Mathematica函数索引

常用的符号与常数

In[n]　第 n 个输入
Out[n] 或 %n　第 n 个输出
%　前一次的输出
%%　倒数第二次的输出
?name　查看变量 name 的信息
??name　查看变量 name 的更为详细的信息
!!filename　显示文件内容
<<filename　读入文件并执行
expr>>filename　将表达式保存到文件
expr>>>filename　将表达式添加到文件
{ }　表使用的括号
[ ]　函数使用的括号
" "　字符串使用的引号
N[expr , n]　求 expr 具有n位数字的近似值
N[expr]　求 expr 具有机器规定的精度的近似值
Clear[$symbol_1$, $symbol_2$, …]　清除一些符号的值
Remove[$symbol_1$, $symbol_2$, …]　清除一些符号
I　虚数单位i
Degree　角度的度
E　自然对数的底 e
Pi　圆周率 π
Infinity　无穷大
ComplexInfinity　复无穷大
Indeterminate　不定值
GoldenRatio　表示 $(\sqrt{5}+1)/2$
b^^nnn　表示 nnn 是一个 b 进位数
nnn`p　表示 nnn 是一个具有 p 位数字的任意精度数
Short[expr]　将输出结果缩略成一行显示
Short[expr , n]　将输出结果缩略成 n 行显示
Alt+, 或 Alt+.　强制中断计算

常用的数学函数

Sin[x]　正弦函数
Cos[x]　余弦函数
Tan[x]　正切函数
Cot[x]　余切函数
Sec[x]　正割函数
Csc[x]　余割函数

ArcSin[x]　反正弦函数
ArcCos[x]　反余弦函数
ArcTan[x]　反正切函数
ArcCot[x]　反余切函数
ArcSec[x]　反正割函数
ArcCsc[x]　反余割函数
Exp[x]　表示 e^x
Log[x]　表示 $\ln x$
Log[a , x]　表示以a为底的对数函数
Sqrt[x]　表示 $\sqrt{x}$
Abs[x]　求实数的绝对值或复数的模
Sign[x]　符号函数
n!　求 n 的阶乘
n!!　求 n 的双阶乘
Binomial[n , k]　求 C_n^k
Max[x_1 , x_2 , …]　一组数的最大值
Min [x_1 , x_2 , …]　一组数的最小值
Re[x]　复数 x 的实部
Im[x]　复数 x 的虚部
Arg[x]　复数 x 的辐角
Conjugate[x]　复数 x 的共轭数
Floor[x]　不超过 x 的最大整数
Ceiling[x]　大于或等于 x 的最小整数
Round[x]　最接近 x 的整数
Mod[m , n]　整数 m 被 n 除的余数
Quotient[m , n]　整数 m 被 n 除的整数部分
GCD[n_1 , n_2 , …]　一组整数的最大公约数
LCM[n_1 , n_2 , …]　一组整数的最小公倍数
FactorInteger[n]　将整数 n 分解成素数的积
Sinh[x]　双曲正弦函数
Cosh[x]　双曲余弦函数
Tanh[x]　双曲正切函数
Coth[x]　双曲余切函数
Sech[x]　双曲正割函数
Csch[x]　双曲余割函数
ArcSinh[x]　反双曲正弦函数
ArcCosh[x]　反双曲余弦函数
ArcTanh[x]　反双曲正切函数
ArcCoth[x]　反双曲余切函数
ArcSech[x]　反双曲正割函数
ArcCsch[x]　反双曲余割函数

代数函数

Simplify[expr]　使用变换化简表达式
FullSimplify[expr]　使用更广泛的变换化简表达式
Assuming[assum , expr]　将条件 assum 临时添加到系统变量 $Assumptions 的原有值中求表达式 expr
Refine[expr , assum]　将条件 assum 临时添加到系统变量 $Assumptions 的原有值中求出表达式 expr
Factor[expr]　因式分解
Collect[expr , x]　合并同类项
Collect[expr , {x_1 , x_2 , …}]　合并 x_1 ， x_2 ， … 的同类项
Expand[expr]　展开表达式
ExpandAll[expr]　展开表达式
ExpandNumerator[expr]　只展开分式的分子
ExpandDenominator[expr]　只展开分式的分母
Cancel[expr]　约去分子、分母的公因式
Apart[expr , var]　将变量var的有理式分解成最简分式的和
TrigExpand[expr]　将三角函数式展开
TrigFactor[expr]　将三角函数式因式分解
TrigReduce[expr]　用倍角化简三角函数式
TrigToExp[expr]　将三角函数式转换成指数形式
ExpToTrig[expr]　将指数形式转换成三角函数式
FunctionExpand[expr]　用于特殊函数的展开
ComplexExpand[expr]　展开复表达式
PowerExpand[expr]　将 $(xy)^p$ 展开成 $x^p y^p$
PolynomialQuotient[p_1 , p_2 , x]　求 x 的多项式 p_1 被 p_2 除的商
PolynomialRemainder[p_1 , p_2 , x]　求 x 的多项式 p_1 被 p_2 除的余式
PolynomialGCD[p_1 , p_2 , …]　求多个多项式的最大公因式
PolynomialLCM[p_1 , p_2 , …]　求多个多项式的最小公倍式
Coefficient[expr , form , n]　多项式 expr 中 form^n 的系数
CoefficientList[poly , var]　求变量为 var 的多项式 poly 的系数表（按升幂排列）
Exponent[expr , form]　多项式 expr 中 form 的最高次数

解方程

Solve[eqns , vars]　对系数按常规约定求出方程（组）的全部解
Solve[eqns , vars , elims]　消去变量 elims 解出变量 vars
Reduce[eqns , vars]　解方程或不等式（组）讨论系数给出所有可能的解
Root[f , k]　求多项式f的第 k 个根
Roots[lhs==rhs , var]　求变量 var 的多项式方程的所有根
Eliminate[eqns , elims]　从一组等式中消去变量 elims
InverseFunction[f]　求函数 f 的反函数
RSolve[eqn, a[n], n]　对未知量 a[n] 解递归方程

