Symbolic Math Toolbox™ Release Notes

How to Contact The MathWorks

`www.mathworks.com`	Web
`comp.soft-sys.matlab`	Newsgroup
`www.mathworks.com/contact_TS.html`	Technical Support

`suggest@mathworks.com`	Product enhancement suggestions
`bugs@mathworks.com`	Bug reports
`doc@mathworks.com`	Documentation error reports
`service@mathworks.com`	Order status, license renewals, passcodes
`info@mathworks.com`	Sales, pricing, and general information

508-647-7000 (Phone)

508-647-7001 (Fax)

The MathWorks, Inc.
3 Apple Hill Drive
Natick, MA 01760-2098

For contact information about worldwide offices, see the MathWorks Web site.

Symbolic Math Toolbox™ Release Notes

Contents

Summary by Version

This table provides quick access to what's new in each version. For clarification, see "Using Release Notes" on page 3.

Version (Release)	New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Latest Version V5.3 (R2009b)	Yes Details	Yes Summary	Bug Reports Includes fixes	Printable Release Notes: PDF Current product documentation
V5.2 (R2009a)	Yes Details	Yes Summary	Bug Reports Includes fixes	No
V5.1 (R2008b)	No	No **Note** If you are upgrading from a version before 4.9, see the release notes for "Version 4.9 (R2007b+) Symbolic Math Toolbox Software" on page 16.	Bug Reports Includes fixes	No

Version (Release)	New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
V5.0 (R2008a+)	No	No **Note** If you are upgrading from a version before 4.9, see the release notes for "Version 4.9 (R2007b+) Symbolic Math Toolbox Software" on page 16.	Bug Reports Includes fixes	No
V4.9 (R2007b+)	Yes Details	Yes Summary	Bug Reports Includes fixes	No
V3.2.3 (R2008a)	No	No	Bug Reports Includes fixes	No
V3.2.2 (R2007b)	No	No	Bug Reports Includes fixes	No
V3.2 (R2007a)	Yes Details	No	Bug Reports Includes fixes	No
V3.1.5 (R2006b)	Yes Details	Yes Summary	Bug Reports Includes fixes	No
V3.1.4 (R2006a)	No	No	Bug Reports Includes fixes	No
V3.1.3 (R14SP3)	No	No	No bug fixes	No
V3.1.2 (R14SP2)	No	No	Bug Reports Includes fixes	No

Version (Release)	New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
V3.1.1 (R14SP1)	No	No	No bug fixes	No
V3.1 (R14)	Yes Details	No	No bug fixes	No

Using Release Notes

Use release notes when upgrading to a newer version to learn about:

- New features
- Changes
- Potential impact on your existing files and practices

Review the release notes for other MathWorks™ products required for this product (for example, MATLAB® or Simulink®). Determine if enhancements, bugs, or compatibility considerations in other products impact you.

If you are upgrading from a software version other than the most recent one, review the current release notes and all interim versions. For example, when you upgrade from V1.0 to V1.2, review the release notes for V1.1 and V1.2.

What Is in the Release Notes

New Features and Changes

- New functionality
- Changes to existing functionality

Version Compatibility Considerations

When a new feature or change introduces a reported incompatibility between versions, the **Compatibility Considerations** subsection explains the impact.

Compatibility issues reported after the product release appear under Bug Reports at The MathWorks™ Web site. Bug fixes can sometimes result in incompatibilities, so review the fixed bugs in Bug Reports for any compatibility impact.

Fixed Bugs and Known Problems

The MathWorks offers a user-searchable Bug Reports database so you can view Bug Reports. The development team updates this database at release time and as more information becomes available. Bug Reports include provisions for any known workarounds or file replacements. Information is available for bugs existing in or fixed in Release 14SP2 or later. Information is not available for all bugs in earlier releases.

Access Bug Reports using your MathWorks Account.

Version 5.3 (R2009b) Symbolic Math Toolbox Software

This table summarizes what's new in Version 5.3 (R2009b):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	Yes–Details labeled as **Compatibility Considerations**, below. See also Summary	Bug Reports Includes fixes	Printable Release Notes: PDF Current product documentation

New features and changes introduced in this version are described here:

Support for Windows x64 and 64-Bit Macintosh

The toolbox now supports 64-bit Windows® and Macintosh® operating systems. If you work in the MuPAD® Notebook Interface on a 64-bit Macintosh operating system, MuPAD runs a 64-bit engine with 32-bit graphical user interfaces, such as notebooks and Editor and Debugger windows.

sym and syms Use Reserved Words as Variable Names

`sym` and `syms` commands now treat reserved MuPAD words, except `pi`, as variable names.

