# Appendix A
# Reference

This appendix lists functions, operators, menu commands, special keystrokes, and other reference material. It is made up of the following sections:

**Menu commands**

**Function keys**

**Greek letters**

**Operators**

**Built-in functions listed alphabetically**

**Predefined variables**

**Suffixes for numbers**

**Arrow and movement keys**

**ASCII codes**

# Menu commands

## File menu

| | | |
|---|---|---|
| **New** | [**Ctrl**]**N** | Open new worksheet based on available templates. |
| **Open...** | [**Ctrl**]**O** | Open existing worksheet or template. |
| **Save** | [**Ctrl**]**S** | Save current worksheet or template. |
| **Save As...** | | Save current worksheet or template under new name. |
| **Close** | [**Ctrl**][**F4**] | Close current worksheet or template. |
| **Collaboratory...** | | Open a dialog box for downloading and uploading messages and Mathcad worksheets from MathSoft's Collaboratory server on the Internet. |
| **Internet Setup...** | | Change various settings associated with Internet access. |
| **Send...** | | Mail current worksheet. |
| **Page Setup...** | | Set margins, suppress printing beyond right margin. |
| **Print Preview...** | | Show worksheet as it will appear when printed. |
| **Print...** | | Print worksheet or selected regions. |
| **Exit** | [**Alt**][**F4**] | Exit Mathcad. |

## Edit menu

| | | |
|---|---|---|
| **Undo** | [**Alt**][**BkSp**] | Undo most recent edit. |
| **Redo** | | Undo the last undo. |
| **Cut** | [**Ctrl**]**X** | Cut selection. |
| **Copy** | [**Ctrl**]**C** | Copy selection. |
| **Paste** | [**Ctrl**]**V** | Insert selection most recently copied or cut. |
| **Paste Special** | | Insert selection most recently copied or cut in other available formats. |
| **Delete** | [**Ctrl**]**D** | Delete selection. |
| **Select All** | | Select every region in the worksheet. |

| | | |
|---|---|---|
| **Find...** | [**Ctrl**][**F5**] | Search for math or text characters. |
| **Replace...** | [**Shift**][**F5**] | Search for and replace math or text characters. |
| **Go to Page...** | | Position the top of a specified page at the top of your window. |
| **Check Spelling...** | | Search text for misspelled words. |
| **Links...** | | Edit OLE links in the worksheet. |

## View menu

| | | |
|---|---|---|
| **Math Palette** | | Hide Math Palette. When Palette is hidden, this menu item is checked. |
| **Toolbar** | | Hide Toolbar. When Toolbar is hidden, this item is checked. |
| **Format Bar** | | Hide Format bar. When Format Bar is hidden, this menu item is checked. |
| **Regions** | | Toggle between boxed and unboxed display of regions. |
| **Zoom...** | | Zoom in for a close-up or out for an overall view. |
| **Refresh** | [**Ctrl**]**R** | Force screen redraw. |
| **Animate...** | | Create an animation clip. |
| **Playback...** | | Play an existing animation clip. |

## Insert menu

| | | |
|---|---|---|
| **Graph ⇒** | | |
| **X-Y Plot** | **@** | Create two-dimensional Cartesian plot. |
| **Polar Plot** | [**Ctrl**]**7** | Create a plot for plotting radius against angle. |
| **Surface Plot** | [**Ctrl**]**2** | Create a plot for displaying a surface in three dimensions. |
| **Contour Plot** | [**Ctrl**]**5** | Create a plot for displaying the level curves of a surface. |
| **3D Scatter Plot** | | Create a three-dimensional scatter plot. |
| **3D Bar Chart** | | Create a three-dimensional bar chart. |
| **Vector Field Plot** | | Create a vector field plot for two-dimensional vectors. |

| | | |
|---|---|---|
| **Matrix...** | [**Ctrl**]**M** | Create a new matrix or insert and delete rows or columns from an existing matrix. |
| **Function...** | [**Ctrl**]**F** | Show a scrolling list of available built-in functions. |
| **Unit...** | [**Ctrl**]**U** | Show a scrolling list of available units. |
| **Picture...** | [**Ctrl**]**T** | Insert a region for graphics import. |
| **Math Region** | | Create a math region inside a text region or at the crosshair location. |
| **Text Region** | **"** | Start a new text region at the crosshair location. |
| **Page Break** | | Insert a hard pagebreak. |
| **Hyperlink ⇒** | | |
| **New** | | Create a hypertext link from the current region to another worksheet on a local file system or on the World Wide Web. |
| **Erase** | | Deactivate any hypertext links associated with the current selection. |
| **Edit** | | Edit a hypertext link associated with the current selection. |
| **Reference...** | | Make variable and function definitions from another worksheet available in the current worksheet. |
| **Component...** | | Insert a data or application component into the worksheet at the crosshair. |
| **Object...** | | Embed or link an OLE object from an OLE server on your system. |

## Format menu

| | |
|---|---|
| **Number...** | Change precision displayed, threshold for scientific notation, and other similar characteristics. |
| **Equation...** | Choose font, size, color, and style used to display math expressions. |
| **Text...** | Choose font, size, color, and style of selected text. |
| **Paragraph...** | Choose indenting and justification style of selected text region. |
| **Style...** | Apply, modify, create, or delete text style. |

| | |
|---|---|
| **Properties...** | Change background color or surrounding box of selected region. Apply calculation options to math region. |
| **Color ⇒** | |
| **Background...** | Change color of worksheet background. |
| **Highlight...** | Change color of highlighted regions. |
| **Annotation...** | Change color of any changes made to an Electronic Book. |
| **Use default palette...** | Enable or disable the display of all bitmap images using a default color palette to reduce screen flashing on 256-color displays. When the default palette is used, this menu item is checked. |
| **Optimize palette...** | Create optimal 256-color default palette for display of all bitmap images in the worksheet. |
| **Graph ⇒** | |
| **X-Y Plot...** | Change characteristics for all Cartesian plots in current worksheet. If a plot is selected, changes will affect only the selected plot. |
| **Polar Plot...** | Change characteristics for all polar plots in current worksheet. If a plot is selected, changes will affect only the selected plot. |
| **3D Plot...** | Change characteristics for the currently selected three-dimensional plot. |
| **Trace...** | Read coordinates directly from the currently selected graph. |
| **Zoom...** | Magnify view of a portion of the currently selected graph. |
| **Separate Regions** | Separate overlapping regions. |
| **Align Regions ⇒** | |
| **Across** | Align selected regions to a horizontal line midway between the highest and the lowest region. |
| **Down** | Align selected regions to a vertical line midway between the rightmost and the leftmost region. |
| **Lock Regions ⇒** | |
| **Set Lock Area** | Define extent of a write-protected area on the worksheet. |
| **Lock Area...** | Write protect selected area. |

| | | |
|---|---|---|
| **Unlock Area...** | | Permit editing in a locked area. |
| **Headers/Footers...** | | Specify headers and footers for printouts. |

## Math menu

| | | |
|---|---|---|
| **Calculate** | [**F9**] | Update all results on screen. |
| **Calculate Worksheet** | | Update all results in worksheet. |
| **Automatic Calculation** | | Toggle between automatic calculation mode in which Mathcad updates screen continuously, and manual mode. |
| **Optimization** | | Turn optimize feature on and off. This feature causes Mathcad to attempt to simplify any expression to the right of ":=" or "=". When the optimize feature is on, this menu item is checked. |
| **Options ⇒** | | |
| **Built-In Variables...** | | Set values of built-in variables. Reset random numbers. |
| **Unit System...** | | Choose unit system for the default display of results. |
| **Dimensions...** | | Choose the names of the fundamental dimensions for the current system of units. |

## Symbolics menu

| | | |
|---|---|---|
| **Evaluate ⇒** | | |
| **Symbolically** | [**Shift**][**F9**] | Carry out symbolic evaluation of an expression. |
| **Floating Point...** | | Return a floating point number for constants rather than a symbolic expression. A dialog box allows you to choose the floating point precision. |
| **Complex** | | Carry out symbolic evaluation of a complex expression. The result is expressed in the form $a + b \cdot i$ . |
| **Simplify** | | Simplify the selected expression, performing arithmetic, canceling common factors, and using basic trigonometric and inverse function identities. |

**Expand** — Expand all powers and products of sums in the selected expression.

**Factor** — Factor the selected expression into a product, if the entire expression can be written as a product. To factor a subexpression of a larger expression, select the subexpression.

**Collect** — Collect terms containing like powers of the selected subexpression, which may be a single variable or a function together with its argument. The result is a polynomial in the selected expression.

**Polynomial Coefficients** — Find the coefficients of the expression when written as a polynomial in the selected variable or function.

**Variable ⇒**

- **Solve** — Find the value of the selected variable that makes the expression containing the variable equal to zero. If you select a variable in an equation or inequality, this command solves the equation or inequality for the selected variable.
- **Substitute** — Substitute the contents of the clipboard for each occurrence of a selected variable in an expression. To use this menu command, first put the expression being substituted in the clipboard by selecting and choosing **Copy** or **Cut**. Then select an occurrence of the variable you are substituting for and choose this menu command.
- **Differentiate** — Differentiate the entire expression containing the selected variable with respect to that variable. Other variables are treated as constants.
- **Integrate** — Integrate the entire expression containing the selected variable with respect to that variable.
- **Expand to Series...** — Derive an expansion series for an expression with respect to the variable you have selected. A dialog box allows you to choose the order of the series.
- **Convert to Partial Fraction** — Generate a partial fraction expansion for an expression by factoring the numerator and denominator with respect to the selected variable.