微积分

Limit[f , x→x_0]　求函数 f　当 x→x_0 时的极限
D[f , var]　求函数 f 对自变量 var 的偏导数
D[f , x_1 , x_2 , …]　求函数 f 对自变量 x_1 ， x_2 ， … 的混合偏导数
D[f , {x_1 , n_1} , {x_2 , n_2} , …]　求函数 f　对自变量的指定阶数的混合偏导数
Dt[f]　求 f 的全微分
Dt[f ,var]　求 f 对自变量 var 的全导数
SetAttributes[c , Constant]　声明 c 是常数
Integrate[f , x]　求 f(x) 的一个原函数
Integrate[f , {x , a , b}]　求 f(x) 的以 a，b 为下、上限的定积分
Integrate[f , {x , a , b} , {y , y_1 , y_2}]　求二重积分
Sum[f , {i , imin , imax}]　求和
Sum[f , {i , imin , imax} , {j , jmin , jmax} , …]　求多重和
Product[f , {i , imin , imax}]　求积
Product[f , {i , imin , imax} , {j , jmin , jmax} , …]　求多重积
Series[f , {x , x_0 , n}]　将 f(x) 在 x_0 处展成幂级数直到 n 次项为止
Series[f , {x , x_0 , n} , {y , y_0 , m}]　将 f(x , y) 先对 y 后对 x 展开
Normal[expr]　将幂级数 expr 去掉余项转换成多项式
SeriesCoefficient[expr , n]　找出幂级数 expr 的 n 次项系数
ComposeSeries[$series_1$, $series_2$, …]　复合幂级数
InverseSeries[f , v]　求幂级数f的反函数的幂级数展开式
FourierSeries[f , x , n]　将 f(x) 展成傅立叶级数直到 n 次项为止
FourierSinSeries[f , x , n]　将 f(x) 奇延拓展开成正弦级数
FourierCosSeries[f , x , n]　将 f(x) 偶延拓展开成余弦级数
DSolve[eqn , y[x] , x]　求微分方程 eqn 的通解
DSolve[{eqn , y[x_0] == y_0} , y[x] , x]　求微分方程满足条件 y[x_0] = y_0 的特解
DSolve[{eqn_1 , eqn_2 , …} , { y_1[x] , y_2[x] , …} , x]　求微分方程组的通解
DSolve[{eqn_1 , … , y_1[x_0] == y_{10} , … } , { y_1[x] , …} , x]　求微分方程组的特解
DSolve[eqn , y , x]　求方程 eqn 的通解y的纯函数形式
Residue[expr , {x , x_0}]　求 expr 在 x_0 点的留数
LaplaceTransform[f , t , s]　求函数 f (t) 的 Laplace 变换返回自变量为s的函数
InverseLaplaceTransform[F, s , t]　求函数 F(s) 的 Laplace 逆变换
FourierTransform[f , t , ω]　求函数 f (t) 的 Fourier 变换返回自变量为 ω 的函数
InverseFourierTransform[F , ω , t]　求函数 F(ω) 的 Fourier 逆变换
Ztransform[f , n , z]　求函数 f (n) 的Z变换
InverseZtransform[F, n , z]　求函数 F(z) 的 Z 逆变换
Minimize[{f , cons} , {x , y , …}]　求函数 f 的满足条件 cons 时的最小值
Maximize[{f , cons} , {x , y , …}]　求函数 f 的满足条件 cons 时的最大值

线性代数

MatrixForm[list]　将表 list 按矩阵形式输出
Array[a , n]　创建一个元素为 a[i] 的有 n 个元素的向量
Array[a , {m , n}]　创建一个 m 行 n 列的矩阵
IdentityMatrix[n]　创建一个 n 阶单位矩阵

DiagonalMatrix[list]　创建一个对角线上为表 list 的元素的方阵
M[[All , j]]　提取矩阵 M 的第 j 列元素组成一个表
Tr[A , List]　提取方阵 A 的对角线元素组成一个表
Dimensions[M]　求矩阵 M 的行列数
Dot[a , b] 或 a.b　求两个矩阵的乘积或两个向量的内积
Cross[a , b]　求两个向量的向量积
Transpose[M]　将矩阵 M 转置
Det[A]　求方阵 A 的行列式
MatrixRank[M]　求矩阵 M 的秩
Inverse[A]　求方阵A的逆矩阵
MatrixPower[A , n]　求 A^n
Eigenvalues[A]　求方阵 A 的全部特征值
Eigenvectors[A]　求方阵 A 的一组线性无关的特征向量
Eigensystem[A]　求全部特征值和对应的线性无关的特征向量组
CharacteristicPolynomial[A , x]　求方阵 A 的特征多项式 p(x)
JordanDecomposition[A]　求方阵A的 Jordan 标准形和过渡矩阵
MatrixExp[A]　求 e^A
RowReduce[M]　消元得到矩阵 M 的行最简形矩阵
NullSpace[M]　求齐次线性方程组 M x = 0 的一个基础解系
LinearSolve[M , b]　求线性方程组 M x = b 的一个特解
Orthogonalize [{v_1 , v_2 , …}]　将向量组{v_1 , v_2 , …}正交化并单位化
Normalize[v]　将向量单位化
Projection[v_1 , v_2]　求向量1在向量2方向上的投影
Norm[expr , p]　求向量或矩阵 expr 的p–范数

数值计算

Fit[data , funs , vars]　对数据用指定函数组进行最小二乘拟合
Fit[data , {1 , x} , x]　求形为 $y = a+bx$ 的近似函数式
Fit[data , {1 , x , x^2} , x]　求形为 $y = a+bx+cx^2$ 的近似函数式
Fit[data , {1 , x , y , x y} , {x , y}]　求形为 $z = a+bx+cy+dxy$ 的近似函数式
Chop[expr，δ]　去掉表达式 exp r的系数中绝对值小于 δ 的项
InterpolatingPolynomial[{{x_1 , f_1} , {x_2 , f_2} , …} , x]　求插值多项式
InterpolatingPolynomial[{f_1 , f_2 , …} , x]　求当自变量为1，2，… 时的插值多项式
InterpolatingPolynomial[{{x_1 , {f_1 , d f_1 , dd f_1 , …}} , …} , x]　规定导数求插值多项式
Interpolation[{{x_1 , f_1} , {x_2 , f_2} , …}]　由数据构造插值函数
Interpolation[{f_1 , f_2 , …}]　求当自变量为1，2，… 时的插值函数
Interpolation[{{x_1 , {f_1 , d f_1 , dd f_1 , …}} , …}]　规定导数求插值函数
ListInterpolation[list , {{x_1 , x_2 , …} , {y_1 , y_2 , …}}]　以表 ist 为函数值求插值函数
ListInterpolation[list , {{xmin , xmax} , {ymin , ymax}}]　自变量取等分点求插值函数
ListInterpolation[{{f_{11} , f_{12} , …} , {f_{21} , f_{22} , …} , …}]　自变量取正整数求插值函数
FunctionInterpolation[expr , {x , xmin , xmax} , {y , ymin , xmax} , …]　求插值函数
NIntegrate[f , {x , xmin , x_1 , x_2 , … , xmax}]　求数值积分其中 x_1 ， x_2 ， … 是奇异点
NIntegrate[f , {x , xmin , xmax} , {y , ymin , ymax} , …]　求多重数值积分

