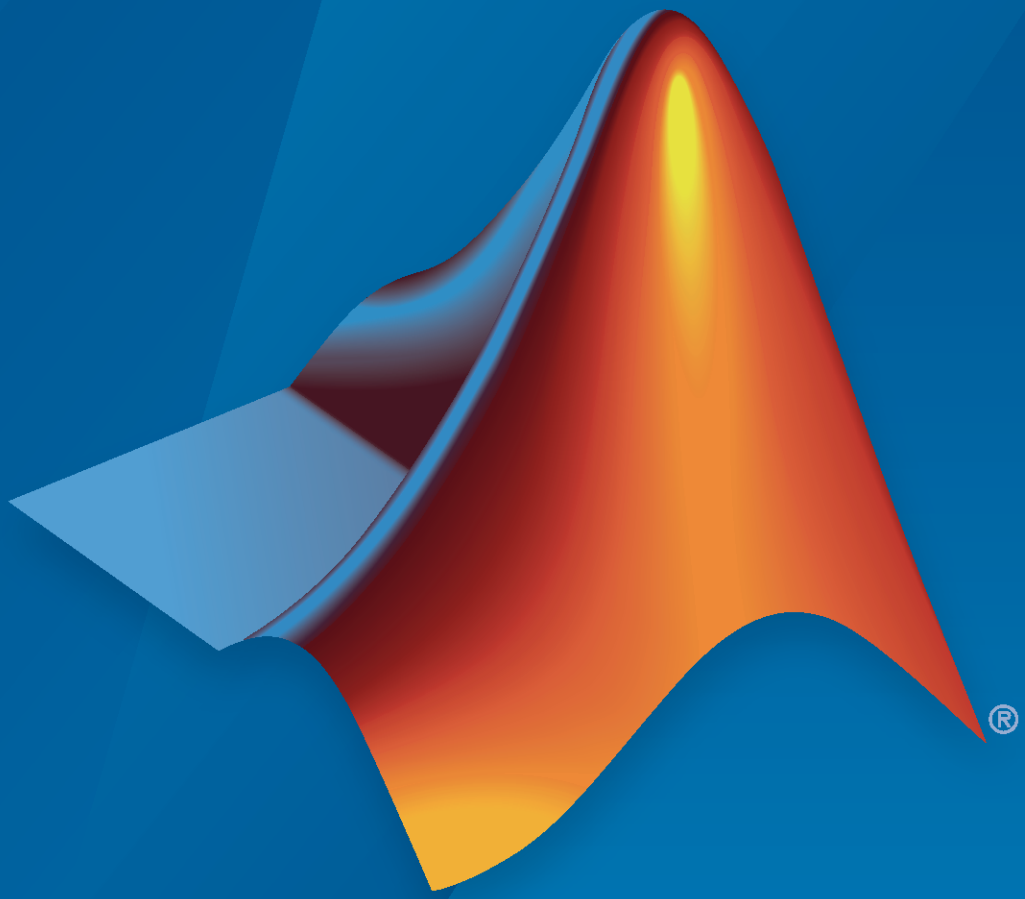
MATLAB®
三维可视化



MATLAB®

R2022b

MathWorks®

如何联系 MathWorks

最新动态: **www.mathworks.com**

销售和服务: **www.mathworks.com/sales_and_services**

用户社区: **www.mathworks.com/matlabcentral**

技术支持: **www.mathworks.com/support/contact_us**

电话: 010-59827000

迈斯沃克软件 (北京) 有限公司
北京市朝阳区望京东园四区 6 号楼
北望金辉大厦 16 层 1604

MATLAB® 三维可视化

修订历史记录

2006 年 3 月	仅限在线版本	MATLAB® 7.2（版本 2006a）中的新内容
2006 年 9 月	仅限在线版本	MATLAB® 7.3（版本 2006b）中的修订内容
2007 年 3 月	仅限在线版本	MATLAB® 7.4（版本 2007a）中的修订内容
2007 年 9 月	仅限在线版本	MATLAB® 7.5（版本 2007b）中的修订内容
2008 年 3 月	仅限在线版本	MATLAB® 7.6（版本 2008a）中的修订内容 本出版物以前是“使用 MATLAB® 图形指南”的一部分。
2008 年 10 月	仅限在线版本	MATLAB® 7.7（版本 2008b）中的修订内容
2009 年 3 月	仅限在线版本	MATLAB® 7.8（版本 2009a）中的修订内容
2009 年 9 月	仅限在线版本	MATLAB® 7.9（版本 2009b）中的修订内容
2010 年 3 月	仅限在线版本	MATLAB® 7.10（版本 2010a）中的修订内容
2010 年 9 月	仅限在线版本	MATLAB 7.11（版本 2010b）中的修订内容
2011 年 4 月	仅限在线版本	MATLAB 7.12（版本 2011a）中的修订内容
2011 年 9 月	仅限在线版本	MATLAB 7.13（版本 2011b）中的修订内容
2012 年 3 月	仅限在线版本	MATLAB 7.14（版本 2012a）中的修订内容
2012 年 9 月	仅限在线版本	MATLAB 8.0（版本 2012b）中的修订内容
2013 年 3 月	仅限在线版本	MATLAB 8.1（版本 2013a）中的修订内容
2013 年 9 月	仅限在线版本	MATLAB 8.2（版本 2013b）中的修订内容
2014 年 3 月	仅限在线版本	MATLAB 8.3（版本 2014a）中的修订内容
2014 年 10 月	仅限在线版本	MATLAB 8.4（版本 2014b）中的修订内容
2015 年 3 月	仅限在线版本	MATLAB 8.5（版本 2015a）中的修订内容
2015 年 9 月	仅限在线版本	MATLAB 8.6（版本 2015b）中的修订内容
2016 年 3 月	仅限在线版本	MATLAB 9.0（版本 2016a）中的修订内容
2016 年 9 月	仅限在线版本	MATLAB 9.1（版本 2016b）中的修订内容
2017 年 3 月	仅限在线版本	MATLAB 9.2（版本 2017a）中的修订内容
2017 年 9 月	仅限在线版本	MATLAB 9.3（版本 2017b）中的修订内容
2018 年 3 月	仅限在线版本	MATLAB 9.4（版本 2018a）中的修订内容
2018 年 9 月	仅限在线版本	MATLAB 9.5（版本 2018b）中的修订内容
2019 年 3 月	仅限在线版本	MATLAB 9.6（版本 2019a）中的修订内容
2019 年 9 月	仅限在线版本	MATLAB 9.7（版本 2019b）中的修订内容
2020 年 3 月	仅限在线版本	MATLAB 9.8（版本 2020a）中的修订内容
2020 年 9 月	仅限在线版本	MATLAB 9.9（版本 2020b）中的修订内容
2021 年 3 月	仅限在线版本	MATLAB 9.10（版本 2021a）中的修订内容
2021 年 9 月	仅限在线版本	MATLAB 9.11（版本 2021b）中的修订内容
2022 年 3 月	仅限在线版本	MATLAB 9.12（版本 2022a）中的修订内容
2022 年 9 月	仅限在线版本	MATLAB 9.13（版本 2022b）中的修订内容

目录

1 曲面图和网格图

创建三维绘图 1-2

更改曲面属性 1-8

创建 MATLAB 徽标 1-17

将数据表示为曲面 1-25
用来绘制数据网格的函数 1-25
用于对数据进行网格化和插值的函数 1-25
网格图和曲面图 1-25
可视化包含两个变量的函数 1-26
非均匀采样数据的曲面图 1-29
重构数据 1-30
参数化曲面 1-32

2 多边形

补片对象简介 2-2
什么是补片对象? 2-2
patch 函数的行为 2-2
创建单个多边形 2-3

多面补片 2-6
示例 - 定义立方体 2-6

3 三维可视化

三维体可视化概述 3-2
三维体数据示例 3-2
选择可视化方法 3-2
创建三维体可视化绘图的步骤 3-3
三维体可视化函数 3-3

可视化标量三维体数据的方法 3-5
什么是标量三维体数据? 3-5
显示 MRI 数据的方法 3-5

利用切片平面探索三维体 **3-11**
切片液体流数据 **3-11**
修改颜色映射 **3-13**

使用等值面连接相等的值 **3-17**
液体流数据中的等值面 **3-17**

使用等值顶为可视化绘图添加环境 **3-19**
什么是等值顶? **3-19**
等值顶的其他应用 **3-20**
定义等值顶 **3-20**
为等值面添加等值顶 **3-20**

可视化向量三维体数据 **3-23**
流线、流粒子、流带、流、流管和圆锥 **3-23**
对向量数据使用标量方法 **3-23**
指定流线图的起点 **3-23**
访问三维体数据的子区域 **3-26**

使用向量数据显示流线图 **3-27**
风的映射数据 **3-27**
1.确定坐标的范围 **3-27**
2.添加切片平面以提供视觉环境 **3-27**
3.在切片平面上添加等高线 **3-27**
4.定义流线的起点 **3-28**
5.定义视图 **3-28**

利用流带显示旋度 **3-29**
流带可以显示哪些信息 **3-29**
1.选择要绘制的数据子集 **3-29**
2.计算旋度角速度和风速 **3-29**
3.创建流带 **3-29**
4.定义视图并添加光照 **3-30**

利用流管显示散度 **3-31**
流管可以显示哪些信息 **3-31**
1.加载数据并计算所需的值 **3-31**
2.绘制切片平面 **3-31**
3.在切片平面上添加等高线 **3-32**
4.创建流管 **3-32**
5.定义视图 **3-32**

创建流粒子动画 **3-34**
随时间变化的抛射体路径 **3-34**
粒子动画可以显示哪些信息 **3-35**
1.指定数据范围的起点 **3-35**
2.创建流线以指示粒子路径 **3-35**
3.定义视图 **3-35**
4.计算流粒子顶点 **3-36**

利用圆锥图显示向量场 **3-38**
圆锥图可以显示哪些信息 **3-38**
1.创建等值面 **3-38**
2.为等值面添加等值顶 **3-38**
3.创建第一组圆锥体 **3-39**

4.创建第二组圆锥体 3-39
5.定义视图 3-39
6.添加光照 3-39

可视化体数据 3-41

可视化四维数据 3-48

显示复杂三维对象 3-54

显示地貌数据 3-62

曲面图和网格图

- “创建三维绘图” (第 1-2 页)
- “更改曲面属性” (第 1-8 页)
- “创建 MATLAB 徽标” (第 1-17 页)
- “将数据表示为曲面” (第 1-25 页)