**Matrix ⇒**

- **Transpose** — Return transpose of the selected matrix.

| | |
|---|---|
| **Invert** | Return symbolic inverse of the selected square matrix. |
| **Determinant** | Return symbolic determinant of the selected square matrix. |
| **Transform ⇒** | |
| **Fourier** | Evaluate Fourier transform of expression with respect to the selected variable. Result is in terms of ω. |
| **Inverse Fourier** | Evaluate the inverse Fourier transform of the expression with respect to the selected variable. Result is in terms of *t*. |
| **Laplace** | Evaluate the Laplace transform of the expression with respect to the selected variable. Result is in terms of *s*. |
| **Inverse Laplace** | Evaluate the inverse Laplace transform of the expression with respect to the selected variable. Result is in terms of *t*. |
| **Z** | Evaluate the *z*-transform of the expression with respect to the selected variable. Result is in terms of *z*. |
| **Inverse Z** | Evaluate the inverse *z*-transform of the expression with respect to the selected variable. Result is in terms of *n*. |
| **Evaluation Style...** | Choose the format for symbolic results. A dialog box presents the options, which include vertically stacked display of results and comments, horizontal display, and display without evaluation comments. |

## Window menu

| | |
|---|---|
| **Cascade** | Stack all worksheet windows neatly, with title bars showing. |
| **Tile Horizontal** | Arrange all worksheet windows horizontally so that they don't overlap. |
| **Tile Vertical** | Arrange all worksheet windows vertically so that they don't overlap. |
| **Arrange Icons** | Arrange worksheet icons neatly along lower left edge of application window |

### Help menu

| | | |
|---|---|---|
| **Mathcad Help...** | [F1] | Show index for on-line help. |
| **Resource Center...** | | Open a window for browsing reference content, QuickSheets, and Web-based worksheets and information. |
| **Tip of the Day...** | | Show a series of helpful hints. Displays automatically at Mathcad start-up unless disabled. |
| **Open Book...** | | Open a Mathcad Electronic Book you have installed. |
| **Using Help...** | | Show instructions for using Help. |
| **About Mathcad...** | | Display version number, serial number, and copyright information. |

# Function keys

| Keys | Actions |
|---|---|
| [**F1**] | Help. |
| [**Shift**][**F1**] | Context sensitive help. |
| [**F2**] | Copy selected region to clipboard. |
| [**F3**] | Cut selected region to clipboard. |
| [**F4**] | Paste contents of clipboard. |
| [**Ctrl**][**F4**] | Close worksheet or template. |
| [**F5**] | Open a worksheet or template. |
| [**Ctrl**][**F5**] | Search for text or math characters. |
| [**Shift**][**F5**] | Replace text or math characters. |
| [**F6**] | Save current worksheet. |
| [**Ctrl**][**F6**] | Make next window active. |
| [**F7**] | Open a new worksheet. |
| [**F9**] | Recalculate everything on the screen. With *READ*, *WRITE* or other file I/O function selected, forces Mathcad to read or write to disk. |

# Greek letters

To type a Greek letter into an equation or into text, press the corresponding roman equivalent from the table below, followed by [**Ctrl**]**G**. Alternatively, use the Greek Symbol Palette.

| Name | Uppercase | Lowercase | Roman equivalent |
|---|---|---|---|
| alpha | Α | α | A |
| beta | Β | β | B |
| chi | Χ | χ | C |
| delta | Δ | δ | D |
| epsilon | Ε | ε | E |
| eta | Η | η | H |
| gamma | **G** | γ | G |
| iota | **I** | ι | I |
| kappa | **K** | κ | K |
| lambda | **L** | λ | L |
| mu | **M** | μ | M |
| nu | **N** | ν | N |
| omega | **W** | ω | W |
| omicron | **O** | ο | O |
| phi | **F** | φ | F |
| phi (alternate) | | ϕ | J |
| pi | **P** | π | P |
| psi | **Y** | ψ | Y |
| rho | **R** | ρ | R |
| sigma | **S** | σ | S |
| tau | **T** | τ | T |
| theta | **Q** | θ | Q |
| theta (alternate) | **J** | | J |
| upsilon | **U** | υ | U |
| xi | **X** | ξ | X |
| zeta | **Z** | ζ | Z |

**Note:** In equations, the Greek letter π is so commonly used that it has its own keyboard shortcut: [**Ctrl**]**P**.

# Operators

In this table:

- **A** and **B** represent arrays, either vector or matrix.
- **u** and **v** represent vectors with real or complex elements.
- **M** represents a square matrix.
- *z* and *w* represent real or complex numbers.
- *x* and *y* represent real numbers.
- *m* and *n* represent integers.
- *i* represents a range variable.
- *S* and any names beginning with *S* represent string expressions.
- *t* represents any variable name.
- *f* represents a function.
- *X* and *Y* represent variables or expressions of any type.

For information about programming operators in Mathcad Professional, see Chapter 18, "Programming." For information about symbolic operators and keywords, see Chapter 17, "Symbolic Calculation."

| Operation | Appearance | Keystroke | Description |
|---|---|---|---|
| **Parentheses** | $(X)$ | `'` | Grouping operator. |
| **Vector Subscript** | $\mathbf{v}_n$ | `[` | Returns indicated element of a vector. |
| **Matrix Subscript** | $\mathbf{A}_{m,n}$ | `[` | Returns indicated element of a matrix. |
| **Superscript** | $\mathbf{A}^{\langle n \rangle}$ | `[Ctrl]6` | Extracts column *n* from array **A**. Returns a vector. |
| **Vectorize** | $\vec{X}$ | `[Ctrl]-` | Forces operations in expression *X* to take place element by element. All vectors or matrices in *X* must be the same size. |
| **Factorial** | $n!$ | `!` | Returns $n \cdot (n-1) \cdot (n-2)\ldots$ The integer *n* cannot be negative. |
| **Complex conjugate** | $\bar{X}$ | `"` | Inverts the sign of the imaginary part of *X*. This keystroke creates a string expression in a blank placeholder. |
| **Transpose** | $\mathbf{A}^{\mathrm{T}}$ | `[Ctrl]1` | Returns a matrix whose rows are the columns of **A** and whose columns are the rows of **A**. **A** can be a vector or a matrix. |

<table>
<tr><td>Power</td><td>$z^w$</td><td>^</td><td>Raises $z$ to the power $w$.</td></tr>
<tr><td>Powers of matrix, matrix inverse</td><td>$\mathbf{M}^n$</td><td>^</td><td>$n$th power of square matrix M (using matrix multiplication). $n$ must be a whole number. $\mathbf{M}^{-1}$ is the inverse of M. Other negative powers are powers of the inverse. Returns a square matrix.</td></tr>
<tr><td>Negation</td><td>$-X$</td><td>-</td><td>Multiplies $X$ by $-1$.</td></tr>
<tr><td>Vector sum</td><td>$\Sigma\mathbf{v}$</td><td>[Ctrl]4</td><td>Sums elements of vector v; returns a scalar.</td></tr>
<tr><td>Square root</td><td>$\sqrt{z}$</td><td>\</td><td>Returns positive square root for positive $z$; principal value for negative or complex $z$.</td></tr>
<tr><td>nth root</td><td>$\sqrt[n]{z}$</td><td>[Ctrl]\</td><td>Returns $n$th root of $z$; returns a real valued root whenever possible.</td></tr>
<tr><td>Magnitude, Absolute value</td><td>$|z|$</td><td>|</td><td>Returns $\sqrt{\mathrm{Re}(z)^2 + \mathrm{Im}(z)^2}$.</td></tr>
<tr><td>Magnitude of vector</td><td>$|\mathbf{v}|$</td><td>|</td><td>Returns the magnitude of the vector v: $\sqrt{\mathbf{v}\cdot\mathbf{v}}$ if all elements in v are real. Returns $\sqrt{\mathbf{v}\cdot\bar{\mathbf{v}}}$ if any element in v is complex.</td></tr>
<tr><td>Determinant</td><td>$|\mathbf{M}|$</td><td>|</td><td>Returns the determinant of the square matrix M. Result is a scalar.</td></tr>
<tr><td>Division</td><td>$\frac{X}{z}$</td><td>/</td><td>Divides the expression $X$ by the non-zero scalar $z$. If $X$ is an array, divides each element by $z$.</td></tr>
<tr><td>Multiplication</td><td>$X \cdot Y$</td><td>*</td><td>Returns the product of $X$ and $Y$ if both $X$ and $Y$ are scalars. Multiplies each element of $Y$ by $X$ if $Y$ is an array and $X$ is a scalar. Returns the dot product (inner product) if $X$ and $Y$ are vectors of the same size. Performs matrix multiplication if $X$ and $Y$ are conformable matrices.</td></tr>
<tr><td>Cross product</td><td>$\mathbf{u} \times \mathbf{v}$</td><td>[Ctrl]8</td><td>Returns cross-product (vector product) for the three-element vectors u and v.</td></tr>
<tr><td>Summation</td><td>$\sum_{i=m}^{n} X$</td><td>[Ctrl]<br>[Shift]4</td><td>Performs summation of $X$ over $= m, m + 1, \ldots, r$. $X$ can be any expression. It need not involve $i$ but it usually does. $m$ and $n$ must be integers.</td></tr>
<tr><td>Product</td><td>$\prod_{i=m}^{n} X$</td><td>[Ctrl]<br>[Shift]3</td><td>Performs iterated product of $X$ for $= m, m + 1, \ldots, r$. $X$ can be any expression. It need not involve $i$ but it usually does. $m$ and $n$ must be integers.</td></tr>
<tr><td>Range sum</td><td>$\sum_{i} X$</td><td>$</td><td>Returns a summation of $X$ over the range variable $i$. $X$ can be any expression. It need not involve $i$ but it usually does.</td></tr>
<tr><td>Range product</td><td>$\prod_{i} X$</td><td>#</td><td>Returns the iterated product of $X$ over the range variable $i$. $X$ can be any expression. It need not involve $i$ but it usually does.</td></tr>
</table>