NSum[f , {i , imin , imax , di}]　求数值和
NProduct[f , {i , imin , imax , di}]　求数值积
NSolve[eqns , vars , n]　求方程（组）的有 n 位精度的数值解
FindRoot[eqn , {x , x_0}]　从 x_0 出发求未知量x的方程 eqn 的一个解
FindRoot[eqn , {x , x_0 , xmin , xmax}]　在指定的范围内求数值解
FindRoot[eqn , {x , {x_0 , x_1}}]　给出两个初值求数值解
FindRoot[{eqn_1 , eqn_2 , …} , {x , x_0} , {y , y_0} , …]　求方程组的数值解
NDSolve[eqns , {y_1 , y_2 , …} , {x , xmin , xmax}]　求常微分方程（组）的近似解
NDSolve[eqns , u , {x , xmin , xmax}, {t , tmin , tmax}]　求未知函数 u (x , t) 的偏微分方程 eqns 的近似解
FindMinimum[f , {x , x_0}]　求函数 f 的一个极小值点和极小值
FindMinimum[f , {x , {x_0 , x_1}}]　给出两个初值求极小值点和极小值
FindMinimum[f , {x , x_0} , {y , y_0} , …]　求多元函数的极小值点和极小值
LinearProgramming[c, m , b]　当 mx≥b 且 x≥0 时求函数 cx 的最小值点 x
NMaximize[f , {x , y , … }]　求自变量为 x ， y ， … 的函数 f 的最大值
NMaximize[{f , cons} , {x , y , … }]　求函数 f 的满足条件 cons 时的最大值
NMinimize[f , {x , y , … }]　求自变量为 x ， y ， … 的函数 f 的最小值
NMinimize[{f , cons} , {x , y , … }]　求函数 f 的满足条件 cons 时的最小值

概率统计

Random[type , range]　产生在指定类型和范围内的具有 n 位数字的随机数
Random[ ]　产生0，1之间的随机实数
Random[Integer]　产生0或1
Random[Complex]　产生单位正方形内的复随机数
RandomChoice[list , n]　从list中随机选取n个元素
SeedRandom[n]　以 n 为随机数发生器的种子
SeedRandom[]　用时间值重新设置种子
BernoulliDistribution[p]　Bernoulli 分布
BinomialDistribution[n , p]　二项分布
GeometricDistribution[p]　几何分布
HypergeometricDistribution[n , M , N]　超几何分布
PoissonDistribution[λ]　Poisson 分布
DiscreteUniformDistribution[m1 , m2]　取值为 $m1 \le k \le m2$ 的整数的均匀分布
NormalDistribution[μ , σ]　正态分布
UniformDistribution[min , max]　均匀分布
ExponentialDistribution[λ]　指数分布
StudentTDistribution[n]　t 分布
ChiSquareDistribution[n]　χ^2 分布
FRatioDistribution[n_1 , n_2]　F 分布
GammaDistribution[r , λ]　Γ 分布
CauchyDistribution[a , b]　Cauchy 分布
BetaDistribution[p , q]　Beta 分布
LogNormalDistribution[μ , σ]　对数正态分布

LaplaceDistribution[μ, λ] Laplace 分布

WeibullDistribution[α, λ] Weibull 分布

ParetoDistribution[A , r] Pareto 分布

RayleighDistribution[σ] Rayleigh 分布

NoncentralChiSquareDistribution[n , λ] 非中心 χ^2 分布

NoncentralStudentTDistribution[n , λ] 非中心 t 分布

NoncentralFRatioDistribution[n_1 , n_2 , λ] 非中心 F 分布

BinormalDistribution[{μ1 , μ2} , {σ1 , σ2} , ρ] 二元正态分布

PDF[dist , x] 求点 x 处的分布dist的密度值

CDF[dist , x] 求点 x 处的分布函数值

InverseCDF[dist , q] 求使 CDF[dist , x]≥q 的最小 x

SurvivalFunction[dist , x] 求随机变量大于x的概率

InverseSurvivalFunction[dist , q] 求使SurvivalFunction[dist , x]≤q 的最小 x

Probability[pred , x Ý dist] 假定 x 服从分布 dist ， 给出满足 pred 的事件概率

Expectation[expr , x Ý dist] 假定 x 服从分布 dist ， 给出 expr 的期望

TransformedDistribution[f[x] , x Ý dist] 求随机变量函数的分布

MarginalDistribution[dist , k] 求多元分布 dist 的第k个分量的边缘分布

RandomVariate[dist , n] 给出服从分布dist的n个伪随机数组成的表

Mean[dist] 求分布dist的期望

Variance[dist] 求方差

StandardDeviation[dist] 求标准差

ExpectedValue[f , dist , x] 求 $\boldsymbol{E}$f (x)

CharacteristicFunction[dist , t] 求特征函数φ(t)

Median[data] 求中值

Mean[data] 求平均值

Variance[data] 求方差（无偏估计）

StandardDeviation[data] 求标准差（无偏估计）

Moment[list , k] 求 k 阶（原点）矩

CentralMoment[list , k] 求 k 阶中心矩

Covariance[x , y] 求 x ， y 的协方差（无偏估计）

Correlation[x , y] 求x ， y 的相关系数

EmpiricalDistribution[list] 由样本值表 list 创建一个经验分布

FindDistributionParameters[data , dist] 由数据data得到分布dist的参数估计值

EstimatedDistribution[data , dist] 由数据data得到分布dist 的估计分布

MeanCI[data , KnownVariance→var] 方差已知的数学期望的区间估计*

MeanCI[data] 方差未知的数学期望的区间估计*

NormalCI[mean , sd] 方差已知的数学期望的区间估计*

StudentTCI[mean , se , dof] 方差未知的数学期望的区间估计*

MeanDifferenceCI[$data_1$, $data_2$, KnownVariance→{var_1 , var_2}] 方差已知的两个数学期望之差的区间估计*