Compatibility Considerations

In previous releases, the reserved words returned MuPAD values. If your code uses the reserved words as MuPAD commands, modify your code and use the `evalin` command with the reserved word as a name. For example, use `evalin(symengine,'beta')`.

The Toolbox Now Displays Floating-Point Results with Their Original Precision

The toolbox now displays the floating-point results with the original precision with which the toolbox returned them.

Compatibility Considerations

In previous releases, the toolbox displayed floating-point results with the current precision. You must update the existing code that relies on the output precision for displaying floating-point numbers. Use `digits` to set the precision you need before computing such results. The toolbox displays the results with the same number of digits it used to compute the results. The toolbox also can increase the specified precision of calculations by several digits.

In previous releases, `sym(A, 'f')` represented numbers in the form `(2^e + N*2^(e - 52))` or `-(2^e + N*2^(e - 52))` , with integers for `N` and `e`, and $N \geq 0$. Now `sym(A, 'f')` displays results in the rational form that actually represents the double-precision floating-point numbers.

New MuPAD Preference Pref::outputDigits Controls Floating-Point Outputs

New preference `Pref::outputDigits` controls the precision MuPAD uses to display floating-point results.

Solver for Ordinary Differential Equations Handles More Equation Types

Enhanced solvers handle more equation types of second-order homogeneous linear ordinary differential equations. The solver demonstrates improved performance.

MuPAD limit Function Supports Limits for Incomplete Gamma Function and Exponential Integral Function

Enhanced limit function now can compute limits for incomplete Gamma function and exponential integral function.

Enhanced Simplification Routines for MuPAD Special Functions

Enhanced simplification routines for MuPAD `hypergeom`, `mejerG`, and `bessel` special functions.

Enhanced MuPAD combine Function for Logarithms

Enhanced `combine` function demonstrates better performance for logarithms.

MuPAD normal Function Accepts New Options

The `normal` command now accepts the options `NoGcd`, `ToCancel`, `Rationalize`, `Recursive`, and `Iterations`. The options control costly operations, such as recognizing greatest common divisors and algebraic dependencies.

Functions and Function Elements Being Removed

Function or Function Element Name	What Happens When You Use the Function or Element?	Use This Instead	Compatibility Considerations
MuPAD `Domain Dom::Ideal`	Warns	`groebner`	Represent ideals as lists, and use functions of the `groebner` package instead.
MuPAD `student` Library	Warns	`plot::Integral` and `linalg`	Use `plot::Integral` and the `linalg` package instead.
`d` in `char(A, d)`	Warns	None	No replacement
MuPAD `relation` option in `simplify`	Warns	None	No replacement

Version 5.2 (R2009a) Symbolic Math Toolbox Software

This table summarizes what's new in Version 5.2 (R2009a):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	Yes–Details labeled as **Compatibility Considerations**, below. See also Summary	Bug Reports Includes fixes	Printable Release Notes: PDF Current product documentation

New features and changes introduced in this version are described here:

• "New Demos" on page 13

dsolve Accepts the New Option IgnoreAnalyticConstraints

The `dsolve` command now accepts the option `IgnoreAnalyticConstraints`. The option controls the level of mathematical rigor that the solver uses on the analytical constraints on the solution. By default, the solver ignores all analytical constraints.

Compatibility Considerations

The results of the `dsolve` command can differ from those returned in the previous release. If you want to obtain the same solutions as in the previous release, set the value of the option `IgnoreAnalyticConstraints` to `none`.

emlBlock Function Generates Embedded MATLAB Function Blocks from Symbolic Objects

The new `emlBlock` command converts symbolic expressions to Embedded MATLAB™ Function Blocks. You can use these blocks in any Simulink installation, even those without a Symbolic Math Toolbox™ license. For more information, see Generating Embedded MATLAB Blocks in the Symbolic Math Toolbox documentation.

matlabFunction Improves Control over Input and Output Parameters

`matlabFunction` now accepts multiple expressions and cell arrays of symbolic arrays as input parameters. The function now allows you to specify the names of the output parameters.