| | | | |
|---|---|---|---|
| **Integral** | $\int_a^b f(t)\,dt$ | **&** | Returns the definite integral of *f*(*t*) over the interval $[a, b]$. *a* and *b* must be real scalars. All variables in the expression *f*(*t*), except the variable of integration *t*, must be defined. The integrand, *f*(*t*), cannot return an array. |
| **Derivative** | $\frac{d}{dt}f(t)$ | **?** | Returns the derivative of *f*(*t*) evaluated at *t*. All variables in the expression *f*(*t*) must be defined. The variable *t* must be a scalar value. The function *f*(*t*) must return a scalar. |
| **nth Derivative** | $\frac{d^n}{dt^n}f(t)$ | **[Ctrl]?** | Returns the *n*th derivative of *f*(*t*) evaluated at *t*. All variables in *f*(*t*) must be defined. The variable *t* must be a scalar value. The function *f*(*t*) must return a scalar. *n* must be an integer between 0 and 5 for numerical evaluation or a positive integer for symbolic evaluation. |
| **Addition** | $X + Y$ | **+** | Scalar addition if *X*, *Y*, or both are scalars. Element by element addition if *X* and *Y* are vectors or matrices of the same size. If *X* is an array and *Y* is a scalar, adds *Y* to each element of *X*. |
| **Subtraction** | $X - Y$ | **–** | Performs scalar subtraction if *X*, *Y*, or both are scalars. Performs element by element subtraction if *X* and *Y* are vectors or matrices of the same size. If *X* is an array and *Y* is a scalar, subtracts *Y* from each element of *X*. |
| **Addition with line break** | $X\ldots$<br>$+Y$ | **[Ctrl][↵]** | Same as addition. Line break is purely cosmetic. |
| **Greater than** | $x > y$,<br>$S1 > S2$ | **>** | For real scalars *x* and *y*, returns 1 if $x > y$, 0 otherwise. For string expressions *S*1 and *S*2, returns 1 if *S*1 strictly follows *S*2 in ASCII order, 0 otherwise. |
| **Less than** | $x < y$,<br>$S1 < S2$ | **<** | For real scalars *x* and *y*, returns 1 if $x < y$, 0 otherwise. For string expressions *S*1 and *S*2, returns 1 if *S*1 strictly precedes *S*2 in ASCII order, 0 otherwise. |
| **Greater than or equal** | $x \geq y$,<br>$S1 \geq S2$ | **[Ctrl]0** | For real scalars *x* and *y*, returns 1 if $x \geq y$, 0 otherwise. For string expressions *S*1 and *S*2, returns 1 if *S*1 follows *S*2 in ASCII order, 0 otherwise. |
| **Less than or equal** | $x \leq y$,<br>$S1 \leq S2$ | **[Ctrl]9** | For real scalars *x* and *y*, returns 1 if $x \leq y$, 0 otherwise. For string expressions *S*1 and *S*2, returns 1 if *S*1 precedes *S*2 in ASCII order, 0 otherwise. |
| **Not equal to** | $z \neq w$,<br>$S1 \neq S2$ | **[Ctrl]3** | For scalars *z* and *w*, returns 1 if $z \neq w$, 0 otherwise.For string expressions *S*1 and *S*2, returns 1 if *S*1 is not character by character identical to *S*2. |
| **Equal to** | $X = Y$ | **[Ctrl]=** | Returns 1 if $X = Y$, 0 otherwise. Appears as a bold = on the screen. |

# Built-in functions listed alphabetically

This section lists Mathcad's built-in functions alphabetically, with a short description of each one. Functions labeled ***Pro*** are available only in Mathcad Professional. For more information, see Chapter 13, "Built-in Functions." Matrix and vector functions are covered in detail in Chapter 10, "Vectors and Matrices." And for more information on Mathcad's differential equation solvers, see Chapter 16, "Solving Differential Equations."

In this table,

- *x* and *y* represent real numbers.
- *z* represents either a real or a complex number.
- *m*, *n*, *i*, *j* and *k* represent integers.
- *S* and any names beginning with *S* represent string expressions.
- **v, u** and any names beginning with **v** represent vectors.
- **A** and **B** represent matrices or vectors.
- **M** and **N** represent square matrices.
- **F** represents a vector-valued function.
- *file* is a string variable that corresponds to a filename or path.

Any function that expects an angle as an argument expects that angle in radians. Similarly, any function that returns an angle as a result, returns the angle in radians. Complex or multivalued functions always return the principal value.

Function names are case sensitive. You must type them as shown here. Names of built-in functions are not, however, font sensitive.

| Function | Returns . . . |
|---|---|
| acos(*z*) | Inverse cosine. Result in radians. Principal value for complex *z*. |
| acosh(*z*) | Inverse hyperbolic cosine. Result in radians. Principal value for complex *z*. |
| angle(*x, y*) | Angle from *x*-axis to (*x, y*). *x* and *y* real. Result in radians. |
| APPEND(*file*) | Single value to append to data file *file*. |
| APPENDPRN(*file*) | Matrix to append to structured data file *file*. |
| arg(*z*) | Angle in complex plane to number *z*. |
| asin(*z*) | Inverse sine. Result in radians. Principal value for complex *z*. |
| asinh(*z*) | Inverse hyperbolic sine. Result in radians. Principal value for complex *z*. |
| atan(*z*) | Inverse tangent. Result in radians. Principal value for complex *z*. |

| | | |
|---|---|---|
| | atanh(*z*) | Inverse hyperbolic tangent. Principal value for complex *z*. |
| | augment(**A**, **B**) | A matrix formed by putting the two argument matrices side by side. **A** and **B** must have the same number of rows. Either or both arguments may be vectors. |
| ***Pro*** | bulstoer(**y**, *x1*, *x2*, *acc*, **D**, *kmx*, *sv*) | Smooth Bulirsch-Stoer differential equation solver, where **y** is the vector of *n* initial values, *n* is the order of the DE or size of the system of DEs, *x1* and *x2* are endpoints of the solution interval, *acc* controls accuracy, **D**(*x*, **y**) prescribes the derivatives, *kmx* is the maximum number of intermediate points, and *sv* is the minimum distance between *x* values (which needn't be equally spaced). Provides good DE solution estimate at *x2*. |
| ***Pro*** | Bulstoer(**y**, *x1*, *x2*, *npts*, **D**) | Smooth Bulirsch-Stoer differential equation solver, where **y** is the vector of *n* initial values, *n* is the order of the DE or size of the system of DEs, *x1* and *x2* are endpoints of the solution interval, *npts* controls the number of rows in the matrix output, and **D**(*x*, **y**) prescribes the derivatives. Provides DE solution at equally spaced *x* values by repeated calls to bulstoer. |
| ***Pro*** | bvalfit(**v1**, **v2**, *x1*, *x2*, *xf*, **D**, **ld1**, **ld2**, **sc**) | Converts a boundary value differential equation to initial/terminal value problems. **v1** is the guess vector for missing initial values at *x1*, **v2** is the guess vector for missing terminal values at *x2*, *x1* and *x2* are endpoints of the solution interval, the *n*-vector **D**(*x*, **y**) prescribes the derivatives, *n* is the order of the DE, the *n*-vector **ld1**(*x1*, **v1**) contains both known initial conditions and guess values from **v1**, the *n*-vector **ld2**(*x2*, **v2**) is similar, the *n*-vector **sc**(*xf*, **y**) measures solution discrepancy at *xf*, where $x1 < xf < x2$. Useful when derivatives have a single discontinuity at *xf*. |
| | ceil(*x*) | Least integer $\geq x$. *x* must be real. |
| | cfft(**A**) | Fast Fourier transform of complex data. Returns an array of same size as its argument. |
| | CFFT(**A**) | Identical to cfft(**A**), except uses a different normalizing factor and sign convention. Returns an array of same size as its argument. |
| ***Pro*** | cholesky(**M**) | A lower triangular matrix **L** such that $\mathbf{L} \cdot \mathbf{L}^{\mathbf{T}} = \mathbf{M}$. This function assumes **M** is symmetric and uses only the upper triangular part of **M**. |
| | cnorm(*x*) | Cumulative normal distribution. Same as pnorm(*x*, 0, 1) |
| | cols(**A**) | Number of columns in array **A**. Returns a scalar. |
| ***Pro*** | concat(*S1*, *S2*) | The string formed by appending string *S2* to the end of string *S1*. |
| ***Pro*** | cond1(**M**) | Condition number of the matrix **M** based on the $L_1$ norm. |
| ***Pro*** | cond2(**M**) | Condition number of the matrix **M** based on the $L_2$ norm. |
| ***Pro*** | conde(**M**) | Condition number of the matrix **M** based on the Euclidean norm. |
| ***Pro*** | condi(**M**) | Condition number of the matrix **M** based on the infinity norm. |

