# Python 的符号计算

刘德华

2023 年 7 月 15 日

# 1 基本操作

### 1.0.1 对符号代入数字求值

[1]:
```
from sympy import *
# 定义符号 xyz
x, y, z = symbols("x y z")
```

[2]:
```
# 定义表达式
expr = cos(x) + 1
# 将 x 替换为 y
expr.subs(x, y)
```

[2]: $\cos(y) + 1$

[3]:
```
# 将 x 替换为 0, 求值
expr.subs(x, 0)
```

[3]: $2$

[4]:
```
# 定义表达式
expr = x**3 + 4*x*y - z
# 代入数字求值
expr.subs([(x, 2), (y, 4), (z, 0)])
```

[4]: $40$

### 1.0.2 将字符串转为表达式

[1]:
```
from sympy import *
# 定义符号 xyz
x, y, z = symbols("x y z")
```

[2]:
```
# 字符串
str_expr = "x**2 + 3*x - 1/2"
# 转为表达式
expr = sympify(str_expr)
expr
```

[2]: $x^2 + 3x - \frac{1}{2}$

[3]:
```
expr.subs(x, 2)
```

[3]: $\frac{19}{2}$

### 1.0.3 对表达式求值（数值结果）

[1]:
```
from sympy import *
# 定义符号 xyz
x, y, z = symbols("x y z")
```

[2]:
```
# 表达式，根号 8
expr = sqrt(8)
expr.evalf()
```

[2]: 2.82842712474619

[3]:
```
# pi 保留 100 位小数
pi.evalf(100)
```

[3]: 3.14159265358979323846264338327950288419716939937510582097494459230781640628620899862803482534211706

[4]:
```
# 表达式
expr = cos(2*x)
# 将 x 替换为 2.4 代入求值
expr.evalf(subs={x: 2.4})
```

[4]: 0.0874989834394464

[5]:
```
# 定义表达式
one = cos(1)**2 + sin(1)**2
(one - 1).evalf()
```

[5]: $-4.0 \cdot 10^{-124}$

[6]:
```
# 精确结果
(one - 1).evalf(chop=True)
```

[6]: 0

### 1.0.4 对表达式在多个点上求值（不使用循环）

[1]:
```
from sympy import *
# 定义符号 xyz
x, y, z = symbols("x y z")
```

[2]:
```
import numpy as np
a = np.arange(10)
# 定义表达式
expr = sin(x)
# 定义函数，输出结果为 numpy 的数组类型
f = lambdify(x, expr, "numpy")
f(a)
```

[2]:
```
array([ 0.        ,  0.84147098,  0.90929743,  0.14112001, -0.7568025 ,
       -0.95892427, -0.2794155 ,  0.6569866 ,  0.98935825,  0.41211849])
```

