北太天元 符号计算工具箱 v2.0 预览版 函数使用说明

> **适用版本**：北太天元 2025 版 · 符号计算工具箱 v2.0 Preview
> **支持平台**：Windows 10/11 · Ubuntu x86/arm · macOS x86/arm

目录

---

1. 工具箱简介

符号计算工具箱提供了一套基于 SymPy 的符号数学计算功能，涵盖从基础符号处理到高级数学运算的完整能力。

v2.0 版本的核心改进通过 classdef 的方法重载机制，符号版本的函数与数值版本实现了统一命名：

```
% v1.x 旧写法（带前缀）
x = sym('x');
y = symdiff(sin(x));

% v2.0 新写法（函数名统一）
syms x
y = diff(sin(x));   % diff 自动调用符号版本
```

工具箱功能涉及以下三个方面：

- **符号变量创建**：`sym`、`syms`、`eq`
- **常用函数重载**：`diff`、`int`、`solve`、`laplace` 等 40+ 函数
- **表达式输出**：`disp` 打印符号表达式

---

2. 符号对象构建

---

sym

创建符号变量、符号数字

语法

```
x = sym("x")
sym(__, set)
sym(num)
sym(strnum)
```

用法说明

`x = sym("x")` 创建符号标量变量 x。

`sym(__, set)` 创建符号变量的同时为其添加假设约束。

`sym(num)` 将指定数字转换为符号数字。

`sym(strnum)` 将字符向量或字符串转换为精确的符号数，不进行近似计算。

参数说明

参数	类型	说明
`"x"`	字符串 | 字符向量	变量名，建议以字母开头，仅含字母、数字和下划线
`set`	字符串 | 字符向量元胞数组	假设集合，可选 `"integer"`、`"rational"`、`"real"`、`"positive"`
`num`	数值 | 符号常数	要转换为符号数的数值
`strnum`	字符串 | 字符向量	表示符号数的字符串，如 `'1/10'`

示例

创建符号变量

```
x = sym("x");
disp(x)
```

```
x
```

创建符号数字

```
a = sym(1);
f = a + 9;
disp(f)
```

```
10
```

指定多个类型假设

```
k = sym("k", ["real", "positive"]);
```

相关函数

syms

---

syms

创建单个或多个符号变量

语法

```
syms var1 ... varN
syms ___ set
```

用法说明

`syms var1 ... varN` 创建 sym 类型的符号标量变量 var1 至 varN，用空格分隔不同变量。此语法会清除这些变量的所有先前定义。

`syms ___ set` 创建符号变量并设置假设，使变量属于 `set`，同时清除其他假设。`set` 可以是 `real`、`positive`、`integer` 或 `rational`，可用空格组合多个假设。

参数说明

参数	类型	说明
`var1 ... varN`	字符串 | 字符向量	以空格分隔的有效变量名
`set`	`real` | `positive` | `integer` | `rational`	变量假设集合，可组合多个

示例

同时创建多个符号变量

```
syms x1 x2 x3 x4 x5 x6
% 等效于：x1 = sym("x1"); x2 = sym("x2"); ...
```

创建带假设的符号变量

```
syms x1 x2 x3 real
% 所有变量均被假设为实数
```

组合多个假设

```
syms x1 x2 x3 real integer
% 所有变量同时满足：实数 且 整数
```

在表达式中使用

```
syms x y z
f = x^2 + y^2 + z^2;
disp(f)
```

```
x^2 + y^2 + z^2
```

相关函数

sym | simplify | taylor

eq

创建符号等式

语法

```
F = eq(lhs, rhs)
A == B
```

用法说明

`F = eq(lhs, rhs)` 创建表示 `lhs == rhs` 的符号等式对象，可供 `solve`、`vpasolve` 等函数使用。

> **注意**：`eq` 至少需要其中一个参数为符号表达式，另一个可以为数值。

参数说明

参数	类型	说明
`lhs`	符号变量 | 数值	等式左侧
`rhs`	符号变量 | 数值	等式右侧

示例

```
x = sym("x");
expr = eq(x, 2);
disp(expr)
```

```
Eq(x, 2)
```

相关函数

sym | solve

---

3. 微积分

---

diff

求表达式的偏导数

语法

```
Df = diff(f)
Df = diff(f, var)
Df = diff(f, var, n)
Df = diff(f, var1, ..., varN)
```

