

In[]:= **points = {{-1, 3}, {0, 0.5}, {0.5, 0}, {1, 1}};**

langrangePoly = InterpolatingPolynomial[points, x]
插值多项式

Out[]:=

$3 + (1 + x) (-1 + (-1 + x) (1.5 + 1. x))$

In[]:= **f25 = langrangePoly /. x -> 0.25**

Out[]:=

0.109375

In[]:= **f75 = langrangePoly /. x -> 0 / 75**

Out[]:=

0.5

points = {{-1, 1.5}, {0, 0}, {0.5, 0}, {1, 0.5}};
langrangePoly = InterpolatingPolynomial[points, x];
插值多项式
f25 = langrangePoly /. x -> -0.25;
f25' = langrangePoly /. x -> 0.25;
{langrangePoly, f25, f25'}

Out[]:=

$\{1.5 + (-0.5 + 1. (-1 + x)) (1 + x), 0.1875, -0.0625\}$

In[]:= **? Fit**

Out[]:=

Symbol

Fit[data, { f_1, \dots, f_n }, { x, y, \dots }] 求变量{ x, y, \dots } 的函数 f_1, \dots, f_n 的 data 列表拟合 $a_1 f_1 + \dots + a_n f_n$.

Fit[{ m, v }] 求最小化设计矩阵 m 的 $\|m.a - v\|$ 的拟合向量 a .

Fit[..., "prop"] 指定应返回哪些拟合属性 prop.

Documentation [Web »](#)

Options {FitRegularization -> None, NormFunction -> Automatic, WorkingPrecision -> Automatic}

Attributes {Protected}

Full Name System`Fit



```

In[*]:= data =
  {{-1.00, 0.22}, {-0.50, 0.80}, {0, 2.0}, {0.25, 2.5}, {0.75, 3.8}, {1.00, 4.2}};

linearFit = Fit[data, {1, x}, x]
          拟合

quadraticFit = Fit[data, {1, x, x^2}, x]
              拟合

ListPlot[data, PlotStyle → Red, PlotLabel → "Data Points",
          绘制点集      绘图样式      红色  绘图标签
          AxesLabel → {"Xi", "yi"}, PlotRange → All]
          坐标轴标签      绘制范围      全部

Show[Plot[linearFit, {x, -1.1, 1.1}, PlotStyle → Blue, PlotLabel → "Linear Fit"],
     显示  绘图      绘图样式      蓝色  绘图标签      拟合
     Plot[quadraticFit, {x, -1.1, 1.1}, PlotStyle → Green, PlotLabel → "Quadratic Fit"],
     绘图      绘图样式      绿色  绘图标签      拟合
     ListPlot[data, PlotStyle → Red, PlotMarkers → Automatic],
     绘制点集      绘图样式      红色  绘制点的标记      自动
     PlotRange → All, AxesLabel → {"Xi", "yi"}]
     绘制范围      全部  坐标轴标签