[3]:
```
# 输出结果为 math 库的类型
f = lambdify(x, expr, "math")
f(0.1)
```

[3]: 0.09983341664682815

### 1.0.5 化简表达式

[1]:
```
from sympy import *
# 定义符号
x, y, z = symbols("x y z")
```

[2]:
```
# 化简表达式
simplify(sin(x)**2 + cos(x)**2)
```

[2]: $1$

[3]:
```
# 化简表达式
simplify((x**3 + x**2 - x - 1)/(x**2 + 2*x + 1))
```

[3]: $x - 1$

[4]:
```
# 化简表达式
simplify(gamma(x)/gamma(x - 2))
```

[4]: $(x - 2)(x - 1)$

[5]:
```
# 化简表达式，无法化简，这是展开后的最简表达式
simplify(x**2 + 2*x + 1)
```

[5]: $x^2 + 2x + 1$

### 1.0.6 多项式展开

[1]:
```
from sympy import *
# 定义符号
x, y, z = symbols("x y z")
```

[2]:
```
# 展开
expand((x + 1)**2)
```

[2]: $x^2 + 2x + 1$

[3]:
```
# 展开
expand((x + 2)*(x - 3))
```

[3]: $x^2 - x - 6$

### 1.0.7 因式分解

[1]:
```
from sympy import *
# 定义符号
x, y, z = symbols("x y z")
```

[2]:
```
# 将多项式因式分解
factor(x**3 - x**2 + x - 1)
```

[2]: $(x - 1)(x^2 + 1)$

[3]:
```
# 将多项式因式分解
factor(x**2*z + 4*x*y*z + 4*y**2*z)
```

[3]: $z\left(x+2y\right)^{2}$

#### 1.0.8 构造关于某个特定变量的多项式

[1]:
```
from sympy import *
# 定义符号
x, y, z = symbols("x y z")
```

[2]:
```
# 定义表达式
expr = x*y + x - 3 + 2*x**2 - z*x**2 + x**3
expr
```

[2]: $x^{3}-x^{2}z+2x^{2}+xy+x-3$

[3]:
```
# 构造关于 x 的多项式
collected_expr = collect(expr, x)
collected_expr
```

[3]: $x^{3}+x^{2}\cdot\left(2-z\right)+x\left(y+1\right)-3$

## 2 化简

#### 2.0.1 分式约分

[1]:
```
from sympy import *
# 定义符号
x, y, z = symbols("x y z")
```

[2]:
```
cancel((x**2 + 2*x + 1)/(x**2 + x))
```

[2]: $\frac{x+1}{x}$

[3]:
```
expr = 1/x + (3*x/2 - 2)/(x - 4)
expr
```

[3]: $\frac{\frac{3x}{2}-2}{x-4}+\frac{1}{x}$

[4]:
```
# 通分约分
cancel(expr)
```

[4]: $\frac{3x^{2}-2x-8}{2x^{2}-8x}$

[5]:
```
expr = (x*y**2 - 2*x*y*z + x*z**2 + y**2 - 2*y*z + z**2)/(x**2 - 1)
expr
```

[5]: $\frac{xy^{2}-2xyz+xz^{2}+y^{2}-2yz+z^{2}}{x^{2}-1}$

[6]:
```
cancel(expr)
```

[6]:
$$\frac{y^2 - 2yz + z^2}{x - 1}$$

### 2.0.2 将分式分解成多个有理分式的和

[1]:
```
from sympy import *
# 定义符号
x, y, z = symbols("x y z")
```

[2]:
```
expr = (4*x**3 + 21*x**2 + 10*x + 12)/(x**4 + 5*x**3 + 5*x**2 + 4*x)
expr
```

[2]:
$$\frac{4x^3 + 21x^2 + 10x + 12}{x^4 + 5x^3 + 5x^2 + 4x}$$

[3]:
```
apart(expr)
```

[3]:
$$\frac{2x - 1}{x^2 + x + 1} - \frac{1}{x + 4} + \frac{3}{x}$$

### 2.0.3 三角函数化简

[1]:
```
from sympy import *
# 定义符号
x, y, z = symbols("x y z")
```

[2]:
```
trigsimp(sin(x)**2 + cos(x)**2)
```

[2]:
$$1$$

[3]:
```
trigsimp(sin(x)**4 - 2*cos(x)**2*sin(x)**2 + cos(x)**4)
```

[3]:
$$\frac{\cos\left(4x\right)}{2} + \frac{1}{2}$$

[4]:
```
trigsimp(sin(x)*tan(x)/sec(x))
```

[4]:
$$\sin^2\left(x\right)$$

[5]:
```
trigsimp(cosh(x)**2 + sinh(x)**2)
```

[5]:
$$\cosh\left(2x\right)$$

[6]:
```
trigsimp(sinh(x)/tanh(x))
```

[6]:
$$\cosh\left(x\right)$$

### 2.0.4 三角函数展开

[1]:
```
from sympy import *
# 定义符号
x, y, z = symbols("x y z")
```

[2]:
```
expand_trig(sin(x + y))
```

[2]:
$$\sin\left(x\right)\cos\left(y\right) + \sin\left(y\right)\cos\left(x\right)$$

[3]:
```
expand_trig(tan(2*x))
```

[3]:
$$\frac{2\tan\left(x\right)}{1 - \tan^2\left(x\right)}$$

### 2.0.5 幂次化简

[1]:
```
from sympy import *
# 定义符号
x, y = symbols("x y", positive=True)
a, b = symbols("a b", real=True)
z, t, c = symbols("z t c")
```

