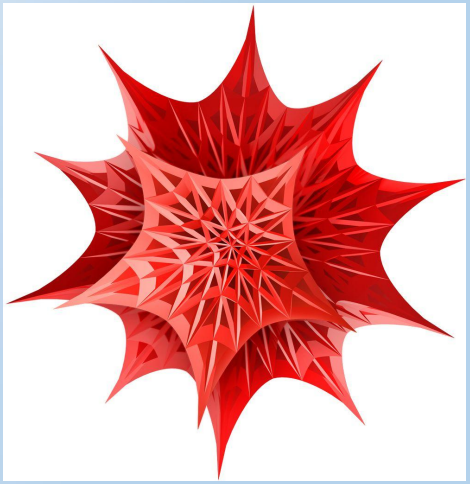
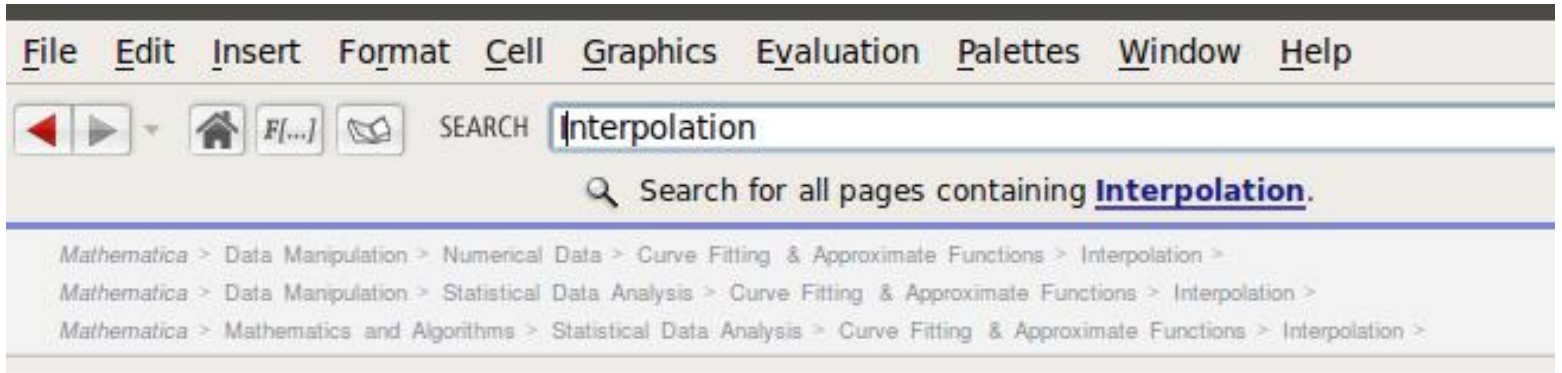


Mathematica（1）

龚明（LQCC，物质楼C812）

计算物理/研，中国科大



Interpolation

`Interpolation[{`f_1`,` f_2`, ...}]`
constructs an interpolation of the function values f_i, assumed to correspond to x values $1, 2, \ldots$.

`Interpolation[{{`x_1`,` f_1`}, {`x_2`,` f_2`}, ...}]`
constructs an interpolation of the function values f_i corresponding to x values x_i.

`Interpolation[{{{`x_1`,` y_1`, ...},` f_1`}, {{`x_2`,` y_2`, ...},` f_2`}, ...}]`
constructs an interpolation of multidimensional data.

`Interpolation[{{{`x_1`, ...},` f_1`,` df_1`, ...}, ...}]`
constructs an interpolation that reproduces derivatives as well as function values.

`Interpolation[`*data*`,` x`]`
find an interpolation of *data* at the point x.

