

# Symbolic Math Toolbox 5.5

---

## Perform mathematics using symbolic computation and variable-precision arithmetic

Symbolic Math Toolbox™ provides tools for solving and manipulating symbolic math expressions and performing variable-precision arithmetic. The toolbox contains hundreds of MATLAB® symbolic functions that leverage the MuPAD® engine for tasks such as differentiation, integration, simplification, transforms, and equation solving.

Symbolic Math Toolbox also includes the MuPAD language, which is optimized for handling and operating on symbolic math expressions. It provides libraries of MuPAD functions in common mathematical areas, such as calculus and linear algebra, as well as specialized areas, such as number theory and combinatorics. You can extend the built-in functionality by writing custom symbolic functions and libraries in the MuPAD language. All functions can be accessed from the MATLAB command line or from the MuPAD notebook interface, where you can manage and document your symbolic computations.

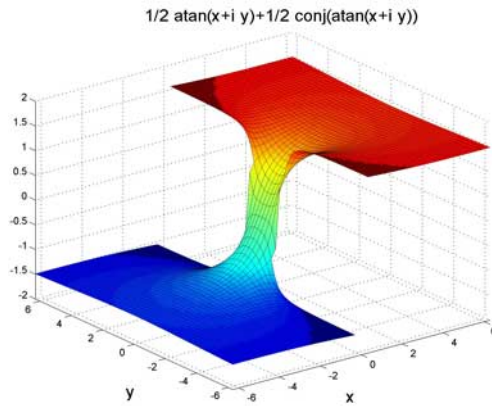
### Key Features

- MATLAB symbolic functions for differentiation, integration, simplification, transforms, and equation solving
- Variable-precision arithmetic capabilities
- MuPAD language for operating on symbolic math expressions
- MuPAD libraries covering common mathematical areas, such as calculus and linear algebra, and specialized areas, such as number theory and combinatorics
- Functions for converting symbolic expressions to MATLAB, C, Fortran, MathML, and TeX
- MuPAD notebook interface with embedded text, graphics, and typeset math for documenting and managing computations performed with the MuPAD language
- MuPAD editor and debugger for writing custom symbolic functions and libraries

### Working with Symbolic Math Toolbox

Symbolic Math Toolbox provides a complete set of tools for symbolic computing that augments the numeric capabilities of MATLAB. The toolbox includes extensive symbolic functionality that you can access directly from the MATLAB command line or from the MuPAD notebook interface. You can extend the functionality available in the toolbox by writing custom symbolic functions or libraries in the MuPAD language.

In addition to performing symbolic math operations, the toolbox provides variable-precision arithmetic functions for computations that require exact control over numeric accuracy. You can evaluate numeric operations to any number of digits.



A surface generated from a symbolic equation. Visualization functions let you plot your symbolic expressions directly from the MATLAB command line.

### Performing Symbolic Computations from MATLAB

Symbolic Math Toolbox lets you perform symbolic computations from the MATLAB command line by defining symbolic math expressions and operating on them. Functions are called using the familiar MATLAB syntax and are available for a wide range of tasks, including differentiation, integration, simplification, transforms, and equation solving. From MATLAB you can also execute statements written in the MuPAD language, which lets you fully access the functionality in the MuPAD libraries that are included in the toolbox.

### Working in the MuPAD Notebook Interface

The MuPAD notebook interface provides an interactive environment for performing symbolic computations using the MuPAD language. It includes a symbol palette for accessing common MuPAD functions, and all results are displayed in typeset math that can be converted into MathML and TeX. You can embed graphics, animations, and descriptive text within your notebook to help manage and document your work.

Symbolic Math Toolbox provides functions for sharing symbolic variables and expressions between the MuPAD notebook interface and the MATLAB workspace, enabling you to merge the work that you do in each environment.