MeanDifferenceCI[$data_1$, $data_2$] 方差未知的两个数学期望之差的区间估计*

VarianceCI[data] 方差的区间估计*

ChiSquareCI[variance , dof] 方差的区间估计*

VarianceRatioCI[$data_1$, $data_2$]　两个方差之比的区间估计*

FRatioCI[ratio , numdof , dendof]　两个方差之比的区间估计*

ZTest[data , σ^2 , μ0]　已知方差σ^2，由数据表data 检验总体数学期望 μ0 ，求出 P 值（基于正态分布）

TTest[data , μ0]　方差未知，由数据表 data 检验总体数学期望 μ0 ，求出 P 值 （基于t分布）

ZTest[{data1 , data2} , { σ_1^2 ， σ_2^2 } , μ0]　方差已知，检验 μ1－μ2 = μ0 ，求出 P 值（基于正态分布）

TTest[{data1 , data2} , μ0]　方差未知，检验 μ1－μ2 = μ0 ，求出 P 值（基于t分布）

VarianceTest[data , σ^2]　由数据表 data 检验总体方差 σ^2，求出 P 值（基于χ^2分布）

▲　VarianceTest[{data1 , data2 }, ratio]　由数据表 data1 和 data2 ，检验两个总体方差之比 ratio = $\frac{\sigma_1^2}{\sigma_2^2}$ ，求出 P 值（基于 F 分布）

DistributionFitTest[data , dist]　由数据表data检验总体是否服从分布 dist ，求出P值

LinearModelFit [data , funs , vars]　由表 data 的数据，求由基函数表 funs 中的函数的线性组合构成的回归方程，其中 funs 中的函数的自变量由表 vars 给出

NonlinearModelFit [data , expr , pars , vars]　参数与最佳拟合结果都与函数 FindFit 相同

ANOVA[data]　由表 data 的数据，完成一元方差分析*

ANOVA[data , model , factors]　由表 data 的数据按照给定的模式 model 对因子 factors 完成多元方差分析*

矩阵分解

LUDecomposition[A]　将方阵 A 进行 LU 分解

LUBackSubstitution[data , b]　由 LUDecomposition 对 A 分解所得数据 data 解线性方程组 Ax = b

CholeskyDecomposition[A]　对 Hermite 正定矩阵 A 进行 Cholesky 分解得到上三角矩阵 U 使 $\overline{U}'U = A$

QRDecomposition[M]　对矩阵 M 进行 QR 分解得到正交（酉）矩阵 Q 和上三角矩阵R 使 $\overline{Q}'R = M$

SchurDecomposition[A]　将一个由近似数组成的方阵 A 进行 Schur 分解得到正交（酉）矩阵 Q 和上三角或分块上三角矩阵 T 使 $QT\overline{Q}' = A$

SingularValueDecomposition[M]　对一个由近似数组成的矩阵 M 进行奇异值分解得到正交（酉）矩阵 U 和 V 、对角矩阵 S 使 $US\overline{V}' = M$

SingularValueList[M]　只返回由矩阵 M 的非零奇异值组成的表

HessenbergDecomposition[A]　将一个由近似数组成的方阵 A 进行 Hessenberg 分解得到正交（酉）矩阵 P 和准上三角矩阵 H 使 $PH\overline{P}' = A$

PseudoInverse[M]　求矩阵 M 的广义逆矩阵 M^+

SparseArray[{pos_1→val_1 , pos_2→val_2 , …}]　建立一个第 pos_i 个元素为 val_i 、其它元素为0的稀疏数组

SparseArray[{pos_1 , pos_2 , …}→{ val_1 , val_2 , …}]　与前一个函数功能相同

SparseArray[data , {d_1 , d_2 , …} , val]　建立一个特殊元素由 data 指定、其它元素的值为

val 、共有共有 $d_1 \times d_2 \times \cdots$ 个元素的稀疏数组
SparseArray[list] 将一个表 list 转换成稀疏数组
Normal[array] 将稀疏数组 array 转换成一个表
ArrayRules[array] 由稀疏数组 array 得到规则表 $\{pos_1 \to val_1, pos_2 \to val_2, \ldots\}$

表

Table[f , {i , imin , imax , stepi} , {j , jmin , jmax , stepj}] 建立通项为 f 的表
Range[min , max , step] 按初值终值步长生成一个表
Range[n] 生成前 n 个自然数的表
Array[a , $\{n_1, n_2, n_3\}$] 创建一个元素为 $a[i_1, i_2, i_3]$ 的表
t[[n]] 或 Part[t , n] 表 t 的第 n 个元素
t[[– n]] 或 Part[t , – n] 表 t 的倒数第 n 个元素
t[[$\{n_1, n_2, \ldots\}$]] 或 Part[t , $\{n_1, n_2, \ldots\}$] 表 t 的第 n_1 ， n_2 ， … 个元素
t[[i , j]] 或 Part[t , i , j] 表 t 的第 i 个子表的第 j 个元素
Length[t] 表 t 的元素的个数
Take[t , n] 提取表 t 的前 n 个元素
Take[t , – n] 提取表 t 的后 n 个元素
Take[t , {m , n}] 提取表 t 的第 m 到第 n 个元素
Insert[t , expr , n] 在表 t 的第 n 个位置插入元素 expr
Insert[t , expr , – n] 在表 t 的倒数第 n 个位置插入元素 expr
Prepend[t , expr] 在表 t 的第1个元素前面插入元素 expr
Append[t , expr] 在表 t 的尾部插入元素 expr
Delete[t , n] 删除表 t 的第 n 个元素
DeleteCases[t , pattern] 删除表 t 中的与模式 pattern 匹配的元素
Drop[t , n] 删除表 t 的前 n 个元素
Drop[t , – n] 删除表 t 的后 n 个元素。
Drop[t , {m , n}] 删除表 t 的第 m 到第 n 个元素
ReplacePart[t , expr , n] 用 expr 替换表 t 的第 n 个元素
ReplacePart[t , expr , – n] 用 expr 替换表 t 的倒数第 n 个元素
ReplacePart[t , expr , {i , j}] 用 expr 替换表 t 的元素 t[[i , j]]
Join[t_1 , t_2] 将表 t_1 和 t_2 连接成一个表
Union[t_1 , t_2] 取表 t_1和 t_2 的并集组成一个表
Intersection[t_1 , t_2] 取表 t_1和 t_2 的交集组成一个表
Union[t] 合并表 t 中的相同元素得到一个表
Complement[eall , e_1 , e_2 , …] 给出在表 eall 中但不在表 e_1 ， e_2 ， … 中的元素组成的表
Partition[t , n] 将表 t 的元素按 n 个一组生成子表
Flatten[t] 展开表 t 的各个子表
Sort[t] 将表 t 的元素按标准顺序排序
Reverse[t] 将表 t 的元素逆向排列
RotateLeft[t , n] 将表 t 的元素循环左移 n 次
RotateRight[t , n] 将表 t 的元素循环右移 n 次
Subsets[t] 给出一个以表 t 的所有子表为元素的表
Subsets[t , {n}] 给出表 t 的由有 n 个元素的子表组成的表

