MathWorks®
Accelerating the pace of engineering and science

Symbolic Math Toolbox: Quick Reference Sheet

Symbolic Variables

syms	Create symbolic variables: `syms x;` Create arrays of symbolic scalar variables: `syms M [2 3];` Create symbolic matrix variables: `syms A [2 3] matrix;`
symmatrix2sym	Convert symbolic matrix variable to array of symbolic scalar variables: `syms A B [2 3] matrix; X = A + B;` `Y = symmatrix2sym(X)`

Calculus

$\int_a^b f(x)\mathrm{d}x$

diff	Differentiation: `syms x t; diff(sin(x^2+t),x)`
int	Definite and indefinite integrals: `syms x z; int(x/(1 + z^2),z)`
release	Evaluate integrals: `syms x; F = int(cos(x),'Hold',true);` `G = release(F)`
limit	Compute limit of symbolic expression: `limit(1/x,x,0,'left')`
taylor	Taylor series: `syms x; taylor(exp(-x))`
series	Puiseux series expansion: `syms x; series(1/sin(x),x)`
symsum	Sum of a series: `syms k n; symsum(k,0,n-1)`
gradient	Gradient vector of scalar function: `syms x y z;` `gradient(x*y + 2*z*x,[x y z])`
jacobian	Jacobian matrix: `syms x y z u v;` `jacobian([x*y*z; y; x+z],[x y z])`
hessian	Hessian matrix of scalar function: `syms x y z;` `hessian(x*y + 2*z*x,[x y z])`
laplacian	Laplacian of scalar function: `syms x y z;` `laplacian(1/x + y^2 + z^3,[x y z])`
divergence	Divergence of vector field: `syms x y z;` `divergence([x^2 2*y z],[x y z])`

Algebra

π

double	Convert symbolic values to double precision: `symN = sym(pi);` `doubleN = double(symN)`
vpa	Control precision of computations with variable-precision arithmetic: `syms x;` `p = sym(pi);` `piVpa = vpa(p)`
subs	Symbolic substitution: `syms a b;` `subs(a^3+b,[a,b],[2,sym('e')])`
solve	Equations and systems solver: `syms a b u v;` `S = solve(u+v==a, u-v==b)`
dsolve	Solve differential equations: `syms y(t) a;` `eqn = diff(y,t)==a*y;` `S = dsolve(eqn)`
pdeCoefficients	Extract PDE Coefficients: `syms u(x,y);` `pdeeq = laplacian(u,[x y])== -3;` `coeffs = pdeCoefficients(pdeeq,u)`
isolate	Isolate variable or expression in equation: `syms a b c x;` `isolate(a*x^2+b*x+c==0,x)`
lhs	Left side (LHS) of equation: `syms x y;` `lhs(x^2 >= y^2)`
rhs	Right side (RHS) of equation: `syms x y;` `rhs(x^2 >= y^2)`
simplify	Algebraic simplification: `syms x;` `simplify(sin(x)^2 + cos(x)^2)`
rewrite	Rewrite expression in terms of another function: `syms x;` `rewrite(tan(x)/cos(x),'sin')`
resultant	Resultant of two polynomials: `syms x y;` `p = x^2+y;` `q = x-2*y;` `resultant(p,q)`

MathWorks
Accelerating the pace of engineering and science

Symbolic Math Toolbox: Quick Reference Sheet

Graphics

`fplot`	Plot symbolic expression or function: `syms x; f(x) = sin(x)/x; fplot(f)`
`fplot3`	Plot 3-D parametric curve: `syms x; fplot3(sin(x),cos(x),log(x))`

`fsurf`	Plot 3-D surface, mesh or contour:
	`syms x y;`
	`f(x,y)=x*exp(-x^2-y^2);`
	`fsurf(f)`

`fmesh`	Plot 3-D mesh:
	`syms x y;`
	`f(x,y)=x*exp(-x^2-y^2);`
	`fmesh(f)`

`fcontour`	Plot contours: `f(x,y)=x*exp(-x^2-y^2); fcontour(f)`

`fimplicit`, `fimplicit3`	Plot implicit symbolic equation or function:
	`syms x y;`
	`fimplicit(y^2-x^2*(x+1),[-2 2])`
	`syms x y z;`
	`fimplicit3(x^2*y*z+y^3-z^3)`

`fanimator`	Create stop-motion animation object:
	`syms y t;`
	`fanimator(@fplot,sin(x+t),[0 t]);`
	`playAnimation`

Functions

`symfun`	Create Symbolic Functions:
	`syms x y;`
	`f = symfun(x+y,[x y]); f(1,2)`

`piecewise`	Piecewise defined expression or function:
	`syms x;`
	`g(x) = piecewise(x<0,-1,x>=0,2); g(3)`

`matlabFunction`	Convert symbolic expression to function handle or file:
	`syms x y;`
	`f = sqrt(x^2 + y^2);`
	`g = matlabFunction(f)`

`matlabFunctionBlock`	Convert symbolic expression to MATLAB function block for Simulink:
	`new_system('my_system');`
	`open_system('my_system');`
	`syms x y z;`
	`f = x^2 + y^2 + z^2;`
	`matlabFunctionBlock('my_system/my_block',f)`

`simscapeEquation`	Convert symbolic expression to Simscape equations:
	`syms t x(t) y(t);`
	`phi = diff(x) + 5*y + sin(t);`
	`simscapeEquation(phi)`

Learn More

www.mathworks.com/products/symbolic

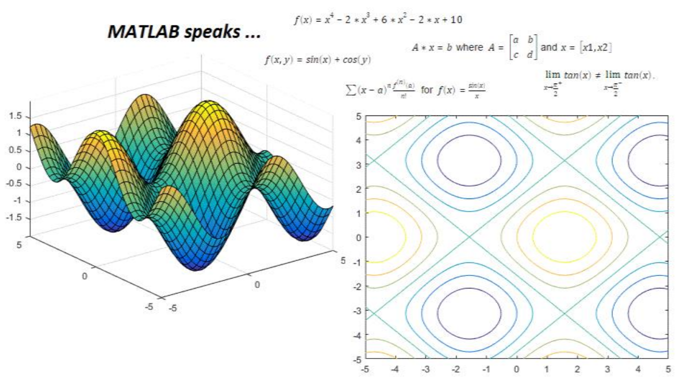
Related Products

Control System Toolbox. Design and analyze control systems.

Financial Toolbox. Analyze financial data

and develop financial models.

Optimization Toolbox. Solve linear, quadratic, conic, integer, and nonlinearoptimization problems



Full MATLAB cheat sheet