*) function 152
- dialects (spell-checker) 66
- DIB
 - See* device-independent bitmap
- dictionaries (spell-checker) 66
- differential equations 176
 - higher order 177, 179
 - partial 186
 - second order 177, 179
 - slowly varying solutions 182
 - smooth systems 182
 - stiff systems 182
 - systems 180
- differentiation
 - See* derivatives
- differentiation variable 133, 135
- digamma* function 302
- dilog* function 301
- dilogarithmic integral 301
- dimensions 108
- Dirac* delta function 301
- disabling equations 64, 118
- display of arrays 198
- display of operators 124
- displayed precision
 - See* decimal places
- distribution functions 164
- division 125, 299
- dlnorm* function 166
- dlogis* function 166
- dnbinom* function 166
- dnorm* function 166
- dot product 129
- double integrals 139
- Down One Level command 205
- dpois* function 166
- drag and drop 10, 25, 50, 71, 73, 116, 242
- dragging regions 10
- drawings
 - See* pictures
- dt* function 166

- dunif* function 166
- dweibull* function 166
- ϵ function 148
- e*, base of natural logarithms 98, 311
- Edit Go to Page command 9
- Edit Links command 73
- editing equations
 - annotated example 42
 - applying an operator 43
 - changing a number 43
 - changing a variable or function name 43
 - compared to word processors 42
 - deleting an operator 46
 - deleting parentheses 49
 - deleting parts of expression 50
 - inserting an operator 44
 - making expression an argument to a function 49
 - moving parts of an expression 49
 - moving/rearranging equations 79
- editing lines 16, 42
- eff* function 174
- effective annual interest rate 174
- Ei* function 301
- eigenvalues 155–156
- eigenvectors 155–156
- Electronic Book
 - browsing history 24
 - copying information from 25
 - creating 89
 - moving around in 23, 26
 - searching for information in 24
 - toolbar 23, 26
- Electronic Books 21, 87, 89
- Email 28, 95
- endpoints for ranges 103
- engineering notation 111
- Enter key 2
- epsilon function 148
- equal sign (=)
 - as Boolean operator 300
 - in numerical calculations 99
 - in solve blocks 159
 - symbolic calculations 246, 259–260
- equality constraints 161
- equations
 - as constraints in solve blocks 161
 - breaking 255, 300
 - calculating results 14, 99
 - color 52
 - disabling calculation for 64, 118
 - dragging and dropping 50
 - effect of range variables in 103
 - errors in 121
 - font 51
 - global definitions 100
 - in text 63
 - locking in area 85
 - order of evaluation 99, 117
 - processing and calculating 13, 117
 - properties 64, 118
 - solving for root 157
 - solving symbolically 259–261
 - solving with solve blocks 159
 - styles 51
 - units in 106
 - variable definition 97
 - wrapping 300
- erf* function 149, 301
- erfc* function 149
- ERR variable and *Minerr* 162
- error (*erf*) function 149–150, 301
- error messages
 - correcting 122
 - custom 187
 - in equations 121
 - in programs 276
 - tracing the source of 121
 - with units 108
- error* string function 187, 276
- Euler's constant 301
- Evaluate Complex command 254
- Evaluate Floating Point command 247, 254
- Evaluate in Place option 255
- Evaluate Symbolically command 254
- Excel
 - calling Mathcad from 295
 - Excel component 284
 - reading an Excel data file 193
 - writing data to an Excel file 203
- exclusive or 125
- exiting Mathcad 20
- exp* function 147
- Expand command 254
- expand in series 248
- expand keyword 247–248, 254
- expand nested arrays 111
- Expand to Series command 247
- Expert Solver 159, 163
- expfit* function 170
- exponent 125, 298
- exponential
 - function 147