```

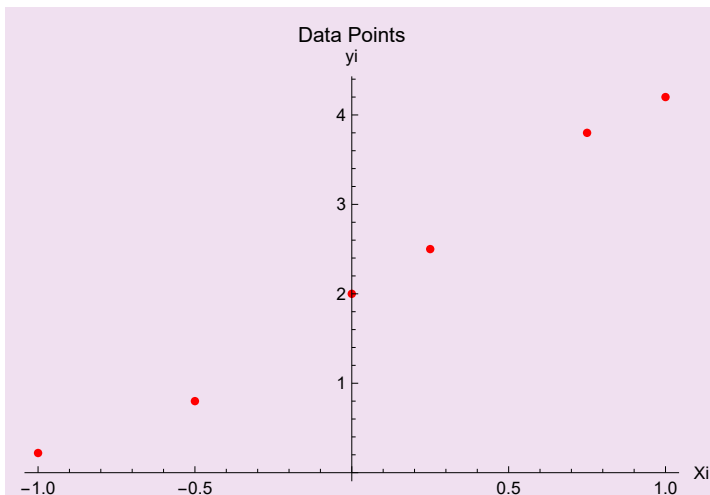
Out[*]=

$$2.07897 + 2.09235 x$$

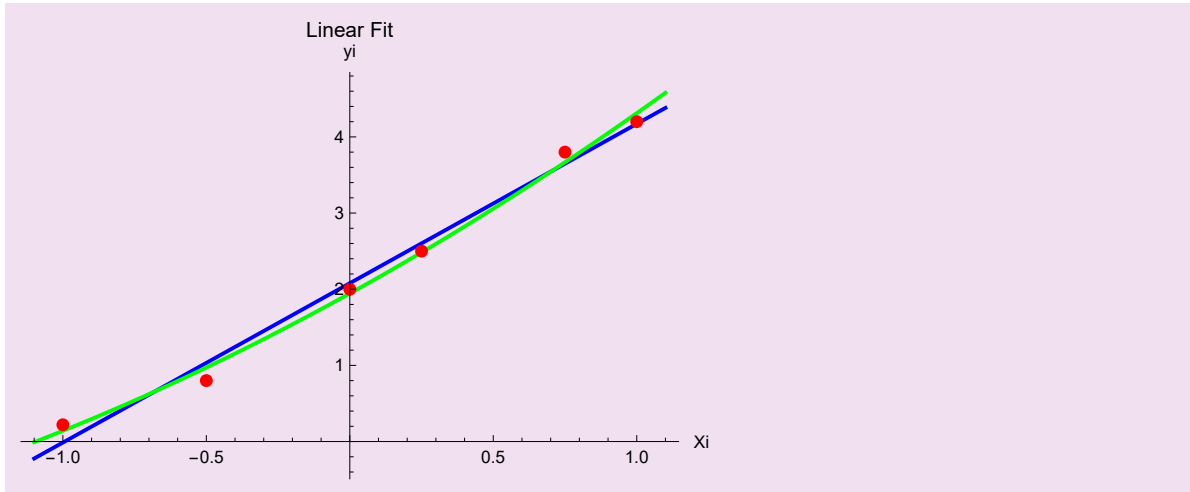
Out[*]=

$$1.94449 + 2.0851 x + 0.28191 x^2$$

Out[*]=



Out[*]=



```
In[*]:= points = {{19, 19.00}, {23, 28.50}, {30, 47.00}, {35, 68.20}, {40, 90.00}};
fit = Fit[points, {1, x^2}, x]
```

拟合

Out[*]=

$$-2.37227 + 0.0573264 x^2$$

```
In[*]:= points = {{0, 2.00}, {1, 2.50}, {2, 4.00}, {3, 6.00}, {4, 8.00}};

fitParams = FindFit[points, a * Exp[b * x], {a, b}, x]
```

求拟合 指数形式

```
fit = a * Exp[b * x] /. fitParams
```

指数形式

Out[*]=

$$\{a \rightarrow 1.94454, b \rightarrow 0.358018\}$$

Out[*]=

$$1.94454 e^{0.358018 x}$$

```
In[*]:= c = {-3, -2};
m = {{1, -2}, {-3, -2}, {-1, 1}};
b = {-4, -14, -3};
l = {0, 0};
solution = LinearProgramming[c, m, b]
```

线性规划

Out[*]=

$$\{4, 1\}$$

```
In[ ]:= c = {2, 3, 4};
m = {{-1, -2, 1}, {-1, -1, 1}, {0, -1, -2}};
b = {10, 60, 12};
solution = LinearProgramming[c, -m, b, {0, 0, 1}]
          线性规划
```

Out[]:=

```
{51, 10, 1}
```

```
(*We make some modifications:
  z' = z - 1, then z' > 0
  min m = 2x+3y+4z'+4
  x = x1-x2, y = y1-y2, z' = z1-z2*)
c = {2, -2, 3, -3, 4, -4, 0, 0};
m = {{1, -1, 2, -2, -1, 1, -1, 0},
      {1, -1, 1, -1, -1, 1, 0, 0}, {0, 0, 1, -1, 2, -2, 0, -1}};
b = {11, 61, 10};
solution = LinearProgramming[c, m, b]
          线性规划
```

LinearProgramming: 该问题是无界的.