利用好help

找各种例子

多实践

In[1]:=
```
InterpolatingPolynomial[{{-1, 4}, {0, 2}, {1, 6}}, x]
```

Out[1]=
$$4 + (1 + x)\ (-2 + 3\ x)$$

```
Clear[x, x1, h]
Fx = InterpolatingPolynomial[{{x1, y1}, {x1 + h, y2}, {x1 + 2 h, y3}}, x]
Integrate[Fx, {x, x1, x1 + 2 h}]
```

$$y1 + (x - x1)\left(\frac{-y1 + y2}{h} + \frac{(-h + x - x1)\left(-\frac{-y1+y2}{h} + \frac{-y2+y3}{h}\right)}{2\ h}\right)$$

$$\frac{h\ y1}{3} + \frac{4\ h\ y2}{3} + \frac{h\ y3}{3}$$

Simpson's Rule

Simpson's Rule was actually invented by Sir Isaac Newton but Thomas Simpson published it again in 1743.

It is more accurate than the Trapezium Rule because it fits a number of parabolas to the curve rather than straight lines.

To use Simpson's Rule, you must have an even number of intervals (or an odd number of ordinates).

Simpson's Rule :

$$\int_a^b y\,dx = \frac{h}{3}((y_0 + y_n) + 4(y_1 + y_3 + \ldots y_{n-1}) + 2(y_2 + y_4 + \ldots y_{n-2}))$$

```
Clear[x, x1, h]
Fx = InterpolatingPolynomial[
   {{x1, y1}, {x1 + h, y2}, {x1 + 2 h, y3}, {x1 + 3 h, y4}}, x]
Integrate[Fx, {x, x1, x1 + 3 h}]
```

$$y1 + (x - x1)\left(\frac{-y1 + y2}{h} + (-h + x - x1)\left(\frac{-\frac{-y1+y2}{h} + \frac{-y2+y3}{h}}{2\,h} + \frac{1}{3\,h}(-2\,h + x - x1)\left(-\frac{-\frac{-y1+y2}{h} + \frac{-y2+y3}{h}}{2\,h} + \frac{-\frac{-y2+y3}{h} + \frac{-y3+y4}{h}}{2\,h}\right)\right)\right)$$

$$\frac{3\,h\,y1}{8} + \frac{9\,h\,y2}{8} + \frac{9\,h\,y3}{8} + \frac{3\,h\,y4}{8}$$

Matrix Operations

The Wolfram Language's matrix operations handle both numeric and symbolic matrices, automatically accessing large numbers of highly efficient algorithms. The Wolfram Language uses state-of-the-art algorithms to work with both dense and sparse matrices, and incorporates a number of powerful original algorithms, especially for high-precision and symbolic matrices.

+, *****, **^**, ... — all automatically work element-wise

Dot (.) — products of matrices, automatically handling row and column vectors

Inverse — matrix inverse (use **LinearSolve** for linear systems)

Transpose — transpose (m^{T}, entered with esc tr esc)

ConjugateTranspose — conjugate transpose ($m^{\dagger}$, entered with esc ct esc)

Tr — trace

Det — determinant

Permanent — permanent

KroneckerProduct — matrix direct product (outer product)

MatrixPower — powers of numeric or symbolic matrices

MatrixExp— matrix exponential

MatrixLog — matrix logarithm

MatrixFunction — general matrix function

Eigenvalues, **Eigenvectors** — exact or approximate eigenvalues and eigenvectors

Eigensystem — eigenvalues and eigenvectors together

CharacteristicPolynomial — symbolic characteristic polynomial

Constructing Matrices

Mathematica offers several ways for constructing matrices:

Table[f,{i,m},{j,n}]
Array[f,{m,n}]
ConstantArray[a,{m,n}]
DiagonalMatrix[list]
IdentityMatrix[n]
Normal[SparseArray[{{i1,j1}->v1,{i2,j2}->v2,…},{m,n}]]
RandomReal[1, {m,n}]