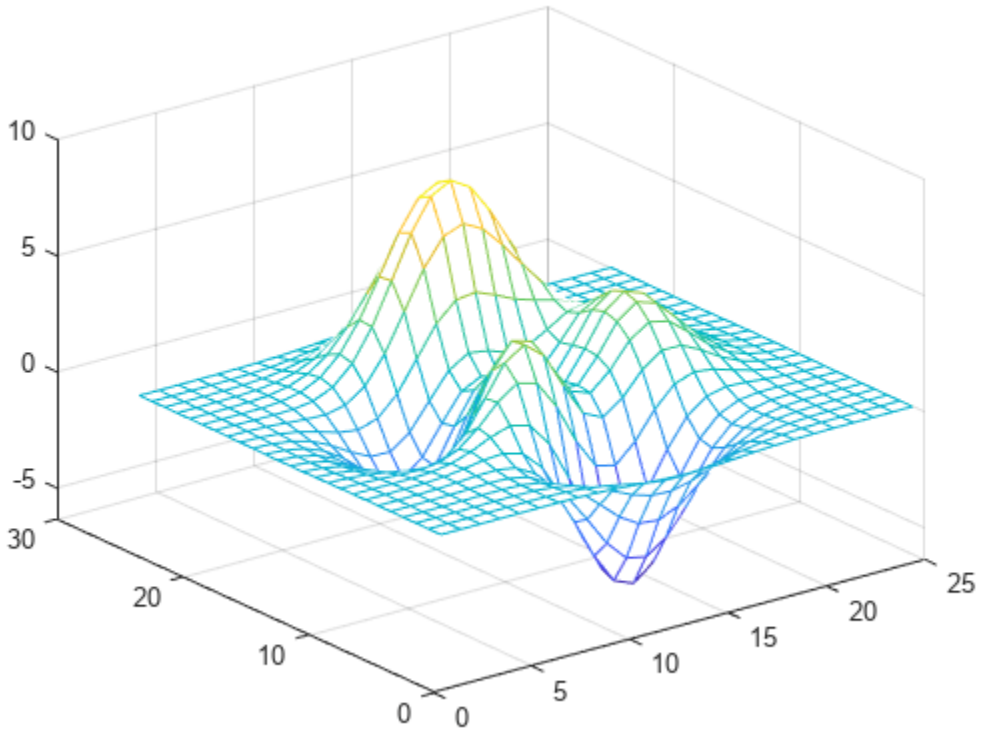
创建三维绘图

以下示例演示如何在 MATLAB® 中创建各种三维绘图。

网格图

mesh 函数可创建线框网格图。默认情况下，网格图颜色与曲面高度成正比。

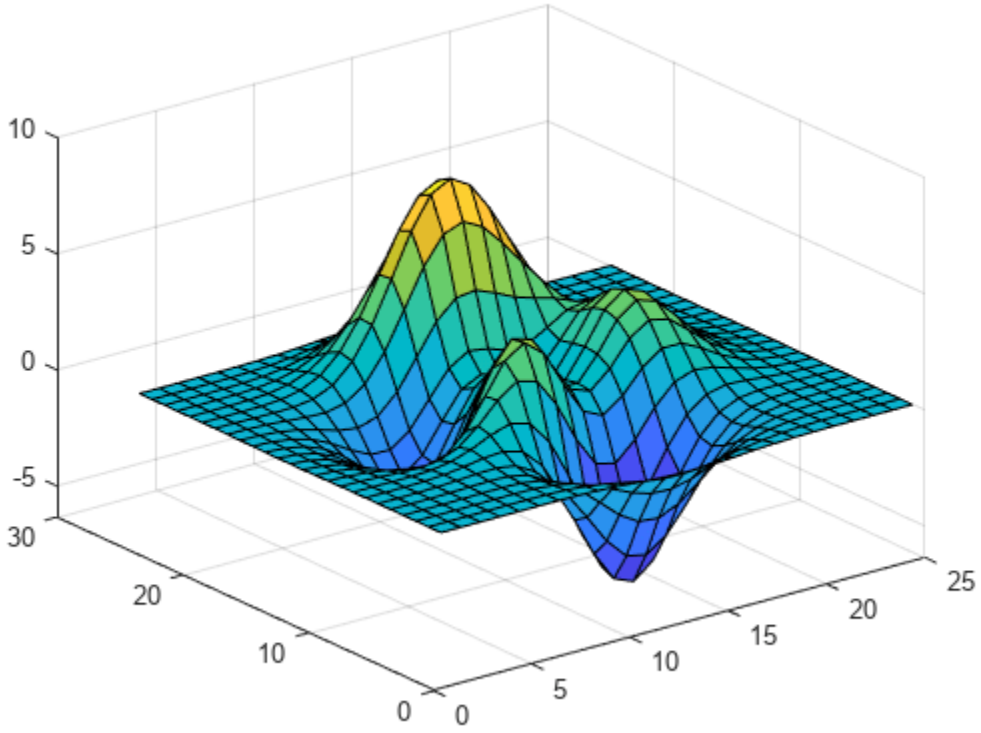
```
z = peaks(25);

figure
mesh(z)
```



曲面图

surf 函数用于创建三维曲面图。

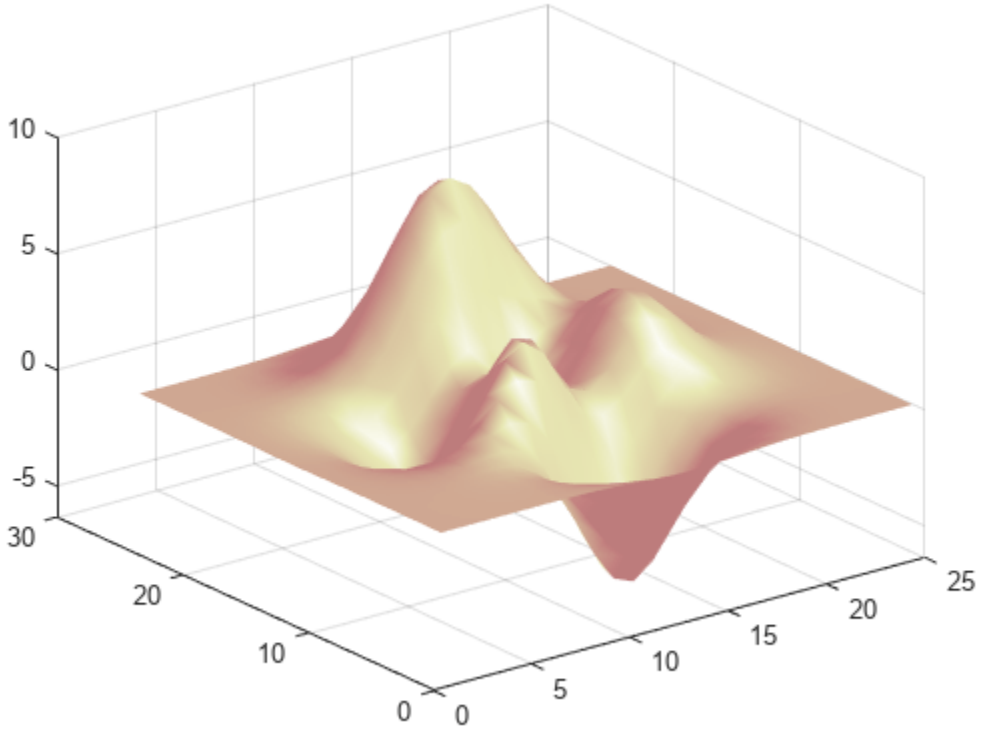
```
surf(z)
```



曲面图（着色）

surfl 函数使用基于颜色图的光照创建曲面图。为了实现更平滑的颜色过渡，请使用具有线性强度变化的颜色图，如 **pink**。

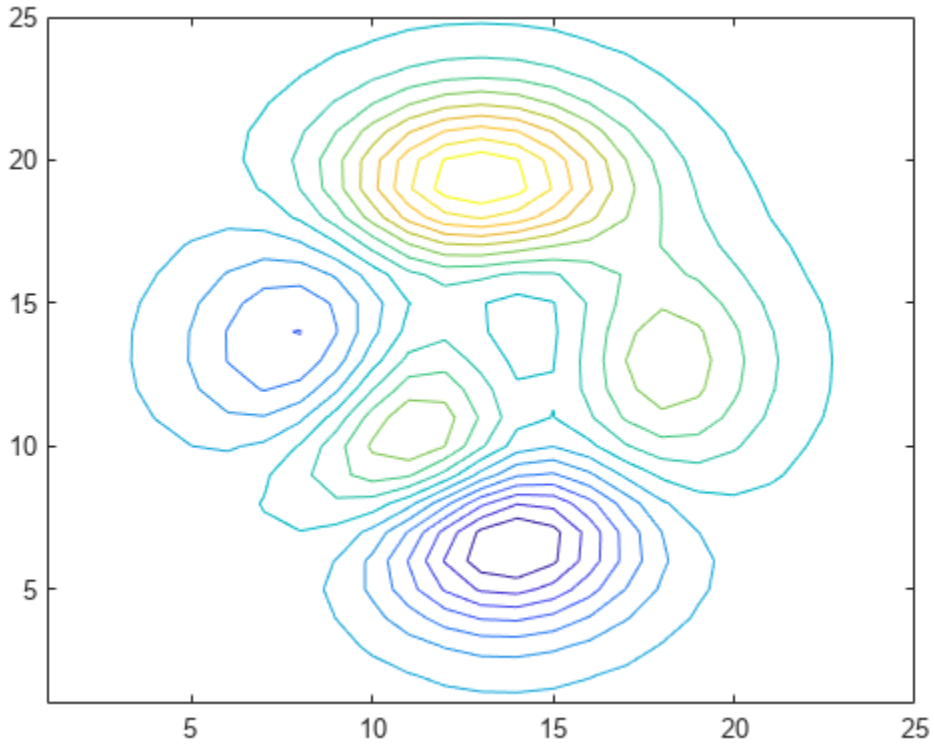
```
surfl(z)
colormap(pink)    % change color map
shading interp    % interpolate colors across lines and faces
```



等高线图

contour 函数用于创建包含常量值等高线的绘图。

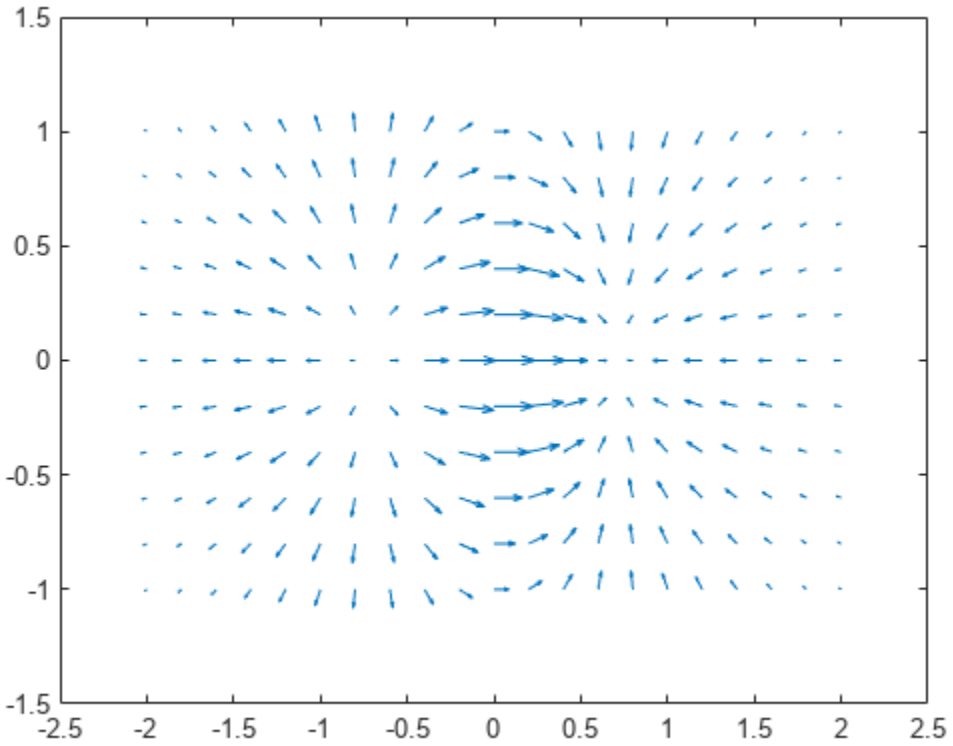
```
contour(z,16)
colormap default    % change color map
```



二维箭头图

quiver 函数将二维向量绘制为箭头。

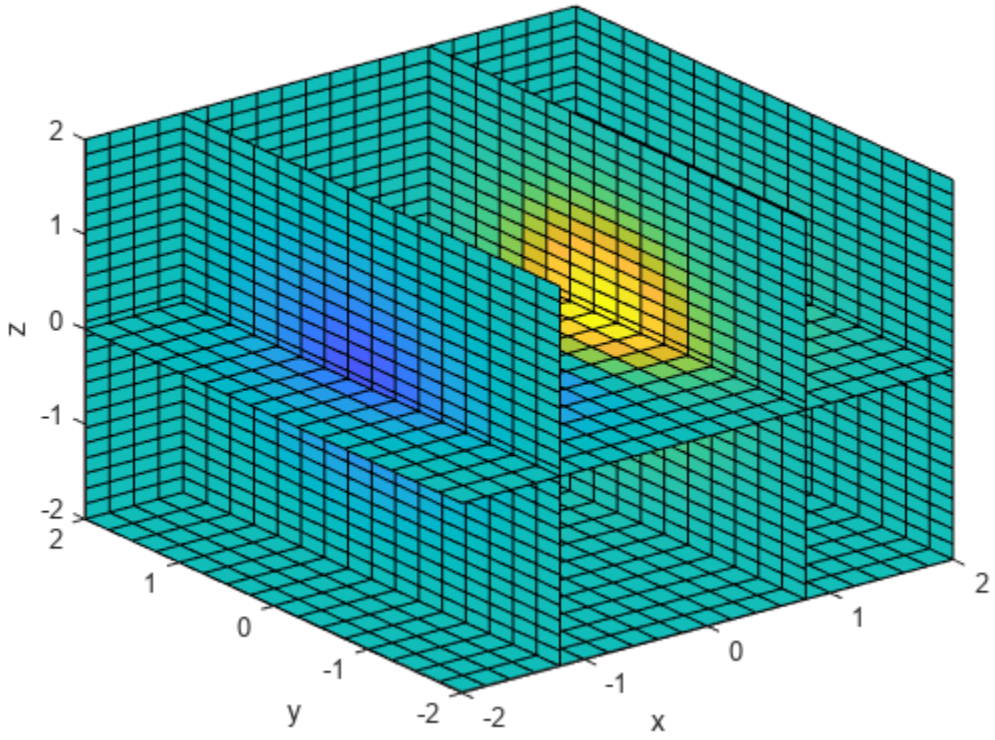
```
x = -2:.2:2;
y = -1:.2:1;

[xx,yy] = meshgrid(x,y);
zz = xx.*exp(-xx.^2-yy.^2);
[px,py] = gradient(zz,.2,.2);

quiver(x,y,px,py)
xlim([-2.5 2.5])    % set limits of x axis
```



三维体的切片图

slice 函数显示三维体数据的切片平面的数据。

```
x = -2:.2:2;
y = -2:.25:2;
z = -2:.16:2;

[x,y,z] = meshgrid(x,y,z);
v = x.*exp(-x.^2-y.^2-z.^2);

xslice = [-1.2,.8,2];    % location of y-z planes
yslice = 2;              % location of x-z plane
zslice = [-2,0];         % location of x-y planes

slice(x,y,z,v,xslice,yslice,zslice)
xlabel('x')
ylabel('y')
zlabel('z')
```


2
1
0
-1
-2
z
2
1
0
-1
-2
y
-2
-1
0
1
2
x

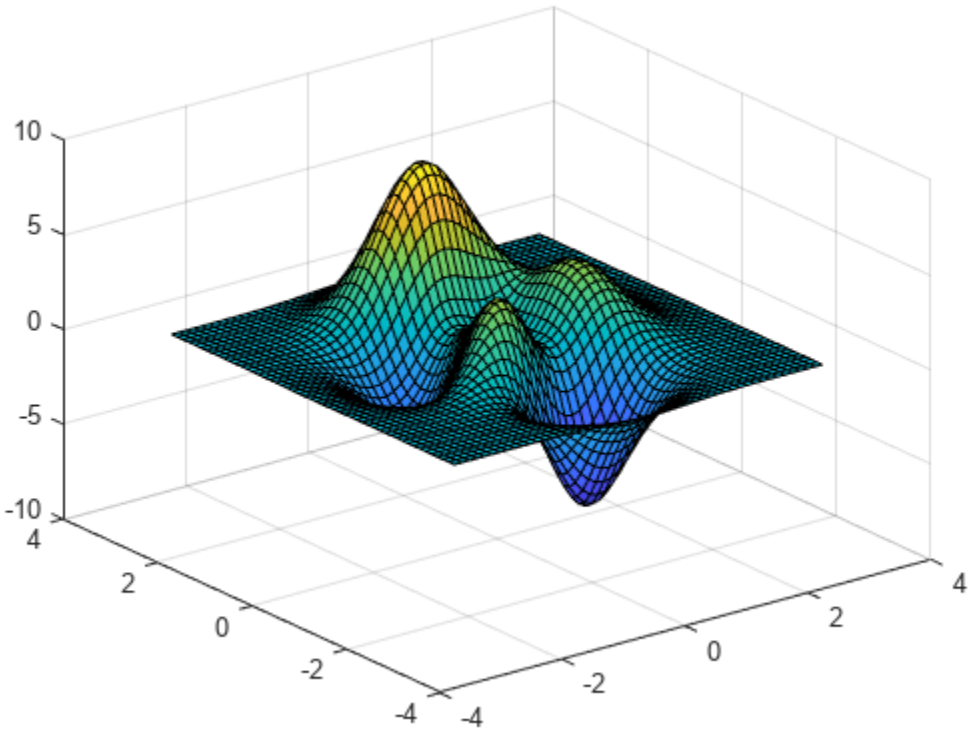
更改曲面属性

以下示例演示如何在 MATLAB® 中获取曲面图的属性，以及如何更改属性值以自定义绘图。

曲面对象

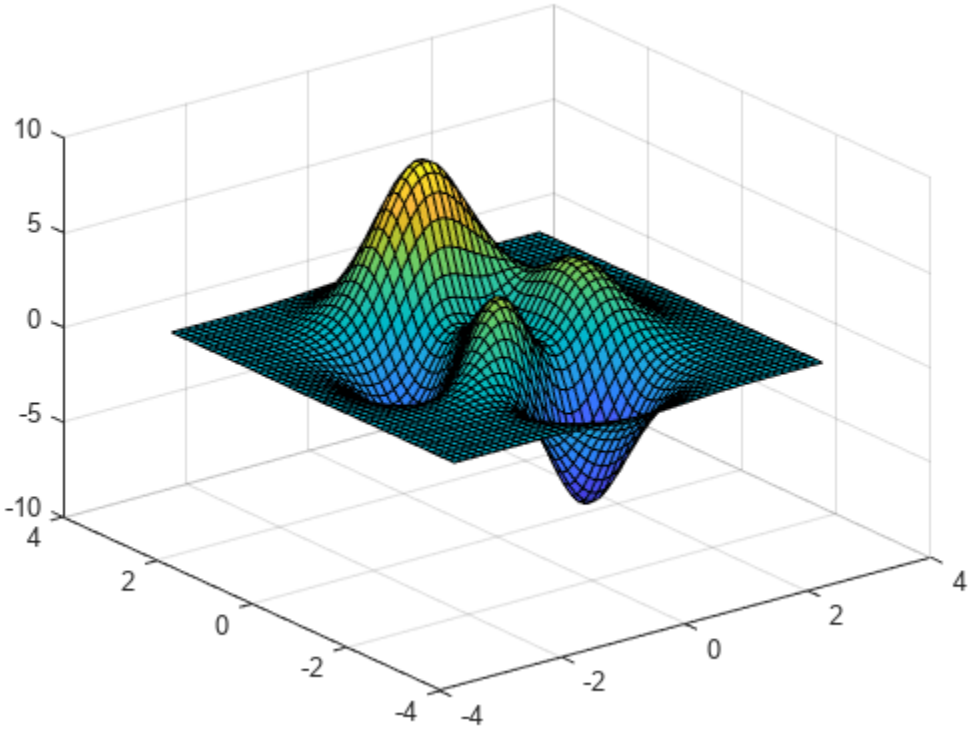
在 MATLAB 中有多种方式可以创建曲面对象。一种方式是使用 **surf**。

```
[X,Y,Z] = peaks(50);

figure
surf(X,Y,Z)
```



与所有图形对象一样，曲面有您可以查看和修改的属性。这些属性具有默认值。下面所示的曲面对象 **s** 显示了最常用的曲面属性，如 **EdgeColor**、**LineStyle**、**FaceColor** 和 **FaceLighting**。

```
s = surf(X,Y,Z)
```




```
s =
  Surface with properties:

      EdgeColor: [0 0 0]
      LineStyle: '-'
      FaceColor: 'flat'
    FaceLighting: 'flat'
      FaceAlpha: 1
          XData: [50x50 double]
          YData: [50x50 double]
          ZData: [50x50 double]
          CData: [50x50 double]

  Show all properties
```