- integral 301
- notation, entering 34
- notation, in displayed results 111
- threshold 111
- exporting
 - worksheets as HTML 77
 - worksheets as RTF 76
- exporting data 203
- expressions
 - applying a function to 49
 - converting to partial fractions 254
 - correcting errors in 122
 - deleting parts of 50
 - error messages in 121
 - evaluating 99
 - expanding 247, 254
 - factoring 247
 - finding the coefficients of 247
 - moving parts of 49
 - on multiple lines 300
 - selecting several 79
 - simplifying 254
 - symbolic evaluation of 246–247
- Extending Mathcad in the Resource Center 22
- Extension Packs 143
- F (function) keys, table of 315
- Φ function 148
- Factor command 254
- factor keyword 247–248
- factorial (!) 125, 298
- Fahrenheit 109, 116
- fast Fourier transform 150
- FFT* function 151
- fft* function 150
- fhyper* function 149
- file access functions 188
- File Read/Write component 193, 203
- File Send command 95
- files
 - opening 76
 - reading data from 188, 193
 - saving 19
 - See also* data files
 - See also* worksheets
 - writing data to 188
- filters
 - for exporting data 203
 - for importing data 193
- finance functions 173
- Find* function 159–160, 261
- first order differential equation 178
- float keyword 247–248
- floor* function 149
- font
 - changing in header or footer 84
 - changing in math 51
 - changing in text 58
- footers 84
- for loop 271
- Format Header/Footer command 84
- Format Properties command 64
- Format Style command 61
- formatting
 - 2D plots 216
 - 3D plots 234
 - numbers in matrices 199
 - operators 124
 - results 110
 - symbolic 255
 - worksheets 83
- Formatting toolbar 8
 - math styles 52
 - text styles 61
- fourier keyword 247, 249, 263
- Fourier transforms
 - numeric functions 150
 - symbolic 247, 249, 263
- FRAME for animation 119
- Fresnel cosine integral 301
- Fresnel sine integral 301
- FTP 28
- functions
 - applying to an expression 49
 - built-in 143
 - business 173
 - colormap 190, 236
 - combinatorics 149
 - complex arithmetic 127
 - defining 16, 104
 - error message 187
 - file access 188
 - finance 173
 - Fourier transform 150
 - hyperbolic 146
 - inserting 143
 - interpolation 167
 - inverse trigonometric 146
 - log and exponential 147
 - matrix decomposition 156
 - number theory 149
 - optimization 157
 - piecewise continuous 148, 188

- population statistics 163
- prediction 167
- probability distribution 164
- recursive 106
- regression 169
- See also* built-in functions
- smoothing 172
- solving 157
- special 149
- statistical 163
- string manipulation 187
- symbolic calculation 301
- tensor 148, 187
- that take vector arguments 152
- to combine arrays 153–154
- to compute angle to a point 148
- to create arrays 154
- to find roots of expressions 157
- to manipulate strings 187
- trigonometric 145
- uniform polyhedra 190
- user-defined 36, 104
- vector and matrix 152
- future value calculations 174
- fv* function 174
- fvadj* function 174
- fvc* function 175
- gamma (Euler’s constant) 301
- gamma* function 149
- Gaussian distribution 166
- gcd* function 149
- generalized
 - eigenvalues 155
 - eigenvectors 155
 - inverse of a matrix 152
 - regression 171
- genfit* function 171
- geninv* function 152
- genvals* function 155
- Given*, in solve blocks 159, 261
- global definitions 100
- gmean* function 163
- Gopher 28
- graphics, inserting 67
- graphing
 - data 212
 - expressions 211
 - functions 211, 224
 - in 2D 207
 - in 3D 223
 - uniform polyhedra 190
 - vector 212
- graphs
 - Axum 283
 - creating 17, 190, 223
 - formatting 18
 - resizing 18
 - See also* plots, 2D or plots, 3D
 - S-PLUS 288
- greater than 125
- greater than or equal to 125
- greatest common divisor 149
- greatest integer function 149
- Greek letters
 - in equations 37
 - in text 57
 - table of 313
- Greek toolbar 38, 57, 313
- guess
 - for root finding 157
 - for solve blocks 159
- guidelines for aligning regions 80
- hard page breaks 83
- HBK file for an Electronic Book 90
- HBK files 22
- headers and footers 84
- heaviside step function 148
- Help
 - context-sensitive 26
 - on-line 26
 - See also* Resource Center *and* technical support
- Her* function 149
- hexadecimal numbers 34, 111
- highlighting equations 82
- hist* function 164
- histograms 164
- history of browsing in Electronic Book 24
- hmean* function 164
- HTML 25, 77
- HTTP 28
- hyperbolic cosine integral 301
- hyperbolic functions 146
- hyperbolic sine integral 302
- hyperlinks
 - and relative paths 88
 - deleting 88
 - editing 88
 - to other file types 89
 - to worksheets 87
- Hypertext Mark-up Language
 - See* HTML
- i* (imaginary unit) 34