Out[]:=

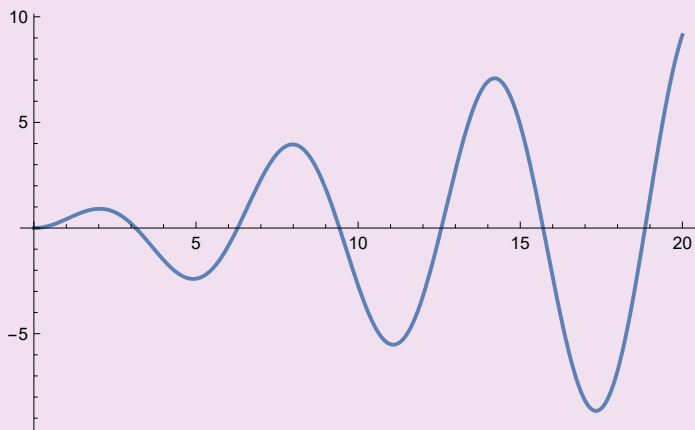
```
{Indeterminate, Indeterminate, Indeterminate, Indeterminate,
  Indeterminate, Indeterminate, Indeterminate, Indeterminate}
```

```
In[ ]:= DSolve[{y''[x] + y[x] == Cos[x], y[0] == 0, y'[0] == 0}, y[x], x]
          求解微分方程          余弦
sol = y[x] /. First[%];
          第一个
Plot[sol, {x, 0, 20}]
          绘图
```

Out[]:=

$$\left\{ \left\{ y[x] \rightarrow \frac{1}{4} \left(-2 \cos[x] + 2 \cos[x]^3 + 2x \sin[x] + \sin[x] \sin[2x] \right) \right\} \right\}$$

Out[]:=



```

In[ ]:= Clear["Global`*"]
清除

solution = NDSolve[
数值求解微分方程组
{

$$u'[t] == 0.09 * u[t] * \left(1 - \frac{u[t]}{20}\right) - 0.45 * u[t] * v[t],$$


$$v'[t] == 0.06 * v[t] * \left(1 - \frac{v[t]}{15}\right) - 0.001 * u[t] * v[t],$$



u[0] == 1.6, v[0] == 1.2}, {u, v}, {t, 20}]

Plot[Evaluate[{u[t], v[t]} /. solution], {t, 0, 20}, PlotLabels -> {"u(t)", "v(t)"},
绘图 计算 数据绘制标签
PlotLegends -> Placed[{"u(t)", "v(t)"}, Above], PlotStyle -> {Blue, Red}]
绘图的图例 放置 上 绘图样式 蓝色 红色

```

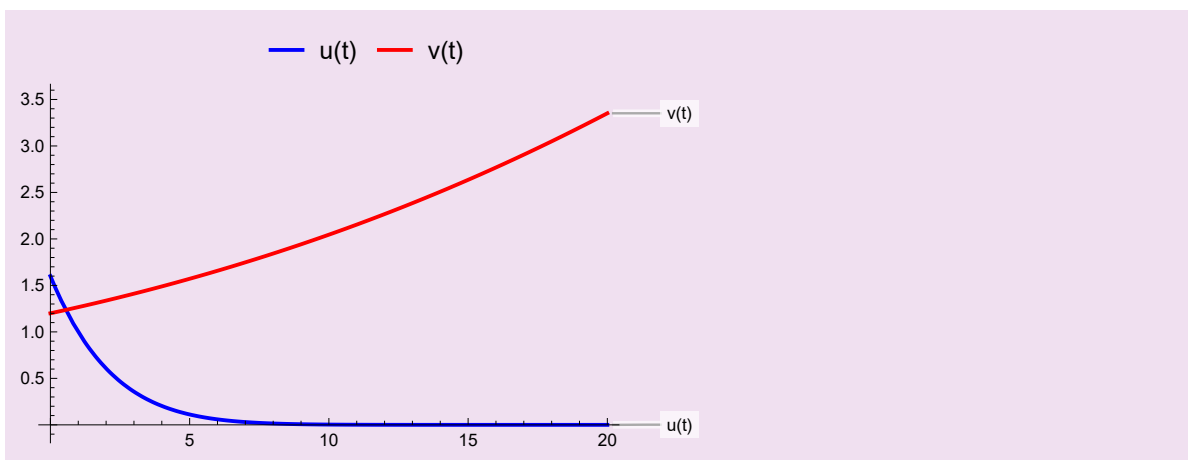
Out[]=

```

{
{
u -> InterpolatingFunction[
+  Domain: {{0., 20.}}
Output: scalar
},
v -> InterpolatingFunction[
+  Domain: {{0., 20.}}
Output: scalar
]}