[2]:
```
powsimp(x**a*x**b)
```

[2]: $x^{a+b}$

[3]:
```
powsimp(x**a*y**a)
```

[3]: $(xy)^{a}$

[4]:
```
# c 没有任何约束，可以是复数
powsimp(t**c*z**c)
```

[4]: $t^{c}z^{c}$

[5]:
```
# 强制合并
powsimp(t**c*z**c, force=True)
```

[5]: $(tz)^{c}$

[6]:
```
(z*t)**2
```

[6]: $t^{2}z^{2}$

[7]:
```
sqrt(x*y)
```

[7]: $\sqrt{x}\sqrt{y}$

### 2.0.6 幂次展开

[1]:
```
from sympy import *
# 定义符号
x, y = symbols("x y", positive=True)
a, b = symbols("a b", real=True)
z, t, c = symbols("z t c")
```

[2]:
```
expand_power_exp(x**(a + b))
```

[2]: $x^{a}x^{b}$

[3]:
```
expand_power_base((x*y)**a)
```

[3]: $x^{a}y^{a}$

### 2.0.7 对数展开

[1]:
```
from sympy import *
# 定义符号
x, y, z = symbols("x y z", positive=True)
n = symbols("n", real=True)
```

[2]:
```
expand_log(log(x*y))
```

[2]: $\log(x) + \log(y)$

[3]:
```
expand_log(log(x/y))
```

[3]: $\log(x) - \log(y)$

[4]:
```
expand_log(log(x**2))
```

[4]: $2\log(x)$

[5]:
```
expand_log(log(x**n))
```

[5]: $n\log(x)$

[6]:
```
expand_log(log(z**2), force=True)
```

[6]: $2\log(z)$

#### 2.0.8 对数合并

[1]:
```
from sympy import *
# 定义符号
x, y, z = symbols("x y z", positive=True)
n = symbols("n", real=True)
```

[2]:
```
logcombine(log(x) + log(y))
```

[2]: $\log(xy)$

[3]:
```
logcombine(n*log(x))
```

[3]: $\log(x^n)$

[4]:
```
logcombine(n*log(z))
```

[4]: $\log(z^n)$

#### 2.0.9 阶乘

[1]:
```
from sympy import *
# 定义符号
x, y, z = symbols("x y z")
k, m, n = symbols("k m n")
```

[2]:
```
factorial(n)
```

[2]: $n!$

#### 2.0.10 组合数

[1]:
```
from sympy import *
# 定义符号
x, y, z = symbols("x y z")
k, m, n = symbols("k m n")
```

[2]:
```
binomial(n, k)
```

[2]: $\binom{n}{k}$

#### 2.0.11 伽马函数

[1]:
```
from sympy import *
# 定义符号
x, y, z = symbols("x y z")
k, m, n = symbols("k m n")
```

[2]:
```
gamma(z)
```

[2]: $\Gamma(z)$

#### 2.0.12 超几何分布函数

[1]:
```
from sympy import *
# 定义符号
x, y, z = symbols("x y z")
k, m, n = symbols("k m n")
```

[2]:
```
hyper([1, 2], [3], z)
```

[2]: ${}_2F_1\left(\left.\begin{matrix}1, 2\\ 3\end{matrix}\right| z\right)$

[3]:
```
# 将上面的函数展开
hyperexpand(hyper([1, 2], [3], z))
```

[3]: $-\frac{2}{z} - \frac{2\log(1-z)}{z^2}$

## 3 微积分

#### 3.0.1 求导

[1]:
```
from sympy import *
x, y, z = symbols("x y z")
```

[2]:
```
# 对 x 求导
diff(cos(x), x)
```

[2]: $-\sin(x)$

[3]:
```
# 求 x 求导
diff(exp(x**2), x)
```

[3]: $2xe^{x^2}$

[4]:
```
# 对 x 求三阶导
diff(x**4, x, 3)
```

[4]: $24x$

[5]:
```
expr = exp(x*y*z)
expr
```

[5]: $e^{xyz}$

[6]:
```
diff(expr, x).diff(y, 2).diff(z, 4)
```

[6]: $x^3y^2\left(x^2y^2z^2\left(xyz+2\right)+4x^2y^2z^2+8xyz\left(xyz+2\right)+36xyz+48\right)e^{xyz}$