| | | |
|---|---|---|
| | corr(**A**, **B**) | Correlation (Pearson's $r$) of two arrays **A** and **B** having the same number of rows and columns. Returns a scalar. |
| | cos($z$) | Cosine. Argument in radians. |
| | cosh($z$) | Hyperbolic cosine. |
| | cot($z$) | Cotangent. Argument in radians. |
| | coth($z$) | Hyperbolic cotangent. |
| | csc($z$) | Cosecant. Argument in radians. |
| | csch($z$) | Hyperbolic cosecant. |
| | csort(**A**, $n$) | Sort rows so as to put column $n$ in ascending order. |
| | cspline(**vx**, **vy**) | Coefficients of cubic spline with cubic ends. **vx** and **vy** are real vectors of same size. Elements of **vx** must be in ascending order. |
| | cspline(**Mxy**, **Mz**) | Vector of second derivatives for data arrays **Mxy** and **Mz**. This vector becomes the first argument of the *interp* function. The resultant surface is cubic at the edges of the region spanned by **Mxy**. |
| | cvar(**A**, **B**) | Covariance of elements in **A** and **B**. **A** and **B** must have the same number of rows and columns. |
| | dbeta($x$, $s_1$, $s_2$) | Probability density for a beta distribution. |
| | dbinom($k$,$n$,$p$) | Binomial distribution of a random variable. |
| | dcauchy($x$,$l$,$s$) | Probability density for the Cauchy distribution. |
| | dchisq($x$, $d$) | Probability density for the chi-squared distribution. |
| | dexp($x$, $r$) | Probability density for the exponential distribution. |
| | dF($x$, $d1$, $d2$) | Probability density for the F distribution. |
| | dgamma($x$, $s$) | Probability density for the gamma distribution. |
| | dgeom($k$, $p$) | $\mathrm{P}(X = k)$ when the random variable $X$ has the geometric distribution. |
| ***Pro*** | diag(**v**) | Diagonal matrix containing on its diagonal the elements of **v**. |
| | dlnorm($x$, $\mu$, $\sigma$) | Probability density for the lognormal distribution. |
| | dlogis($x$, $l$, $s$) | Probability density for the logistic distribution. |
| | dnbinom($k$, $n$, $p$) | $\mathrm{P}(X = k)$ when the random variable $X$ has the negative binomial distribution. |
| | dnorm($x$, $\mu$, $\sigma$) | Probability density for the normal distribution. |
| | dpois($k$, $\lambda$) | $\mathrm{P}(X = k)$ when the random variable $X$ has the Poisson distribution. |
| | dt($x$, $d$) | Probability density for Student's $t$ distribution. |
| | dunif($x$, $a$, $b$) | Probability density for the uniform distribution. |
| | dweibull($x$, $s$) | Probability density for the Weibull distribution. |

| | | |
|---|---|---|
| | eigenvals(**M**) | A vector of eigenvalues for the matrix **M**. |
| | eigenvec(**M**, *z*) | A vector containing the normalized eigenvector corresponding to the eigenvalue *z* of the square matrix **M**. |
| ***Pro*** | eigenvecs(**M**) | A matrix containing the normalized eigenvectors corresponding to the eigenvalues of the matrix **M**. The *n*th column of the matrix is the eigenvector corresponding to the *n*th eigenvalue returned by *eigenvals*. |
| | erf(*z*) | Error function. |
| ***Pro*** | error(*S*) | The error message string *S*. |
| | exp(*z*) | Exponential: $e^z$ |
| | find(*var1*, *var2*, . . .) | Values of *var1*, *var2*, . . . that solve the system of equations. Returns a scalar if only one argument; otherwise, returns a vector of answers. |
| | fft(**v**) | Fast Fourier transform of real data. **v** must be a real vector with $2^n$ elements, where *n* is an integer. Returns a vector of size $2^{n-1}+1$. |
| | FFT(**v**) | Identical to fft(**v**), except uses a different normalizing factor and sign convention. **v** must be a real vector with $2^n$ elements, where *n* is an integer. Returns a vector of size $2^{n-1}+1$. |
| | floor(*x*) | Greatest integer $\leq x$. *x* must be real. |
| | genfit(**vx**, **vy**, **vg**, **F**) | A vector containing the parameters that make a function *f* of *x* and *n* parameters $u_0, u_1, \ldots, u_r$ best approximate the data in **vx** and **vy**. **F** is a function that returns an $n+1$ element vector containing *f* and its partial derivatives with respect to its *n* parameters. **vx** and **vy** must be the same size. **vg** is an *n* element vector of guess values for the *n* parameters. |
| ***Pro*** | geninv(**A**) | Matrix **L**, the left inverse of matrix **A**, such that $\mathbf{L} \cdot \mathbf{A} = \mathbf{I}$, where **I** is the identity matrix having the same size as **A**. Matrix **A** is an $m \times n$ real-valued matrix, where $m \geq n$. |
| ***Pro*** | genvals(**M**, **N**) | Vector **v** of computed eigenvalues each of which satisfies the generalized eigenvalue problem $\mathbf{M} \cdot \mathbf{x} = v_i \cdot \mathbf{N} \cdot \mathbf{x}$. Matrices **M** and **N** contain real values. Vector **x** contains the corresponding eigenvectors. |
| ***Pro*** | genvecs(**M**, **N**) | A matrix containing the normalized eigenvectors corresponding to the eigenvalues in **v**, the vector returned by *genvals*. The *n*th column of this matrix is the eigenvector **x** satisfying the generalized eigenvalue problem $\mathbf{M} \cdot \mathbf{x} = v_n \cdot \mathbf{N} \cdot \mathbf{x}$. Matrices **M** and **N** contain real values. |
| | hist(**intervals**, **data**) | Histogram. **intervals** is a vector of interval limits, in ascending order. **data** is an array of data. Returns a vector of size one less than the size of **intervals**, showing how many points of **data** fall in each interval. |
| | I0(*x*) | Bessel function $\mathrm{I}_0(x)$. Argument must be real. |

| | | |
|---|---|---|
| | I1(*x*) | Bessel function $I_1(x)$ . Argument must be real. |
| | In(*m*, *x*) | Bessel function $I_m(x)$ . *x* must be real; $0 \le m \le 100$ . |
| | icfft(**A**) | Inverse Fourier transform corresponding to *cfft*. Returns an array of the same size as its argument. |
| | ICFFT(**A**) | Inverse transform corresponding to *CFFT*. Returns an array of the same size as its argument. |
| | identity(*n*) | Identity matrix of size *n*. *n* must be a positive integer. |
| | if(*cond*, *x*, *y*) | Conditional: returns *x* or *y* depending on value of *cond*. If *cond* is true (non-zero), returns *x*. If *cond* is false (zero), returns *y*. |
| | ifft(**v**) | Inverse Fourier transform corresponding to *fft*. Takes a vector of size $1 + 2^{n-1}$ , where *n* is an integer. Returns a real vector of size $2^n$ . |
| | IFFT(**v**) | Inverse transform corresponding to *FFT*. Takes a vector of size $1 + 2^{n-1}$ , where *n* is an integer. Returns a real vector of size $2^n$ . |
| | Im(*z*) | Imaginary part of complex number *z*. Also works on vectors and matrix arguments. |
| | intercept(**vx**, **vy**) | Intercept of regression line. Takes two vector arguments **vx** and **vy** of same size. The elements of **vx** must be in ascending order. Returns a scalar: the *y*-intercept of the regression line. |
| | interp(**vs**, **vx**, **vy**, *x*) | Interpolated value from spline coefficients. Takes three vector arguments **vs**, **vx**, and **vy** of same size and a scalar *x* at which to interpolate; returns a scalar. The elements of **vx** should be in ascending order. **vs** should be a vector computed with *cspline*, *lspline*, or *pspline*. |
| | interp(**vs**, **Mxy**, **Mz**, *v*) | Spline interpolated value of **Mz** at the *x* and *y* coordinates specified in **v**. |
| ***Pro*** | iwave(**v**) | Inverse wavelet transform corresponding to *wave*. Takes a $2^n$ element vector of real data, where *n* is an integer. |
| | J0(*x*) | Bessel function $J_0(x)$ . Argument must be real. |
| | J1(*x*) | Bessel function $J_1(x)$ . Argument must be real. |
| | Jn(*m*, *x*) | Bessel function $J_m(x)$ . *x* must be real; $0 \le m \le 100$ . |
| | K0(*x*) | Bessel function $K_0(x)$ . Argument must be positive. |
| | K1(*x*) | Bessel function $K_1(x)$ . Argument must be positive. |
| | Kn(*m*, *x*) | Bessel function $K_m(x)$ . *x* must be positive; $0 \le m \le 100$ . |
| ***Pro*** | ksmooth(**vx**,**vy**, *b*) | 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. |
| | last(**v**) | Index of last element in vector **v**. Returns a scalar. |
| | length(**v**) | Number of elements in vector **v**. Returns a scalar. |