```

Out[]=



```

In[ ]:= heatEquation = D[u[x, t], t] == D[u[x, t], {x, 2}];
          偏导          偏导

initialCondition = u[x, 0] == 0;

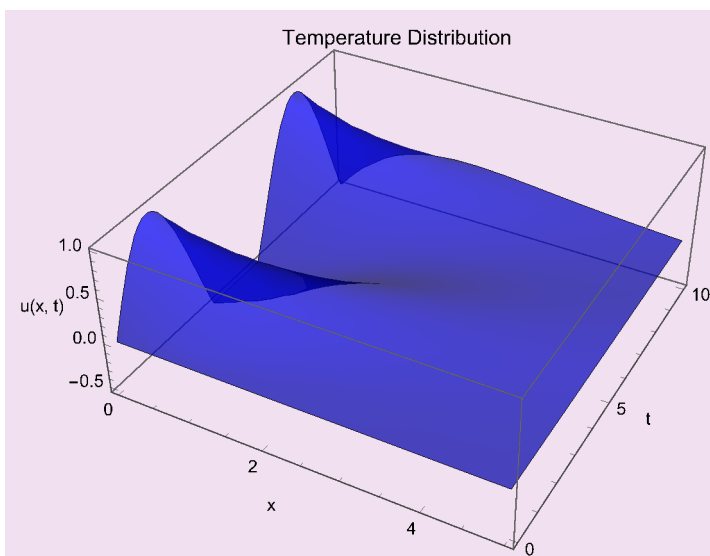
boundaryConditions = {u[0, t] == Sin[t], u[5, t] == 0};
                   正弦

solution = NDSolve[
          数值求解微分方程组
    {heatEquation, initialCondition, boundaryConditions}, u, {x, 0, 5}, {t, 0, 10}];

Plot3D[Evaluate[u[x, t] /. solution], {x, 0, 5}, {t, 0, 10},
  绘制... 计算
  PlotLabel -> "Temperature Distribution", AxesLabel -> {"x", "t", "u(x, t)"},
  绘图标签          坐标轴标签
  Mesh -> None, PlotStyle -> Directive[Opacity[0.7], Blue]]
  网格 无 绘图样式 指令 不透明度 蓝色

```

Out[]=



```

In[*]:= ClearAll[u, x, t]
清除全部

(*定义波动方程*) len = 10; (*定义计算区域的长度*)
pde = D[u[x, t], {t, 2}] == D[u[x, t], {x, 2}];
偏导 偏导

(*设置初始条件*)
ic = {u[x, 0] == Exp[-x^2], Derivative[0, 1][u][x, 0] == 0}; (*初始位移和速度*)
指数形式 导数

(*设置边界条件: 周期性边界条件*)
bc = {u[-len, t] == u[len, t],
      Derivative[1, 0][u][-len, t] == Derivative[1, 0][u][len, t]};
导数 导数

(*使用 NDSolve 求解波动方程*)
数值求解微分方程组
solution = NDSolve[{pde, ic[[1]], ic[[2]], bc[[1]], bc[[2]]}, u, {x, -len, len}, {t, 0, 40}]
数值求解微分方程组

```

Out[*]=

```

{{u -> InterpolatingFunction[
  {
    +  Domain: {{-10., 10.}, {0., 40.}}
    Output: scalar
  ]
}}

```

```

In[ ]:= (*可视化*) uSolution[x_, t_] := u[x, t] /. solution[[1]];
Plot3D[uSolution[x, t], {x, -len, len}, {t, 0, 40}, PlotRange -> All,
|绘制三维图形|绘制范围|全部
AxesLabel -> {"x", "t", "u(x, t)"}, MeshFunctions -> {#3 &}, Mesh -> 10]
|坐标轴标签|网格函数|网格

```

Out[]=