Subsets[t , n]　给出表 t 的由有至多 n 个元素的子表组成的表
Subsets[t , {m , n}]　给出表 t 的子表组成的表，其中子表的元素个数在 m 和 n 之间

绘图函数

Plot[f(x) , {x , a , b}]　绘制函数 f(x) 在区间 [a , b] 上的图形
Plot[{f_1(x) , f_2(x) , …}, {x , a , b}]　同时绘制多个函数的图形
ParametricPlot[{x(t) , y(t)} , {t , a , b}]　绘制由参数方程给出的函数的图形
ParametricPlot[{{x_1(t) , y_1(t)} , {x_2(t) , y_2(t)} , …} , {t , a , b}]　同时绘制多条曲线
ListPlot[{y_1 , y_2 , …}]　画出点列 (i , y_i)
ListPlot[{{x_1 , y_1 } , { x_2 , y_2 } , …}]　画出点列 (x_i , y_i)
ContourPlot[f , {x , xmin , xmax} , {y , ymin , ymax}]　绘制函数 f 的等值线图
ContourPlot[f = = g , {x , xmin , xmax} , {y , ymin , ymax}]　绘制f = g 的等高线
DensityPlot[f , {x , xmin , xmax} , {y , ymin , ymax}]　绘制函数 f 的密度图
PolarPlot[r(θ), {θ , α , β}]　绘制由极坐标方程 r = r(θ) 确定的曲线，其中θ 的取值范围是区间[α , β]
PolarPlot[{r_1(θ) , r_2 (θ) , …} , {θ , α , β}]　同时绘制多条曲线
RegionPlot [ineqs , {x , xmin , xmax} , {y , ymin , ymax}]　绘制由不等式（组）ineqs 所确定的平面区域
ImplicitPlot[eqn , ranges , options]　绘制隐函数方程 eqn 所确定的曲线*
InequalityPlot[ineqs , {x , xmin , xmax} , {y , ymin , ymax}]　绘制由不等式（组）ineqs 所确定的平面区域*
ComplexInequalityPlot[ineqs , {z , xmin , xmax}]　绘制由复变量 z 的不等式（组）ineqs 所确定的平面区域*
BarChart[list]　由表list 给出的数据画出条形图
PieChart[list]　由表list 给出的数据绘制饼图
VectorPlot [{fx , fy } , {x , xmin , xmax} , {y , ymin , ymax}]　由已知的向量函数在指定的区域中绘制向量场
Plot3D[f , {x , xmin , xmax} , {y , ymin , ymax}]　绘制由函数f给出的曲面
ListPlot3D[{{x_1 , y_1, z_1} , {x_2 ,y_2 , z_2} , …}]　产生一个曲面，在 (x_i , y_i,) 处的高度为 z_i
ParametricPlot3D[{x(t) , y(t) , z(t)} , {t , a , b}]　绘制由三维参数方程给出的曲线
ParametricPlot3D[{x(u , v) , y(u , v) , z(u , v)} , {u , umin , umax} , {v , vmin , vmax}]　绘制由参数方程给出的曲面
SphericalPlot3D[r (φ ， θ) , { φ ， φ min ， φ max} , { θ ， θ min ， θ max}]　其中r 是 φ 和 θ 的函数，而 x = r sin φ cos θ ， y = r sin φ sin θ ， z = r cos φ
RevolutionPlot3D[f [x] , {x , xmin, xmax}]　将 oxz 平面上方程为 z = f [x] 的曲线绕 z 轴旋转一周生成曲面
RevolutionPlot3D[{x [t] , z [t]},{t , xmin, xmax}]　oxz 平面上的曲线方程由参数式 x = x [t] ， z = z [t] 给出
RevolutionPlot3D[{x [t] , y [t] , z [t]},{t , xmin, xmax}]　空间的曲线方程由参数式 x = x [t] ， y = y[t] ， z = z [t] 给出
RegionPlot3D[ineqs , {x , xmin , xmax} , {y , ymin , ymax} , {z , zmin , zmax}]　绘制由不等式（组）ineqs 所表示的区域的边界曲面
InequalityPlot3D[ineqs , {x , xmin , xmax} , {y , ymin , ymax} , {z , zmin , zmax}]　绘制由不

等式（组）ineqs 所表示的区域的边界曲面*
VectorPlot3D [{P , Q , R} , {x , xmin , xmax}, {y , ymin , ymax}, {z , zmin , zmax}] 用于绘制三维向量场，其中 P ， Q ， R 是向量的坐标，它们是 x ， y ， z 的函数
Shadow[g] 将三维图形对象g投影到坐标平面上*
Evaluate[expr] 求 expr 的值
Manipulate[带参数的表达式 , 参数的值域] 进行动态绘图与计算
Tooltip[expr , label] 当鼠标指向 expr 的显示位置时，产生一个显示为 label 的提示条