- I₀*, *I₁*, and *I_n* Bessel functions 147
- ibeta* function 149
- ICFFT* function 150
- icfft* function 150
- identity* function 152
- identity matrix 152
- if* function 148, 188
- if statement 269
- ifft* and *icfft* functions 150
- IFFT* function 151
- Im* function 148
- image file
 - BMP format 68, 189
 - in headers and footers 84
- imaginary numbers
 - entering 34
 - symbol for 34, 111
- imaginary value 111
- implied multiplication 42, 108, 312
- importing data 193, 196
- impulse function 148
- incompatible units (error message) 108
- increments for ranges 103
- indefinite integral 258
- indented paragraphs 60
- index for an Electronic Book 91
- index variables
 - See range variables
- inequalities 125, 300
 - as constraints in solve blocks 161
- infinity (∞) 37, 98, 311
- in-line division 129
- inner product 129
- in-place activation 71, 282
- Input Table component 195–196
- input to a component 281
- Insert Area command 85
- Insert Function command 143
- Insert Hyperlink command 87
- Insert key 44, 56
- Insert Link command 87
- Insert Math Region command 63
- Insert Matrix command
 - to create array 35, 191
 - to resize array 192
- Insert Object command 10, 71, 116
- Insert Reference command 87, 279
- Insert Unit command 108, 113
- inserting
 - blank lines 80
 - characters 43
 - equations in text 63
 - functions 49
 - graphic objects 71
 - graphics computationally linked 74
 - hyperlinks 87
 - math region 63
 - minus sign in front of expression 47
 - parentheses around expression 48
 - pictures 67
 - text 55
 - units 108
 - worksheet 279
- insertion point 12
- installation instructions 5
- integral transforms
 - Fourier 247, 263
 - Laplace 247, 263
 - z 247, 263
- integrals 136, 299
 - algorithms 136
 - AutoSelect 136
 - contour 138
 - double 139
 - indefinite 258
 - symbolic evaluation of 258
 - tolerance 138
 - variable limits 137
- integration
 - adaptive 137
 - improper endpoints 137
 - infinite limits 137
 - Romberg 137
- IntelliMouse support 9, 244
- intercept* function 169
- interest rate 174
- interest rate calculations 173
- internal rate of return 176
- International System of units (SI) 114
- Internet
 - access 27
 - Collaboratory 28
 - Web browsing 25
- Internet Explorer
 - See Microsoft Internet Explorer
- Internet setup 28
- interp* function 167
- interpolation
 - cubic spline 167
 - functions 167
 - linear 167
- interrupting calculations in progress 118

- inverse
 - cumulative distributions 164
 - Fourier transform 247, 249, 263
 - hyperbolic functions 146
 - Laplace transform 247, 249, 263
 - matrix 130, 298
 - trigonometric functions 146
 - wavelet transform 152
 - z-transform 247, 249, 263
- invfourier keyword 247, 249, 263
- invlaplace keyword 247, 249, 263
- invztrans keyword 247, 249, 263
- ipmt* function 175
- irr* function 176
- IsArray* function 187
- IsScalar* function 187
- IsString* function 187
- iterated product 130, 299
- iterated sum 130
- iteration
 - in programs 270
 - with range variables 15
- iwave* function 152
- j* (imaginary unit) 34
- J0*, *J1*, and *Jn* Bessel functions 147
- Jac* function 150
- Jacobian matrix 182
- JavaScript 292
- js* and *ys* spherical Bessel functions 147
- JScript 292
- K0*, *K1*, and *Kn* Bessel functions 147
- keywords, symbolic 247–250
- Kronecker’s delta function 148
- ksmooth* function 173
- kurt* function 164
- Lag* function 150
- Lambert’s *W* function 302
- laplace keyword 247, 249, 263
- Laplace transforms 247, 249, 263
- Laplace’s equation 186
- last* function 152
- Laurent series 247
- lcm* function 149
- least common multiple 149
- least integer function 149
- left inverse of a matrix 152
- Leg* function 150
- length* function 152
- less than 125
- less than or equal to 125
- lgsfit* function 170
- limits, evaluating 124, 259
- line break
 - in equation 300
 - in text 56
- line* function 169
- linear
 - interpolation 167
 - prediction 167
 - programming 159
 - regression 169
 - system solver and optimizer 159
 - systems of differential equations 180
 - systems of equations 156, 159
- linfit* function 171
- link
 - See also* hyperlinks
 - to objects 71
 - to other worksheets 87, 279
- linterp* function 167
- literal subscripts 39
- ln* (natural log) function 147
- LoadColormap* function 190, 236
- local result format 17
- lockable area
 - See* area
- locked calculations 85–86
- locking and unlocking an area 85
- loess* function 169
- log* function 147
- logarithms and exponential functions 147
- logfit* function 170
- logical operators
 - See* Boolean operators
- long equations 255, 300
- looping
 - for loop 271
 - while loop 272
- lsolve* function 156
- lspline* function 167
- lu* function 156
- magnitude
 - complex numbers 127, 299
 - vector 130
- mailing worksheets 95
- manual mode 117
- margins 83
- Math Optimization command 266
- Math Options command 98
- math styles
 - applying 52
 - Constants 51