### 3.0.2 积分

[1]:
```
from sympy import *
x, y, z = symbols("x y z")
```

[2]:
```
integrate(cos(x), x)
```

[2]: $\sin\left(x\right)$

[3]:
```
# 定积分
integrate(exp(-x), (x, 0, oo))
```

[3]: $1$

[4]:
```
# 二重积分
integrate(exp(-x**2 - y**2), (x, -oo, oo), (y, -oo, oo))
```

[4]: $\pi$

[5]:
```
# 无法求解这个积分
expr = integrate(x**x, x)
expr
```

[5]: $\int x^x\,dx$

[6]:
```
# 求解不出来
expr.doit()
```

[6]: $\int x^x\,dx$

[7]:
```
# 创建未求值的积分
expr = Integral(log(x)**2, x)
expr
```

[7]: $\int \log\left(x\right)^2\,dx$

[8]:
```
# 求解
expr.doit()
```

[8]: $x\log\left(x\right)^2-2x\log\left(x\right)+2x$

[9]:
```
integ = Integral(
    (x**4 + x**2*exp(x) - x**2 - 2*x*exp(x) -
     2*x - exp(x))*exp(x)/((x - 1)**2*(x + 1)**2*(exp(x) + 1)),
    x
)
```

```
integ
```

[9]:
$$\int \frac{\left(x^4 + x^2 e^x - x^2 - 2xe^x - 2x - e^x\right) e^x}{\left(x-1\right)^2 \left(x+1\right)^2 \left(e^x + 1\right)}\, dx$$

[10]:
```
integ.doit()
```

[10]:
$$\log\left(e^x + 1\right) + \frac{e^x}{x^2 - 1}$$

[11]:
```
integ = Integral(sin(x**2), x)
integ
```

[11]:
$$\int \sin\left(x^2\right) dx$$

[12]:
```
integ.doit()
```

[12]:
$$\frac{3\sqrt{2}\sqrt{\pi} S\left(\frac{\sqrt{2}x}{\sqrt{\pi}}\right) \Gamma\left(\frac{3}{4}\right)}{8\Gamma\left(\frac{7}{4}\right)}$$

### 3.0.3 极限

[1]:
```
from sympy import *
x, y, z = symbols("x y z")
```

[2]:
```
limit(sin(x)/x, x, 0)
```

[2]:
$$1$$

[3]:
```
expr = x**2/exp(x)
expr.subs(x, oo)
```

[3]:
NaN

[4]:
```
limit(expr, x, oo)
```

[4]:
$$0$$

[5]:
```
expr = Limit((cos(x) - 1)/x, x, 0)
expr
```

[5]:
$$\lim_{x \to 0^+}\left(\frac{\cos\left(x\right) - 1}{x}\right)$$

[6]:
```
expr.doit()
```

[6]:
$$0$$

[7]:
```
# 右极限
limit(1/x, x, 0, '+')
```

[7]:
$$\infty$$

[8]:
```
# 左极限
limit(1/x, x, 0, '-')
```

[8]:
$$-\infty$$

#### 3.0.4 级数展开

[1]:
```
from sympy import *
x, y, z = symbols("x y z")
```

[2]:
```
expr = exp(sin(x))
# 在零处展开四阶
expr.series(x, 0, 4)
```

[2]:
$$1 + x + \frac{x^2}{2} + O\left(x^4\right)$$

[3]:
```
# 省略无穷小量
expr.series(x, 0, 4).removeO()
```

[3]:
$$\frac{x^2}{2} + x + 1$$

[4]:
```
# 在 x=6 处展开
exp(x - 6).series(x, x0=6)
```

[4]:
$$-5 + \frac{(x-6)^2}{2} + \frac{(x-6)^3}{6} + \frac{(x-6)^4}{24} + \frac{(x-6)^5}{120} + x + O\left((x-6)^6; x \to 6\right)$$

## 4 矩阵

#### 4.0.1 创建矩阵

[1]:
```
from sympy import *
```

[2]:
```
Matrix([[1, -1], [3, 4], [0, 2]])
```

[2]:
$$\begin{bmatrix} 1 & -1 \\ 3 & 4 \\ 0 & 2 \end{bmatrix}$$

[3]:
```
Matrix([1, 2, 3])
```

[3]:
$$\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

[4]:
```
# 矩阵乘法
M = Matrix([[1, 2, 3], [3, 2, 1]])
N = Matrix([0, 1, 1])
M*N
```

[4]:
$$\begin{bmatrix} 5 \\ 3 \end{bmatrix}$$

#### 4.0.2 获取矩阵的行列数

[1]:
```
from sympy import *
```

[2]:
```
M = Matrix([[1, 2, 3], [-2, 0, 4]])
M
```

[2]:
$$\begin{bmatrix} 1 & 2 & 3 \\ -2 & 0 & 4 \end{bmatrix}$$

[3]:
```
shape(M)
```

[3]: (2, 3)