Compatibility Considerations

In previous releases, the default name of an output variable was RESULT. Now the default names of the output variables coincide with the names you use to call `matlabFunction`. You must update existing code that relies on the default output name `RESULT`. You can change your code using any of these methods:

- Define the name of an output variable as `RESULT`.
- Change the name of an input variable to `RESULT`.
- Throughout your code change the variable name from `RESULT` to the input name.

Enhancements to Object-Oriented Programming Capabilities

The Symbolic Math Toolbox product uses some object-oriented programming features to implement symbolic objects. Major enhancements to object-oriented programming capabilities enable easier development and maintenance of large applications and data structures. For a full description of object-oriented features, see the MATLAB Object-Oriented Programming documentation.

Compatibility Considerations

It is no longer possible to add methods to `@sym` by creating a `@sym` directory containing custom methods.

For an empty `x`, `sym(x)` returns a symbolic object of the same size as `x`. In previous releases, `sym(x)` returned a symbolic object of size 0-by-0 for an empty `x`.

generate::MATLAB Function Converts MuPAD Expressions to MATLAB Code

The new `generate::MATLAB` command converts MuPAD expressions, equations, and matrices to MATLAB formatted strings.

MuPAD IgnoreAnalyticConstraints Option Specifies That Core Functions Apply Common Algebraic Assumptions to Simplify Results

The new `IgnoreAnalyticConstraints` option allows the use of a set of simplified mathematical rules when solving equations, simplifying expressions, or integrating. For example, this option applies practical, but not generally correct rules for combining logarithms: $\ln(a) + \ln(b) = \ln(a \cdot b)$

MuPAD Outputs Contain Abbreviations for Better Readability

The new default format of presenting results enhances readability of long output expressions by using abbreviations.

MuPAD Solver for Ordinary Differential Equations Handles More Equation Types

The solver now can handle more than 200 additional types of second-order ordinary differential equations. The solver demonstrates improved performance.

MuPAD limit Function Now Can Compute Limits for Piecewise Functions

The enhanced `limit` function computes limits of piecewise functions including bidirectional and one-sided limits.

New and Improved MuPAD Special Functions

MuPAD includes the following new special functions:

- `laguerreL` represents Laguerre's L function.
- `erfc(x,n)` returns iterated integrals of the complementary error function.
- `meijerG` represents the Meijer G function.

The `hypergeom` special function demonstrates better performance.

New Calling Syntax for Test Report Function prog::tcov

The `prog::tcov` function that inspects the data collected during the code execution has the new syntax and set of options.

Compatibility Considerations

The new syntax is not valid in MuPAD versions earlier than 5.2. MuPAD 5.2 does not support the earlier syntax.

New Demos

To see new demos that use MuPAD Notebook Interface, type `mupadDemo` at the MATLAB command line or click MuPAD Notebooks Demo.

Version 5.1 (R2008b) Symbolic Math Toolbox Software

This table summarizes what's new in Version 5.1 (R2008b):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
No	No	Bug Reports Includes fixes	Printable Release Notes: PDF Current product documentation

There are no new features or changes in this version.

Note If you are upgrading from a version before 4.9, see the release notes for "Version 4.9 (R2007b+) Symbolic Math Toolbox Software" on page 16.

Version 5.0 (R2008a+) Symbolic Math Toolbox Software

This table summarizes what's new in Version 5.0 (R2008a+):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
No	No	Bug Reports Includes fixes	Printable Release Notes: PDF Current product documentation

There are no new features or changes in this version.

Note If you are upgrading from a version before 4.9, see the release notes for "Version 4.9 (R2007b+) Symbolic Math Toolbox Software" on page 16.

Version 4.9 (R2007b+) Symbolic Math Toolbox Software

This table summarizes what's new in Version 4.9 (R2007b+):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	Yes–Details labeled as **Compatibility Considerations**, below. See also Summary.	Bug Reports Includes fixes	Printable Release Notes: PDF Current product documentation

New features and changes introduced in this version are described here:

MuPAD Engine Replaces Maple Engine

The default Symbolic Math Toolbox engine is now the MuPAD engine. For more information, see the "MuPAD in Symbolic Math Toolbox" chapter in the Symbolic Math Toolbox User's Guide.