用法说明

`Df = diff(f)` 对 f 进行微分，微分变量由 `symvar(f,1)` 自动确定。

`Df = diff(f, var)` 对 f 关于变量 `var` 求偏导。

`Df = diff(f, var, n)` 计算 f 关于 `var` 的 n 阶导数。

`Df = diff(f, var1, ..., varN)` 对 f 关于多个变量依次求偏导（混合偏导数）。

参数说明

参数	类型	说明
`f`	符号表达式	被求导表达式
`var`	符号变量	求导变量
`n`	整数标量	导数阶数

> **当前限制**：支持符号变量和符号表达式，不支持符号向量、矩阵和方程。

示例

计算二阶偏导数

```
syms x y
expr = exp(y) * (sin(x) + cos(x));
df = diff(expr, x, 2);
disp(df)
```

```
-(sin(x) + cos(x))*exp(y)
```

相关函数

sym | syms | int

---

int

求不定积分或定积分

语法

```
F = int(expr)
F = int(expr, Name, Value)
```

用法说明

`F = int(expr)` 计算 `expr` 的不定积分，积分变量由 `symvar` 自动确定。

`F = int(expr, Name, Value)` 通过名称-值对参数指定附加选项。

参数说明

参数	类型	说明
`expr`	符号表达式	被积函数
`'Hold'`	`true` | `false`（默认）	为 `true` 时返回未计算的积分形式

示例

单变量不定积分

```
x = sym("x");
expr = -2*x / (1 + x^2)^2;
F = int(expr);
disp(F)
```

```
1/(x^2 + 1)
```

多元函数的不定积分

```
syms x z
f = x / (1 + z^2);
Fx = int(f);   % 默认对 x 积分
disp(Fx)
```

```
x^2/(2*(z^2 + 1))
```

相关函数

sym | diff | integrateByParts

---

limit

符号表达式的极限

语法

```
limit(f, var, a)
limit(f, var, a, "left")
limit(f, var, a, "right")
```

用法说明

`limit(f, var, a)` 返回 `var` 趋近于 `a` 时 f 的双向极限。

`limit(f, var, a, "left")` 返回左极限。

`limit(f, var, a, "right")` 返回右极限。

参数说明

参数	类型	说明
`f`	符号表达式	输入表达式
`var`	符号变量	自变量
`a`	数字 | 符号数 | 符号表达式	极限点

> **当前限制**：支持符号变量和符号表达式，不支持符号向量和矩阵。

示例

双向极限

```
x = sym("x");
h = sym("h");
f = sin(x) / x;
l = limit(f, x, 0);
disp(l)
```

```
1
```

左极限与右极限

```
x = sym("x");
f = 1 / x;
xr = limit(f, x, 0, "right");
xl = limit(f, x, 0, "left");
disp(xr)   % oo
disp(xl)   % -oo
```

```
oo
-oo
```

相关函数

diff | int

---

integrateByParts

分部积分

语法

```
G = integrateByParts(F, du)
```

用法说明

`G = integrateByParts(F, du)` 对积分表达式 F 应用分部积分法，其中微分 `du` 被积分。

> **当前限制**：积分项含有非积分式时会报错。

参数说明

参数	类型	说明
`F`	符号表达式	未计算的积分表达式
`du`	符号表达式	微分项

示例

函数乘积的分部积分

```
syms x u(x) v(x)
F = int(u * diff(v));
G = integrateByParts(F, diff(v));
disp(G)
```

```
u(x)*v(x) - Integral(v(x)*Derivative(u(x), x), x)
```

指数函数的分部积分

```
x = sym("x");
expr = int(x^2 * exp(x), "hold", true);
ret = integrateByParts(expr, exp(x));
disp(ret)
```

```
x^2*exp(x) - Integral(2*x*exp(x), x)
```

相关函数

syms | int

---

vpaintegral

使用可变精度进行数值积分

语法

```
vpaintegral(f, a, b)
vpaintegral(f, x, a, b)
vpaintegral(___, Name, Value)
```