获取个别曲面属性

若要访问个别属性，请使用圆点表示法语法 **object.PropertyName**。例如，返回曲面的 **FaceColor** 属性。

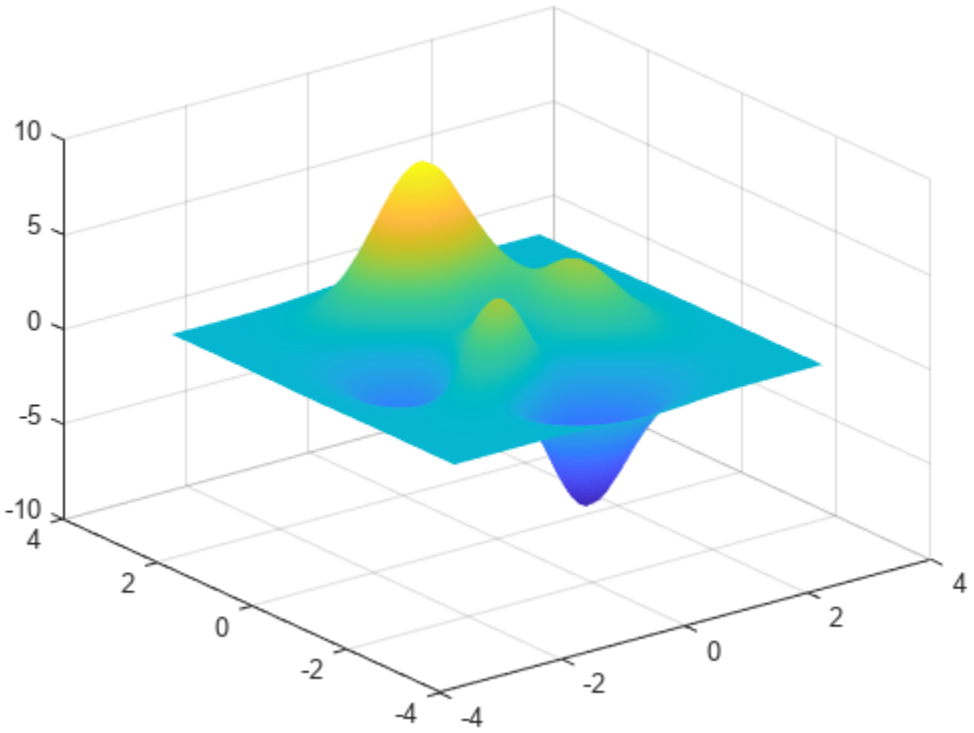
```
s.FaceColor
```

```
ans =
'flat'
```

更改常用的曲面属性

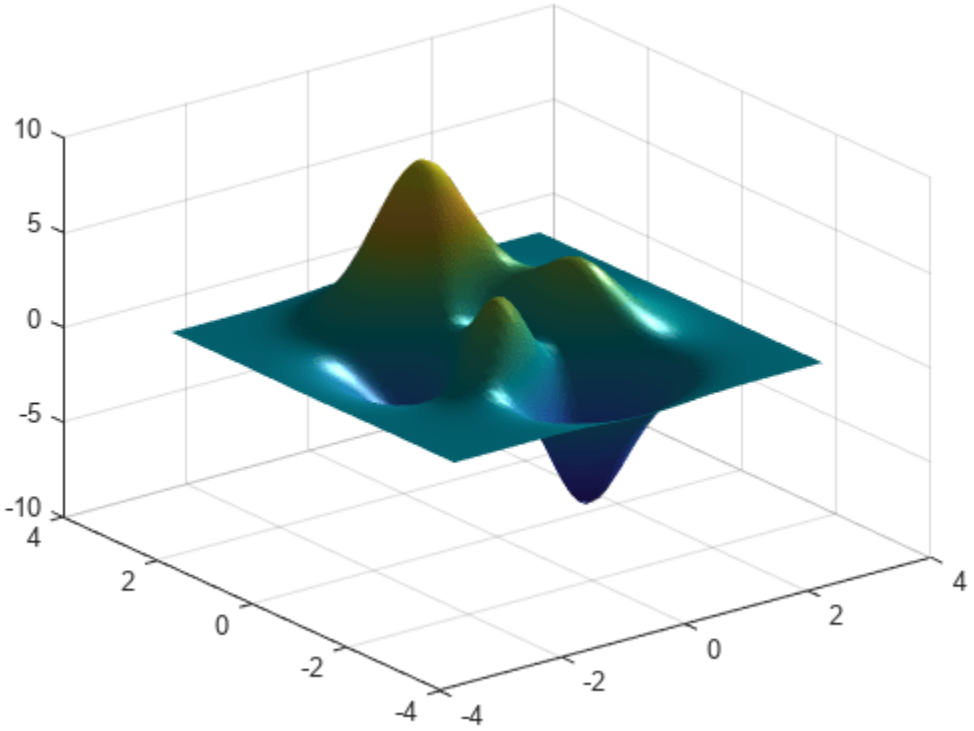
有些函数可用于更改曲面属性。例如，使用 **shading** 函数控制曲面着色。

```
shading interp    % interpolate the colormap across the surface face
```



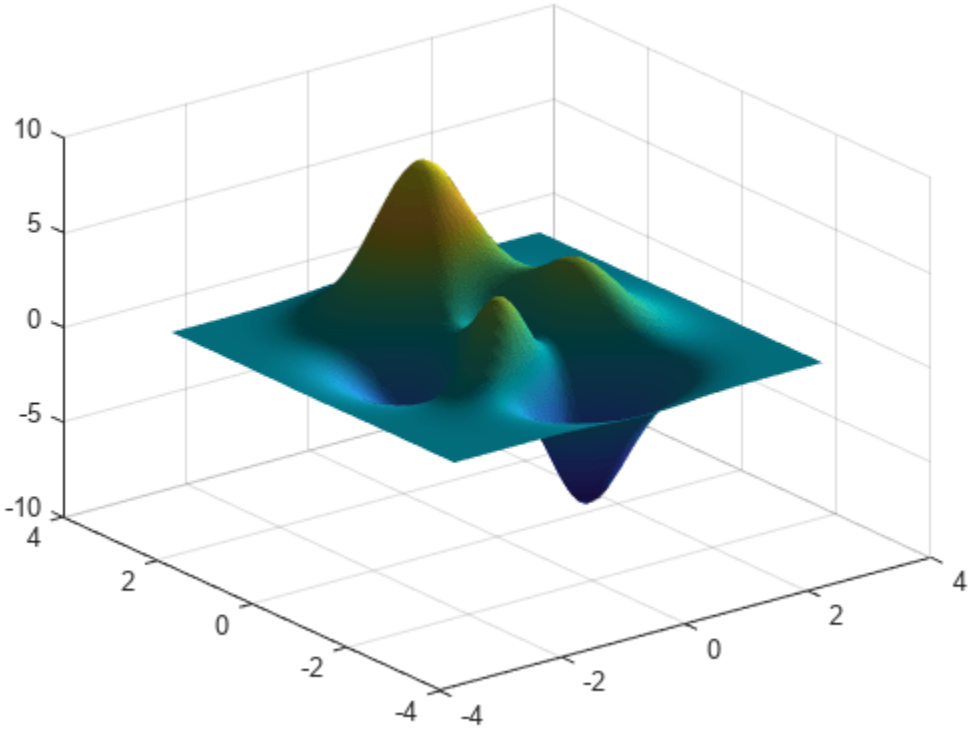
使用 **lighting** 函数调整曲面的光照特性。要使 **lighting** 产生效果，您必须创建一个光源对象来照亮曲面。

```
light           % create a light
lighting gouraud    % preferred method for lighting curved surfaces
```



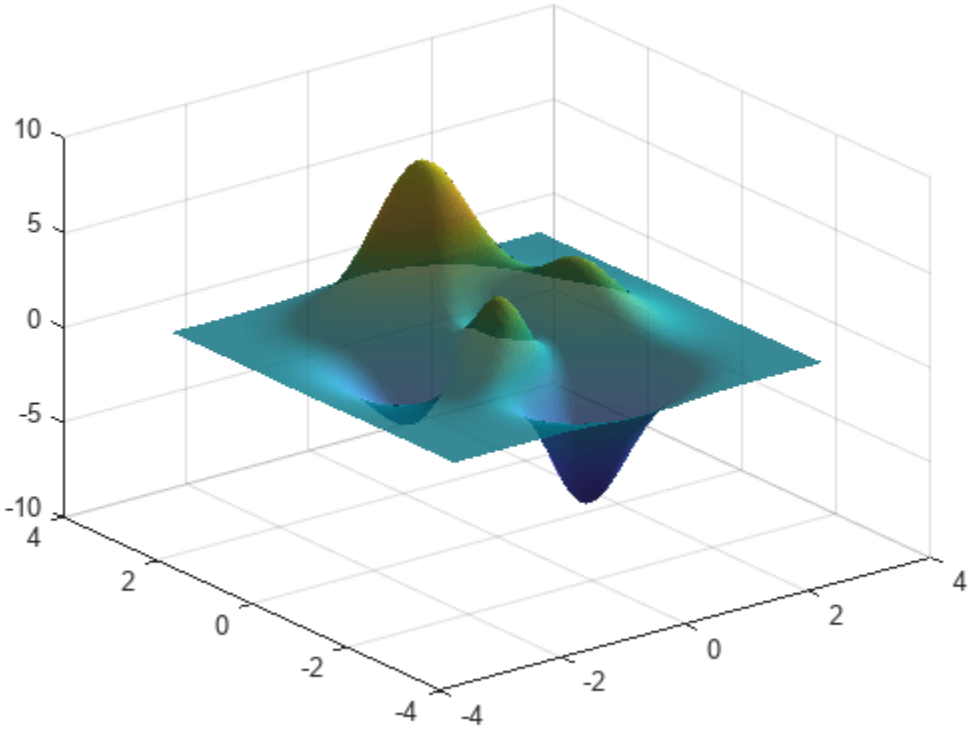
若要更改曲面的反射属性，请使用 **material** 函数。

```
material dull    % set material to be dull, no specular highlights
```



若要为当前坐标区中的所有对象设置透明度，请使用 **alpha** 函数。此函数将透明度设置为介于 1 和 0 之间的任意值，其中 1 表示完全不透明，0 表示完全透明。

```
alpha(0.8)    % set transparency to 0.8
```

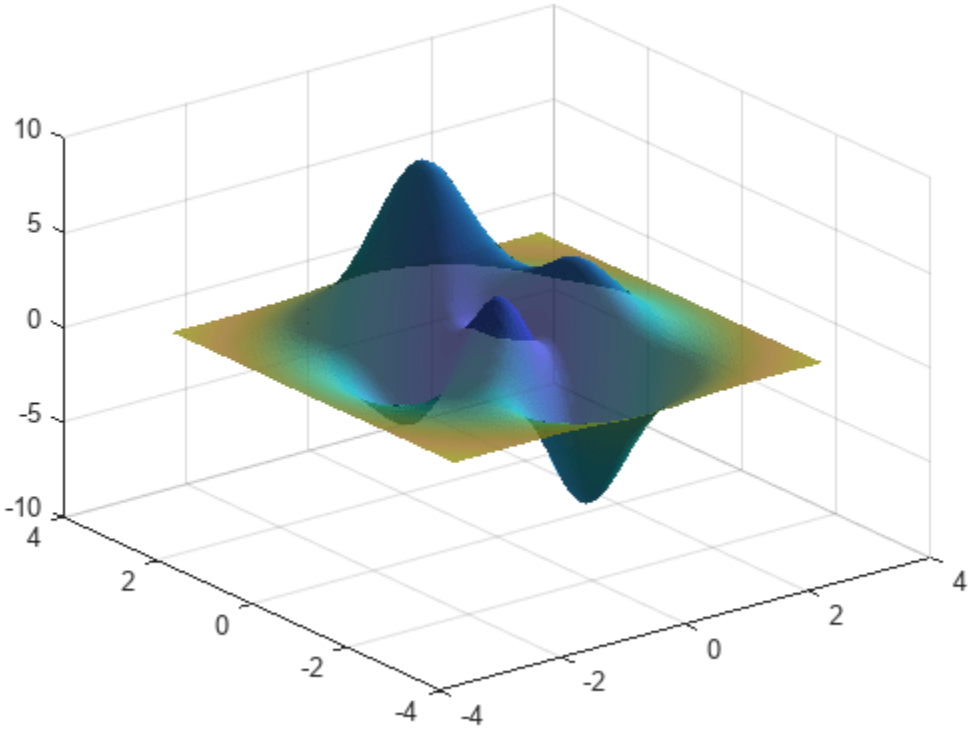


更改其他曲面属性

若要自定义曲面的外观，请使用圆点表示法更改属性值。

CData 定义曲面顶点的颜色。**FaceColor** 属性指示如何从顶点颜色确定曲面颜色。

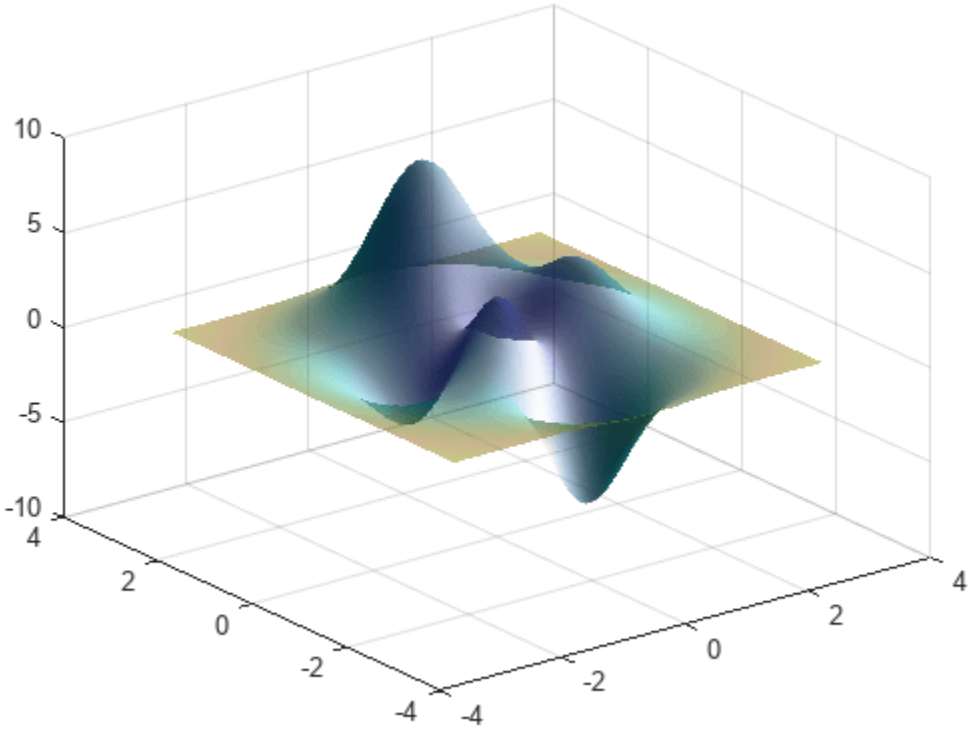
```
s.CData = hypot(X,Y);     % set color data
```




```
s.FaceColor = 'interp';    % interpolate to get face colors
```

AlphaData 定义曲面每个顶点的透明度。FaceAlpha 属性指示如何从顶点透明度确定曲面透明度。

```
s.AlphaData = gradient(Z);    % set vertex transparencies
s.FaceAlpha = 'interp';       % interpolate to get face transparencies
```