Compatibility Considerations

The new engine causes many computed results to differ from those returned by previous versions of Symbolic Math Toolbox software.

General Differences.

- Many computations return in a permuted order (such as `a + b` instead of `b + a`).
- Some computations return in a different, mathematically equivalent form (such as `(cos(x))^2` instead of `1 - (sin(x))^2`).
- `diff(dirac(t))` returns `dirac(t,1)` instead of `dirac(1,t)`.
- `sym(x,'f')` no longer produces strings of the form `hex digits*2^n`. Instead the strings have the form `(2^e+N*2^(e-52))`, where `N` and `e` are integers.
- For toolbox calculations, some symbols can only be used as symbolic variables, and not in strings: `E`, `I`, `D`, `O`, `beta`, `zeta`, `theta`, `psi`, `gamma`, `Ci`, `Si`, and `Ei`. This is because those symbols represent MuPAD reserved words, and are interpreted as the MuPAD word if you pass them as strings. The words `Ci`, `Si`, `Ei` represent special mathematical functions: the cosine integral, sine integral, and exponential integral respectively.
- Error and warning message IDs may have changed.
- Performance of numerical integration is slower than in previous versions.
- Subexpressions, calculated by the `subexpr` function, may be different than in previous versions.
- The `pretty` function no longer uses partial subexpressions (with syntax `%n`).

Calculus.

- `Int` no longer evaluates some integrals, including many involving Bessel functions.
- `symsum(sin(k*pi)/k,0,n)` no longer evaluates to `pi`.

Linear Algebra.

- The output of `colspace` may differ from previous versions, but it is mathematically equivalent.
- The `eig` function may return eigenvalues in a different order than previous versions. Expressions returned by `eig` may be larger than in previous versions.
- The `jordan` function may return diagonal subblocks in a different order than previous versions.
- `svd` may return singular values in a different order than previous versions.

Simplification.

- The `coeffs` function may return multivariable terms in a different order than in previous versions.
- The `expand` function may return some trig and exponential expressions differently than in previous versions.
- The `simplify` function involving radicals and powers make fewer assumptions on unknown symbols than in previous versions.
- The `subexpr` function may choose a different subexpression to be the common subexpression than in previous versions.
- Subexpressions no longer have partial subexpressions (previous syntax `%n`).
- The `solve` function returns solutions with higher multiplicity only when solving a single polynomial.
- `acot(-x) = -acot(x)` instead of `pi - acot(x)` as in previous versions.
- `acoth(-x) = -acoth(x)` instead of `2*acoth(0) - acoth(x)` as in previous versions.
- The `simple` function has several differences:
  - The `'how'` value `combine(trig)` has been replaced with `combine(sincos)`, `combine(sinhcosh)`, and `combine(ln)`.
  - The `'how'` values involving `convert` have been replaced with `rewrite`.
  - A new `'how'` value of `mlsimplify(100)` indicates the MuPAD function `Simplify(...,Steps=100)` simplified the expression.

  - Simplifications such as `(sin(x)^2)^(1/2)` to `sin(x)` are no longer performed, since the MuPAD language is careful not to make assumptions about the sign of `sin(x)`.

Conversion.

- Arithmetic involving the `vpa` function uses the current number of digits of precision. Variable precision arithmetic may have different rounding behaviors, and answers may differ in trailing digits (trailing zeros are now suppressed).
- The `char` function returns strings using MuPAD syntax instead of Maple™ syntax.
- Testing equality does not compare strings as in previous versions; the symbolic engine equality test is used.
- Saving and loading symbolic expressions is compatible with previous versions, except when the symbolic contents use syntax or functions that differ between Maple or MuPAD engines. For example, suppose you save the symbolic object `sym('transform::fourier(f,x,w)')`, which has MuPAD syntax. You get a MATLAB error if you try to open the object while using a Maple engine.
- LaTeX output from the `latex` function may look different than before.
- C and Fortran code generated with the `ccode` and `fortran` functions may be different than before. In particular, generated files have intermediate expressions as “optimized” code. For more information, see the “Generating C or Fortran Code” section of the User’s Guide.
- `pretty` output may look different than before.