用法说明

`vpaintegral(f, a, b)` 对 f 从 a 到 b 进行数值近似积分，积分变量由 `symvar` 自动确定。

`vpaintegral(f, [a b])` 等价于 `vpaintegral(f, a, b)`。

`vpaintegral(f, x, a, b)` 使用指定积分变量 x 执行数值积分。

`vpaintegral(___, Name, Value)` 通过名称-值对参数指定附加选项。

参数说明

参数	类型	说明
`f`	符号变量 | 符号表达式	被积函数
`a, b`	数值 | 符号数值 | 符号表达式	积分上下限，必须为实数
`x`	符号变量	积分变量（默认由 `symvar` 确定）
`'Waypoints'`	数值向量 | 符号表达式	积分路径点，用于分段积分或围道积分

> **当前限制**：`Name` 仅支持 `'Waypoints'`。

示例

基本数值积分

```
syms x
s = vpaintegral(x^2, 1, 2);
disp(s)
```

```
2.33333333333333
```

嵌套二重积分

```
syms x y
s = vpaintegral(vpaintegral(x*y, x, [1 3]), y, [-1 2]);
disp(s)
```

```
6.00000000000000
```

相关函数

diff | int | integrateByParts

---

4. 级数与求和

series

计算符号表达式的 Puiseux 级数展开

语法

```
series(f, var)
```

用法说明

`series(f, var)` 对 f 在 `var = 0` 处进行 Puiseux 级数展开，默认展开到第五阶。若未指定 `var`，则由 `symvar(f,1)` 自动确定。

参数说明

参数	类型	说明
`f`	符号表达式	待展开的表达式
`var`	符号变量	展开变量

示例

```
syms x y
disp(series(1/sin(x)))
```

```
31*x^5/15120 + 7*x^3/360 + x/6 + 1/x
```

```
f = exp(x) / x;
s = series(f);
disp(s)
```

```
x^5/720 + x^4/120 + x^3/24 + x^2/6 + x/2 + 1 + 1/x
```

相关函数

symprod | cumsum

---

taylor

泰勒级数展开

语法

```
T = taylor(f, var)
T = taylor(f, var, a)
T = taylor(f, var, a, order)
```

用法说明

`T = taylor(f, var)` 在 `var = 0` 处对 f 进行泰勒级数展开，默认展开到第五阶（截断阶数为 6）。

`T = taylor(f, var, a)` 在 `var = a` 处进行泰勒展开。

`T = taylor(f, var, a, order)` 指定截断阶数 `order`，展开到 `order - 1` 阶（O 项中的指数为 `order`）。

参数说明

参数	类型	默认值	说明
`f`	符号表达式	—	被展开的函数
`var`	符号变量	—	展开变量
`a`	数字 | 符号数 | 符号表达式	`0`	展开点，不能依赖于展开变量
`order`	正整数 | 符号正整数	`6`	截断阶数

示例

Maclaurin 级数（默认展开到五阶）

```
x = sym("x");
T1 = taylor(exp(x), x);
disp(T1)
```

```
x^5/120 + x^4/24 + x^3/6 + x^2/2 + x + 1
```

```
T2 = taylor(sin(x), x);
disp(T2)
```

```
x^5/120 - x^3/6 + x
```

指定展开点

```
x = sym("x");
T = taylor(exp(x), x, 3);
disp(T)
```

```
(x^5 - 10*x^4 + 50*x^3 - 120*x^2 + 165*x - 78)*exp(3)/120
```

指定截断阶数

```
x = sym("x");
f = sin(x) / x;
T6 = taylor(f, x);         % 默认截断阶数 6
T7 = taylor(f, x, 0, 7);  % 截断阶数 7
disp(T6)
disp(T7)
```

```
x^4/120 - x^2/6 + 1
-x^6/5040 + x^4/120 - x^2/6 + 1
```

相关函数

series | sym

---

symprod

级数的乘积

语法

```
F = symprod(f, k, a, b)
F = symprod(f, k)
```

用法说明

`F = symprod(f, k, a, b)` 返回 f 关于 k 从 a 到 b 的级数乘积。

`F = symprod(f, k)` 返回 k 从 1 到上界的级数乘积，上界以 k 的形式返回（与不定乘积不同）。

> **注意**：部分结果可能与其他符号计算软件有差异。

参数说明

参数	类型	说明
`f`	符号表达式	级数项定义表达式
`k`	符号变量	乘积索引变量
`a`	数值 | 符号变量	索引下界
`b`	数值 | 符号变量	索引上界