获取所有曲面属性

MATLAB 中的图形对象有许多属性。若要查看曲面的所有属性，请使用 **get** 命令。

```
get(s)
```

```
    AlignVertexCenters: off
           AlphaData: [50x50 double]
     AlphaDataMapping: 'scaled'
      AmbientStrength: 0.3000
           Annotation: [1x1 matlab.graphics.eventdata.Annotation]
     BackFaceLighting: 'reverselit'
        BeingDeleted: off
          BusyAction: 'queue'
       ButtonDownFcn: ''
             CData: [50x50 double]
        CDataMapping: 'scaled'
          CDataMode: 'manual'
        CDataSource: ''
          Children: [0x0 GraphicsPlaceholder]
          Clipping: on
        ContextMenu: [0x0 GraphicsPlaceholder]
          CreateFcn: ''
    DataTipTemplate: [1x1 matlab.graphics.datatip.DataTipTemplate]
          DeleteFcn: ''
    DiffuseStrength: 0.8000
        DisplayName: ''
          EdgeAlpha: 1
```

```
            EdgeColor: 'none'
            EdgeLighting: 'none'
            FaceAlpha: 'interp'
            FaceColor: 'interp'
            FaceLighting: 'gouraud'
            FaceNormals: [49x49x3 double]
            FaceNormalsMode: 'auto'
            HandleVisibility: 'on'
            HitTest: on
            Interruptible: on
            LineStyle: '-'
            LineWidth: 0.5000
            Marker: 'none'
            MarkerEdgeColor: 'auto'
            MarkerFaceColor: 'none'
            MarkerSize: 6
            MeshStyle: 'both'
            Parent: [1x1 Axes]
            PickableParts: 'visible'
            Selected: off
            SelectionHighlight: on
            SpecularColorReflectance: 1
            SpecularExponent: 10
            SpecularStrength: 0
            Tag: ''
            Type: 'surface'
            UserData: []
            VertexNormals: [50x50x3 double]
            VertexNormalsMode: 'auto'
            Visible: on
            XData: [50x50 double]
            XDataMode: 'manual'
            XDataSource: ''
            YData: [50x50 double]
            YDataMode: 'manual'
            YDataSource: ''
            ZData: [50x50 double]
            ZDataSource: ''
```

创建 MATLAB 徽标

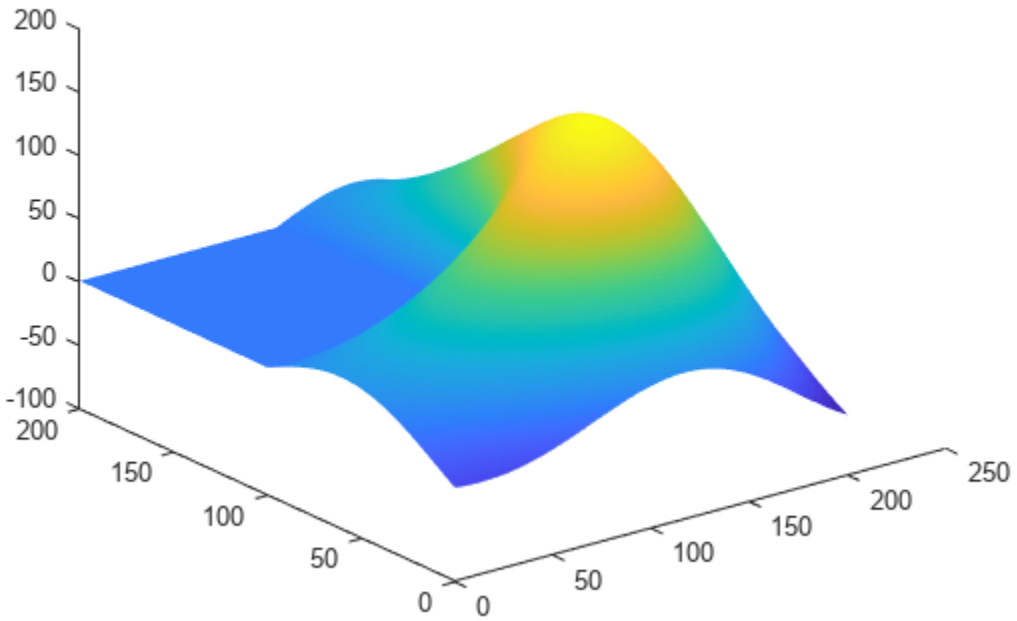
以下示例演示如何创建和显示 MATLAB® 徽标。

使用 **membrane** 命令生成徽标的曲面数据。

```
L = 160*membrane(1,100);
```

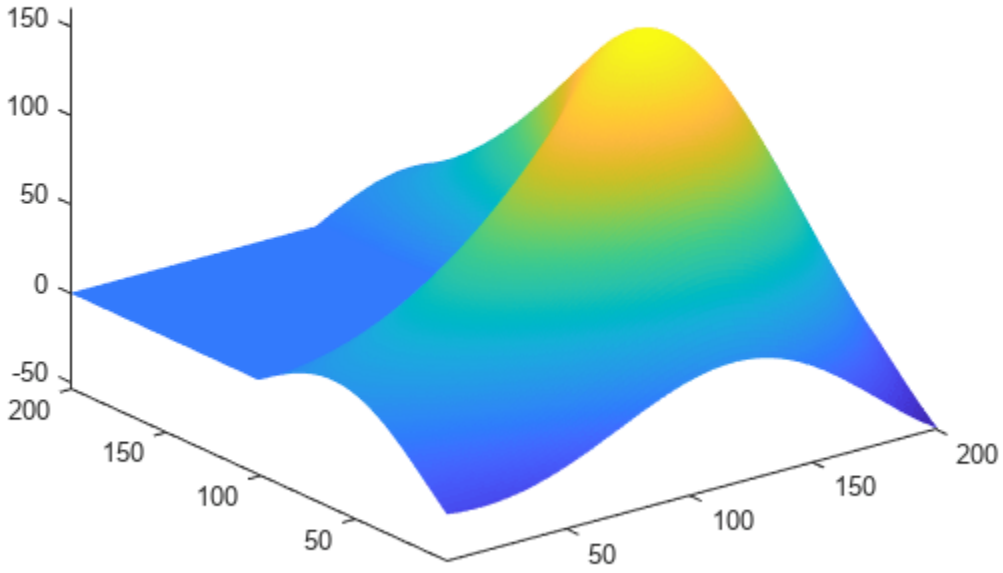
创建一个图窗和一套坐标区以显示徽标。然后，使用通过 **membrane** 命令得到的点创建徽标的曲面。关闭曲面中的线条。

```
f = figure;
ax = axes;

s = surface(L);
s.EdgeColor = 'none';
view(3)
```



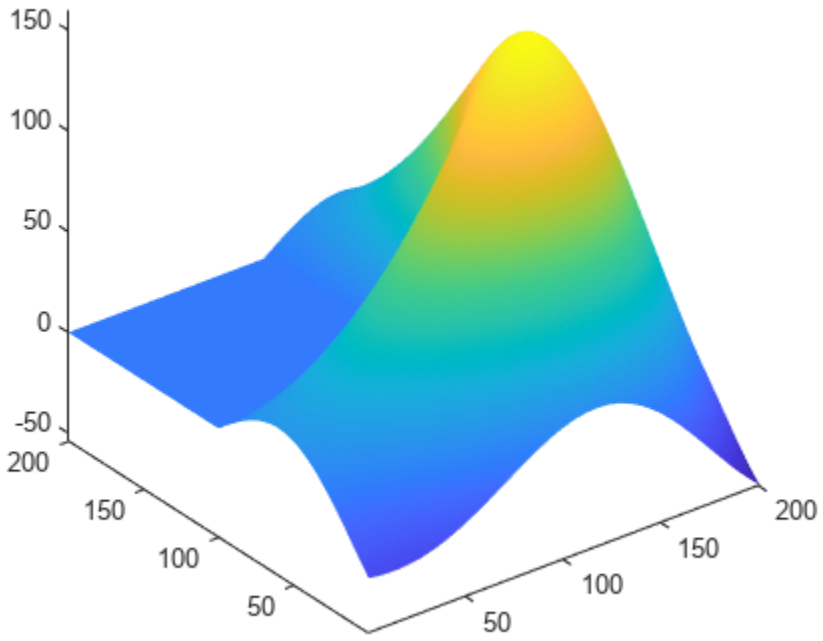
调整坐标区范围，使坐标区紧密围绕在徽标周围。

```
ax.XLim = [1 201];
ax.YLim = [1 201];
ax.ZLim = [-53.4 160];
```



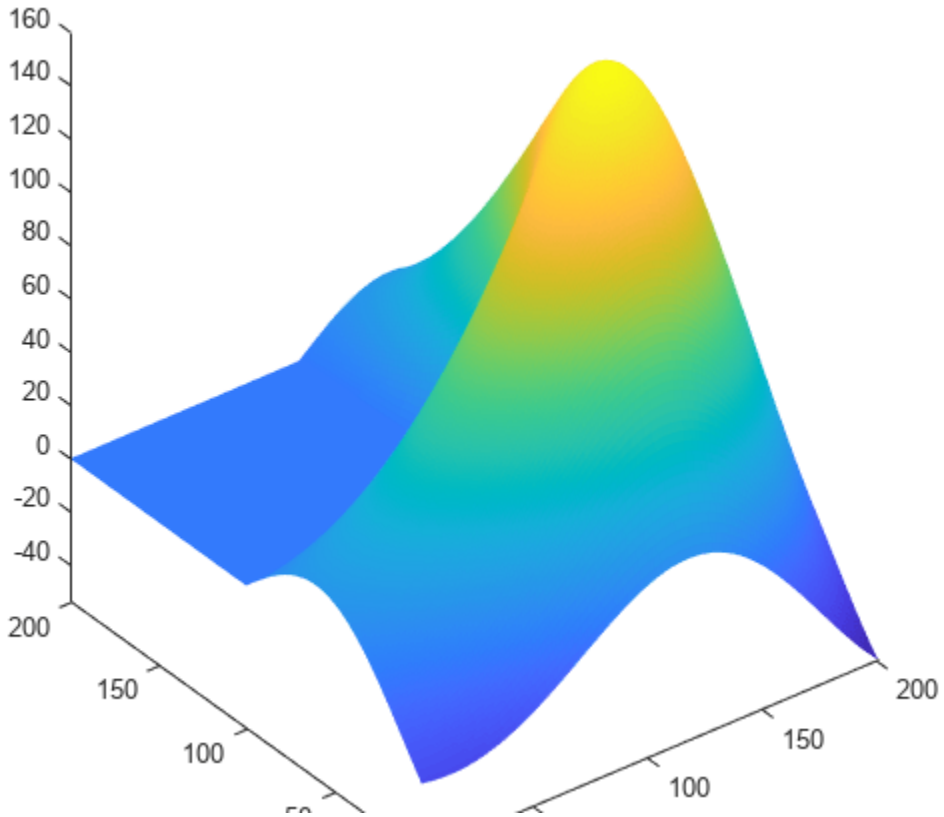
使用坐标区的相机属性调整徽标视图。相机属性控制三维场景的视图，就像带有缩放镜头的相机一样。

```
ax.CameraPosition = [-145.5 -229.7 283.6];
ax.CameraTarget = [77.4 60.2 63.9];
ax.CameraUpVector = [0 0 1];
ax.CameraViewAngle = 36.7;
```



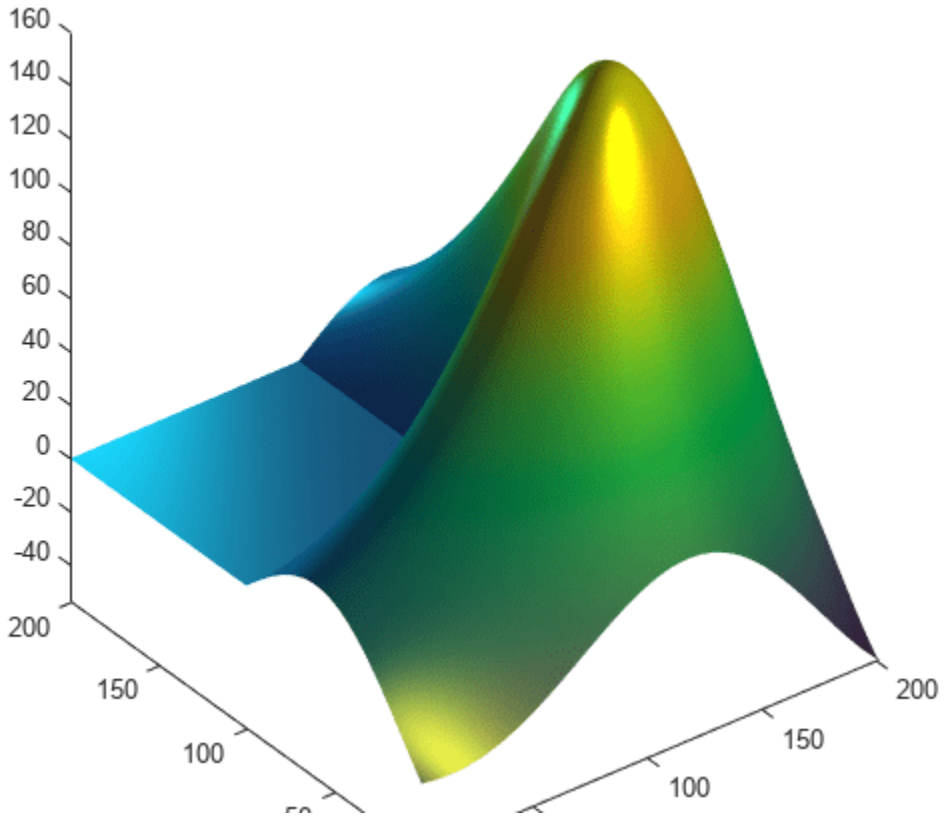
更改坐标区的位置和 x、y 和 z 纵横比以填充图窗窗口中的额外空间。

```
ax.Position = [0 0 1 1];
ax.DataAspectRatio = [1 1 .9];
```



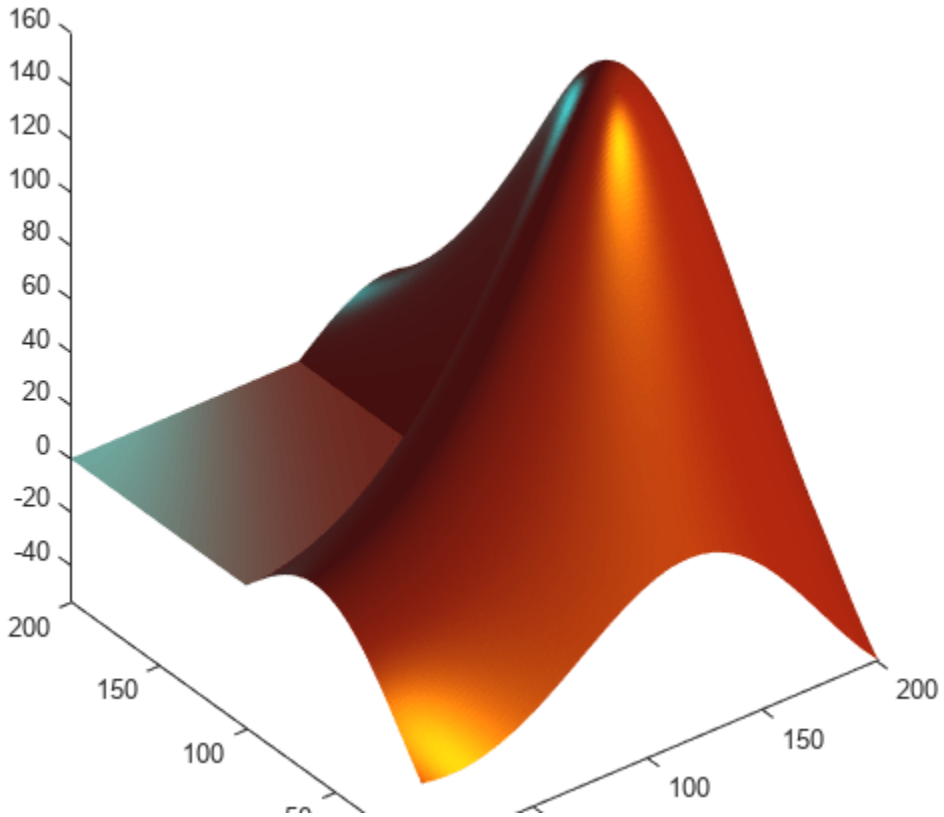
创建光源以照亮徽标。光源本身不可见，但可设置其属性以更改坐标区中任何填充或曲面对象的外观。

```
l1 = light;
l1.Position = [160 400 80];
l1.Style = 'local';
l1.Color = [0 0.8 0.8];

l2 = light;
l2.Position = [.5 -1 .4];
l2.Color = [0.8 0.8 0];
```