### 4.0.3 获取矩阵的某一行或者某一列

[1]:
```
from sympy import *
```

[2]:
```
M = Matrix([[1, 2, 3], [-2, 0, 4]])
M
```

[2]:
$$\begin{bmatrix} 1 & 2 & 3 \\ -2 & 0 & 4 \end{bmatrix}$$

[3]:
```
# 第一行
M.row(0)
```

[3]:
$$\begin{bmatrix} 1 & 2 & 3 \end{bmatrix}$$

[4]:
```
# 最后一列
M.col(-1)
```

[4]:
$$\begin{bmatrix} 3 \\ 4 \end{bmatrix}$$

### 4.0.4 删除矩阵的某一行或者某一列

[1]:
```
from sympy import *
```

[2]:
```
M = Matrix([[1, 2, 3], [-2, 0, 4]])
M
```

[2]:
$$\begin{bmatrix} 1 & 2 & 3 \\ -2 & 0 & 4 \end{bmatrix}$$

[3]:
```
# 删除第一列
M.col_del(0)
M
```

[3]:
$$\begin{bmatrix} 2 & 3 \\ 0 & 4 \end{bmatrix}$$

[4]:
```
# 删除第二行
M.row_del(1)
M
```

[4]:
$$\begin{bmatrix} 2 & 3 \end{bmatrix}$$

#### 4.0.5 在矩阵中插入某一行或者某一列

[1]:
```
from sympy import *
```

[2]:
```
M = Matrix([[1, 2, 3], [-2, 0, 4]])
M
```

[2]:
$$\begin{bmatrix} 1 & 2 & 3 \\ -2 & 0 & 4 \end{bmatrix}$$

[3]:
```
# 插入行到第二行的位置
M = M.row_insert(1, Matrix([[0, 4, 3]]))
M
```

[3]:
$$\begin{bmatrix} 1 & 2 & 3 \\ 0 & 4 & 3 \\ -2 & 0 & 4 \end{bmatrix}$$

[4]:
```
# 插入列到第一列的位置
M = M.col_insert(0, Matrix([1, -2, 0]))
M
```

[4]:
$$\begin{bmatrix} 1 & 1 & 2 & 3 \\ -2 & 0 & 4 & 3 \\ 0 & -2 & 0 & 4 \end{bmatrix}$$

#### 4.0.6 矩阵的加减乘“除”运算

[1]:
```
from sympy import *
```

[2]:
```
M = Matrix([[1, 3], [-2, 3]])
N = Matrix([[0, 3], [0, 7]])
```

[3]:
```
M
```

[3]:
$$\begin{bmatrix} 1 & 3 \\ -2 & 3 \end{bmatrix}$$

[4]:
```
N
```

[4]:
$$\begin{bmatrix} 0 & 3 \\ 0 & 7 \end{bmatrix}$$

[5]:
```
M + N
```

[5]:
$$\begin{bmatrix} 1 & 6 \\ -2 & 10 \end{bmatrix}$$

[6]:
```
# 矩阵的乘法
M*N
```

[6]:
$$\begin{bmatrix} 0 & 24 \\ 0 & 15 \end{bmatrix}$$

[7]:
```
# 矩阵的数乘
3*M
```

[7]:
$$\begin{bmatrix} 3 & 9 \\ -6 & 9 \end{bmatrix}$$

[8]:
```
# 矩阵的幂乘
M**2
```

[8]:
$$\begin{bmatrix} -5 & 12 \\ -8 & 3 \end{bmatrix}$$

[9]:
```
# 矩阵的逆
M**-1
```

[9]:
$$\begin{bmatrix} \frac{1}{3} & -\frac{1}{3} \\ \frac{2}{9} & \frac{1}{9} \end{bmatrix}$$