Equation Solving.

- `solve` returns solutions with higher multiplicity only when solving a single polynomial.
- `solve` may return a different number of solutions than before.
- Some calls to `dsolve` that used to return results involving `lambertw` now return no solution.
- `dsolve` can now use the variable `C`.

- Some `dsolve` results are more complete (more cases are returned).
- Some `dsolve` results are less complete (not all previous answers are found).
- `finverse` may be able to find inverses for different classes of functions than before.
- When `finverse` fails to find an explicit inverse, it produces different output than before.

Transforms.

- Fourier and inverse Fourier transforms return the MuPAD form `transform::fourier` when they cannot be evaluated. For example,

```
h = sin(x)/exp(x^2);
FF = fourier(h)

FF =
transform::fourier(sin(x)/exp(x^2), x, -w)
```

  The reason for this behavior is the MuPAD definition of Fourier transform and inverse Fourier transform differ from their Symbolic Math Toolbox counterparts by the sign in the exponent:

	Symbolic Math Toolbox definition	**MuPAD definition**
Fourier transform	$F(w) = \int_{-\infty}^{\infty} f(x)e^{-iwx}dx$	$F(w) = \int_{-\infty}^{\infty} f(x)e^{iwx}dx$
Inverse Fourier transform	$f(x) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(w)e^{iwx}dw$	$f(x) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(w)e^{-iwx}dw$

- Several Fourier transforms can no longer be calculated, especially those involving Bessel functions.
- `ztrans` and `iztrans` may return more complicated expressions than before.

Special Mathematical Functions.

- The three-parameter Riemann Zeta function is no longer supported.
- heaviside(0) = 0.5; in previous versions it was undefined.

maple.

- The `maple`, `mhelp`, and `procread` functions error, unless a Maple engine is installed and selected with `symengine`.

New MuPAD Language and Libraries Supplant Extended Symbolic Math Toolbox Software

The functionality of the MuPAD language, together with the included libraries, goes far beyond that of the previous Symbolic Math Toolbox software. However, it is not identical to that of the previous Extended Symbolic Math Toolbox™ software. The differences between these software packages are beyond the scope of these release notes. More information is available in the "Differences in Maple and MuPAD Syntax" section of the User's Guide.

You can access the MuPAD language in several ways:

- To learn the commands, syntax, and functionality of the language, use the MuPAD Help browser, or read the Tutorial.
- Use a MuPAD notebook, which contains an integrated help system for the language syntax.
- Use the new `evalin` function or `feval` function to access the MuPAD language at the MATLAB command line. More detail is available in the "Calling MuPAD Functions at the MATLAB Command Line" section of the User's Guide.

New MuPAD Help Viewer (GUI)

The MuPAD help viewer contains complete documentation of the MuPAD language, and of the MuPAD Notebook Interface. For more information, see the "Getting Help for MuPAD" section of the User's Guide.

New MuPAD Notebook Interface (GUI)

A MuPAD notebook is an interface for performing symbolic math computations with embedded math notation, graphics, animations, and text. It also enables you to share, document, and publish your calculations and graphics. For example, the MuPAD help viewer is essentially a special MuPAD notebook. For more information, see the "Calculating in a MuPAD Notebook" section of the User's Guide.

New MuPAD Editor and Debugger (GUI)

The MuPAD Editor GUI enables you to write custom symbolic functions and libraries in the MuPAD language. The Debugger enables you to test your code. For more information, consult the MuPAD help viewer.

New Functionality for Communication Between MATLAB Workspace and MuPAD

Function	Use
`doc(symengine,...)`	Access the MuPAD Help browser.
`evalin(symengine,...)`	Use MuPAD functionality in the MATLAB workspace.
`feval(symengine,...)`	Use MuPAD functionality in the MATLAB workspace.
`getVar`	Copy expressions residing in a MuPAD notebook into the MATLAB workspace.
`mupad`	Launch a MuPAD notebook .
`mupadwelcome`	Access MuPAD GUIs .
`reset(symengine,...)`	Clear the MuPAD engine for the MATLAB workspace.
`setVar`	Copy expressions residing in the MATLAB workspace into a MuPAD notebook.
`symvar`	Produce a list of symbolic objects in an expression.