| | | |
|---|---|---|
| | linfit(**vx**, **vy**, **F**) | A vector containing the coefficients used to create a linear combination of the functions in **F** which best approximates the data in **vx** and **vy**. **F** is a function that returns a vector. |
| | linterp(**vx**, **vy**, *x*) | Linearly interpolated value. Takes two vector arguments **vx** and **vy** of same size and a scalar *x* at which to interpolate; returns a scalar. The elements of **vx** should be in ascending order. |
| | ln(*z*) | Natural logarithm of *z* (to base *e*). Returns principal value (imaginary part between $\pi$ and $-\pi$) for complex *z*. |
| ***Pro*** | loess(**vx**, **vy**, *span*) | Vector required by the *interp* function to find the set of second order polynomials that best fit particular neighborhoods of data points specified in arrays **vx** and **vy**. **vx** is an *m* element vector containing *x* coordinates. **vy** is an *m* element vector containing the *y* coordinates corresponding to the *m* points specified in **vx**. The argument *span* ($span > 0$) specifies how large a neighborhood *loess* will consider in performing this local regression. |
| ***Pro*** | loess(**Mxy**, **vz**, *span*) | Vector required by the *interp* function to find the set of second order polynomials that best fit particular neighborhoods of data points specified in arrays **Mxy** and **vz**. **Mxy** is an $m \times 2$ matrix containing *x*-*y* coordinates. **vz** is an *m* element vector containing the *z* coordinates corresponding to the *m* points specified in **Mxy**. The argument *span* ($span > 0$) specifies how large a neighborhood *loess* will consider in performing this local regression. |
| | log(*z*) | Common logarithm of *z* (to base 10). |
| ***Pro*** | lsolve(**M**, **v**) | Solution vector **x** such that $\mathbf{M} \cdot \mathbf{x} = \mathbf{v}$. |
| | lspline(**vx**, **vy**) | Coefficients of cubic spline with linear ends. **vx** and **vy** are real vectors of same size. Elements of **vx** must be in ascending order. |
| | lspline(**Mxy**, **Mz**) | Vector of second derivatives for data arrays **Mxy** and **Mz**. This vector becomes the first argument of the *interp* function. The resultant surface is linear at the boundaries of the region spanned by **Mxy**. |
| ***Pro*** | lu(**M**) | One 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 $\mathbf{P} \cdot \mathbf{M} = \mathbf{L} \cdot \mathbf{U}$. **L** and **U** are lower and upper triangular respectively. |
| | matrix(*m*, *n*, *f*) | Creates a matrix in which the *ij*th element contains $f(i, j)$ where $= 0, 1, \ldots, m-1$ and $= 0, 1, \ldots, n-1$. |
| | max(**A**) | Largest element in **A**. Returns a scalar. If **A** is complex, returns max(Re(**A**)) + *i*·max(Im(**A**)). |
| | mean(**A**) | Mean of elements of an array **A**. Returns a scalar. |
| | median(**A**) | Median of elements in array **A**. Returns a scalar. |
| | medsmooth(**vy**,*n*) | An *m*-element vector created by smoothing **vy** with running medians. **vy** is an *m*-element vector of real numbers. The smoothing window has size *n*. |

| | | |
|---|---|---|
| | min(**A**) | Smallest element in **A**. Returns a scalar. If **A** is complex, returns min(Re(**A**)) + *i*.min(Im(**A**)). |
| | minerr(*var1, var2, . . .*) | Values of *var1, var2, . . .* coming closest to solving the system of equations. Returns a scalar if only one argument; otherwise, returns a vector of answers. |
| | mod(*x, modulus*) | Remainder on dividing *x* by *modulus*. Arguments must be real. Result has same sign as *x*. |
| ***Pro*** | multigrid(**M**, *ncycle*) | Solves the Poisson partial differential equation over a planar square region. The $n \times n$ matrix **M** gives source function values, where $n - 1$ is a power of 2 and zero boundary conditions on all four edges are assumed. Cycle control variable *ncycle* is usually 1 or 2. Different algorithm and faster than relax, which is more general. |
| ***Pro*** | norm1(**M**) | The $L_1$ norm of the matrix **M**. |
| ***Pro*** | norm2(**M**) | The $L_2$ norm of the matrix **M**. |
| ***Pro*** | norme(**M**) | The Euclidean norm of the matrix **M**. |
| ***Pro*** | normi(**M**) | The infinity norm of the matrix **M**. |
| ***Pro*** | num2str(*z*) | The string whose characters correspond to the decimal value of the number *z*. |
| | pbeta($x$, $s_1$, $s_2$) | Cumulative beta distribution with shape parameters $s_1$ and $s_2$. $s_1, s_2 > 0$ |
| | pbinom(*k*, *n*, *p*) | Cumulative binomial distribution for *k* successes in *n* trials. |
| | pcauchy(*x*, *l*, *s*) | Cumulative Cauchy distribution with scale parameters *l* and *s*. |
| | pchisq(*x*, *d*) | Cumulative chi-squared distribution in which $d > 0$ is the degrees of freedom and $x > 0$. |
| | pexp(*x*, *r*) | Cumulative exponential distribution in which $r > 0$ is the rate and $x > 0$. |
| | pF($x$, $d_1$, $d_2$) | Cumulative F distribution in which $d_1, d_2 > 0$ are the degrees of freedom. $x > 0$. |
| | pgamma(*x*, *s*) | Cumulative gamma distribution in which $s > 0$ is the shape parameter. $x > 0$. |
| | pgeom(*k*, *p*) | Cumulative geometric distribution. *p* is the probability of success. $k \geq 0$ and $0 < p \leq 1$. |
| | plnorm($x$, $\mu$, $\sigma$) | Cumulative lognormal distribution in which $\mu$ is the logmean, $\sigma$ is the logdeviation, and $x > 0$. |
| | plogis(*x*, *l*, *s*) | Cumulative logistic distribution. *l* is the location parameter. $s > 0$ is the scale parameter. |
| | pnbinom(*k*, *n*, *p*) | Cumulative negative binomial distribution in which $n > 0$ and $0 < p \leq 1$. |
| | pnorm($x$, $\mu$, $\sigma$) | Cumulative normal distribution with mean $\mu$ and standard deviation $\sigma$. |

| | | |
|---|---|---|
| | polyroots(**v**) | Roots of the *n*th degree polynomial whose coefficients are in **v**, a vector of length $n + 1$. |
| | ppois($k$, $\lambda$) | Cumulative Poisson distribution. $\lambda > 0$. |
| ***Pro*** | predict(**v**, $m$, $n$) | A vector of $n$ predicted values based on $m$ consecutive elements in **v,** a vector whose values represent samples taken at equal intervals. |
| | pspline(**vx**, **vy**) | Coefficients of cubic spline with parabolic ends. **vx** and **vy** are real vectors of same size. Elements of **vx** must be in ascending order. |
| | pspline(**Mxy**, **Mz**) | Vector of second derivatives for data arrays **Mxy** and **Mz**. This vector becomes the first argument of the *interp* function. The resultant surface is parabolic at the boundaries of the region spanned by **Mxy**. |
| | pt($x$, $d$) | Cumulative Student's $t$ distribution. $d$ is the degrees of freedom. $x > 0$ and $d > 0$. |
| | punif($x$, $a$, $b$) | Cumulative uniform distribution. $b$ and $a$ are the endpoints of the interval. $a < b$. |
| | pweibull($x$, $s$) | Cumulative Weibull distribution. $s > 0$. |
| | qbeta($p$, $s_1$, $s_2$) | Inverse beta distribution with shape parameters $s_1$ and $s_2$. $0 \le p \le 1$ and $s_1, s_2 > 0$. |
| | qbinom($p$, $n$, $q$) | Number of successes in $n$ trials of the Bernoulli process such that the probability of that number of successes is $p$. $q$ is the probability of success on a single trial. $0 \le q \le 1$ and $0 \le p \le 1$. |
| | qcauchy($p$, $l$, $s$) | Inverse Cauchy distribution with scale parameters $l$ and $s$. $s > 0$ and $0 < p < 1$. |
| | qchisq($p$, $d$) | Inverse chi-squared distribution in which $d > 0$ is the degrees of freedom. $0 \le p < 1$. |
| | qexp($p$, $r$) | Inverse exponential distribution in which $r > 0$ is the rate. $0 \le p < 1$. |
| | qF($p$, $d_1$, $d_2$) | Inverse F distribution in which $d_1, d_2 > 0$ are the degrees of freedom. $0 \le p < 1$. |
| | qgamma($p$, $s$) | Inverse gamma distribution in which $s > 0$ is the shape parameter. $0 \le p < 1$. |
| | qgeom($p$, $q$) | Inverse geometric distribution. $q$ is the probability of success on a single trial. $0 < p < 1$ and $0 \le q < 1$. |
| | qlnorm($p$, $\mu$, $\sigma$) | Inverse log normal distribution in which $\mu$ is the log of the mean, $\sigma > 0$ is the log of the standard deviation. $0 \le p < 1$. |
| | qlogis($p$, $l$, $s$) | Inverse logistic distribution. $l$ is the location parameter. $s > 0$ is the scale parameter. $0 \le p < 1$. |
| | qnbinom($p$, $n$, $q$) | Inverse negative binomial distribution with size $n$ and probability of failure $q$. $0 \le q \le 1$ and $0 \le p \le 1$. |