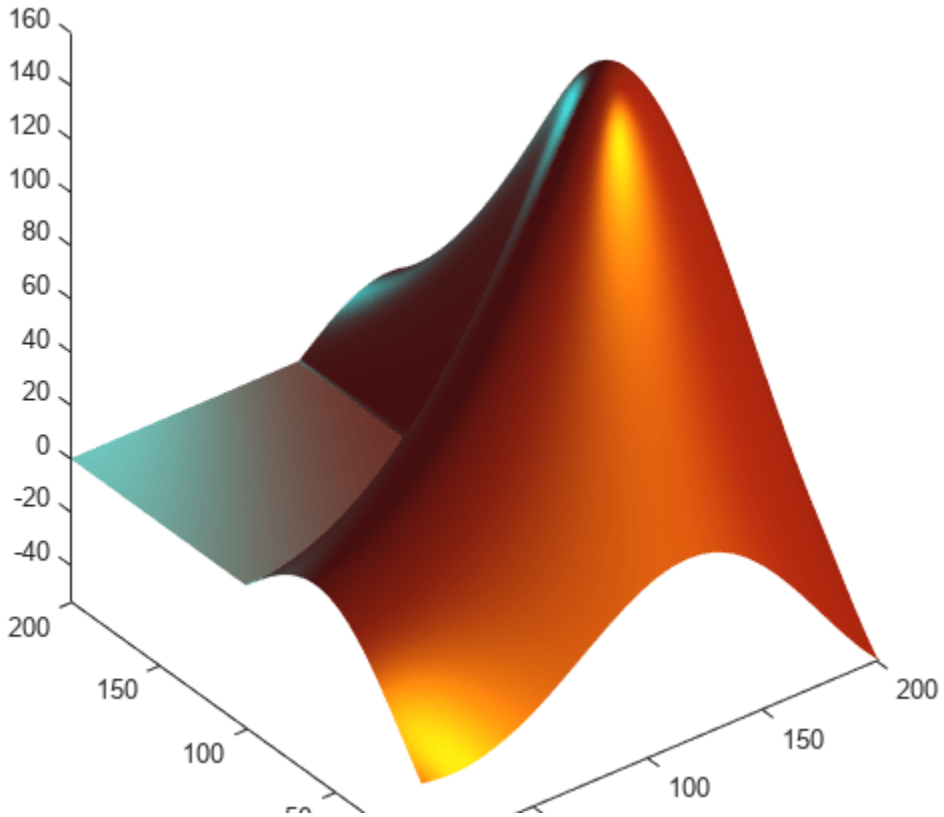
更改徽标的颜色。

```
s.FaceColor = [0.9 0.2 0.2];
```



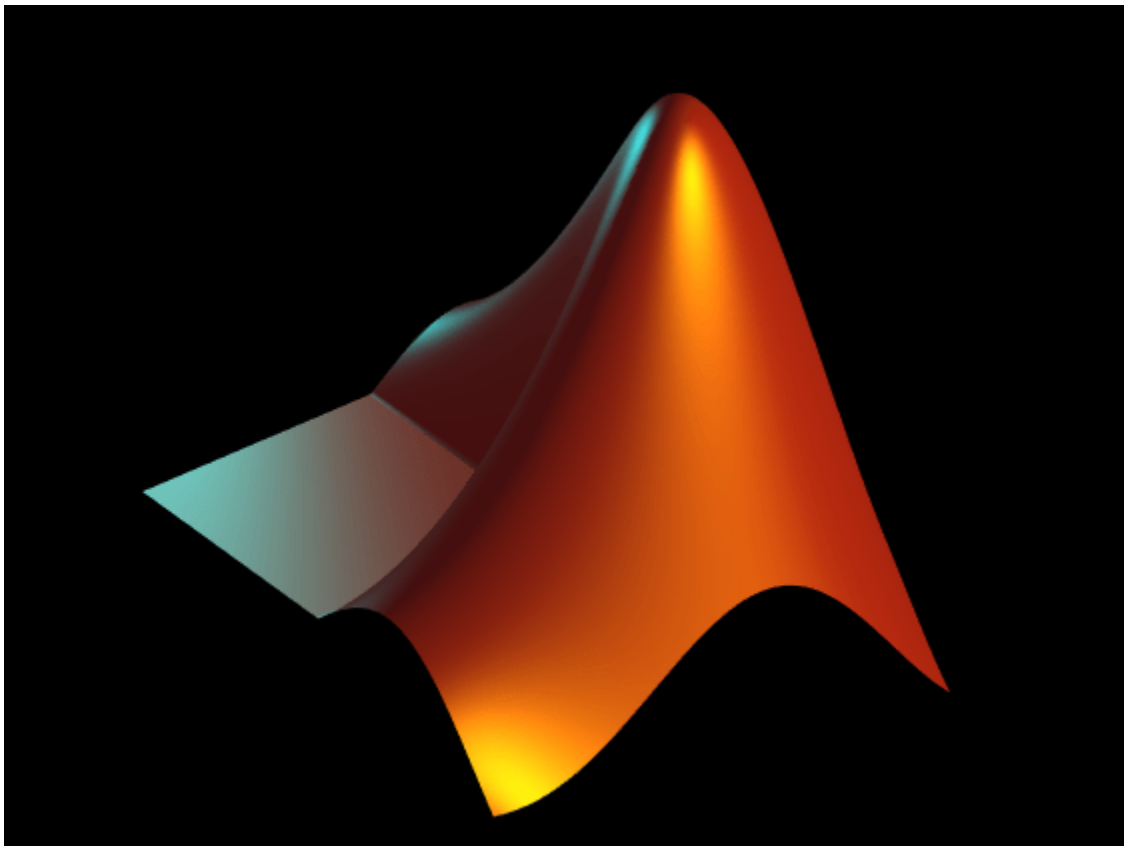
使用曲面的光照和镜面（反射）属性来控制光照效果。

```
s.FaceLighting = 'gouraud';
s.AmbientStrength = 0.3;
s.DiffuseStrength = 0.6;
s.BackFaceLighting = 'lit';

s.SpecularStrength = 1;
s.SpecularColorReflectance = 1;
s.SpecularExponent = 7;
```



关闭轴以查看最终结果。

```
axis off
f.Color = 'black';
```



将数据表示为曲面

本节内容
“用来绘制数据网格的函数” （第 1-25 页） “用于对数据进行网格化和插值的函数” （第 1-25 页） “网格图和曲面图” （第 1-25 页） “可视化包含两个变量的函数” （第 1-26 页） “非均匀采样数据的曲面图” （第 1-29 页） “重构数据” （第 1-30 页） “参数化曲面” （第 1-32 页）

用来绘制数据网格的函数

MATLAB 图形通过 x-y 平面中的矩形网格上方的点的 z 坐标来定义曲面。通过用直线连接相邻点来形成绘图。曲面图可用于可视化因太大而无法以数字形式显示的矩阵，还可用于绘制包含两个变量的函数。

MATLAB 可以创建不同形式的曲面图。网格图是指仅对连接定义点的线条进行着色的线框曲面图。曲面图对连接线和面都进行着色。下表列出了曲面图的各种形式。

函数	用于创建
mesh、**surf**	曲面图
meshc、**surfc**	下方带有等高线图的曲面图
meshz	带帷幕图（参考平面）的曲面图
pcolor	单一着色平面图（值仅与颜色成比例）
surfl	从指定方向照亮的曲面图
surface	用于创建曲面图形对象的低级函数（高级函数的基础）

用于对数据进行网格化和插值的函数

当您需要对数据进行重构和插值以便将它们表示为曲面时，这些函数很有用。

函数	用于创建
meshgrid	二维和三维空间中的矩形网格
griddata	散点数据插值
griddedInterpolant	网格数据插值
scatteredInterpolant	散点数据插值

有关如何对数据进行插值的讨论，请参阅“插入网格数据”和“内插散点数据”。

网格图和曲面图

mesh 和 **surf** 命令可创建矩阵数据的三维曲面图。如果 **Z** 是矩阵，其元素 **Z(i,j)** 定义曲面在基础 **(i,j)** 网格的上方的高度，则

mesh(Z)

生成曲面的彩色线框视图并在三维视图中显示。类似地，

surf(Z)

生成曲面的着色分面视图并在三维视图中显示。通常这些分面为四边形，每个分面为一种固定颜色，边为黑色网格线，但使用 **shading** 命令可以消除网格线 (**shading flat**) 或选择对整个分面进行插补着色 (**shading interp**)。

曲面对象属性用于进一步控制曲面的视觉外观。您可以指定边的线型、顶点标记、面的颜色以及光照特性等。

可视化包含两个变量的函数

1. 要显示包含两个变量的函数 z = f(**x**,**y**)，需要生成 **X** 和 **Y** 矩阵，它们分别由函数域内重复的行和列组成。您将使用这两个矩阵来计算和绘制该函数。
2. **meshgrid** 函数将 **x** 和 **y** 两个向量指定的域转换为矩阵 **X** 和 **Y**。然后，您可以使用这两个矩阵来计算包含两个变量的函数：**X** 的行是向量 **x** 的副本，**Y** 的列是向量 **y** 的副本。

例 1.1. 示例：说明 meshgrid 的用法

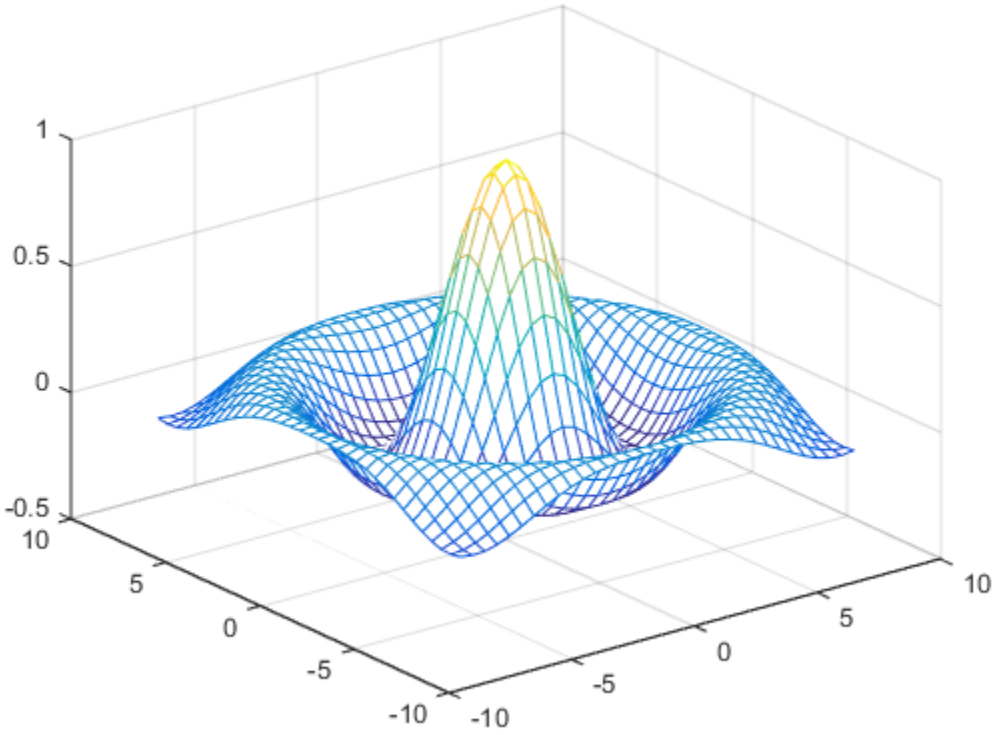
下面以 **sin(r)/r**（即 **sinc** 函数）为例说明 **meshgrid** 的用法。要计算此函数在 x 和 y 都为 -8 到 8 之间时的值，只需向 **meshgrid** 传递一个向量参数，此参数将用于完成两个方向上的计算。

```
[X,Y] = meshgrid(-8:.5:8);
R = sqrt(X.^2 + Y.^2) + eps;
```

矩阵 **R** 包含到矩阵中心（即原点）之间的距离。加上 **eps** 是为了防止被零除（下一步），否则将在数据中生成 **Inf** 值。

构造 **sinc** 函数并使用 **mesh** 绘制 **Z** 值将生成一个三维曲面。

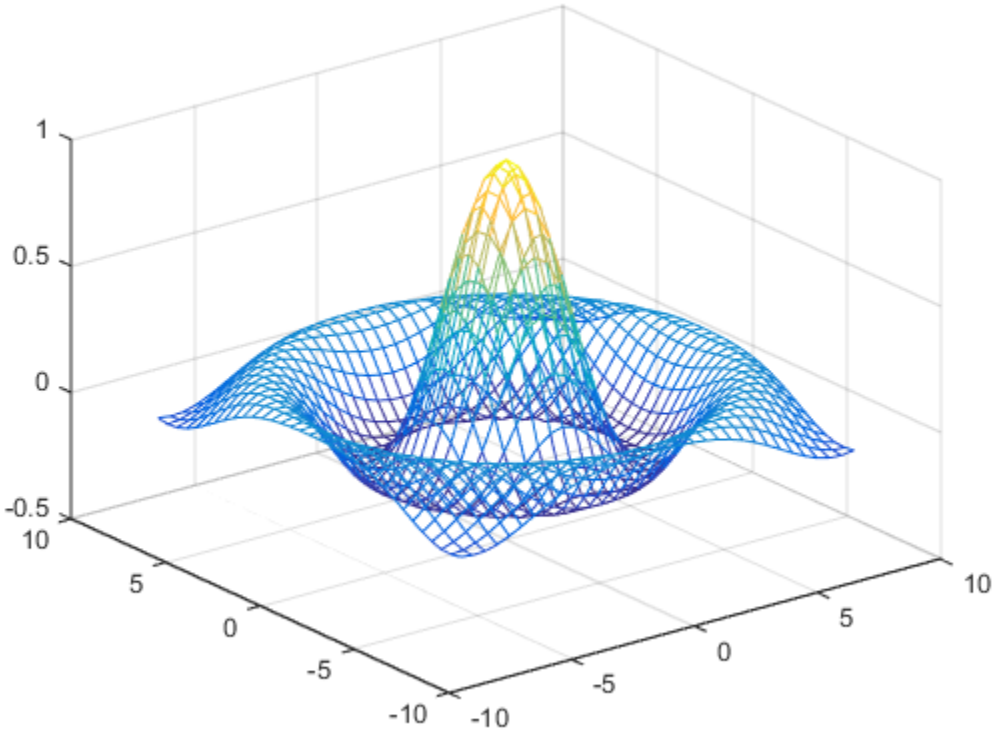
```
Z = sin(R)./R;
figure
mesh(X,Y,Z)
```