- editing 51
 - saving 53
 - Variables 51
- Math toolbar 8, 38
- Mathcad
 - accessing from other applications 294
- Mathcad 6, 7, or 8 77
- Mathcad Document object model 295
- Mathcad OLE automation objects 294
- Mathcad Overview 22
- Mathcad Web Library 22
- MathConnex 291
- MathSoft
 - contacting 6
- MathSoft home page 25
- MATLAB component 291
- matrices
 - adding/deleting rows or columns 192
 - as array elements 204
 - calculations by element 201
 - combining 153–154
 - condition number 153
 - creating 35
 - creating from bitmaps 189
 - creating with components 281
 - decomposition functions 156
 - defining by formula 192
 - defining with two range variables 192
 - definition of 35, 191
 - determinant 130, 262, 299
 - displayed as pictures 67
 - displayed as scrolling output tables 198
 - extracting a column 196
 - extracting a submatrix 154
 - extracting elements 196
 - functions for 152
 - inverting 130, 298
 - limits on size 191, 199
 - matrix arithmetic 129
 - norm 153
 - norm of 153
 - numbering elements 197
 - operators for 127
 - ORIGIN used with 197
 - plotting in contour plot 231
 - plotting in surface plot 228
 - raising to a power 298
 - rank 153
 - See also* arrays
 - singular values 157
 - sorting by row or column 157
 - special characteristics 153
 - start with row and column zero 197
 - subscripts 196
 - trace 153
 - transpose 130, 298
- Matrix Determinant command 262–263
- Matrix display style 111
- matrix* function 154
- Matrix Invert command 263
- matrix subscript 298
- Matrix Transpose command 262–263
- max* function 152
- Maximize* function 160
- MCD file 75
- MCT file 75
- mean
 - arithmetic 164
 - geometric 163
 - harmonic 164
- mean* function 164
- measurement for the ruler 80
- medfit* function 169
- median* function 164
- metafile 69
- mhyper* function 150
- Microsoft Internet Explorer 22, 25, 292
- Microsoft Office 73, 284
- min* function 152
- Minerr* function 160, 162
- Minimize* function 160
- minus sign
 - for negation 125, 298
 - inserting in front of expression 47
- MIP
 - See* mixed integer programming
- mirr* function 176
- mixed integer programming 163
- MKS units 114, 309
- mod* function 149
- mode
 - See* automatic mode, manual mode
- mode* function 164
- moving
 - crosshair 10, 314
 - editing lines 43, 314
 - insertion point 43, 314
 - regions 10
- multigrd* function 187
- multiple integrals 139
- multiple roots
 - finding with *polyroots* 158

- finding with solve blocks 162
- multiple summations 131
- multiplication 42, 299
 - implied 42, 108, 312
- multivalued functions 112
- names of variables and functions 36
- natural logarithm 147
- negating an expression 47, 298
- nested arrays
 - defining 204
 - displaying 204
 - expanding 111, 205
- new features in Mathcad 4
- nom* function 174
- nominal interest rate 174
- nonlinear systems of equations 159
- nonscalar value (error message) 102
- norm
 - of matrix 153
 - of vector 130, 299
- norm1* and *norm2* functions 153
- normal distribution 166
- norme* and *normi* functions 153
- not 125
- not converging (error) 138
- not equal to 125
- notation in this *User's Guide* 1
- nper* function 173
- npv* function 176
- nth* order derivative 135
- nth* root 125, 299
- num2str* function 187
- number format
 - See* result format
- number theory functions 149
- numbered paragraphs 60
- numbers 33
 - binary 34, 111
 - complex 34
 - decimal 111
 - displayed as zero 111
 - exponential notation for 34, 111
 - format for computed results 110
 - formatting 16, 110
 - hexadecimal 34, 111
 - imaginary 34
 - octal 34, 111
 - radix (base) for results 111
 - rounding 149
- numerical methods
 - differentiation 133, 135
 - integration 136
 - root finding 158
 - solving and optimization 162
- object linking and embedding
 - See* OLE
- object model, Mathcad Document 295
- objects
 - embedding 71
 - linking 71
- octal numbers 34, 111
- odesolve* function 177
- OLE
 - automation 292, 294
 - drag and drop 73
 - editing links 73
 - in-place activation 71, 73, 282
 - scripting objects 293
 - via components 280
- on error statement 275
- on-line resources 21
- OpenGL 223
- opening worksheets 76
- operator placeholder 47
- operators
 - Boolean 125
 - changing the display of 124
 - customizing 140
 - defined 39
 - deleting 46
 - derivative 133, 256
 - for complex numbers 127
 - for vectors and matrices 127
 - indefinite integral 258
 - inserting 44
 - integral 136
 - iterated product 130
 - iterated sum 130
 - logical 125, 148
 - nth* order derivative 135
 - replacing 47
 - symbolic 124
 - table of 298
 - toolbars 8, 123
 - vector sum 132
- Optimize Palette command 70
- optimizers 159
- or 125
- or, exclusive 125
- order
 - of derivative 135
 - of polynomial regression 169