图形可选参数

PlotRange 指定作图的范围
AspectRatio 指定图形的高宽比
Axes 用于指定是否显示坐标轴
AxesOrigin 用于指定两个坐标轴的交点位置
AxesLabel 给坐标轴加上标记
Ticks 用于给坐标轴加上刻度或给坐标轴上的点加标记
AxesStyle 设置坐标轴的颜色和线宽等选项
Frame 给图形加框
GridLines 加网格线
Background 用于指定背景颜色
PlotLabel 在图形上方居中加注释
DisplayFunction 指定如何显示图形或声音
PlotStyle 用于规定曲线的线型和颜色
PlotPoints 规定作图时使用样本点的数目
Boxed 是否给图形加立体框
BoxRatios[r_x , r_y , r_z] 给出3个方向上的长度比
Mesh 说明在曲面上是否画网格
ColorFunction 决定曲面的颜色
ViewPoint 设置观察点

图形表达式

Graphics[list] 一般二维图形
Graphics3D[list] 一般三维图形
Show[{g_1 , g_2 , …} , options] 将多个图形 g_1 ， g_2 ， … 组合成一个图形进行显示
Show[GraphicsGrid [[list]] 将多个图形按行列排列同时显示，其中 list 是按矩阵给出的由图形组成的表

图形元素

Point[{x , y}] 坐标为 (x , y) 的点
Line[{{x_1 , y_1 } , { x_2 , y_2 } , …}] 顺次连接各点的折线
Circle[{x , y} , r] 圆心坐标为 (x , y) 且半径为 r 的圆
Circle[{x , y} , r , {θ_1 , θ_2}] 从角 θ_1 到角 θ_2 的圆弧

Circle[{x , y} , {a , b}]]　半轴为 a、b 的椭圆
Circle[{x , y} , {a , b} , { θ_1 , θ_2}]]　椭圆弧
Rectangle[{xmin , ymin } , {xmax , ymax}]　按指定颜色填充的矩形
Polygon[{{x_1 , y_1 } , { x_2 , y_2 } , …}]　填充多边形。
Disk[{x , y} , r]　填充圆。
Text["text" , {x , y}]　以点 (x , y) 为中心在图上标注字符串
Point[{x , y , z}]　坐标为 (x , y , z) 的点
Line[{{x_1 , y_1 , z_1 } , { x_2 , y_2 , z_2 } , …}]　顺次连接各点的折线
Cuboid[{xmin , ymin , zmin} , {xmax , ymax , zmax}]　立方体
Polygon[{{x_1 , y_1 , z_1 } , { x_2 , y_2 , z_2 } , …}]　填充多边形。
Text["text" , {x , y , z}]　以点 (x , y , z) 为中心在图上标注字符串
RGBColor[r , g , b]　RGB 颜色
GrayLevel[k]　灰度
Hue[h , s , b]　由色调、饱和度和亮度给出颜色
PointSize[d]　点的大小
Thickness[r]　线宽
Dashing[{r_1 , r_2 , …}]　虚线长度

动画和声音

Animate[带参数的表达式 , 参数的值域]　由“带参数的表达式”生成动画
ListAnimate[list]　由表 list 生成动画
Play[f , {t , tmin , tmax}]　播放函数 f 产生的声音
ListPlay[{a_1 , a_2 , …}]　播放幅值数列产生的声音
EmitSound　播放声音
SampleRate　采样频率
SampleDepth　表示每一个幅值时所用的字节数
PlayRange　播放的幅值范围

函数与变换规则

Function[x , body] 或 body&　表示纯函数
#n　表示纯函数的第 n 个自变量
##　表示纯函数的所有自变量
##n　表示纯函数的从第 n 个起往后的所有自变量
Attributes[f]　查看名为 f 的函数的属性
Attributes[f] = {$attr_1$, $atttr_2$, …}　设置函数 f 的属性
Attributes[f] = {}　设置函数无任何属性
SetAttributes[f , {$attr_1$, $atttr_2$, …}]　添加属性到 f 的属性表中
ClearAttributes[f , {$attr_1$, $atttr_2$, …} }　清除 f 的指定属性
Unset[lhs] 或 lhs=.　清除一个规则
Clear[f]　清除 f 的定义式但不清除属性
ClearAll[f]　清除 f 的定义式和属性但不清除符号 f
Remove[f]　清除符号 f

Protect[f]　给函数 f 加上 Protected 属性
Unprotect[f]　清除函数 f 的 Protected 属性
expr /. Rules 或 ReplaceAll[expr , rules]　对表达式中各项尝试使用一次规则表中的规则
expr //. Rules 或 ReplaceRepeated[expr , rules]　反复使用规则表中的规则直到结果不变
Replace[expr , rules]　对整个表达式试用一次规则表中的规则
Replace[expr , rules , levelspec]　对由层号 levelspec 指定的表达式的各层试用规则表中的规则
Level[expr , levelspec]　查看位于层号指定各层的所有子表达式
Depth[expr]　返回一个数等于表达式的层数加1
ReplaceList[expr , rules]　对整个表达式试用规则表中的每一个规则
ReplaceList[expr , rules , n]　给出至多有 n 个结果的表
pattern /; condition　给模式附加条件
FullForm[expr]　返回表达式的完全形式
Head[expr]　返回表达式的头
expr//f　后缀形式
f@expr　前缀形式
Apply[f , expr] 或f@@expr　将表达式 expr 的头换成 f
Apply[f , expr , levelspec]　用 f 替换 expr 的层号 levelspec 指定的各层的头
Map[f , expr] 或f/@expr　将 f 作用到表达式 expr 的第一层元素上
Map[f , expr , levelspec]　将 f 作用到层号 levelspec 指定的各层元素上
MapAt[f , expr , n]　将 f 作用到表达式 expr 的第 n 个元素上
MapAt[f , expr , {i , j}]　将 f 作用到两层表达式的元素 expr[[i , j]] 上
MapAt[f , expr , {{i_1 , j_1} , {i_2 , j_2} , …}]　将f作用到两层表达式 expr 的多个元素上
MapThread[f , {{x_1 , x_2 , …} , {y_1 , y_2 , …}}]　得到二元函数值的集合{f [x_1 , y_1] , f [x_2 , y_2] , …}
Cases[list , pattern]　从表 list 中选出与模式 pattern 匹配的所有元素得到一个新表
Cases[list , pattern –> rhs]　对表 list 中与模式 pattern 匹配的所有元素使用变换法则，返回由所得到的 rsh 的值组成的一个新表
Cases[expr , pattern]　从表达式 expr 的第1层元素中选出与模式 pattern 匹配的所有元素得到一个新表
Cases[expr , pattern –> rhs]　对表达式 expr 的第1层元素中与模式 pattern 匹配的所有元素使用变换法则，返回由所得到的 rsh 的值组成的一个新表
Except[c]　表示一个除了表达式c以外的任何表达式
Except[c , pattern]　表示一个除了表达式c以外的模式pattern
Select[list , crit]　从表 list 中选出满足判断函数的所有元素得到一个新表
Select[list , crit , n]　从表中选出满足判断函数的前 n 个元素得到一个新表
Select[expr , crit]　从表达式 expr 的第1层元素中删除不满足判断函数 crit 的元素得到一个新表达式（操作是对表达式 expr 的完全形式进行的）
SetOptions[f , $option_1$→$value_1$, …]　设置函数 f 的一个或多个可选参数的默认值
Options[f]={$option_1$→$value_1$, …}　建立函数 f 的全部可选参数默认值的集合
Options[f]　显示函数 f 的全部可选参数默认值的集合
Options[f , option]　显示可选参数 option 的默认值
Piecewise[{{val1 , cond1} , {val1 , cond1} , …}]　表示一个在条件 condi 规定的区域内的函数值为 vali 的分段函数