隐线消除

默认情况下，MATLAB 会消除网格图中不可见的隐线，即使网格图的面未填充也是如此。您可以使用 **hidden** 命令禁用隐线消除并使网格图的面透明：

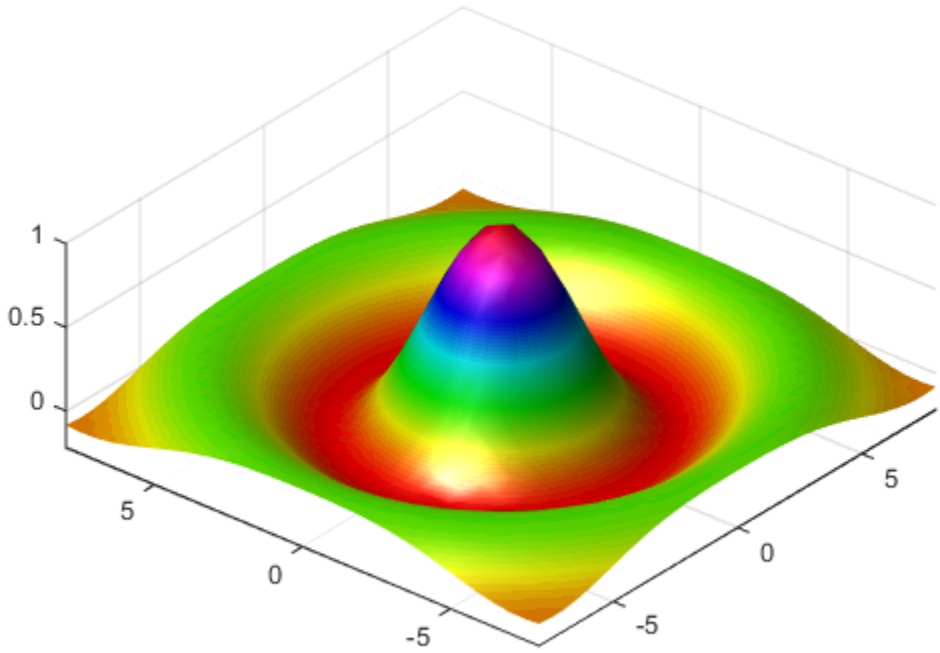
```
hidden off
```



渲染曲面形状

MATLAB 提供了许多方法来增强图形所含信息的显示。例如，下面这个 **sinc** 函数图使用的数据与上一个图相同，但它利用光照、视图调整和不同的颜色图来渲染所绘制函数（**daspect**、**axis**、**view** 和 **camlight**）的形状。

```
figure
colormap hsv
surf(X,Y,Z,'FaceColor','interp',...
   'EdgeColor','none',...
   'FaceLighting','gouraud')
daspect([5 5 1])
axis tight
view(-50,30)
camlight left
```



有关曲面图的详细信息，请参阅 **surf** 函数。

非均匀采样数据的曲面图

您可以使用 **meshgrid** 创建要用于计算和绘制 **sinc** 函数的均匀采样数据点的网格。然后，MATLAB 通过连接相邻矩阵元素来形成四边形网格，从而生成曲面图。

要根据非均匀采样数据生成曲面图，请使用 **scatteredInterpolant** 在等间距处进行插值，然后按常规方式使用 **mesh** 和 **surf**。

示例 - 在曲面上显示非均匀数据

此示例计算 **sinc** 函数在特定范围内随机点处的值，然后生成均匀采样的数据以显示为曲面图。此过程涉及以下任务：

- 使用 **linspace** 在非均匀采样的数据范围内生成等间距值。
- 使用 **meshgrid** 并利用 **linspace** 的输出生成绘图网格。
- 使用 **scatteredInterpolant** 将按 **meshgrid** 返回的等间距网格对非固定间隔采样数据进行插值。
- 使用绘图函数显示数据。

1 在 [-8, 8] 范围内生成非均匀采样的数据，并利用它来计算函数：

```
x = rand(100,1)*16 - 8;
y = rand(100,1)*16 - 8;
```

```
r = sqrt(x.^2 + y.^2) + eps;
z = sin(r)./r;
```

2 **linspace** 函数为创建具有所需元素数的等间距数据提供了一种便捷的方式。下面的语句在随机数据范围内生成向量，分辨率与前面的 **sinc** 示例中 -8:.5:8 语句生成的分辨率相同：

```
xlin = linspace(min(x),max(x),33);
ylin = linspace(min(y),max(y),33);
```

3 现在使用这些点生成等间距网格：

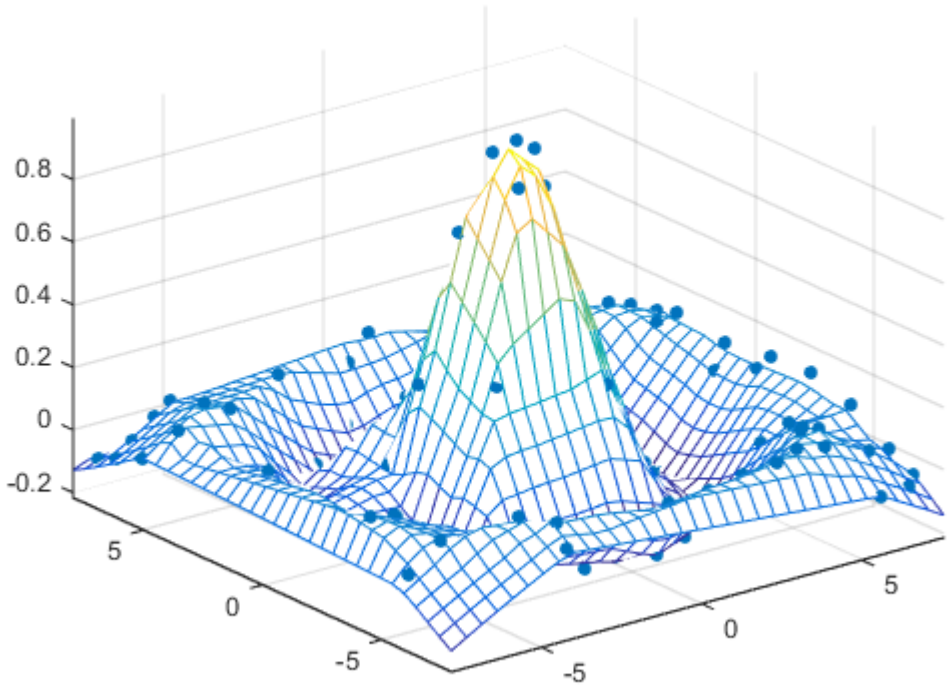
```
[X,Y] = meshgrid(xlin,ylin);
```

4 此过程的关键是基于函数在原始数据点的值（本例中为随机数据点），使用 **scatteredInterpolant** 在等间距点处进行函数值插值。此语句使用默认的线性插值生成新数据：

```
f = scatteredInterpolant(x,y,z);
Z = f(X,Y);
```

5 绘制插值和非均匀数据以生成：

```
figure
mesh(X,Y,Z) %interpolated
axis tight; hold on
plot3(x,y,z,'.','MarkerSize',15) %nonuniform
```



重构数据

假设您有一个数据集，其中包含以下 (X, Y, Z) 三元组：

X	Y	Z
1	1	152
2	1	89
3	1	100
4	1	100
5	1	100
1	2	103
2	2	0
3	2	100
4	2	100
5	2	100
1	3	89
2	3	13
3	3	100
4	3	100
5	3	100
1	4	115
2	4	100
3	4	187
4	4	200
5	4	111
1	5	100
2	5	85
3	5	111
4	5	97
5	5	48

您可以先通过调整数据结构，使用各种 MATLAB 图形函数（如 **surf**、**contour** 和 **stem3**）来表示这些向量形式的数据。使用 (X, Y) 值定义 x-y 平面上存在 Z 值的点处的坐标。**reshape** 和 **transpose** 函数可以调整数据结构，使 (X, Y, Z) 三元组形成矩形网格：

```
x = reshape(X,5,5)';
y = reshape(Y,5,5)';
z = reshape(Z,5,5)';
```

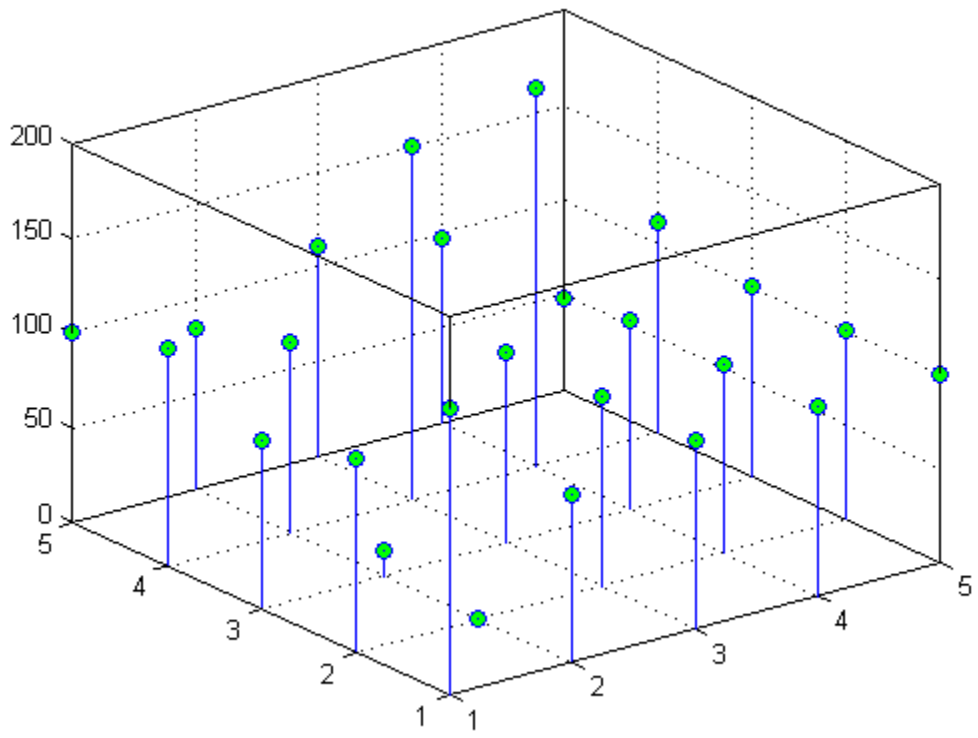
重构会产生三个 5×5 数组：

```
x =

     1     2     3     4     5
     1     2     3     4     5
     1     2     3     4     5
     1     2     3     4     5
     1     2     3     4     5
```

```
y =

     1     1     1     1     1
     2     2     2     2     2
     3     3     3     3     3
     4     4     4     4     4
     5     5     5     5     5

z =

   152    89   100   100   100
   103     0   100   100   100
    89    13   100   100   100
   115   100   187   200   111
   100    85   111    97    48
```

现在您可以相对于 X 和 Y 来表示 Z 的值。例如，创建一个三维针状图：

```
stem3(x,y,z,'MarkerFaceColor','g')
```



参数化曲面

用来绘制曲面的函数可以使用两个额外的向量或矩阵参数来表示具有特定 x 和 y 数据的曲面。如果 **Z** 是 m×n 矩阵，其中 **x** 是 n 向量，**y** 是 m 向量，则

```
mesh(x,y,Z,C)
```

表示顶点颜色为 **C(i,j)** 并位于以下点的网格曲面

(x(j), y(i), Z(i,j))

其中 **x** 对应于 **Z** 的各列，**y** 对应于各行。

更常见的是，如果 **X**、**Y**、**Z** 和 **C** 是维度相同的矩阵，则

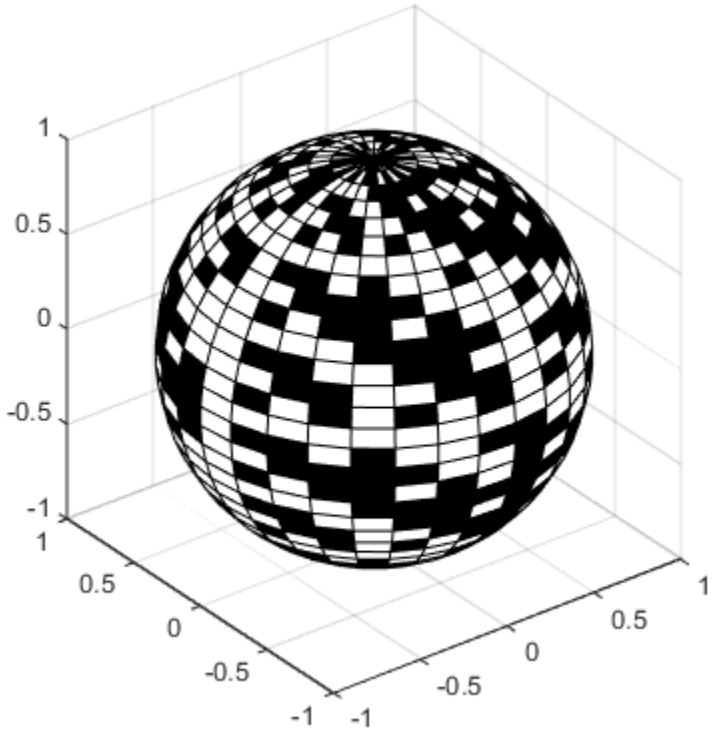
mesh(X,Y,Z,C)

表示顶点颜色为 **C(i,j)** 并位于以下点的网格曲面

(X(i,j), Y(i,j), Z(i,j))

此示例使用球面坐标来绘制球体，并使用 Hadamard 矩阵（信号处理编码理论中使用的一种正交矩阵）中的加号和减号图案进行着色。向量 **theta** 和 **phi** 的范围分别为 -π ≤ **theta** ≤ π 和 -π/2 ≤ **phi** ≤ π/2。由于 **theta** 是行向量而 **phi** 是列向量，因此产生矩阵 **X**、**Y** 和 **Z** 的乘法是向量外积。

```
figure
k = 5;
n = 2^k-1;
theta = pi*(-n:2:n)/n;
phi = (pi/2)*(-n:2:n)'/n;
X = cos(phi)*cos(theta);
Y = cos(phi)*sin(theta);
Z = sin(phi)*ones(size(theta));
colormap([0 0 0;1 1 1])
C = hadamard(2^k);
surf(X,Y,Z,C)
axis square
```


1
0.5
0
-0.5
-1
1
0.5
0
-0.5
-1
1
0.5
0
-0.5
-1

多边形

- “补片对象简介” (第 2-2 页)
- “多面补片” (第 2-6 页)

补片对象简介

本节内容
“什么是补片对象？” （第 2-2 页） “patch 函数的行为” （第 2-2 页） “创建单个多边形” （第 2-3 页）

什么是补片对象?