For more information, see the “Integration of MuPAD and MATLAB” section of the User’s Guide.

New symengine Command for Choosing a Maple Engine

If you own a compatible version of a Maple software, you can choose to have Symbolic Math Toolbox software use the Maple engine instead of a MuPAD engine. You might want to do this if you have existing Maple programs. Choose the engine by entering `symengine` at the MATLAB command line; this brings up a GUI for making your choice. For more information, see “Choosing a Maple or MuPAD Engine” in the User’s Guide.

New matlabFunction Generates M-Files and Function Handles

The new `matlabFunction` converts symbolic expressions to function handles or M-files. You can use these function handles and files in any MATLAB installation, even those without a Symbolic Math Toolbox license. For more information, see “Generating MATLAB Function Handles and M-Files” in the User’s Guide.

Version 3.2.3 (R2008a) Symbolic Math Toolbox and Extended Symbolic Math Toolbox Software

This table summarizes what's new in Version 3.2.3 (R2008a):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
No	No	Bug Reports Includes fixes	No

There are no new features or changes in this version.

Version 3.2.2 (R2007b) Symbolic Math Toolbox and Extended Symbolic Math Toolbox Software

This table summarizes what's new in Version 3.2.2 (R2007b):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
No	No	Bug Reports Includes fixes	No

There are no new features or changes in this version.

Version 3.2 (R2007a) Symbolic Math Toolbox and Extended Symbolic Math Toolbox Software

This table summarizes what's new in Version 3.2 (R2007a):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	No	Bug Reports Includes fixes	No

New features and changes introduced in this version are described here:

Maple10 Access Added for Linux 64–bit Processors and Intel Macintosh Platforms

MATLAB now supports Maple Version 10 on 32-bit Windows, 32- and 64-bit Linux® platforms, and the Intel® and PowerPC® Macintosh platforms.

Version 3.1.5 (R2006b) Symbolic Math Toolbox and Extended Symbolic Math Toolbox Software

This table summarizes what's new in version 3.1.5 (R2006b):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	Yes–Details labeled as **Compatibility Considerations**, below. See also Summary.	Bug Reports Includes fixes	No

New features and changes introduced in this version are described here:

Change in call to code generation package using the maple function

Calling a function in code generation package using Maple software now requires you to explicitly include the package name. For example,

```
maple('codegen[fortran](x^2-4)');
```

The generated code output using these methods is unaffected by this change.

Compatibility Considerations

In previous versions, functions in the code generation package of Maple software were made automatically available using the Maple `with` command, and did not require the package name. For example

```
maple('fortran(x^2-4)');
```

This sometimes caused a conflict when assigning to Maple variables having the same name as a function in the code generation package.

Version 3.1 (R14) Symbolic Math Toolbox and Extended Symbolic Math Toolbox Software

This table summarizes what's new in version 3.1 (R14):

New Features and Changes	Version Compatibility Considerations	Fixed Bugs and Known Problems	Related Documentation at Web Site
Yes Details below	No	No	No

New features and changes introduced in this version are described here:

Rounding Operations

The following new functions perform rounding operations on symbolic arrays:

- `ceil` — Round a number x to the nearest integer greater than or equal to x.
- `fix` — Round toward zero.
- `floor` — Round a number x to the nearest integer less than or equal to x.
- `frac` — Compute the fractional part of a number.
- `round` — Round a number to the nearest integer.

For example,

```
x = sym([2.5; -9.639])
[fix(x) floor(x) round(x) ceil(x) frac(x)]

x =
        5/2
 -9639/1000

ans =
[         2,         2,         3,         3,       1/2]
[        -9,       -10,       -10,        -9, -639/1000]
```

Quotient and Remainder for Division of Integers and Polynomials

The new function `quorem` computes the quotient and remainder for division of integers and polynomials. For example,

```
syms x y
p = x^3-2*x+5
[q,r] = quorem(x^5,p)

p =
x^3-2*x+5

q =
x^2+2

r =
-5*x^2-10+4*x
```

Dirac and Step Functions

The following new functions compute the Dirac delta and Heaviside functions:

- `dirac` — Compute the Dirac delta function.
- `heaviside` — Compute the Heaviside step function.