Piecewise[{{val1 , cond1} , …} , val]　当不满足任何条件 condi 时函数值为 val（默认值是0）
Boole[expr]　当表达式 expr 成立时函数值为1、当表达式 expr 不成立时函数值为0

程序控制

If [test , then , else , unknown]　If 型条件结构
Which[$test_1$, $value_1$, $test_2$, $value_2$, …]　Which 型条件结构
Switch[expr , $form_1$, $value_1$, $form_2$, $value_2$, …]　Switch 型条件结构
For[start , test , incr , body]　For 型循环结构
While[test , body]　While 型循环结构
Do[body , {i , imin , imax } , {j , jmin , jmax } , …]　Do 型循环结构
Nest[f , expr , n]　求 f [… f [f [expr]] …]（共复合 n 次）
NestList[f , expr , n]　返回一个有 n+1 项的表 {expr , f [expr] , f [f [expr]] , …}
FixedPoint[f , expr]　反复迭代直到不再变化为止
Return[expr]　中断一个函数的求值过程返回 expr
Return[ ]　返回结果为 Null
Goto[tag]　由 Goto[tag] 处跳转到 Label[tag] 处再向后执行
Break[ ]　退出本层的循环
Continue[ ]　跳转到下一次循环
Module[{x = x_0, y = y_0 , …} , body]　模块结构
Block[{x = x_0 , y = y_0 , …} , body]　块结构
Context[ ]　给出当前的上下文
Context[s]　给出符号 s 的上下文
BeginPackage["context`"]　将 $Context 变成 context` 并将 $ContextPath 变成 {context` , System`}
BeginPackage["context`" , {"$need_1$`" , "$need_2$`" ，…}]　在 $ContextPath 中还添加其他上下文
EndPackage[ ]　结束 Package 回到以前的状态并将新的上下文添加到 $ContextPath 中
Begin["`Private`"]　将 $Context 变成子上下文 context` Private`
End[ ]　将 $Context 返回到上下文 context`
DeclarePackage["context`" , {"$name_1$`" , "$name_2$`" ， …}]　声明一个能被自动装入的程序包
On[ ]　打开对所有变量和函数的跟踪
On[$name_1$, $name_2$, …]　打开对名为 $name_1$ ， $name_2$ ， … 的变量或函数的跟踪
Off[ ]　关闭对所有变量和函数的跟踪
Off[$name_1$, $name_2$, …]　关闭对名为 $name_1$ ， $name_2$ ， … 的变量或函数的跟踪
Trace[expr]　生成计算表达式 expr 时所有中间结果组成的表
Trace[expr , form]　生成仅包含与模式 form 匹配的那些中间表达式组成的表

判断函数

Equal[$expr_1$, $expr_2$] 或 ==　判定两个表达式相等
SameQ[$expr_1$, $expr_2$] 或 ===　判定两个表达式相等
FreeQ[expr , form]　判定在表达式 expr 中没有子表达式与 form 匹配
MemberQ[expr , form]　判定在表达式 expr 中有第1层元素与 form 匹配

MemberQ[expr , form , levelspec]　判定在表达式 expr 的由层号 levelspec 指定的层中有元素与 form 匹配
NumericQ[expr]　判定表达式 expr 是一个数量
NumberQ[expr]　判定表达式 expr 是一个数
IntegerQ[expr]　判定表达式 expr 是一个整数
EvenQ[expr]　判定表达式 expr 是一个偶数
OddQ[expr]　判定表达式 expr 是一个奇数
PrimeQ[expr]　判定表达式 expr 是一个素数
PolynomialQ[expr , {x_1 , x_2 , …}]　判定表达式 expr 是一个多元多项式