[10]:
```
# 转置
M.T
```

[10]:
$$\begin{bmatrix} 1 & -2 \\ 3 & 3 \end{bmatrix}$$

#### 4.0.7 构造单位阵

[1]:
```
from sympy import *
```

[2]:
```
eye(3)
```

[2]:
$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

[3]:
```
eye(4)
```

[3]:
$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

#### 4.0.8 构造零矩阵

[1]:
```
from sympy import *
```

[2]:
```
zeros(2, 3)
```

[2]:
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

#### 4.0.9 构造一矩阵

[1]:
```
from sympy import *
```

[2]:
```
ones(3, 2)
```

[2]:
$$\begin{bmatrix} 1 & 1 \\ 1 & 1 \\ 1 & 1 \end{bmatrix}$$

#### 4.0.10 构造对角矩阵

[1]:
```
from sympy import *
```

[2]:
```
diag(1, 2, 3)
```

[2]:
$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix}$$

[3]:
```
# 对角块矩阵
diag(-1, ones(2, 2), Matrix([5, 7, 5]))
```

[3]:
$$\begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 5 \\ 0 & 0 & 0 & 7 \\ 0 & 0 & 0 & 5 \end{bmatrix}$$

#### 4.0.11 行列式

[1]:
```
from sympy import *
```

[2]:
```
M = Matrix([[1, 0, 1], [2, -1, 3], [4, 3, 2]])
M
```

[2]:
$$\begin{bmatrix} 1 & 0 & 1 \\ 2 & -1 & 3 \\ 4 & 3 & 2 \end{bmatrix}$$

[3]:
```
M.det()
```

[3]:
$$-1$$

#### 4.0.12 最简行阶梯形矩阵

[1]:
```
from sympy import *
```

[2]:
```
M = Matrix([[1, 0, 1, 3], [2, 3, 4, 7], [-1, -3, -3, -4]])
M
```

[2]:
$$\begin{bmatrix} 1 & 0 & 1 & 3 \\ 2 & 3 & 4 & 7 \\ -1 & -3 & -3 & -4 \end{bmatrix}$$

[3]:
```
M.rref()[0]
```

[3]:
$$\begin{bmatrix} 1 & 0 & 1 & 3 \\ 0 & 1 & \frac{2}{3} & \frac{1}{3} \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

[4]:
```
M.rref()[1]
```

[4]:
```
(0, 1)
```

#### 4.0.13 矩阵的核空间（零空间）

[1]:
```
from sympy import *
```

[2]:
```
M = Matrix([[1, 2, 3, 0, 0], [4, 10, 0, 0, 1]])
M
```

[2]:
$$\begin{bmatrix} 1 & 2 & 3 & 0 & 0 \\ 4 & 10 & 0 & 0 & 1 \end{bmatrix}$$

[3]:
```
for i in M.nullspace():
    print(i)
```

```
Matrix([[-15], [6], [1], [0], [0]])
Matrix([[0], [0], [0], [1], [0]])
Matrix([[1], [-1/2], [0], [0], [1]])
```

[4]:
```
M.nullspace()[0]
```

[4]:
$$\begin{bmatrix} -15 \\ 6 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

[5]:
```
M.nullspace()[1]
```

[5]:
$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

[6]:
```
M.nullspace()[2]
```

[6]:
$$\begin{bmatrix} 1 \\ -\frac{1}{2} \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

#### 4.0.14 矩阵的列向量的线性组合构成的空间

[1]:
```
from sympy import *
```

[2]:
```
M = Matrix([[1, 1, 2], [2 ,1 , 3], [3 , 1, 4]])
M
```

[2]:
$$\begin{bmatrix}1 & 1 & 2\\2 & 1 & 3\\3 & 1 & 4\end{bmatrix}$$

[3]:
```
for i in M.columnspace():
    print(i)
```

```
Matrix([[1], [2], [3]])
Matrix([[1], [1], [1]])
```

[4]:
```
M.columnspace()[0]
```

[4]:
$$\begin{bmatrix}1\\2\\3\end{bmatrix}$$

[5]:
```
M.columnspace()[1]
```

[5]:
$$\begin{bmatrix}1\\1\\1\end{bmatrix}$$

#### 4.0.15 矩阵特征值

[1]:
```
from sympy import *
```

[2]:
```
M = Matrix([[3, -2,  4, -2], [5,  3, -3, -2], [5, -2,  2, -2], [5, -2, -3,  3]])
M
```