| | | |
|---|---|---|
| | qnorm($p$, $\mu$, $\sigma$) | Inverse normal distribution with mean $\mu$ and standard deviation $\sigma$. $0 < p < 1$ and $\sigma > 0$. |
| | qpois($p$, $\lambda$) | Inverse Poisson distribution. $\lambda > 0$ and $0 \le p \le 1$. |
| ***Pro*** | qr(**A**) | 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 $\mathbf{A} = \mathbf{Q} \cdot \mathbf{R}$, where **A** is a real-valued array. |
| | qt($p$, $d$) | Inverse Student's *t* distribution. *d* is the degrees of freedom. $d > 0$ and $0 < p < 1$. |
| | qunif($p$, $a$, $b$) | Inverse uniform distribution. *b* and *a* are the endpoints of the interval. $a < b$ and $0 \le p \le 1$. |
| | qweibull($p$, $s$) | Inverse Weibull distribution. $s > 0$ and $0 < p < 1$. |
| | rank(**A**) | The rank of real-valued matrix **A**. |
| | rbeta($m$, $s_1$, $s_2$) | Vector of *m* random numbers having the beta distribution. $s_1, s_2 > 0$ are the shape parameters. |
| | rbinom($m$, $n$, $p$) | Vector of *m* random numbers having the binomial distribution. $0 \le p \le 1$. *n* is an integer satisfying $n > 0$. |
| | rcauchy($m$, $l$, $s$) | Vector of *m* random numbers having the Cauchy distribution. *l* and $s > 0$ are scale parameters. |
| | rchisq($m$, $d$) | Vector of *m* random numbers having the chi-squared distribution. $d > 0$ is the degrees of freedom. |
| | Re($z$) | Real part of complex number *z*. |
| | READ(*file*) | Single value read from data file *file*. |
| ***Pro*** | READ_BLUE(*file*) | Matrix corresponding to the blue color component in image file *file*. |
| | READBMP(*file*) | Matrix containing a grayscale representation of the image in BMP image file *file*. |
| ***Pro*** | READ_GREEN(*file*) | Matrix corresponding to the green color component in image file *file*. |
| ***Pro*** | READ_HLS(*file*) | Matrix in which the color information in image file *file* is represented by the appropriate values of hue, saturation, and value. |
| ***Pro*** | READ_HLS_HUE(*file*) | Matrix corresponding to the hue color component in image file *file*. |
| ***Pro*** | READ_HLS_LIGHT(*file*) | Matrix corresponding to the lightness color component in image file *file*. |
| ***Pro*** | READ_HLS_SAT(*file*) | Matrix corresponding to the saturation color component in image file *file*. |
| ***Pro*** | READ_HSV(*file*) | Matrix in which the color information in image file *file* is represented by the appropriate values of hue, lightness, and saturation. |
| ***Pro*** | READ_HSV_HUE(*file*) | Matrix corresponding to the hue color component in image file *file*. |

| | | |
|---|---|---|
| ***Pro*** | READ_HSV_SAT(*file*) | Matrix corresponding to the saturation color component in image file *file*. |
| ***Pro*** | READ_HSV_VALUE(*file*) | Matrix corresponding to the value color component in image file *file*. |
| ***Pro*** | READ_IMAGE(*file*) | Matrix containing a grayscale representation of the image file *file*. *file* may be in BMP, GIF, JPG, or TGA format. |
| | READPRN(*file*) | Matrix read from structured data file *file*. |
| ***Pro*** | READ_RED(*file*) | Matrix corresponding to the red color component in image file *file*. |
| | READRGB(*file*) | Array in which the color information in image file *file* is represented by the appropriate values of red, green, and blue. This array is formed by combining the three arrays giving the red, green, and blue components of the image into a single array with three times as many columns as the image. |
| | regress(**vx**, **vy**, *n*) | Vector required by the *interp* function to find the *n*th order polynomial that best fits data arrays **vx** and **vy**. **vx** is an *m* element vector containing *x* coordinates. **vy** is an *m* element vector containing the *y* coordinates corresponding to the *m* points specified in **vx**. |
| | regress(**Mxy**,**vz**, *n*) | Vector required by the *interp* function to find the *n*th order polynomial that best fits data arrays **Mxy** and **vz**. **Mxy** is an $m \times 2$ matrix containing *x*-*y* coordinates. **vz** is an *m* element vector containing the *z* coordinates corresponding to the *m* points specified in **Mxy**. |
| ***Pro*** | relax(**A, B, C, D, E, F**, **U**, *rjac*) | Solves the Poisson partial differential equation over a planar square region. $n \times n$ matrices **A**, **B**, **C**, **D**, and **E** specify coefficients for linearly approximating the Laplacian operator at each of $n^2$ gridpoints, $n \times n$ matrix **F** gives source function values, $n \times n$ matrix **U** prescribes boundary values along all four edges and guesses for interior values, and 0 < *rjac* <1 is the Jacobi spectral radius. More general than multigrid, which is faster. |
| | reverse(**v**) | Reverse order of elements in **v**. |
| | rexp(*m*, *r*) | Vector of *m* random numbers having the exponential distribution. $r > 0$ is the rate. |
| | rF($m$, $d_1$, $d_2$) | Vector of *m* random numbers having the F distribution. $d_1$, $d_2 > 0$ are the degrees of freedom. |
| | rgamma(*m*, *s*) | Vector of *m* random numbers having the gamma distribution. $s > 0$ is the shape parameter. |
| | rgeom(*m*, *p*) | Vector of *m* random numbers having the geometric distribution. $0 < p \leq 1$ . |

| | | |
|---|---|---|
| ***Pro*** | rkadapt(**y**, *x1*, *x2*, *acc*, **D**, *kmx*, *sv*) | Slowly varying Runge-Kutta differential equation solver, where **y** is the vector of *n* initial values, *n* is the order of the DE or size of the system of DEs, *x1* and *x2* are endpoints of the solution interval, *acc* controls accuracy, **D**(*x*, **y**) prescribes the derivatives, *kmx* is the maximum number of intermediate points, and *sv* is the minimum distance between *x* values (which needn't be equally spaced). Provides good DE solution estimate at *x2*. |
| ***Pro*** | Rkadapt(**y**, *x1*, *x2*, *npts*, **D**) | Slowly varying Runge-Kutta differential equation solver, where **y** is the vector of *n* initial values, *n* is the order of the DE or size of the system of DEs, *x1* and *x2* are endpoints of the solution interval, *npts* controls the number of rows in the matrix output, and **D**(*x*, **y**) prescribes the derivatives. Provides DE solution at equally-spaced *x* values by repeated calls to rkadapt. |
| | rkfixed(**y**, *x1*, *x2*, *npts*, **D**) | Standard Runge-Kutta differential equation solver, where **y** is the vector of *n* initial values, *n* is the order of the DE or size of the system of DEs, *x1* and *x2* are the endpoints of the solution interval, *npts* controls the number of rows in the matrix output, and **D**(*x*, **y**) prescribes the derivatives. Provides DE solution at equally spaced *x* values. |
| | rlnorm(*m*, $\mu$, $\sigma$ ) | Vector of *m* random numbers having the lognormal distribution in which $\mu$ is the logmean and $\sigma > 0$ is the logdeviation. |
| | rlogis(*m*, *l*, *s*) | Vector of *m* random numbers having the logistic distribution in which *l* is the location parameter and $s > 0$ is the scale parameter. |
| | rnbinom(*m*, *n*, *p*) | Vector of *m* random numbers having the negative binomial distribution. $0 < p \le 1$ . *n* is an integer satisfying $n > 0$ . |
| | rnd(*x*) | Random number between 0 and *x*, *x* real. Identical to runif(1, 0, *x*) if $x > 0$ . |
| | rnorm(*m*, $\mu$, $\sigma$ ) | Vector of *m* random numbers having the normal distribution with mean $\mu$ and standard deviation $\sigma > 0$ . |
| | root(*expr*, *var*) | Value of *var* where *expr* is zero. |
| | rows(**A**) | Number of rows in array **A**. Returns a scalar. |
| | rpois(*m*, $\lambda$) | Vector of *m* random numbers having the Poisson distribution. $\lambda > 0$. |
| | rref(**A**) | A matrix representing the row-reduced echelon form of **A**. |
| | rsort(**A**, *n*) | Sort columns so as to put row *n* in ascending order. |
| | rt(*m*, *d*) | Vector of *m* random numbers having Student's *t* distribution. $d > 0$ . |
| | runif(*m*, *a*, *b*) | Vector of *m* random numbers having the uniform distribution in which *b* and *a* are the endpoints of the interval and $a < b$ . |
| | rweibull(*m*, *s*) | Vector of *m* random numbers having the Weibull distribution in which $s > 0$ is the shape parameter. |