输入输出函数

Get["name" , Path→{"Path$_1$" , "Path$_2$" , …}]　调入文件时指明查找路径
Read["name" , type]　按指定类型读取文件的一个数据
Read["name" , {type$_1$, type$_2$, …}]　按指定类型一次读取多个数据
ReadList["name" , type]　按指定类型读取文件的全部数据返回一个表
ReadList["name" , {type$_1$, type$_2$, …}]　按第2个参数的指示将多个数据组成子表
Input[ ]　返回用户由键盘输入的表达式
Input["prompt"]　在弹出窗口的上半部显示提示字符串
InputString[ ]　限制输入内容为字符串
InputString["prompt"]　带有提示字符串
Import ["name.ext"]　从一个文件中输入该文件的内容
Put[expr$_1$, expr$_2$, … , "name"]　将表达式保存到文件中
PutAppend[expr$_1$, expr$_2$, … , "name"]　将表达式添加到文件中
Save["name" , f , g , …]　添加变量或函数到文件中
Export ["name.ext" , expr]　保存表达式到文件中
Sow[val]　为函数 Reap 设立需要显示的值 val
Reap[expr]　求表达式 expr 的值并返回 expr 中由函数 Sow 设立的值组成的一个表
Sow[val , tag]　为函数 Reap 设立需要显示的值 val 并为该值加上标记 tag
Reap[expr , patt]　求表达式 expr 的值并返回 expr 中由函数 Sow 设立的与 patt 匹配的标记 tag 的值组成的一个表。
Sow[val , { tag $_1$, tag $_2$, …}]　给 val 加上多个标记
Reap[expr , { patt $_1$, patt $_2$, …}]　允许多个匹配
Reap[expr , patt , f]　返回 {expr , {f[tag$_1$, {e_{11} , e_{12} , …}] , …} }
FractionBox[x, y]　表示分式 $\frac{x}{y}$
SqrtBox[*x*]　表示 $\sqrt{x}$
RadicalBox[*x*, *n*]　表示 $\sqrt[n]{x}$
SuperscriptBox[*x*, y]　表示 x^y
SubscriptBox[*x*, *y*]　表示 x_y
SubsuperscriptBox[*x*, *y*, *z*]　表示 x_y^z
OverscriptBox[*x*, *y*]　表示 $\overset{y}{x}$

UnderscriptBox[*x*, *y*]　表示 $\underset{y}{x}$

UnderoverscriptBox[*x*, *y*, *z*]　表示 $\overset{z}{\underset{y}{x}}$

DisplayForm[*boxes*]　用于显示 Box 类的表达式
RowBox[{box$_1$, box$_2$, …}]　在一行中顺序排列多个表达式
GridBox[{{box$_{11}$, box$_{12}$, …} , {box$_{21}$, box$_{22}$, …} , …}]　在多行中顺序排列多个表达式
HoldForm[expr]　保持表达式 expr 的未求值形式
Encode["source" , "dest"]　将名为 source 的程序文件加密生成一个名为 dest 的程序文件
Encode["source" , "dest" , MachineID → "ID"]　生成带有计算机识别码 ID 的加密文件
Encode["source", "dest" , “key”]　生成带有密钥 key 的加密文件

字符串函数

ToString[expr]　将表达式转换成字符串
ToExpression["string"]　将字符串转换成表达式
StringLength["string"]　求字符串的长度
"s$_1$"<>"s$_2$"<>…　或 StringJoin["s$_1$"<>"s$_2$"<>…]　连接字符串
StringReverse["string"]　串的反转
StringTake["string", {m, n}]　提取第 m 到第 n 个字符得到一个子串
StringDrop["string", {m , n}]　去掉第 m 到第 n 个字符得到一个子串
StringInsert["string", "snew" , { n$_1$, n$_2$, …}]　在 string 的第 n$_1$ ，n$_2$ ，… 个位置上插入snew
StringReplace["string", {"s$_1$"→"p$_1$" , "s$_2$"→"p$_2$" , …}]　用 p$_i$ 替换 s$_i$
StringReplacePart["string", "snew" , {m , n}]　将 string 的第 m 到第 n 个字符换成 snew

其他

Rationalize[x , dx]　给出 x 的误差小于 dx 的有理数近似值
Precision[x]　给出数 x 的有效数字位数（相对精度）
Accuracy[x]　给出数 x 的在小数点右边的数字位数（绝对精度）
StandardForm[expr]　按标准格式输出表达式 expr
TraditionalForm[expr]　按传统格式输出表达式 expr
InputForm[expr]　按输入格式输出表达式 expr
OutputForm[expr]　按输出格式输出表达式 expr
CForm[expr]　将表达式 expr 翻译成 C 语言的表达式
FortranForm[expr]　将表达式 expr 翻译成 Fortran 语言的表达式
Date[ ]　显示当前的日期和时间
Date[z]　给出z时区的当前日期和时间
TimeZone[ ]　给出计算机所采用的时区
AbsoluteTime[ ]　给出从1900年1月1日起以秒计算的绝对时间
Todate[time]　将 绝对时间 time 转换成日期与时间
Fromdate[date]　将 日期与时间 date 转换成绝对时间
Timing[expr]　计算表达式 expr 并返回计算所花费的时间
Pause[n]　暂停至少 n 秒
TimeConstrained[expr , t , failexpr]　计算超时后强制中止计算并返回 failexpr

Compile[{x_1 , x_2 , …} , expr]　建立一个经过编译的函数用于计算表达式 expr
Compile[{{x_1 , t_1 , n_1} , …} , expr]　编译时规定数组 x_1 的类型为 t_1 层数为 n_1
Compile[vars , expr , {{p_1 , pt_1} , …}]　编译时规定表达式 expr 的子表达式 p_1 的类型为pt_1
Directory[ ]　给出当前工作目录
SetDirectory["dir"]　设置 dir 为当前工作目录
ResetDirectory[ ]　返回以前的工作目录
FileNames[ ]　列出当前工作目录中的所有文件和子目录
FileNames["form"]　列出当前工作目录中的所有与 form 匹配的文件或子目录

常用的系统变量

$Assumptions　用于设定 Assumptions 选项的默认值
$Context　当前的上下文
$ContextPath　当前能够被 Mathematica 自动查找使用的所有上下文的表
$Packages　已经被调入的所有程序包的上下文的表
$RecursionLimit　给出能够使用的递归层数
$IterationLimit　给出计算链的最大长度
$Post　每次将表达式求值后都将该函数作用到所得结果上
$PrePrint　每次将表达式的求值结果赋给 Out[n] 之后再将该函数作用到所得结果上
$TimeUnit　在一个计算机系统上以秒为单位的最小时间间隔。
$MaxNumber　在一个计算机系统上能表示的最大任意精度数的量级
$MinNumber　在一个计算机系统上能表示的最小任意精度正数的量级
$MaxMachineNumber　在一个计算机系统上能表示的最大机器精度数
$MinMachineNumber　在一个计算机系统上能表示的最小机器精度正数
$MachinePrecision　机器精度数的精度
$MaxExtraPrecision　附加的精度的最大值
$MachineID　正在运行 Mathematica 的计算机的唯一识别代码串

说明　有些函数是外部函数（加有星号），需要调入相应的程序包才能使用。还有少量在前面各章中没有被介绍的函数，只要看索引中的解释就会使用它们了。