- of worksheet evaluation 100
- ORIGIN variable 98, 197
- output from a component 281
- output table 198
 - alignment 199
 - resizing 199
 - versus matrix display style 111
- overlapping regions 81
- overtyping text 56
- overview of Mathcad features 4, 22
- page
 - breaks, inserting and deleting 83
 - headers and footers 84
 - length 83
 - numbering 84
- Page Setup dialog box 83, 93
- palettes, color, for bitmaps 70
- paragraphs 55
 - bullets 60
 - hanging indent 60
 - indenting 60
 - numbers 60
 - properties 59
 - tab stops 60
 - text alignment in 60
- parametric surface plots
 - creating 229–230
 - See also* plots, 3D
- parentheses
 - deleting from expression 49
 - inserting into an expression 48
- partial differential equations 186
- partial fractions 247
- password protecting an area 85
- Paste command 11, 72
- Paste Special command 11, 69, 72
- pasting
 - bitmaps 69
 - device-independent bitmaps 69
 - from Clipboard 49, 69
 - metafiles 69
 - OLE objects 11, 72
- payment calculations 175
- pbeta* function 165
- pbinom* function 165
- pcauchy* function 165
- pchisq* function 165
- Pearson's correlation coefficient 163
- pending computations 117–118
- percent 98, 311
- permut* function 149
- permutations 148
- personal
 - dictionary (spell-checker) 66
 - QuickSheets 141
- pF* function 165
- pgamma* function 165
- pgeom* function 166
- phypergeom* function 166
- pi (Π , product symbol) 130
- pi (3.14159...) 43, 98, 311, 313
- picture operator 67, 203
- pictures
 - border on 70
 - creating from bitmap file 68
 - creating from matrix 68
 - creating using SmartSketch 286
 - formatting 70
 - importing into an array 189
 - pasted from Clipboard 69
 - resizing 70
- Piecewise 148
- piecewise continuous functions 148
- placeholder 12, 33
- placeholder for an operator 47
- Playback command 120
- plnorm* function 166
- plogis* function 166
- plots 3D
 - graphing functions 224
- plots, 2D
 - autoscaling of axis limits 217
 - changing perspective 220
 - copying format from existing plot 218
 - creating 208
 - default formats 218–219
 - formatting 216
 - graphing expressions 209
 - graphing functions 209
 - graphing vectors 213
 - multiple traces on 210
 - of data 214
 - QuickPlot 18
 - read-out of coordinates 221
 - reference lines in 218
 - resizing 18
 - setting axis or data limits 217
 - setting default formats 219
 - Show Markers 218
 - titles and labels 216
 - traces on 210
 - tracing coordinates on 221

- zooming 220
- plots, 3D 223
 - 3D Plot Format dialog box 235
 - 3D Plot Wizard 224
 - annotations 241
 - backplanes 235
 - color 236, 240
 - colormaps 236–237
 - contour lines 238
 - contour plots 231
 - converting 241
 - creating 223
 - examples 224, 228
 - fill color 236
 - filling contours 237
 - fog 234
 - formatting 234
 - graphic annotations on 241
 - lighting 240
 - line color 239
 - lines 238
 - multiple plots on 233
 - OpenGL graphics 223
 - parametric surface plots 229–230
 - point color 240
 - point symbols 240
 - QuickPlot 224
 - resizing 18
 - rotating 243
 - space curves 227
 - spinning 244
 - surface plots 225, 227–228
 - text on 241
 - titles 236
 - uniform polyhedra 190
 - vector field plots 232
 - wireframe 238
 - zooming 243–244
- pmt* function 175
- pnorm* function 166
- Poisson's equation 186
- polar plots
 - creating 208
 - formatting 216
 - See also* plots, 2D
- polygamma* function 302
- polyhedra 190
- Polyhedron* function 190
- PolyLookup* function 190
- polynomial
 - finding the roots of 158
 - regression 169–170
- Polynomial Coefficients command 247
- polyroots* function 157–158
- population standard deviation 164
- population statistics 163
- population variance 164
- popup hyperlink 87
- pop-up menu
 - 3D plots 242
 - animation playback 120
 - component 282
 - Excel component 285
 - Input Table component 195
 - integration 137
 - MATLAB component 291
 - Scriptable Object component 294
 - SmartSketch component 286
 - solving 163
 - S-PLUS Graph component 289
 - S-PLUS Script component 290
 - Web browsing 26
- pop-up window, creating 87
- power 125, 298
- ppmt* function 175
- ppois* function 166
- precision, internal 110
- predefined variables 98
- predict* function 167
- prediction, linear 167
- present value calculations 176
- principal branch of function 112
- Print Preview command 94
- printing 20, 93
 - and calculation of worksheet 118
 - and pagination 84
 - blank pages in 83, 94
 - color 82
 - print preview 94
 - wide worksheets 93
- PRNCOLWIDTH variable 98, 311
- PRNPRECISION variable 98, 311
- probability distributions 164
- processing equations 13, 117–118
 - results of 117
- product 130, 299
 - cross product 130
 - dot product 129
 - iterated 130
- programs 267
 - adding lines 267
 - break statement 273