补片图形对象由一个或多个相连或不相连的多边形组成。补片对建模真实世界对象（如飞机或汽车）以及绘制任意形状的二维或三维多边形非常有用。

相对而言，曲面对象是由四边形组成的矩形网格，更适合显示平面布局图，例如具有两个变量的数学函数的值、矩形平面中数据的等高线或者参数化曲面（如球体）。

有一些 MATLAB 函数可以创建补片对象 - **fill**、**fill3**、**isosurface**、**isocaps**、某些 **contour** 函数以及 **patch**。本节主要介绍如何使用 **patch** 函数。

您可以通过指定顶点坐标和某种形式的颜色数据来定义补片。补片支持各种着色选项，这些选项对于可视化几何形状上叠加的数据很有用。

指定补片的方法有两种：

- 指定每个多边形的顶点坐标，这些顶点连接起来形成补片
- 指定每个唯一顶点的坐标以及一个定义如何连接这些顶点以形成面的矩阵

第二种方法是多面补片的首选，因为它通常需要较少的数据来定义补片，而且由多个面共享的顶点只需定义一次。本节提供了这两种方法的示例。

patch 函数的行为

有两种形式的 **patch** 函数 - 高级语法和低级语法。根据您使用的语法，**patch** 函数的行为有所不同。

高级语法

当您使用高级语法时，MATLAB 会根据您指定的颜色数据自动确定如何为每个面着色。高级语法允许您省略 x、y 和 z 坐标以及颜色数据的属性名称，只要按正确顺序指定这些参数即可。

```
patch(x-coordinates,y-coordinates,z-coordinates,colordata)
```

但您必须指定颜色数据，这样 MATLAB 才能确定要使用的着色类型。如果不指定颜色数据，MATLAB 将返回错误。

```
x = [0 1 1 0];
y = [0 0 1 1];
patch(x,y)
Error using patch
Not enough input arguments.
```

低级语法

低级语法仅接受属性名称/属性值对组作为参数，并且不会自动为面着色，除非您还更改了 **FaceColor** 属性的值。例如，下面的语句

```
patch('XData',x,'YData',y)
```

绘制一个面颜色为黑色的补片，因为 **FaceColor** 属性的出厂默认值为黑色。

```
get(groot,'FactoryPatchFaceColor')
ans =
     0     0     0
```

有关如何获取属性的出厂默认值和用户默认值的信息，请参阅《MATLAB 函数参考》中的 Patch 列表以及 **get** 命令。

解释颜色参数

当您使用高级语法时，MATLAB 会将第三个（如果存在 z 坐标，则为第四个）参数解释为颜色数据。如果您打算使用 x、y 和 z 坐标来定义补片，但未指定颜色，则 MATLAB 会将 z 坐标解释为颜色数据，并绘制一个二维补片。例如，

```
patch(x,y,1:length(x))
```

将绘制一个所有顶点的 z 值均为 0 的补片，并会根据顶点颜色来插补着色（因为每个顶点一种颜色），而

```
patch(x,y,1:length(x),'y')
```

则会绘制一个各个顶点的 z 值递增的补片，颜色为黄色。

“补片数据与颜色图的关系”提供了有关补片着色选项的更多信息。

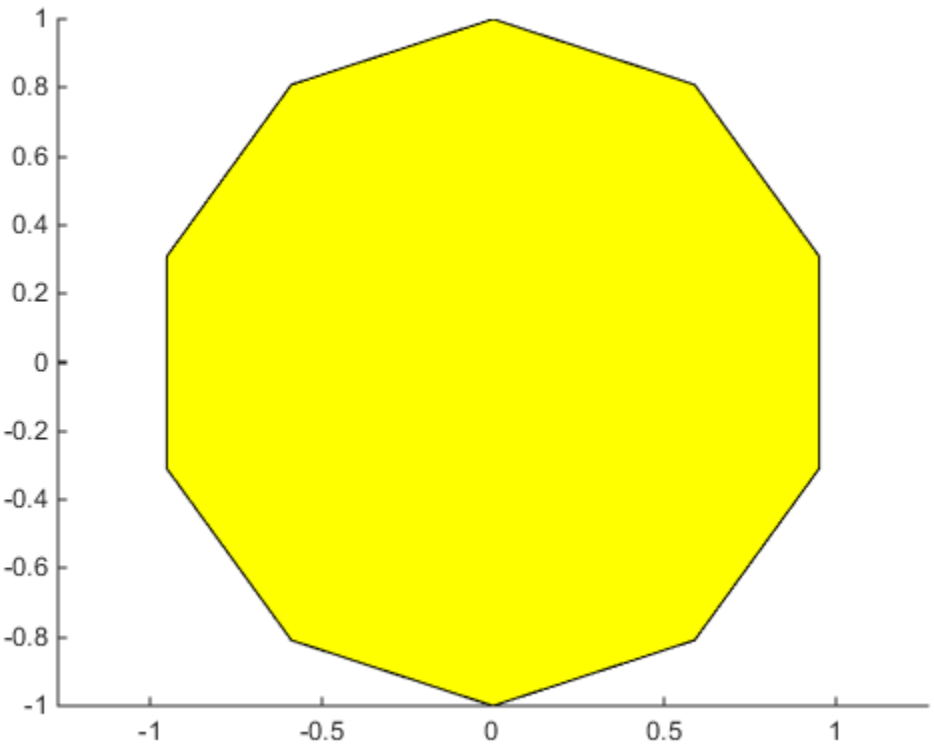
创建单个多边形

多边形是只有一个面的补片。要创建多边形，请按以下形式的语句指定顶点坐标和颜色数据

```
patch(x-coordinates,y-coordinates,[z-coordinates],colordata)
```

例如，下面的语句显示一个面为黄色、边为黑色的十边形。**axis equal** 命令会生成一个正多边形。

```
t = 0:pi/5:2*pi;
figure
patch(sin(t),cos(t),'y')
axis equal
```

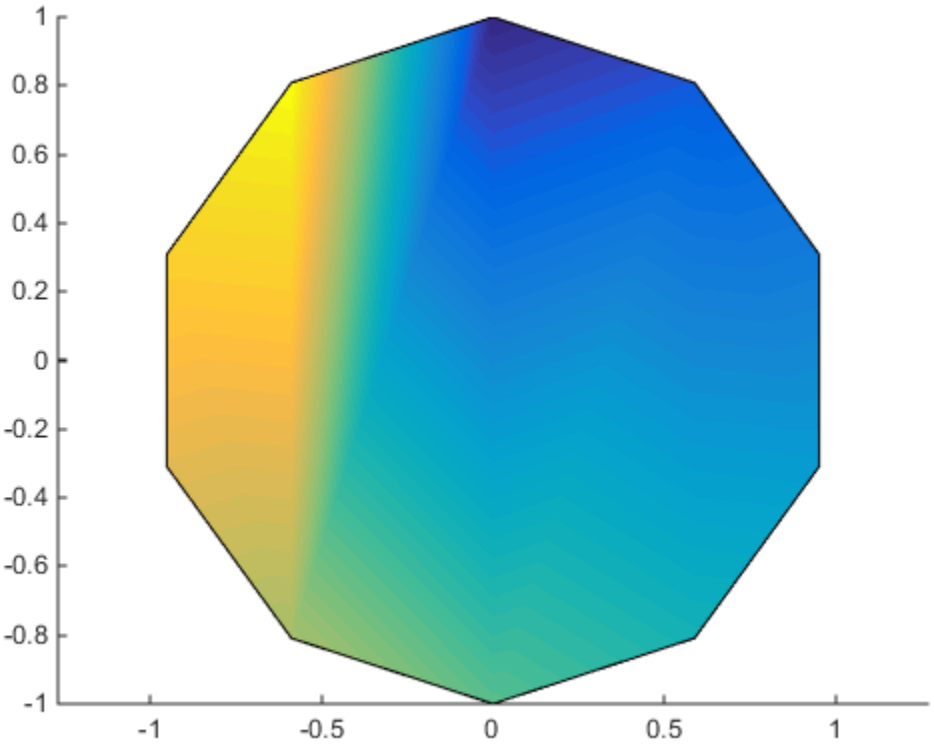


第一个和最后一个顶点不需要重合，MATLAB 会自动闭合补片的每个多边形面。事实上，通常每个顶点最好只定义一次，尤其是在使用插补法对面进行着色时。

对面进行插补着色

您可以控制补片着色的许多方面。例如，您可以提供一系列数值，将每个顶点的颜色映射到图窗颜色图中的一种颜色，而不是只指定一种颜色。

```
a = t(1:length(t)-1); %remove redundant vertex definition
figure
patch(sin(a),cos(a),1:length(a),'FaceColor','interp')
axis equal
```



MATLAB 现在可在补片的整个面上进行颜色插值。同样，您可以将边的颜色指定为插补着色来为补片的边着色。命令如下：

```
patch(sin(a),cos(a),1:length(a),'EdgeColor','interp')
```

“补片数据与颜色图的关系”提供了有关补片着色选项的更多信息。

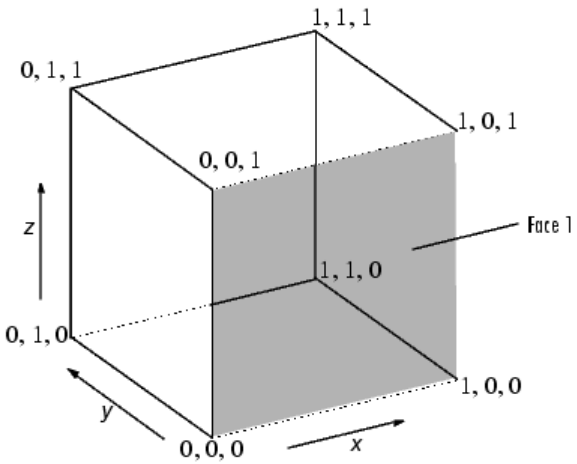
多面补片

示例 - 定义立方体

立方体由八个顶点定义，它们形成六个面。下图显示了定义边长为一个单位的立方体的各个顶点的 x、y 和 z 坐标。

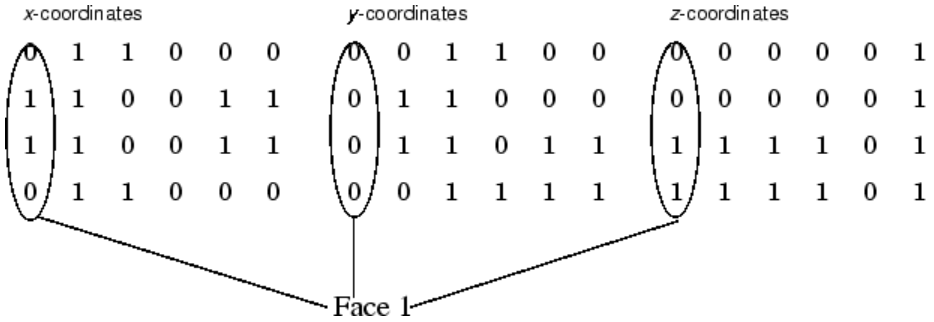
如果您将 x、y 和 z 坐标参数指定为向量，它们将呈现出一个按顺序连接各点的多边形。如果这些参数是矩阵，MATLAB 会为每一列绘制一个多边形，从而形成一个具有多个面的补片。这些面不需要连接，并且可以自相交。

您也可以指定每个唯一顶点的坐标以及将它们连接成面的顺序。本节中的示例对这两种方法均进行了说明。



指定 X、Y 和 Z 坐标

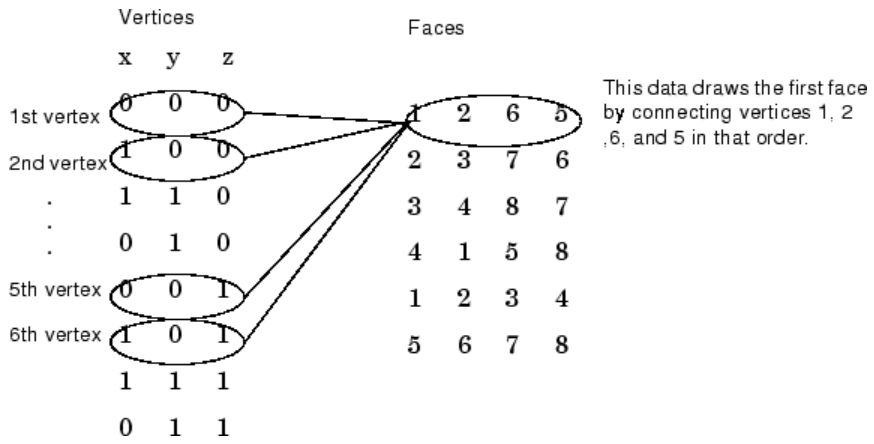
六个面中每个面都有四个顶点。由于不需要闭合每个多边形（即第一个和最后一个顶点不需要相同），因此可以为每个 x、y 和 z 坐标使用一个 4×6 矩阵来定义该立方体。



矩阵的每一列指定一个不同的面。虽然只有八个顶点，但您必须指定 24 个顶点才能定义所有六个面。由于每个面与其他四个面共享顶点，因此您可以每个顶点只定义一次，然后指定连接它们的顺序以形成每个面，这样可以提高定义补片的效率。补片的 **Vertices** 和 **Faces** 属性就是以这种方式定义补片的。

指定面和顶点

这些矩阵使用 **Vertices** 和 **Faces** 指定立方体。



当补片包含很多面时，使用顶点/面方法可以节省大量计算机内存。这种方法需要使用正式的 **patch** 函数语法，为 **Vertices** 和 **Faces** 属性显式赋值。例如，

```
patch('Vertices',vertex_matrix,'Faces',faces_matrix)
```

由于高级语法不会自动分配面或边颜色，因此您必须设置适当的属性，以生成面和边的颜色不是默认的白色和黑色的补片。

面的单一着色

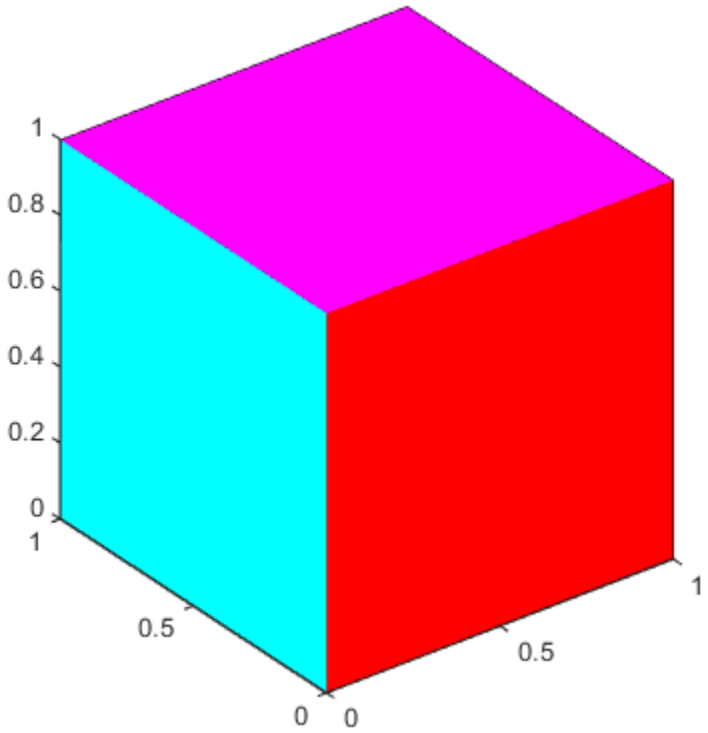
面的单一着色是为每个面指定一种颜色的结果。例如，下面的语句使用顶点/面方法和 **FaceVertexCData** 属性来定义颜色，为每个面指定一种颜色并将 **FaceColor** 属性设置为 **flat**。

```
vert = [0 0 0;1 0 0;1 1 0;0 1 0;0 0 1;1 0 1;1 1 1;0 1 1];
fac = [1 2 6 5;2 3 7 6;3 4 8 7;4 1 5 8;1 2 3 4;5 6 7 8];
patch('Vertices',vert,'Faces',fac,...
    'FaceVertexCData',hsv(6),'FaceColor','flat')
```

调整坐标区：

```
view(3)
axis vis3d
```

由于使用 **FaceVertexCData** 属性指定的真彩色与 MATLAB 颜色图（即由 RGB 值组成的 n×3 数组）的格式相同，因此本示例使用 **hsv** 颜色图生成单一着色所需的六种颜色。

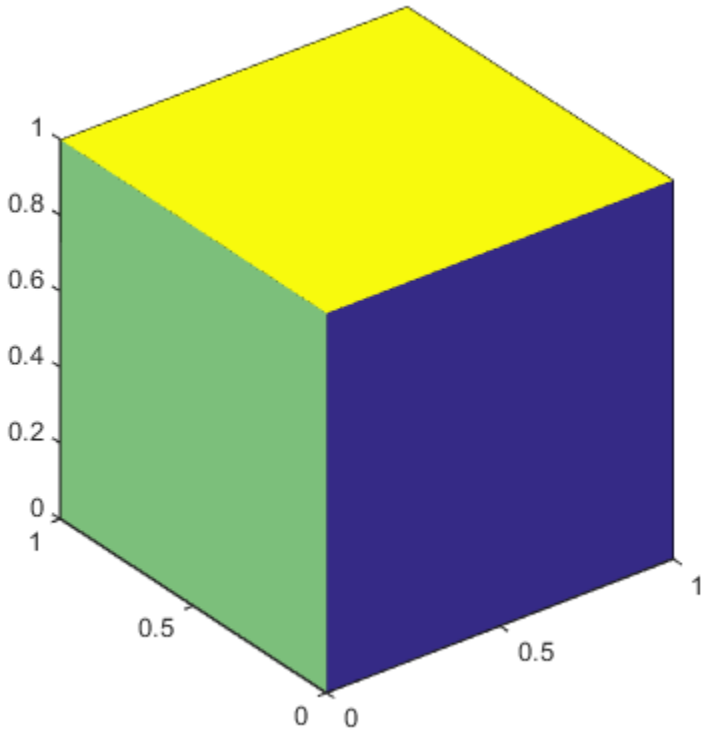


要将面颜色映射到当前颜色图，需要为 **FaceVertexCData** 属性指定一个 n×1 数组：

```
patch('Vertices',vert,'Faces',fac,...
    'FaceVertexCData',(1:6)','FaceColor','flat')
```