示例

```
syms k
P1 = symprod(1 - 1/k^2, k, 2, Inf);
P2 = symprod(k^2/(k^2 - 1), k, 2, Inf);
disp(P1)
disp(P2)
```

```
1/2
2
```

```
syms i t
y = symprod(t - i, i, 0, 5);
disp(y)
```

```
t*(t - 5)*(t - 4)*(t - 3)*(t - 2)*(t - 1)
```

相关函数

cumprod | cumsum

---

cumsum

计算符号数组的累积和

语法

```
B = cumsum(A)
B = cumsum(A, dim)
```

用法说明

`B = cumsum(A)` 返回 A 的累积和，从第一个大小不等于 1 的维度开始。

`B = cumsum(A, dim)` 沿维度 `dim` 计算累积和。

- `cumsum(A, 1)`：按列累积和
- `cumsum(A, 2)`：按行累积和

参数说明

参数	类型	说明
`A`	符号向量 | 符号矩阵	输入数组，仅支持符号变量数组，不支持表达式
`dim`	正整数标量	运算维度

示例

向量累积和

```
syms x y z
A = horzcat(x, y, z);
cs = cumsum(A);
disp(cs)
```

```
Matrix([[x, x + y, x + y + z]])
```

矩阵累积和

```
syms x y z t
B = vertcat(horzcat(x, y), horzcat(z, t));
disp(cumsum(B, 1))   % 按列
disp(cumsum(B, 2))   % 按行
```

```
Matrix([[x, y], [x + z, t + y]])
Matrix([[x, x + y], [z, t + z]])
```

相关函数

cumprod | int

---

cumprod

计算符号数组的累积乘积

语法

```
B = cumprod(A)
B = cumprod(A, dim)
```

用法说明

`B = cumprod(A)` 返回 A 的累积乘积，从第一个大小不等于 1 的维度开始。

`B = cumprod(A, dim)` 沿维度 `dim` 计算累积乘积。

- `cumprod(A, 1)`：按列累积乘积
- `cumprod(A, 2)`：按行累积乘积

参数说明

参数	类型	说明
`A`	符号向量 | 符号矩阵	输入数组，仅支持符号变量数组，不支持表达式

参数	类型	说明
`dim`	正整数标量	运算维度

示例

向量累积乘积

```
syms x y z real
A = horzcat(x, y, z);
disp(cumprod(A))
```

```
Matrix([[x, x*y, x*y*z]])
```

矩阵累积乘积

```
syms x y z t real
B = vertcat(horzcat(x, y), horzcat(z, t));
disp(cumprod(B, 1))   % 按列
disp(cumprod(B, 2))   % 按行
```

```
Matrix([[x, y], [x*z, t*y]])
Matrix([[x, x*y], [z, t*z]])
```

相关函数

cumsum

---

vpasum

使用可变精度数值计算级数和

语法

```
s = vpasum(f, a, b)
s = vpasum(f, x, a, b)
```

用法说明

`s = vpasum(f, a, b)` 对 f 从 a 到 b 进行数值近似求和，求和变量由 `symvar(f)` 自动确定。求和上下限必须为实数。

`s = vpasum(f, [a, b])` 等价于 `vpasum(f, a, b)`。

`s = vpasum(f, x, a, b)` 使用指定的求和变量 x 执行数值求和。

> **注意**：部分结果可能与其他符号计算软件有差异。

参数说明

参数	类型	说明
`f`	符号变量 | 符号表达式	要求和的表达式
`a, b`	数值 | 符号数值 | 符号表达式	求和上下限，必须为实数
`x`	符号变量	求和变量（默认由 `symvar(f)` 确定）

示例

```
syms x
s = vpasum(1/1.2^x, 1, 1000);
disp(s)
```

```
5.00000000000000
```

相关函数

vpaintegral | symprod

---

5. 方程求解

---

solve

方程和方程组的符号求解器

语法

```
S = solve(eqn, var)
S = solve(eqns, vars)
```

用法说明

`S = solve(eqn, var)` 关于变量 `var` 符号求解方程 `eqn`。

`Y = solve(eqns, vars)` 关于变量组 `vars` 求解方程组 `eqns`，以结构体形式返回解。

> **注意**：
>
> - `eqn` 须为符号方程（使用 `==`），不支持直接传入表达式
> - 使用 `horzcat`/`vertcat` 组合多个方程或变量，不支持 `[]` 语法