- continue statement 273
- controlling or interrupting 272
- defining 267
- error handling 274
- error messages in 276
- for loop 271
- generating symbolic results 269
- if statement 269
- local assignment 267
- looping 270
- nested 276
- on error statement 275
- output of 267
- recursion 278
- return statement 274
- statements 267
- subroutines 276
- symbolic evaluation of 269
- while loop 272
- properties
 - of components 282
 - region 81
- proxy server 28
- Psi* functions 302
- pspline* function 167
- pt* function 166
- punif* function 166
- pvr* function 176
- pweibull* function 166
- pwrfit* function 170
- qbeta* function 165
- qbinom* function 165
- qcauchy* function 165
- qchisq* function 165
- qF* function 165
- qgamma* function 165
- qgeom* function 166
- qhypergeom* function 166
- qlnorm* function 166
- qlogis* function 166
- qnorm* function 166
- qpois* function 166
- qr* function 156
- qt* function 166
- quadratic equation solving 163
- QuickPlot 18, 209, 224
- QuickSheets 21
 - See also Resource Center
 - storing custom operators 141
- qunif* function 166
- qweibull* function 166
- radians
 - converting to degrees 116, 145
 - trig functions 145
- radix of displayed results 111
- range variables
 - array calculations with 201
 - creating arrays with 192
 - defining 15, 102–103
 - fundamental principle for 103
 - how Mathcad evaluates equations with 103
 - setting endpoints and increments 103
- rank* function 153
- rater* function 173
- rbeta* function 165
- rbinom* function 165
- rcauchy* function 165
- rchisq* function 165
- Re* function 148
- READ_BLUE* function 189
- READ_GREEN* function 189
- READ_HLS* function 189
- READ_HLS_HUE* function 189
- READ_HLS_LIGHT* function 189
- READ_HLS_SAT* function 189
- READ_HSV* function 189
- READ_HSV_HUE* function 189
- READ_HSV_SAT* function 189
- READ_HSV_VALUE* function 189
- READ_IMAGE* function 189
- READ_RED* function 189
- READBMP* function 189
- reading a data file 193
- READPRN* function 188
- READRGB* function 189
- recursion 106
- reduced-row echelon form 152
- reference tables in the Resource Center 22
- references
 - and relative paths 280
 - to other worksheets 279
- regions 10
 - aligning 79
 - blank space between 10
 - copying 10
 - deleting 11
 - dragging 10
 - dragging across documents 11
 - equation 10
 - locking 85
 - moving 10
 - overlapping 81

- putting borders around 10
 - separating 81
 - text 55
 - unlocking 86
 - viewing 10, 81
- regress* function 169
- regression
 - functions 169
 - generalized 171
 - linear 169
 - localized 170
 - multivariate 170
 - polynomial 169–170
 - using linear combinations of functions 171
- relational operators 148
- relative paths
 - for hyperlinks 88
 - for references 280
- relax* function 186
- Repaginate Now command 84
- replacing characters in math or text 65
- replacing operators 47
- reports
 - for a solve block 163
- resizing
 - graphs 18
 - pictures 70
- Resource Center 21
 - accessing worksheets on Web 25
 - bookmarks 26
 - Mathcad Web Library 22
 - Web browsing in 25
- resources, on-line 21
- result format 110
- results
 - calculating 14
 - calculating with equations 99, 110
 - complex 112
 - copying 116
 - formatting 110
 - units and dimensions in 113
 - wrapping 255
- return statement 274
- reverse* function 157
- rF* function 165
- rgamma* function 165
- rgeom* function 166
- rhypergeom* function 166
- rich text format (RTF) 76
- right page margin 83
- rkfixed* function 178
- rlnorm* function 166
- rlogis* function 166
- rnd* function 166
- rnorm* function 166
- root* function 157
- roots
 - finding 157
 - finding multiple with solve blocks 162
 - finding symbolically 259
 - numerical approximations used 158
 - of polynomials 158
- round* function 149
- rounding numbers 149
- row vectors
 - See* vectors
- rows* function 152
- rpois* function 166
- rref* function 152
- rsort* function 157
- rt* function 166
- RTF file 75
 - See also* rich text format
- ruler
 - for formatting a worksheet 79
 - for formatting text 60
- runif* function 166
- rweibull* function 166
- sample standard deviation 164
- sample variance 164
- Save As dialog box 19
- SaveColormap* function 190, 236
- saving
 - new file 19
 - templates 78
 - worksheets 19, 75
- sbval* function 184
- scalar
 - addition 129
 - definition of 35, 191
 - division 129
 - multiplication 129
- scatter plots (3D)
 - formatting 234
 - See also* plots, 3D
- scientific notation 111
- Scriptable Object component 292
 - object model 293
- search
 - Electronic Book 24
 - in equations 64
 - in text 64