IdentityMatrix[3]//MatrixForm

Out[3]//MatrixForm=

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

IdentityMatrix[3]//TraditionalForm

Out[4]//MatrixForm=

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

```
A = {{1, 2, 3}, {-1, 3, 0}}
A // MatrixForm
B = {{1, 2.0, 3}, {-1, 3, 0}, {1, 0, -3}}
B // MatrixForm
Eigenvalues[B]
```

```
{{1, 2, 3}, {-1, 3, 0}}
```

atrixForm=

$$\begin{pmatrix} 1 & 2 & 3 \\ -1 & 3 & 0 \end{pmatrix}$$

```
{{1, 2., 3}, {-1, 3, 0}, {1, 0, -3}}
```

atrixForm=

$$\begin{pmatrix} 1 & 2. & 3 \\ -1 & 3 & 0 \\ 1 & 0 & -3 \end{pmatrix}$$

```
{-3.61061, 2.3053 + 1.15441 i, 2.3053 - 1.15441 i}
```

```
B // MatrixForm
Eigenvalues[B]
Eigenvectors[B]
Eigensystem[B]
```

1. 掌握Eigenvalues, Eigenvectors, Eigensystem
2. 可以计算任意方阵

latrixForm=

$$\begin{pmatrix} 1 & 2. & 3 \\ -1 & 3 & 0 \\ 1 & 0 & -3 \end{pmatrix}$$

```
{-3.61061, 2.3053 + 1.15441 i, 2.3053 - 1.15441 i}
```

```
{{0.519524, 0.0785895, -0.850834},
 {0.79435 + 0. i, 0.303996 + 0.505163 i, 0.142959 - 0.0311072 i},
 {0.79435 + 0. i, 0.303996 - 0.505163 i, 0.142959 + 0.0311072 i}}
```

```
{{-3.61061, 2.3053 + 1.15441 i, 2.3053 - 1.15441 i},
 {{0.519524, 0.0785895, -0.850834},
  {0.79435 + 0. i, 0.303996 + 0.505163 i, 0.142959 - 0.0311072 i},
  {0.79435 + 0. i, 0.303996 - 0.505163 i, 0.142959 + 0.0311072 i}}}
```

```
A // MatrixForm
Eigenvalues[A]
Eigenvectors[A]
Eigensystem[A]
```

对于非方阵，会出错。

latrixForm=

$$\begin{pmatrix} 1 & 2 & 3 \\ -1 & 3 & 0 \end{pmatrix}$$

Eigenvalues::matsq :
Argument {{1, 2, 3}, {−1, 3, 0}} at position 1 is not a non−empty square matrix. ≫

```
Eigenvalues[{{1, 2, 3}, {-1, 3, 0}}]
```

Eigenvectors::matsq :
Argument {{1, 2, 3}, {−1, 3, 0}} at position 1 is not a non−empty square matrix. ≫

```
Eigenvectors[{{1, 2, 3}, {-1, 3, 0}}]
```

Eigensystem::matsq :
Argument {{1, 2, 3}, {−1, 3, 0}} at position 1 is not a non−empty square matrix. ≫

```
Eigensystem[{{1, 2, 3}, {-1, 3, 0}}]
```

掌握矩阵的QR分解。

```
In[138]:= A // MatrixForm
          RES = QRDecomposition[A]
          Q = RES[[1]]
          R = RES[[2]]
          Q // MatrixForm
          R // MatrixForm
```

Out[138]//MatrixForm=

$$\begin{pmatrix} 1 & 2 & 3 \\ -1 & 3 & 0 \end{pmatrix}$$

Out[139]= $\left\{\left\{\left\{\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\right\}, \left\{\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right\}\right\}, \left\{\left\{\sqrt{2}, -\frac{3}{\sqrt{2}} + \sqrt{2}, \frac{3}{\sqrt{2}}\right\}, \left\{0, \frac{5}{\sqrt{2}}, \frac{3}{\sqrt{2}}\right\}\right\}\right\}$

Out[140]= $\left\{\left\{\frac{1}{\sqrt{2}}, -\frac{1}{\sqrt{2}}\right\}, \left\{\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right\}\right\}$

Out[141]= $\left\{\left\{\sqrt{2}, -\frac{3}{\sqrt{2}} + \sqrt{2}, \frac{3}{\sqrt{2}}\right\}, \left\{0, \frac{5}{\sqrt{2}}, \frac{3}{\sqrt{2}}\right\}\right\}$