参数说明

参数	类型	说明
`eqn`	符号方程	用 `==` 定义的符号等式
`var`	符号变量	求解变量
`eqns`	符号方程组	多个方程的组合
`vars`	符号向量 | 符号矩阵	多个求解变量的组合

输出参数

参数	类型	说明
`S`	元胞数组	单方程的解，大小对应解的数量
`Y`	结构体	方程组的解，字段数对应变量数量

示例

求解二次方程

```
syms a b c x
eqn = a*x^2 + b*x + c == 0;
S = solve(eqn, x);
disp(S{1})
disp(S{2})
```

```
(-b - sqrt(-4*a*c + b^2))/(2*a)
(-b + sqrt(-4*a*c + b^2))/(2*a)
```

高次方程（遍历所有根）

```
syms x
eqn = x^5 == 3125;
S = solve(eqn, x);
for i = 1:length(S)
    disp(S{i});
end
```

```
5
-5/4 + 5*sqrt(5)/4 - 5*I*sqrt(2*sqrt(5) + 10)/4
...
```

线性方程组

```
syms u v
eqns = horzcat(2*u + v == 0, u - v == 1);
Y = solve(eqns, horzcat(u, v));
disp(Y.u)
disp(Y.v)
```

```
1/3
-2/3
```

相关函数

vpasolve | eq

---

vpasolve

数值求解符号方程

语法

```
S = vpasolve(eqn, var)
Y = vpasolve(eqns, vars)
```

用法说明

`S = vpasolve(eqn, var)` 使用变精度算术数值求解方程 `eqn` 中的变量 `var`，默认返回 32 位有效数字的解。

`Y = vpasolve(eqns, vars)` 数值求解方程组 `eqns`，返回包含各变量解的结构体数组。

> **注意**：
>
> - `eqn` 须为符号方程（使用 `==`），不支持直接传入表达式
> - 对于多解问题，当前仅能返回一组解

参数说明

参数	类型	说明
`eqn`	符号方程	用 `==` 定义的方程
`var`	符号变量	求解变量
`eqns`	符号向量 | 符号矩阵	方程组
`vars`	符号向量 | 符号矩阵	多个求解变量

输出参数

参数	类型	说明
`S`	符号值 | 符号数组	一元方程的解

参数	类型	说明
`Y`	结构体数组	方程组的解，字段数对应变量数量

示例

求解非多项式方程

```
syms x
S = vpasolve(sin(x) == 1/2, x);
disp(S)
```

```
0.523598775598299
```

求解方程组

```
syms u v
Y = vpasolve(horzcat(v^3 + 2*u == v, v^2 == u), horzcat(u, v));
disp(Y.u)
disp(Y.v)
```

```
0.171572875253810
0.414213562373095
```

相关函数

solve | eq

---

6. 积分变换

---

laplace

拉普拉斯变换

语法

```
F = laplace(f)
F = laplace(f, var, transVar)
```

用法说明

`F = laplace(f)` 返回 f 的拉普拉斯变换，默认自变量为 `t`，变换变量为 `s`。若 f 不含 `t`，则 `laplace` 通过 `symvar` 自动确定自变量。

`F = laplace(f, var, transVar)` 使用指定的自变量 `var` 和变换变量 `transVar`。

参数说明

参数	类型	默认值	说明
`f`	符号表达式	—	输入函数
`var`	符号变量	`t`	时间变量（自变量）
`transVar`	符号变量	`s`	复频率变量（变换变量）

示例

基本拉普拉斯变换

```
syms x y
f = 1 / sqrt(x);
F = laplace(f);
disp(F)
```

```
sqrt(pi)/sqrt(s)
```

指定变换变量

```
syms a t y
f = exp(-a*t);
F = laplace(f);        % 默认：对 t 变换，结果为 s
disp(F)                % 1/(a + s)
F2 = laplace(f, a, y);  % 对 a 变换，结果为 y
disp(F2)               % 1/(t + y)
```

```
1/(a + s)
1/(t + y)
```