| | | |
|---|---|---|
| ***Pro*** | sbval(**v**, *x1*, *x2*, **D**, **ld**, **sc**) | Converts a boundary value differential equation to an initial value problem. **v** is the guess vector for missing initial values, *x1* and *x2* are endpoints of the solution interval, the *n*-vector **D**(*x*, **y**) prescribes the derivatives, *n* is the order of the DE, the *n*-vector **ld**(*x1*, **v**) contains both known initial conditions and guess values from **v**, and the *n*-vector **sc**(*x2*, **y**) measures solution discrepancy at *x2*. Good when derivatives are continuous throughout. |
| | sec(*z*) | Secant. Argument in radians. |
| | sech(*z*) | Hyperbolic secant. |
| ***Pro*** | search(*S1*, *SubS*, *m*) | The starting position of the substring *SubS* in *S1* beginning from position *m*. Returns –1 if the substring is not found. $m \geq 0$. |
| | sin(*z*) | Sine. Argument in radians. |
| | sinh(*z*) | Hyperbolic sine. |
| | slope(**vx**, **vy**) | Slope of regression line. Takes two vector arguments **vx** and **vy** of the same size. The elements of **vx** must be in ascending order. |
| | sort(**v**) | Sort elements in vector **v**. |
| | stack(**A**, **B**) | Array formed by placing **A** above **B**. The arrays **A** and **B** must have the same number of columns. |
| | stdev(**A**) | Standard deviation of elements of **A**. Uses *n* in the denominator. Returns a scalar. |
| | Stdev(**A**) | Sample standard deviation of elements of **A**. Uses $n - 1$ in the denominator. Returns a scalar. |
| ***Pro*** | stiffb(**y**, *x1*, *x2*, *acc*, **D**, **J**, *kmx*, *sv*) | Stiff Bulirsch-Stoer differential equation solver, where **y** is the vector of n initial values, *n* is the order of the DE or size of the system of DEs, *x1* and *x2* are endpoints of the solution interval, *acc* controls accuracy, **D**(*x*, **y**) prescribes the derivatives, **J**(*x*, **y**) is the Jacobian matrix prescribing the second derivatives, *kmx* is the maximum number of intermediate points, and *sv* is the minimum distance between *x* values (which needn't be equally spaced). Provides good DE solution estimate at *x2*. |
| ***Pro*** | Stiffb(**y**, *x1*, *x2*, *npts*, **D**, **J**) | Stiff Bulirsch-Stoer differential equation solver, where **y** is the vector of *n* initial values, *n* is the order of the DE or size of the system of DEs, *x1* and *x2* are endpoints of the solution interval, *npts* controls the number of rows in the matrix output, **D**(*x*, **y**) prescribes the derivatives, and **J**(*x*, **y**) is the Jacobian matrix prescribing the second derivatives. Provides DE solution at equally-spaced *x* values by repeated calls to stiffb. |
| ***Pro*** | stiffr(**y**, *x1*, *x2*, *acc*, **D**, **J**, *kmx*, *sv*) | Stiff Rosenbrock differential equation solver, where **y** is the vector of n initial values, *n* is the order of the DE or size of the system of DEs, *x1* and *x2* are endpoints of the solution interval, *acc* controls accuracy, **D**(*x*, **y**) prescribes the derivatives, **J**(*x*, **y**) is the Jacobian matrix prescribes the second derivatives, *kmx* is the maximum number of intermediate points, and *sv* is the minimum distance between *x* values (which needn't be equally spaced). Provides good DE solution estimate at *x2*. |

| | | |
|---|---|---|
| ***Pro*** | Stiffr(**y**, *x1*, *x2*, *npts*, **D**, **J**) | Stiff Rosenbrock differential equation solver, where **y** is the vector of *n* initial values, *n* is the order of the DE or size of the system of DEs, *x1* and *x2* are endpoints of the solution interval, *npts* controls the number of rows in the matrix output, **D**(*x*, **y**) prescribes the derivatives, and **J**(*x*, **y**) is the Jacobian matrix prescribing the second derivatives. Provides DE solution at equally spaced *x* values by repeated calls to stiffr. |
| ***Pro*** | str2num(*S*) | A numerical constant formed by converting the characters in *S* into a number. Characters in *S* must constitute a real, complex, floating point, or e-format number. Spaces are ignored. |
| ***Pro*** | str2vec(*S*) | A vector of ASCII codes corresponding to the characters in string *S*. |
| ***Pro*** | strlen(*S*) | The number of characters in string *S*. |
| | submatrix(**A**, *ir*, *jr*, *ic*, *jc*) | Submatrix of **A** consisting of all elements common to rows *ir* through *jr* and columns *ic* through *jc*. To maintain order of rows and/or columns, make sure $ir \le jr$ and $ic \le jc$, otherwise order of rows and/or columns will be reversed. |
| ***Pro*** | substr(*S*, *m*, *n*) | A substring of *S* beginning with the character in the *m*th position and having at most *n* characters. $m, n \ge 0$. |
| ***Pro*** | supsmooth(**vx**, **vy**) | An *n*-element vector created by the piecewise use of a symmetric *k*-nearest neighbor linear least square 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. |
| ***Pro*** | svd(**A**) | One 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 $\mathbf{A} = \mathbf{U} \cdot \text{diag}(\mathbf{s}) \cdot \mathbf{V}^{\mathbf{T}}$, where **s** is the vector returned by svds(**A**). **A** is an $m \times n$ array of real values, where $m \ge n$. |
| ***Pro*** | svds(**A**) | A vector containing the singular values of the $m \times n$ real-valued array **A**, where $m \ge n$. |
| | tan(*z*) | Tangent. Argument in radians. |
| | tanh(*z*) | Hyperbolic tangent. |
| | tr(**M**) | Trace of square matrix **M**: sum of diagonal elements. |
| | until(*x*, *y*) | Returns *y* until *x* is negative. |
| | var(**A**) | Variance of elements of **A**. Uses *n* in the denominator. Returns a scalar. |
| | Var(**A**) | Sample variance of elements of **A**. Uses $n - 1$ in the denominator. Returns a scalar. |
| ***Pro*** | vec2str(**v**) | The string formed by converting the vector **v** of ASCII codes to characters. The elements of **v** must be integers between 0 and 255. |
| ***Pro*** | wave(**v**) | Discrete wavelet transform of real data using Daubechies four-coefficient wavelet filter. Vector **v** must contain $2^n$ real values, where *n* is an integer. |

| | | |
|---|---|---|
| | WRITE(*file*) | Single value written to a data file *file*. |
| | WRITEBMP(*file*) | Grayscale BMP image file *file* out of a matrix. |
| ***Pro*** | WRITE_HLS(*file*) | Color BMP image file *file* out of an array formed by juxtaposing the three arrays giving the hue, lightness, and saturation values of an image. |
| ***Pro*** | WRITE_HSV(*file*) | Color BMP image file *file* out of an array formed by juxtaposing the three arrays giving the hue, saturation, and value components of an image. |
| | WRITEPRN(*file*) | Structured data file out of a matrix. |
| | WRITERGB(*file*) | Colored BMP file out of an array formed by juxtaposing the three arrays giving the red, green, and blue values of an image. |
| | Y0(*x*) | Bessel function $Y_0(x)$ . Argument must be positive. |
| | Y1(*x*) | Bessel function $Y_1(x)$ . Argument must be positive. |
| | Yn(*m*, *x*) | Bessel function $Y_m(x)$ . $x$ must be positive; $0 \le m \le 100$ . |
| | δ(*x*, *y*) | Kronecker delta function. Returns 1 if $x=y$; otherwise, returns 0. (To type δ, press **d**[**Ctrl**]**G**) |
| | ε(*i*, *j*, *k*) | Completely antisymmetric tensor of rank three. *i*, *j*, and *k* must be integers between 0 and 2 (or between ORIGIN and ORIGIN+2). Result is 0 if any two are the same, 1 if the three arguments are an even permutation of (0 1 2), and –1 if the arguments are an odd permutation of (0 1 2). (To type ε, press **e**[**Ctrl**]**G**) |
| | Γ(*z*) | Euler's gamma function. (To type Γ, press **G**[**Ctrl**]**G**) |
| | Φ(*x*) | Heaviside step function. Returns 1 if $x \ge 0$; otherwise, returns 0. (To type Φ, press **F**[**Ctrl**]**G**) |

# Predefined variables

Mathcad's predefined variables are listed here, together with their default starting values.