[2]:
$$\begin{bmatrix}3 & -2 & 4 & -2\\5 & 3 & -3 & -2\\5 & -2 & 2 & -2\\5 & -2 & -3 & 3\end{bmatrix}$$

[3]:
```
M.eigenvals()
```

[3]:
```
{3: 1, -2: 1, 5: 2}
```

#### 4.0.16 矩阵特征向量

[1]:
```
from sympy import *
```

[2]:
```
M = Matrix([[3, -2,  4, -2], [5,  3, -3, -2], [5, -2,  2, -2], [5, -2, -3,  3]])
M
```

[2]:
$$\begin{bmatrix} 3 & -2 & 4 & -2 \\ 5 & 3 & -3 & -2 \\ 5 & -2 & 2 & -2 \\ 5 & -2 & -3 & 3 \end{bmatrix}$$

[3]:
```
M.eigenvects()
```

[3]:
```
[(-2,
  1,
  [Matrix([
   [0],
   [1],
   [1],
   [1]])]),
 (3,
  1,
  [Matrix([
   [1],
   [1],
   [1],
   [1]])]),
 (5,
  2,
  [Matrix([
   [1],
   [1],
   [1],
   [0]]),
   Matrix([
   [ 0],
   [-1],
   [ 0],
   [ 1]])])]
```

#### 4.0.17 矩阵相似对角化

[1]:
```
from sympy import *
```

[2]:
```
M = Matrix([[3, -2,  4, -2], [5,  3, -3, -2], [5, -2,  2, -2], [5, -2, -3,  3]])
M
```

[2]:
$$\begin{bmatrix} 3 & -2 & 4 & -2 \\ 5 & 3 & -3 & -2 \\ 5 & -2 & 2 & -2 \\ 5 & -2 & -3 & 3 \end{bmatrix}$$

[3]:
```
P, D = M.diagonalize()
```

[4]:
```
P
```

[4]:
$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & -1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix}$$

[5]:
```
D
```

[5]:
$$\begin{bmatrix} -2 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \\ 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 5 \end{bmatrix}$$

[6]:
```
P*D*P**-1
```

[6]:
$$\begin{bmatrix} 3 & -2 & 4 & -2 \\ 5 & 3 & -3 & -2 \\ 5 & -2 & 2 & -2 \\ 5 & -2 & -3 & 3 \end{bmatrix}$$

[7]:
```
P*D*P**-1 == M
```

[7]:
```
True
```

# 5 解方程

### 5.0.1 求解一元方程

[1]:
```
from sympy import *
# 定义变量
x, y = symbols("x y")
```

[2]:
```
# 求解方程
solve(x**2 - y, x)
```

[2]:
```
[-sqrt(y), sqrt(y)]
```

[3]:
```
solveset(x**2 - y, x)
```

[3]:
$$\{-\sqrt{y}, \sqrt{y}\}$$

### 5.0.2 求解多元方程组

[1]:
```
from sympy import *
# 定义变量
x, y, z = symbols("x y z")
```

[2]:
```
solve([x + y - 2*z, y + 4*z], [x, y])
```

[2]:
```
{x: 6*z, y: -4*z}
```

### 5.0.3 求解微分方程

[1]:
```
from sympy import *
# 定义变量
x, y, z = symbols("x y z")
```

[2]:
```
# 定义函数
y = Function("y")
# Solve the ODE
result = dsolve(Derivative(y(x), x, x) + 9*y(x), y(x))
result
```

[2]: $$y(x) = C_1 \sin(3x) + C_2 \cos(3x)$$

[3]:
```
# 检验结果
checkodesol(Derivative(y(x), x, x) + 9*y(x), result)
```

[3]: (True, 0)

### 5.0.4 求解线性方程组

[1]:
```
from sympy import *
c, d, e = symbols("c, d, e")
```

[2]:
```
A = Matrix([[c,d], [1, -e]])
A
```

[2]: $$\begin{bmatrix} c & d \\ 1 & -e \end{bmatrix}$$

[3]:
```
b = Matrix([2, 0])
b
```

[3]: $$\begin{bmatrix} 2 \\ 0 \end{bmatrix}$$

[4]:
```
A.solve(b)
```

[4]: $$\begin{bmatrix} \frac{2e}{ce+d} \\ \frac{2}{ce+d} \end{bmatrix}$$