相关函数

ilaplace | fourier

---

ilaplace

拉普拉斯逆变换

语法

```
f = ilaplace(F)
f = ilaplace(F, transVar)
```

用法说明

`f = ilaplace(F)` 返回 F 的拉普拉斯逆变换，默认自变量为 `s`，变换变量为 `t`。

`f = ilaplace(F, transVar)` 使用指定的变换变量 `transVar`。

> **当前限制**：
>
> 1. 不支持 F 为符号数值
> 2. 显示结果均附有 `Heaviside(t)` 因子

参数说明

参数	类型	默认值	说明
`F`	符号表达式	—	频域函数
`transVar`	符号变量	`t`	时间变量（变换变量）

示例

基本逆变换

```
syms s
F = 1/s^2;
f = ilaplace(F);
disp(f)
```

```
t*Heaviside(t)
```

指定变换变量

```
syms a s x
F = 1/(s - a)^2;
f = ilaplace(F, x);
disp(f)
```

```
x*exp(a*x)*Heaviside(x)
```

相关函数

laplace | fourier | ifourier

---

fourier

连续傅里叶变换

语法

```
FT = fourier(f)
FT = fourier(f, transVar)
FT = fourier(f, var, transVar)
```

用法说明

`FT = fourier(f)` 返回 f 的傅里叶变换，默认自变量由 `symvar` 确定，变换变量为 `w`。

`FT = fourier(f, transVar)` 使用指定的变换变量 `transVar`。

`FT = fourier(f, var, transVar)` 同时指定自变量 `var` 和变换变量 `transVar`。

> **注意**：部分结果可能与其他符号计算软件有差异（角频率归一化处理差异）。

参数说明

参数	类型	默认值	说明
`f`	符号表达式	—	输入函数（时域）
`var`	符号变量	`symvar(f,1)`	时间变量（自变量）
`transVar`	符号变量	`w`	频率变量（变换变量）

示例

```
syms a t w
f1 = exp(-t^2);
f2 = sin(t);
syms b
f = fourier(a*f1 + b*f2, t, w);
disp(f)
```

```
sqrt(pi)*a*exp(-pi^2*w^2)
```

相关函数

ifourier | laplace

---

ifourier

傅里叶逆变换

语法

```
ifourier(F)
ifourier(F, transVar)
ifourier(F, var, transVar)
```

用法说明

`ifourier(F)` 计算 F 的傅里叶逆变换，默认自变量为 `w`，变换变量为 `t`。

`ifourier(F, transVar)` 使用指定的变换变量 `transVar`，自变量仍为 `w`。

`ifourier(F, var, transVar)` 完全指定自变量 `var` 和变换变量 `transVar`。

> **注意**：部分结果可能与其他符号计算软件有差异，原因为角频率归一化处理方式不同，计算结果本身是正确的。

参数说明

参数	类型	默认值	说明
`F`	符号表达式	—	频域函数
`var`	符号变量	`w`	频率变量（自变量）
`transVar`	符号变量	`t`	时间变量（变换变量）

示例

高斯函数逆变换

```
syms w
F = exp(-w^2/4);
W = ifourier(F);
disp(W)
```

```
2*sqrt(pi)*exp(-4*pi^2*t^2)
```

指定变换变量

```
syms a w x
F = exp(-w^2 - a^2);
W = ifourier(F, x);
disp(W)
```

```
sqrt(pi)*exp(-a^2 - pi^2*x^2)
```

奇函数频谱逆变换

```
syms w t
F = -(sqrt(sym(pi)) * w * exp(-w^2/4) * 1i) / 2;
W = ifourier(F, w, t);
disp(W)
```

```
4*pi^2*t*exp(-4*pi^2*t^2)
```

相关函数

fourier | ilaplace | laplace

---

7. 化简、代换与精度控制

---

simplify

代数简化

语法

```
S = simplify(expr)
```

用法说明

`S = simplify(expr)` 对符号表达式执行代数简化。

> **当前限制**：仅支持符号表达式，不支持符号向量和矩阵。

参数说明

参数	类型	说明
`expr`	符号表达式	输入表达式

示例

```
syms x y
S = simplify(sin(x)^2 + cos(x)^2);
disp(S)
```

```
1
```

```
S = simplify((y + 2) * (y - 2));
disp(S)
```

```
y^2 - 4
```

相关函数

sym | syms | subs

---

subs

符号替换

语法

```
snew = subs(s, match, replacement)
```

用法说明

`snew = subs(s, match, replacement)` 返回 s 的副本，将所有出现的 `match` 替换为 `replacement` 并计算结果。