| Constant=Value | Meaning |
|---|---|
| $\pi = 3.14159...$ | Pi. Mathcad uses the value of π to 15 digits. To type π, press [**Ctrl**]**p**. |
| $e = 2.71828...$ | The base of natural logarithms. Mathcad uses the value of *e* to 15 digits. |
| $\infty = 10^{307}$ | Infinity. This symbol represents values larger than the largest real number representable in Mathcad (about $10^{307}$ ). Do not use this variable in place of actual infinities in numerical formulas. To type ∞, press [**Ctrl**]**Z**. |
| % = 0.01 | Percent. Use in expressions like **10*%** (appears as $10 \cdot \%$ ) or as a scaling unit at the end of an equation with an equals sign. |
| **TOL** = $10^{-3}$ | Tolerance. The tolerance Mathcad uses in numerical approximation algorithms (integrals, equation solving, etc.). For more information, see the section on the specific operation in question. |
| **ORIGIN** = 0 | Array origin. Specifies the index of the first element in arrays. |
| **PRNCOLWIDTH** = 8 | Column width used in writing files with *WRITEPRN* function. |
| **PRNPRECISION** = 4 | Number of significant digits used when writing files with the *WRITEPRN* function. |
| **FRAME** = 0 | Used as a counter for creating animation clips. |
| ***Pro*** **CWD** = "<system path>" | String corresponding to the working directory of the worksheet. |
| ***Pro*** **in***n* = 0, **out***n* = 0 | Input and output variables (**in0**, **in1**, **out0**, **out1**, etc.) in a Mathcad component in a MathConnex system. See the *MathConnex Getting Started Guide* for details. |

## Suffixes for numbers

The table below shows how Mathcad interprets numbers. (A number is any sequence of characters beginning with a digit.)

| Suffix | Examples | Meaning |
|---|---|---|
| i *or* j | 4i, 1j, 3 + 1.5j | Imaginary |
| H *or* h | 0aH, 8BCh | Hexadecimal |
| O *or* o | 57O, 100o | Octal |
| L *or* l | 1L, –2.54l | Standard length unit |
| M *or* m | 1M, 2.2m | Standard mass unit |
| T *or* t | 1T, 3600t | Standard time unit |
| Q *or* q | 1Q, –100q | Standard charge unit |
| K *or* k | 1K, –273k | Standard absolute temperature unit |
| U *or* u | 1U, 125u | Standard luminosity unit in SI unit system |
| S *or* s | 1S, 6.97s | Standard substance unit in SI unit system |

# *Arrow and movement keys*

| Keys | Actions |
|---|---|
| [↑] | Move crosshair up.<br>In math: move editing lines up.<br>In text: move insertion point up to previous line. |
| [↓] | Move crosshair down.<br>In math: move editing lines down.<br>In text: move insertion point down to next line. |
| [←] | Move crosshair left.<br>In math: select left operand.<br>In text: move insertion point one character to the left. |
| [→] | Move crosshair right.<br>In math: select right operand.<br>In text: move insertion point one character to the right. |
| [**Shift**][↑] | In math: move crosshair outside and above equation.<br>In text: highlight from insertion point up to previous line. |
| [**Shift**][↓] | In math: move crosshair outside and below equation.<br>In text: highlight from insertion point down to next line. |
| [**Shift**][←] | In math: move crosshair outside and to the left of equation.<br>In text: highlight towards the left of the insertion point, character by character. |
| [**Shift**][→] | In math: move crosshair outside and to the right of equation.<br>In text: highlight towards the right of the insertion point, character by character. |
| [**Ctrl**][↑] | In text: move insertion point to the beginning of a line. |
| [**Ctrl**][↓] | In text: move insertion point to the beginning of next line. |
| [**Ctrl**][←] | In text: move insertion point left to the beginning of a word. |
| [**Ctrl**][→] | In text: move insertion point to the beginning of next word. |
| [**Ctrl**][**Shift**][↑] | In text: highlight from insertion point up to the beginning of a line. |
| [**Ctrl**][**Shift**][↓] | In text: highlight from insertion point to end of the current line. |
| [**Ctrl**][**Shift**][←] | Highlight left from insertion point to the beginning of a word. |
| [**Ctrl**][**Shift**][→] | Highlight from insertion point to beginning of the next word. |

| Keys | Actions |
|---|---|
| [**Space**] | In math: cycles through different states of the editing lines. |
| [**Tab**] | In text: inserts a five-character space.<br>In array or plot: move to next placeholder. |
| [**Shift**][**Tab**] | In array or plot: move to previous placeholder. |
| [**PgUp**] | Move up 5 lines. |
| [**PgDn**] | Move down 5 lines. |
| [**Ctrl**][**PgUp**] | Move 80% up the window. |
| [**Ctrl**][**PgDn**] | Move 80% down the window. |
| [**Shift**][**PgUp**] | Move up to previous pagebreak. |
| [**Shift**][**PgDn**] | Move down to next pagebreak. |
| [**Home**] | In equation, move to beginning previous region. In text, move to beginning of current line. |
| [**End**] | In equation, move to next region. In text, move to end of current line. |
| [**Ctrl**][**Home**] | Scroll to beginning of worksheet. In text, move insertion point to beginning of text region or paragraph. |
| [**Ctrl**][**End**] | Scroll to end of worksheet. In text, move insertion point to end of text region or paragraph. |
| [↵] | In text: start new line.<br>In equation or plot: move crosshair below region, even with left edge of region. |

# ASCII codes

Decimal ASCII codes from 32 to 255. Characters corresponding to codes 0–31 are nonprinting, as are characters in the table indicated by “*npc*.”

| Code | Character | Code | Character | Code | Character | Code | Character | Code | Character |
|---|---|---|---|---|---|---|---|---|---|
| 32 | [space] | 80 | P | 130 | ‚ | 182 | ¶ | 230 | æ |
| 33 | ! | 81 | Q | 131 | ƒ | 183 | · | 231 | ç |
| 34 | " | 82 | R | 132 | „ | 184 | ¸ | 232 | è |
| 35 | # | 83 | S | 133 | … | 185 | ¹ | 233 | é |
| 36 | $ | 84 | T | 134 | † | 186 | º | 234 | ê |
| 37 | % | 85 | U | 135 | ‡ | 187 | » | 235 | ë |
| 38 | & | 86 | V | 136 | ˆ | 188 | ¼ | 236 | ì |
| 39 | ' | 87 | W | 137 | ‰ | 189 | ½ | 237 | í |
| 40 | ( | 88 | X | 138 | Š | 190 | ¾ | 238 | î |
| 41 | ) | 89 | Y | 139 | ‹ | 191 | ¿ | 239 | ï |
| 42 | * | 90 | Z | 140 | Œ | 192 | À | 240 | ð |
| 43 | + | 91 | [ | 141–4 | *npc* | 193 | Á | 241 | ñ |
| 44 | , | 92 | \ | 145 | ‘ | 194 | Â | 242 | ò |
| 45 | - | 93 | ] | 146 | ’ | 195 | Ã | 243 | ó |
| 46 | . | 94 | ^ | 147 | “ | 196 | Ä | 244 | ô |
| 47 | / | 95 | _ | 148 | ” | 197 | Å | 245 | õ |
| 48 | 0 | 96 | ` | 149 | • | 198 | Æ | 246 | ö |
| 49 | 1 | 97 | a | 150 | – | 199 | Ç | 247 | ÷ |
| 50 | 2 | 98 | b | 151 | — | 200 | È | 248 | ø |
| 51 | 3 | 99 | c | 152 | ˜ | 201 | É | 249 | ù |
| 52 | 4 | 100 | d | 153 | ™ | 202 | Ê | 250 | ú |
| 53 | 5 | 101 | e | 154 | š | 203 | Ë | 251 | û |
| 54 | 6 | 102 | f | 155 | › | 204 | Ì | 252 | ü |
| 55 | 7 | 103 | g | 156 | œ | 205 | Í | 253 | ý |
| 56 | 8 | 104 | h | 157–8 | *npc* | 206 | Î | 254 | þ |
| 57 | 9 | 105 | i | 159 | Ÿ | 207 | Ï | 255 | ÿ |
| 58 | : | 106 | j | 160 | *npc* | 208 | Ð | | |
| 59 | ; | 107 | k | 161 | ¡ | 209 | Ñ | | |
| 60 | < | 108 | l | 162 | ¢ | 210 | Ò | | |
| 61 | = | 109 | m | 163 | £ | 211 | Ó | | |
| 62 | > | 110 | n | 164 | ¤ | 212 | Ô | | |
| 63 | ? | 111 | o | 165 | ¥ | 213 | Õ | | |
| 64 | @ | 112 | p | 166 | ¦ | 214 | Ö | | |
| 65 | A | 113 | q | 167 | § | 215 | × | | |
| 66 | B | 114 | r | 168 | ¨ | 216 | Ø | | |
| 67 | C | 115 | s | 169 | © | 217 | Ù | | |
| 68 | D | 116 | t | 170 | ª | 218 | Ú | | |
| 69 | E | 117 | u | 171 | « | 219 | Û | | |
| 70 | F | 118 | v | 172 | ¬ | 220 | Ü | | |
| 71 | G | 119 | w | 173 | - | 221 | Ý | | |
| 72 | H | 120 | x | 174 | ® | 222 | Þ | | |
| 73 | I | 121 | y | 175 | ¯ | 223 | ß | | |
| 74 | J | 122 | z | 176 | ° | 224 | à | | |
| 75 | K | 123 | { | 177 | ± | 225 | á | | |
| 76 | L | 124 | \| | 178 | ² | 226 | â | | |
| 77 | M | 125 | } | 179 | ³ | 227 | ã | | |
| 78 | N | 126 | ~ | 180 | ´ | 228 | ä | | |
| 79 | O | 127–9 | *npc* | 181 | µ | 229 | å | | |