Out[142]//MatrixForm=

$$\begin{pmatrix} \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix}$$

Out[143]//MatrixForm=

$$\begin{pmatrix} \sqrt{2} & -\frac{3}{\sqrt{2}} + \sqrt{2} & \frac{3}{\sqrt{2}} \\ 0 & \frac{5}{\sqrt{2}} & \frac{3}{\sqrt{2}} \end{pmatrix}$$

```
In[189]:= F = {{5, 2, 0}, {2, 2, 3}, {0, 3, 9}}
          F // MatrixForm
          L = CholeskyDecomposition[F]
          L // MatrixForm
          ConjugateTranspose[L].L - F

Out[189]= {{5, 2, 0}, {2, 2, 3}, {0, 3, 9}}
```

Out[190]//MatrixForm=

$$\begin{pmatrix} 5 & 2 & 0 \\ 2 & 2 & 3 \\ 0 & 3 & 9 \end{pmatrix}$$

Out[191]= $\left\{\left\{\sqrt{5}, \frac{2}{\sqrt{5}}, 0\right\}, \left\{0, \sqrt{\frac{6}{5}}, \sqrt{\frac{15}{2}}\right\}, \left\{0, 0, \sqrt{\frac{3}{2}}\right\}\right\}$

Out[192]//MatrixForm=

$$\begin{pmatrix} \sqrt{5} & \frac{2}{\sqrt{5}} & 0 \\ 0 & \sqrt{\frac{6}{5}} & \sqrt{\frac{15}{2}} \\ 0 & 0 & \sqrt{\frac{3}{2}} \end{pmatrix}$$

```
Out[193]= {{0, 0, 0}, {0, 0, 0}, {0, 0, 0}}
```

Cholesky【乔列斯基】分解是把一个对称正定的矩阵表示成一个下三角矩阵L和其转置的乘积的分解。它要求矩阵的所有特征值必须大于零，故分解的下三角的对角元也是大于零的。Cholesky分解法又称平方根法，是当A为实对称正定矩阵时，LU三角分解法的变形。

$$A = LL^T$$

In[240]:=
```
T = {{1, 2}, {3, 4}};
{lu, p, c} = LUDecomposition[T]
lu // MatrixForm
l = lu * SparseArray[{i_, j_} /; j < i → 1, {2, 2}] + IdentityMatrix[2]
u = lu * SparseArray[{i_, j_} /; j ≥ i → 1, {2, 2}]
(l.u - T) // MatrixForm
```

Out[241]= {{{1, 2}, {3, -2}}, {1, 2}, 1}

Out[242]//MatrixForm=

$$\begin{pmatrix} 1 & 2 \\ 3 & -2 \end{pmatrix}$$

Out[243]= {{1, 0}, {3, 1}}

Out[244]= {{1, 2}, {0, -2}}

Out[245]//MatrixForm=

$$\begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$$

LU分解稍微复杂一点点
命令在help中可以找到

SingularValueDecomposition

In[277]:=
```
A = {{1, 2, 3}, {-1, 3, 0.0}};
A // MatrixForm
{u, w, v} = SingularValueDecomposition[A]
u.w.Transpose[v] // MatrixForm
```

ut[278]//MatrixForm=

$$\begin{pmatrix} 1 & 2 & 3 \\ -1 & 3 & 0. \end{pmatrix}$$

Out[279]=
```
{{{-0.828067, 0.560629}, {-0.560629, -0.828067}},
 {{4.16955, 0., 0.}, {0., 2.57193, 0.}}, {{-0.0641408, 0.539943, -0.839254},
  {-0.800571, -0.52993, -0.279751}, {-0.595796, 0.653939, 0.466252}}}
```

ut[280]//MatrixForm=

$$\begin{pmatrix} 1. & 2. & 3. \\ -1. & 3. & 0. \end{pmatrix}$$