**Gibbs Phenomenon - MuPAD**

The following sum is a Fourier representation of a periodic step function:

$$f(x) = \sum_{k=1}^{\infty} \frac{\sin((2k-1)x)}{2k-1}$$

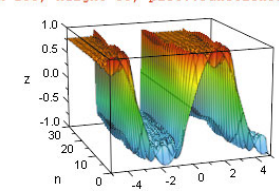
We wish to show the convergence of this function for  $k=1:n$  and compare the result to the original function  $f(x)$ . We start by assigning the value of  $f_n(x)$  to the MuPAD identifier  $f_n$ .

```
f_n := hold(sum(sin((2*k-1)*x)/(2*k-1), k = 1..floor(n)))
```

$$\sum_{k=1}^n \frac{\sin((2k-1)x)}{2k-1}$$

We then plot the sequence of  $f_n$  in a 3 dimensional view of  $z$  against  $x$  and  $n$ . The "overshoot" at the discontinuities is the Gibbs phenomenon.

```
plot(f_n, x=-5..5, n=1..30, #3, FillColorType=Rainbow, Width=100, Height=50, plot::Function3d::Submesh={5,1})
```



The amount of overshoot is  $\int_0^{\pi} \frac{\sin(x)}{2x} dx - \frac{\pi}{4}$ . We will compute this integral.

```
overshoot := int(sin(x)/x, x=0..PI)/2 - PI/4
```

$$\frac{\text{Si}(\pi)}{2} - \frac{\pi}{4}$$

Command Bar

General Math

Plot Commands

Not Connected Text INS

MuPAD notebook that analyzes the Gibbs phenomenon of a periodic step function. MuPAD notebooks let you perform and document symbolic computations.

## Programming in the MuPAD Language

Symbolic Math Toolbox provides an editor, a debugger, and other programming utilities for authoring custom symbolic functions and libraries in the MuPAD language. The MuPAD language supports multiple programming styles, including imperative, functional, and object-oriented programming. The language treats variables as symbolic by default, and it is optimized for handling and operating on symbolic math expressions.

```

rotate3d.mu - MuPAD Editor
File Edit Search View Window Help
Courier New
Rotate := proc(a,U)
    // a is the angle through which to rotate
    // U is a 3 by 1 matrix (a column vector)
    option escape;
    local p, q, r, S, result;
begin
    // Extract the coordinate data from U
    p:=U[1,1]; q:=U[2,1]; r:=U[3,1];
    // Normalize the coordinates of U
    S:=Dom::Quaternion([cos(a/2),
    p/sqrt(p^2+q^2+r^2)*sin(a/2),
    q/sqrt(p^2+q^2+r^2)*sin(a/2),
    r/sqrt(p^2+q^2+r^2)*sin(a/2)]);
    result := proc(V)
        // V is a 3 by 1 matrix (a column vector) to be rotated
        local T, x, y, z, X, Y, Z, Q;
    begin
        Q := Dom::Quaternion;
        x:=V[1,1];y:=V[2,1];z:=V[3,1];
        T:=S*Q([0,x,y,z])*Q::conjugate(S);
        X:=Q::scalarprod(T,Q([0, 1, 0, 0]));
        Y:=Q::scalarprod(T,Q([0, 0, 1, 0]));
        Z:=Q::scalarprod(T,Q([0, 0, 0, 1]));
        return(Dom::Matrix(Dom::Real)(3, 1,[X,Y,Z]));
    end_proc;
    return(result);
end_proc;

```

MuPAD procedure for rotating a vector in three dimensions. The MuPAD language is optimized for handling and operating on symbolic math expressions.

## Resources

### Product Details, Demos, and System Requirements

[www.mathworks.com/products/symbolic](http://www.mathworks.com/products/symbolic)

### Trial Software

[www.mathworks.com/trialrequest](http://www.mathworks.com/trialrequest)

### Sales

[www.mathworks.com/contactsales](http://www.mathworks.com/contactsales)

### Technical Support

[www.mathworks.com/support](http://www.mathworks.com/support)

### Online User Community

[www.mathworks.com/matlabcentral](http://www.mathworks.com/matlabcentral)

### Training Services

[www.mathworks.com/training](http://www.mathworks.com/training)

### Third-Party Products and Services

[www.mathworks.com/connections](http://www.mathworks.com/connections)

### Worldwide Contacts

[www.mathworks.com/contact](http://www.mathworks.com/contact)