For example,

```
dirac([-1 0 1])

ans =
     0   Inf     0

heaviside([-1 0 1])

ans =
     0   NaN     1
```

Sorting Symbolic Expressions

The new function sort sorts symbolic expressions. For example,

```
syms a b c d e x
sort([a c e b d])

ans =
[ a, b, c, d, e]
sort([a c e b d]*x.^(0:4).')

ans =
x^4*d+x^3*b+e*x^2+x*c+a
```

Coefficients of Multivariable Expressions

The new function coeffs computes coefficients of a multivariate polynomial. For example,

```
syms c t x y
t = 2 + (3 + 4*log(x))^2 - 5*log(x);
coeffs(expand(t))

ans =
[ 11, 19, 16]

z = 3*x^2*y^2 + 5*x*y^3
[c,t] = coeffs(z,y)

z =
3*x^2*y^2+5*x*y^3
```

```
c =
[ 3*x^2,   5*x]

t =
[ y^2, y^3]
```

Multidimensional Symbolic Arrays

The new function reshape reshapes symbolic arrays. For example,

```
syms x
A = reshape(x.^(1:9),1,3,3)

A(:,:,1) =
[   x, x^2, x^3]

A(:,:,2) =
[ x^4, x^5, x^6]

A(:,:,3) =
[ x^7, x^8, x^9]
```

Conversion to Nondouble Numeric Data Types

The following new functions enable you to convert symbolic arrays to nondouble numeric data types:

- `int8` — Convert a symbolic matrix to signed 8-bit integers.
- `int16` — Convert a symbolic matrix to signed 16-bit integers.
- `int32` — Convert a symbolic matrix to signed 32-bit integers.
- `int64` — Convert a symbolic matrix to signed 64-bit integers.
- `single` — Convert a number to single precision.
- `uint8` — Convert a symbolic matrix to unsigned 8-bit integers.
- `uint16` — Convert a symbolic matrix to unsigned 16-bit integers.
- `uint32` — Convert a symbolic matrix to unsigned 32-bit integers.
- `uint64` — Convert a symbolic matrix to unsigned 64-bit integers.

Logarithms to Base 2 and Base 10

The following new functions enable you to compute the logarithm of symbolic arrays to base 2 and base 10:

- `log10` — Compute base 10 logarithm.
- `log2` — Compute base 2 logarithm.

Modulus After Division

The new function mod computes modulus after division. For example,

```
syms x
mod(x^3-2*x+999,10)

ans =
x^3+8*x+9
```

Compatibility Summary for Symbolic Math Toolbox and Extended Symbolic Math Toolbox Software

This table summarizes new features and changes that might cause incompatibilities when you upgrade from an earlier version, or when you use files on multiple versions. Details are provided with the description of the new feature or change.

Version (Release)	New Features and Changes with Version Compatibility Impact
Latest Version V5.3 (R2009b)	See the **Compatibility Considerations** subheading for each of these new features or changes: • "sym and syms Use Reserved Words as Variable Names" on page 6 • "The Toolbox Now Displays Floating-Point Results with Their Original Precision" on page 6 • "Functions and Function Elements Being Removed" on page 7
V5.2 (R2009a)	See the **Compatibility Considerations** subheading for each of these new features or changes: • "dsolve Accepts the New Option IgnoreAnalyticConstraints" on page 10 • "matlabFunction Improves Control over Input and Output Parameters" on page 10

Version (Release)	New Features and Changes with Version Compatibility Impact
	• "Enhancements to Object-Oriented Programming Capabilities" on page 11 • "New Calling Syntax for Test Report Function prog::tcov" on page 12
V5.1 (R2008b)	None
V5.0 (R2008a+)	None
V4.9 (R2007b+)	See the **Compatibility Considerations** subheading for each of these new features or changes: • "MuPAD Engine Replaces Maple Engine" on page 16
V3.2.3 (R2008a)	None
V3.2.2 (R2007b)	None
V3.2 (R2007a)	None
V3.1.5 (R2006b)	See the **Compatibility Considerations** subheading for each of these new features or changes: • "Change in call to code generation package using the maple function" on page 27
V3.1.4 (R2006a)	None
V3.1.3 (R14SP3)	None
V3.1.1 (R14SP1)	None
V3.1 (R14)	None