- Search Book command 24
- search* function 187
- sec* function 145
- sech* function 146
- second derivatives, calculating 135
- second order differential equations 179
- selecting
 - graphs 18
 - math expression 45
 - page break 83
 - regions 10
 - text 56
- selection rectangle 10
- semicolon, in range variable definitions 102
- Separate Regions command 81, 84
- separating overlapping regions 81, 84
- series 247
- series keyword 248
- Shi* function 302
- Show Border option 70
- Si* function 302
- sigma (summation symbol) 130
 - for vector 130
- sign* function 148
- signum* function 148
- Simplify command 254
- simplify keyword 248, 254
- simultaneous equations, solving numerically 156, 159
- sin* function 145
- sine integral 302
- sinfit* function 170
- singular value decomposition 157
- sinh* function 146
- skew* function 164
- slope* function (linear regression) 169
- SmartSketch component 74, 286
- smooth systems (differential equations) 182
- smoothing of data 172
- soft page breaks 83
- solve blocks 159, 177
 - constraints in 161
 - definition of 159
 - expressions allowed in 161
 - finding multiple solutions 162
 - Given* in 159
 - reports for 163
 - tolerance 162
 - using to solve differential equations 177
 - using to solve numerically 159
 - using to solve symbolically 261
 - values returned by 160
- solve keyword 248, 259–260
- Solving and Optimization Extension Pack 159
- solving equations 159
 - AutoSelect of algorithm 162
 - linear systems 162
 - nonlinear systems 162
 - See also* solve blocks
 - with *lsolve* 156
 - with *root* function 157
 - with solve blocks 159, 261
 - with Solve for Variable 259
 - with solve keyword 259
- sort* function 157
- sorting vectors and matrices 157
- space curves
 - creating 227
 - See also* plots, 3D
- spaces, inserting or deleting 80
- special functions 149
- spell-checking 65
- spline functions 167–168
- S-PLUS Graph component 288
- S-PLUS Script component 289
- spreadsheets, exchanging data with 280
- square root 299
- stack* function 154–155
- stack overflow error 106
- standard deviation of a data set 164
- standard normal distribution 166
- statistics
 - cubic spline interpolation 167
 - cumulative distribution functions 164
 - functions 163
 - generalized linear regression 171
 - interpolation 167
 - inverse cumulative distributions 164
 - linear interpolation 167
 - linear prediction 167
 - linear regression 169
 - multivariate polynomial regression 170
 - polynomial regression 169
 - probability density functions 164
- stderr* function 169
- Stdev* and *stdev* functions 164
- step function 148
- step size
 - for differential equation solving 182
 - for iteration 103
- str2num* function 187
- str2vec* function 187
- strings

- arguments to file access functions 188
- as elements of vectors 35
- comparing 127
- converting to numbers and vectors 187
- defining 35
- editing 46
- evaluating 35
- manipulating 187
- variables 35
- strlen* function 187
- styles
 - math 51
 - text 61
- submatrix* function 154
- subroutines 276
- subscripts
 - in text 59
 - last element function 152
 - literal 39
 - non-numeric 39
 - ORIGIN used with 197
 - start with zero 197
 - vector and matrix 298
- Substitute for Variable command 247
- substitute keyword 247–248
- substr* function 188
- subtraction 125, 300
- summation 299
 - iterated 130
 - of vector elements 130
 - variable upper limit 132
- superscript
 - array 130, 298
 - to get column from matrix 197
- supsmooth* function 173
- surface plots
 - creating 225, 227–228
 - formatting 234
 - parametric 229–230
 - See also* plots, 3D
- svd* function 157
- svds* function 157
- symbolic
 - equal sign 246
 - evaluation 246
 - evaluation of programs 269
 - keywords 247
- Symbolics menu commands 254
- system requirements for Mathcad 5
- tab stops in a worksheet 79
- table of contents for an Electronic Book 90
- tables of data 195
- tabs in a paragraph 60
- tan* function 145
- tanh* function 146
- Taylor series 247
- Tcheb* function 150
- technical support 6
- temperature conversions 109, 116
- templates 75
 - creating new 77
 - modifying 78
 - See also* QuickSheets
 - used to save calculation mode 117
 - using to create a worksheet 75
- text 55
 - alignment 60
 - bullets in 60
 - changing font 58
 - color 59
 - editing 58
 - entering 14
 - Greek letters in 57
 - inserting equations in 63
 - moving 57
 - moving insertion point in 56
 - Occupy Page Width option 58
 - Push Regions Down As You Type option 59
 - regions 55
 - selecting 56
 - spell-checking 65
 - styles 61
- text box 14, 55
- text regions 55
 - changing width 57
 - creating 14, 55
 - editing 58
 - how to exit 15, 55
- text ruler 60
- text styles 61
 - applying 61
 - creating 62
 - modifying 61
- tilde (~), used in global definitions 100
- time in header or footer 84
- Tip of the Day 27
- TOL variable 98, 311
 - and integrals 138
 - and solve blocks 162
- tolerance
 - See* TOL variable *and* CTOL variable
- toolbars 8, 123