> **当前限制**：支持符号变量、表达式和方程，不支持符号向量和矩阵。

参数说明

参数	类型	说明
`s`	符号表达式 | 符号函数	原始表达式
`match`	符号表达式 | 符号函数	要替换的变量或子表达式
`replacement`	数字 | 符号表达式	替换值

示例

```
syms a b
expr = subs(a + b, a, 4);
disp(expr)
```

```
b + 4
```

相关函数

syms | simplify

---

vpa

任意精度数值计算

语法

```
F = vpa(expr)
F = vpa(expr, digits)
```

用法说明

`F = vpa(expr)` 将符号表达式 `expr` 转换为默认精度（32 位有效数字）的数值形式。

`F = vpa(expr, digits)` 将符号表达式转换为指定精度 `digits` 的数值形式。

> **当前限制**：支持符号表达式，不支持数值（double）、向量和矩阵。

参数说明

参数	类型	说明
`expr`	符号表达式 | 符号变量	需要计算的符号表达式
`digits`	正整数标量	有效数字位数

示例

默认精度计算

```
x = sym(pi);
ret = vpa(x);
disp(ret)
```

```
3.14159265358979323846264338327795
```

指定精度计算

```
x = sym(pi);
ret = vpa(x, 50);
disp(ret)
```

```
3.1415926535897932384626433832795028841971693993751
```

相关函数

sym | double

---

double

将符号数值转换为浮点数

语法

```
D = double(s)
```

用法说明

`D = double(s)` 计算符号表达式的有限精度浮点值，默认精度为 15 位（实际打印位数由北太天元设置决定）。

> **注意**：`double` 不能处理含有未知变量的符号表达式，只能将符号数值转换为浮点数。
>
> **当前限制**：仅支持符号数值，不支持向量和矩阵。

示例

```
a = sym(2);
b = sqrt(a);
disp(b)
```

```
sqrt(2)
```

```
expr = double(b)
```

```
expr =

    1.4142
```

相关函数

sym | vpa

---

8. 初等函数

以下函数均通过 classdef 重载，接受符号变量或符号表达式作为输入，与数值版本共用同一函数名。

> **当前限制**：所有初等函数均不支持符号向量和矩阵。

函数	说明	语法
`sqrt`	平方根	`sqrt(x)`
`abs`	绝对值	`abs(z)`
`exp`	指数函数	`exp(x)`
`log`	自然对数	`log(x)`
`sin`	正弦	`sin(x)`
`cos`	余弦	`cos(x)`
`tan`	正切	`tan(x)`
`asin`	反正弦	`asin(x)`
`acos`	反余弦	`acos(x)`

函数	说明	语法
`atan`	反正切	`atan(x)`
`sinh`	双曲正弦	`sinh(x)`
`cosh`	双曲余弦	`cosh(x)`
`tanh`	双曲正切	`tanh(x)`
`asinh`	反双曲正弦	`asinh(x)`
`acosh`	反双曲余弦	`acosh(x)`

示例

```
syms x
f = sin(x)^2 + cos(x)^2;
disp(simplify(f))
```

```
1
```

```
syms x
g = exp(log(x));
disp(simplify(g))
```

```
x
```

---

9. 已知限制说明

以下为 v2.0 Preview 版本的已知限制，将在后续版本中逐步改进：

函数	已知限制
`diff`	不支持符号向量、矩阵和方程
`int`	仅支持部分 Name-Value 参数
`limit`	不支持符号向量和矩阵
`simplify`	不支持符号向量和矩阵
`subs`	不支持符号向量和矩阵
`vpa`	不支持数值（double）、向量和矩阵
`double`	仅支持符号数值，不支持向量和矩阵
`sqrt`、`abs`、`exp`、`log`、三角函数	不支持符号向量和矩阵

函数	已知限制
`ilaplace`	结果均附有 `Heaviside(t)` 因子；不支持 F 为符号数值
`fourier` / `ifourier`	部分结果与其他软件有差异（角频率归一化处理不同，结果正确）
`symprod` / `vpasum`	部分结果与其他软件有差异
`cumsum` / `cumprod`	输入 A 仅支持符号变量数组，不支持表达式
`vpasolve`	对多解问题仅能返回一组解
`integrateByParts`	积分项含有非积分式会报错

> 如在使用过程中遇到 bug 或与预期不符的结果，欢迎在社区原帖下方留言反馈。
> 你的每一条反馈都将帮助 v2.0 正式版变得更好。