- customizing 9
- Electronic Book 23, 26
- operator 123
- programming 267
- Web 25
- top-to-bottom evaluation 99
- tr* function 153
- traces, on 2D plots 210
- tracing the source of an error 121
- trailing zeros 111
- training 23
- transcendental functions 145
- transforms
 - discrete Fourier 150
 - Fourier (symbolic) 247, 263
 - Laplace 247, 263
 - symbolic 247
 - wavelet 152
 - z 247, 263
- transpose of matrix 130, 262, 298
- trig keyword modifier 251
- trigonometric functions 145
 - inserting without parentheses 142
 - with degrees and radians 116
- trunc* function 149
- truncation functions 149
- two-point boundary value problems 183
- typing over text 56
- U.S. Customary units 114
- Ucheb* function 150
- undefined variables 99, 101
- uniform polyhedra 190
- Uniform Resource Locator
 - See URL
- unit impulse function 148
- unit step function 148
- units
 - alternative definitions 113
 - base units 113
 - CGS system 114
 - common sources of error 108
 - converting calculated results 115
 - default 107
 - defining 109, 113
 - dimensional consistency 108
 - errors in dimensions 108
 - in calculated values 113
 - in equations 106
 - metric 114
 - MKS system 114
 - of variable or expression 187
 - placeholder 113
 - prefixes 114
 - SI 114
 - simplifying 111
 - U.S. customary 114
- UnitsOf* function 109, 187
- Up One Level command 205
- update
 - window manually 118
 - worksheet 118
 - worksheet window 118
- URL
 - Collaboratory 28
 - MathSoft home page 25
- Use Default Palette command 70
- user-defined functions 104
 - evaluating variables in 105
 - valid names 36
- Var* and *var* functions 164
- Variable Differentiate command 256
- Variable Integrate command 258
- Variable Solve command 260
- Variable Substitute command 254
- variables
 - changing the font style of 51
 - defining 13, 97
 - global definitions of 100
 - in red 101, 121
 - matrices 35, 191
 - names 36
 - predefined 98
 - range variables 15, 101
 - string 35
 - substituting for 247
 - undefined 121
 - vectors 35, 191
- variance of a data set 164
- VBScript 292
- vec2str* function 188
- vector field plots
 - creating 232
 - from complex matrices 232
 - See also plots, 3D
- vector product 299
- vector subscript 298
- vector sum operator 132
- vectorize operator 201, 298
 - effect of 201
 - how to type 201
 - properties of 201
- vectors

- as array elements 204
- calculations by element 201
- column vectors 130
- combining 154
- creating 35, 191
- cross product 130
- displayed as scrolling output tables 198
- dot product 129
- functions for 152
- graphing 213
- limit on size 191
- magnitude 130, 299
- numbering elements 197
- operators for 127
- ORIGIN used with 197
- row 130
- See also* arrays
- sorting elements 157
- start with element zero 197
- subscripts 196
- sum elements of 130, 298
- undefined elements filled with zeros 197
- vector arithmetic 129
- vectorize operator 201
- View Animate command 119
- View Zoom command 9
- Visual Basic in Microsoft Excel 295
- Visual Basic Scripting Edition 292
- W* function 302
- wait message 117
- wave function 152
- wavelet transforms 152
- Web
 - See* World Wide Web
- Web Store 23
- Web toolbar 25
- while loops 272
- windows
 - update results manually 118
 - working with 9
 - zooming in and out of 9
- Windows keystrokes 9, 315
- wireframe, in 3D plots 238
- Wizards
 - for inserting 3D plots 224
 - for inserting a component 281
- worksheet ruler 79
- worksheets
 - closing 20
 - exporting as HTML 77
 - exporting as RTF 76
 - formatting 83
 - gathering in an Electronic Book 89
 - hyperlinking 87
 - in pop-up window 87
 - including by reference 87
 - opening 76
 - opening from Internet 26
 - order of evaluation 99
 - printing 20, 93
 - referencing in another worksheet 279
 - saving 19, 75
 - saving as templates 78
 - saving in an earlier format 77
 - sending by Email 95
- World Wide Web
 - accessing 25
 - bookmarks for browsing 26
 - browsing 25, 77
 - Collaboratory 28
 - HTML browsing 25, 77
 - MathSoft home page 25
 - toolbar 25
- wrapping equations 300
- WRITE* function 188
- WRITEBMP* function 189
- WRITEPRN* function 188
- WRITERGB* function 189–190
- writing data to a data file 203
- WWW
 - See* World Wide Web
- Wythoff symbol for a polyhedron 190
- xor 125
- X-Y plots
 - creating 208
 - formatting 216
 - QuickPlot 18
 - See also* plots, 2D
- Y0, Y1, and Yn* Bessel functions 148
- y-intercept 169
- zero threshold 111
- zeros of expressions or functions
 - See* roots
- Zeta* function 302
- zooming
 - 2D plots 220
 - 3D plots 244
 - windows 9
- ztrans keyword 247, 249, 263
- z-transforms 247, 249, 263