调整坐标区：

```
view(3)
axis vis3d
```



面的插补着色

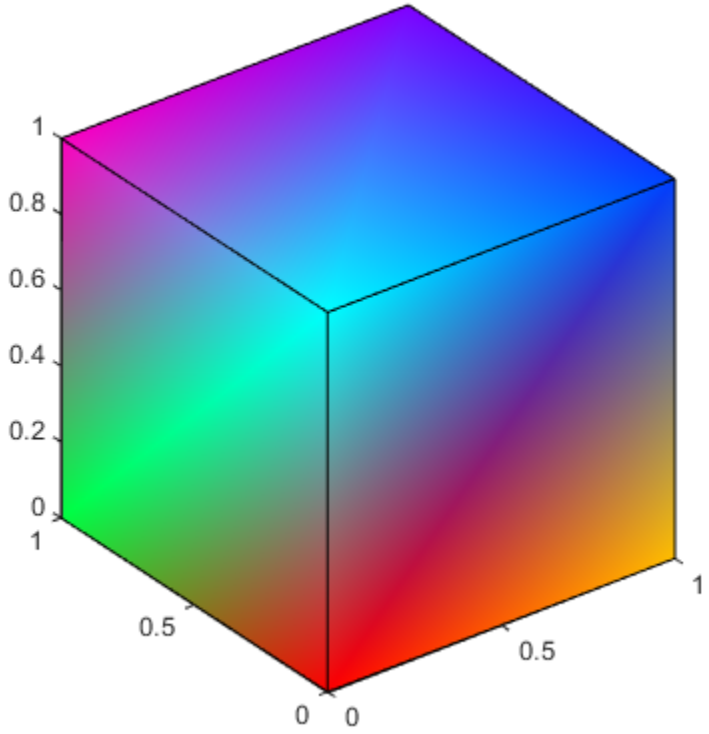
面的插补着色意味着每个面的顶点颜色定义从一个顶点到下一个顶点的颜色过渡。要在顶点之间插补颜色，必须为每个顶点指定一种颜色并将 **FaceColor** 属性设置为 **interp**。

```
patch('Vertices',vert,'Faces',fac,...
    'FaceVertexCData',hsv(8),'FaceColor','interp')
```

调整坐标区：

```
view(3)
axis vis3d
```

生成一个通过顶点的颜色对每个面进行插补着色的立方体。



要使用 x、y、z、c 方法指定相同的着色，c 必须是一个 m×n×3 数组，其中 x、y 和 z 的维度为 m×n。

下图显示了 **FaceVertexCData** 和 **CData** 属性之间的对应关系。

```
FaceVertexCData =
1.00 0.00 0.00
1.00 0.75 0.00
0.50 1.00 0.00
0.00 1.00 0.25
0.00 1.00 1.00
0.00 0.25 1.00
0.50 0.00 1.00
1.00 0.00 0.75
Red Green Blue
```

```
CData(:,:,1) =
1.00 1.00 0.50 0.00 1.00 0.00
1.00 0.50 0.00 1.00 1.00 0.00
0.00 0.50 1.00 0.00 0.50 0.50
0.00 0.00 0.50 1.00 0.00 1.00
CData(:,:,2) =
0.00 0.75 1.00 1.00 0.00 1.00
0.75 1.00 1.00 0.00 0.75 0.25
0.25 0.00 0.00 1.00 1.00 0.00
1.00 0.25 0.00 0.00 1.00 0.00
CData(:,:,3) =
0.00 0.00 0.00 0.25 0.00 1.00
0.00 0.00 0.25 0.00 0.00 1.00
1.00 1.00 0.75 1.00 0.00 1.00
1.00 1.00 1.00 0.75 0.25 0.75
```

Red page

Green page

Blue page

“补片数据与颜色图的关系”中详细介绍了着色方法。

三维可视化

- “三维体可视化概述” （第 3-2 页）
- “可视化标量三维体数据的方法” （第 3-5 页）
- “利用切片平面探索三维体” （第 3-11 页）
- “使用等值面连接相等的值” （第 3-17 页）
- “使用等值顶为可视化绘图添加环境” （第 3-19 页）
- “可视化向量三维体数据” （第 3-23 页）
- “使用向量数据显示流线图” （第 3-27 页）
- “利用流带显示旋度” （第 3-29 页）
- “利用流管显示散度” （第 3-31 页）
- “创建流粒子动画” （第 3-34 页）
- “利用圆锥图显示向量场” （第 3-38 页）
- “可视化体数据” （第 3-41 页）
- “可视化四维数据” （第 3-48 页）
- “显示复杂三维对象” （第 3-54 页）
- “显示地貌数据” （第 3-62 页）

三维体可视化概述

本节内容
“三维体数据示例” （第 3-2 页） “选择可视化方法” （第 3-2 页） “创建三维体可视化绘图的步骤” （第 3-3 页） “三维体可视化函数” （第 3-3 页）

三维体数据示例

三维体可视化是指为三维网格上定义的数据集创建图形表示。三维体数据集的特点是它们是由标量或向量数据组成的多维数组。这些数据通常在网格结构上定义，表示在三维空间采样的值。有两种基本类型的三维体数据：

- 标量三维体数据的每个点包含一个值。
- 向量三维体数据的每个点包含两个或三个值，它们定义一个向量的分量。

flow 生成的数据就是标量三维体数据的一个示例。流数据表示一股浸没射流在一个无限大的水箱内的速度剖面图。键入以下命令

```
[x,y,z,v] = flow;
```

将生成四个三维数组。**x**、**y** 和 **z** 数组指定数组 **v** 中标量值的坐标。

wind 数据集是向量三维体数据的一个示例，这些数据代表北美地区的气流。您可以使用以下命令将这些数据加载到 MATLAB 工作区：

```
load wind
```

此数据集包含六个三维数组：**x**、**y** 和 **z** 是数组 **u**、**v** 和 **w** 的坐标数据，后三个数组是三维体中每个点的向量分量。

选择可视化方法

选择哪一种三维体数据可视化方法取决于您的数据类型以及您要了解的内容。一般来说：

- 标量数据最好用等值面、切片平面和等高线切片进行展示。
- 向量数据表示每个点的模和方向，最好通过流线图（流粒子、流带和流管）、圆锥图和箭头图显示。然而，大多数可视化绘图都综合使用多种方法，以便最好地展示数据的内容。

下面各节的内容介绍如何对典型的三维体数据应用各种方法。

数据插值和网格化数据

利用 MATLAB 提供的函数，您可以对数据插值和调整结构，以做好可视化准备。有关详细信息，请参阅以下章节：

- “插入网格数据”

- “内插散点数据”

创建三维体可视化绘图的步骤

创建有效的可视化绘图需要多个步骤来合成最终的场景。这些步骤基本分为四种：

1 确定数据的特性。绘制三维体数据通常需要了解有关坐标和数据值范围的知识。

2 选择合适的绘图例程。本节的信息可以帮助您选择正确的方法。

3 定义视图。通过仔细合成场景，可以大大地丰富复杂三维图所传达的信息。观察技巧包括调整相机位置、指定纵横比和投影类型、放大或缩小等。

4 添加光照并指定着色。光照是增强曲面形状可见性并为三维体图提供三维透视的有效手段。颜色既可以传达不变的数据值，也可以传达变化的数据值。

三维体可视化函数

MATLAB 提供的函数允许您应用各种三维体可视化方法。下表根据适用的数据类型（标量或向量）将这些函数分为两类。每个函数的参考页提供了函数的用法示例。

适合标量数据的函数

函数	用途
contourslice	在三维体切片平面中绘制等高线
isocaps	计算等值面端帽几何图
isocolors	计算等值面顶点的颜色
isonormals	计算等值面顶点的法向量
isosurface	从三维体数据中提取等值面数据
patch	创建补片（多面多边形）图形对象
reducepatch	减少补片面的数量
reducevolume	减少三维体数据集中的元素个数
shrinkfaces	减少每个补片面的大小
slice	在三维体中绘制切片平面
smooth3	平滑处理三维数据
surf2patch	将曲面数据转换为补片数据
subvolume	提取三维体数据集的子集

适合向量数据的函数

函数	用途
coneplot	在三维向量场中以圆锥体形式绘制速度向量
curl	计算三维向量场的旋度和角速度
divergence	计算三维向量场的散度
interpstreamspeed	根据向量场的模进行流线顶点插值
streamline	根据二维或三维向量数据绘制流线图

函数	用途
streamparticles	根据向量三维体数据绘制流粒子图
streamribbon	根据向量三维体数据绘制流带图
streamslice	根据向量三维体数据绘制间隔合适的流线图
streamtube	根据向量三维体数据绘制流管图
stream2	计算二维流线数据
stream3	计算三维流线数据
volumebounds	返回三维体（标量和向量）的坐标和颜色范围

可视化标量三维体数据的方法

本节内容
“什么是标量三维体数据？” （第 3-5 页）
“显示 MRI 数据的方法” （第 3-5 页）

什么是标量三维体数据?

典型的标量三维体数据由一个三维数据数组和三个维度相同的坐标数组组成。坐标数组指定每个数据点的 x、y 和 z 坐标。

坐标的单位取决于数据的类型。例如，流数据的坐标单位可能是英寸，数据单位是 psi。

许多 MATLAB 函数对于可视化标量数据很有用：

- 切片平面通过将数据值映射到颜色，为了解数据值在三维体内的分布提供了一种途径。您可以按任意角度创建切片平面，也可以使用非平面切片。（有关如何使用切片平面的说明，请参阅 **slice**、三维体切片 （第 3-11 页）示例以及用来显示环境 （第 3-19 页）的切片平面。）您可以指定用来为等值面着色的数据，以便通过颜色和曲面形状显示不同的信息（请参阅 **isocolors**）。
- 等高线切片是在三维体内的特定坐标处绘制的等高线图。等高线图让您能够看到给定平面内哪些位置的数据值是相等的。有关示例，请参阅 **contourslice**。
- 等值面是通过使用等值点作为 **patch** 图形对象的顶点而构成的曲面。

显示 MRI 数据的方法

- “更改数据格式” （第 3-6 页）
- “显示 MRI 数据的图像” （第 3-6 页）
- “显示二维等高线切片” （第 3-7 页）
- “显示三维等高线切片” （第 3-7 页）
- “将等值面应用于 MRI 数据” （第 3-8 页）
- “添加等值顶以显示切割曲面” （第 3-9 页）
- “定义视图” （第 3-9 页）
- “添加光照” （第 3-9 页）

标量数据的示例包括磁共振成像 (MRI) 数据。这种数据通常包含在一个三维体（例如人体）内采集的多个切片平面。MATLAB 包括一个 MRI 数据集，其中包含人类头部的 27 个图像切片。本示例说明适用于 MRI 数据的以下可视化方法：

- 代表头部切片的一系列二维图像 （第 3-6 页）
- 在数据内任意位置采集的二维 （第 3-7 页）和三维 （第 3-7 页）等高线切片
- 显示内部横截面的具有等值顶的等值面 （第 3-8 页）

更改数据格式

MRI 数据 **D** 以 128×128×1×27 数组的形式存储。第三个数组维度通常用于表示图像的颜色数据。但是，由于这些是索引图像（颜色图 **map** 也会加载），因此第三个维度没有信息，可以使用 **squeeze** 命令删除。结果就是一个 128×128×27 数组。

第一步是加载数据并将数据数组从四维转换为三维。

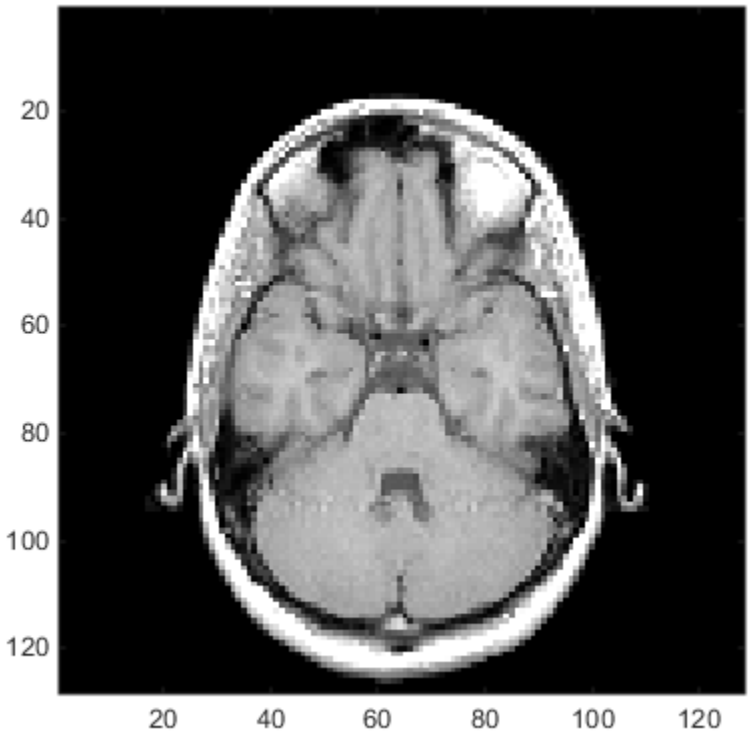
```
load mri
D = squeeze(D);
```

显示 MRI 数据的图像

要显示其中一个 MRI 图像，请使用 **image** 命令：

- 创建一个新的 **figure** 函数，它使用与数据一起加载的 MRI 颜色图：
- 对数据数组进行索引，以获得第八个图像的数据。
- 调整 **axis** 缩放。

```
figure
colormap(map)
image_num = 8;
image(D(:,:,image_num))
axis image
```



保存 x 和 y 坐标轴范围，在本示例的下一部分使用：

```
x = xlim;
y = ylim;
```

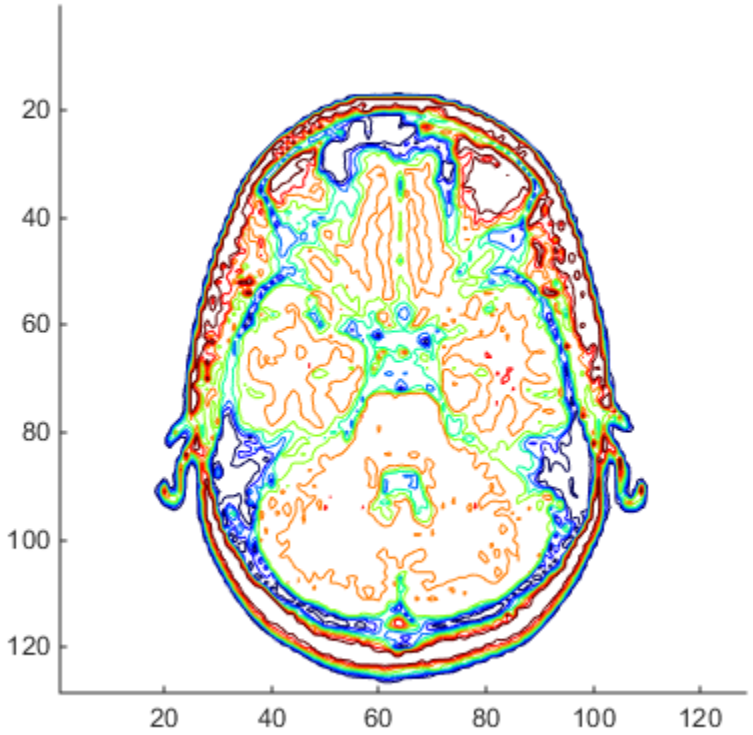
显示二维等高线切片

将 MRI 数据可视化为三维体数据，因为它是从三维对象中逐步采集的切片集合。使用 **contourslice** 显示三维体切片的等高线图。创建与本示例第一部分中的图像具有相同方向和大小的等高线图：

- 调整 y 轴方向 (**axis**)。
- 设置范围（**xlim**、**ylim**）。
- 设置数据纵横比 (**daspect**)。

为了提高细节的可见性，此等高线图使用 **jet** 颜色图。**brighten** 函数可降低颜色值的亮度。

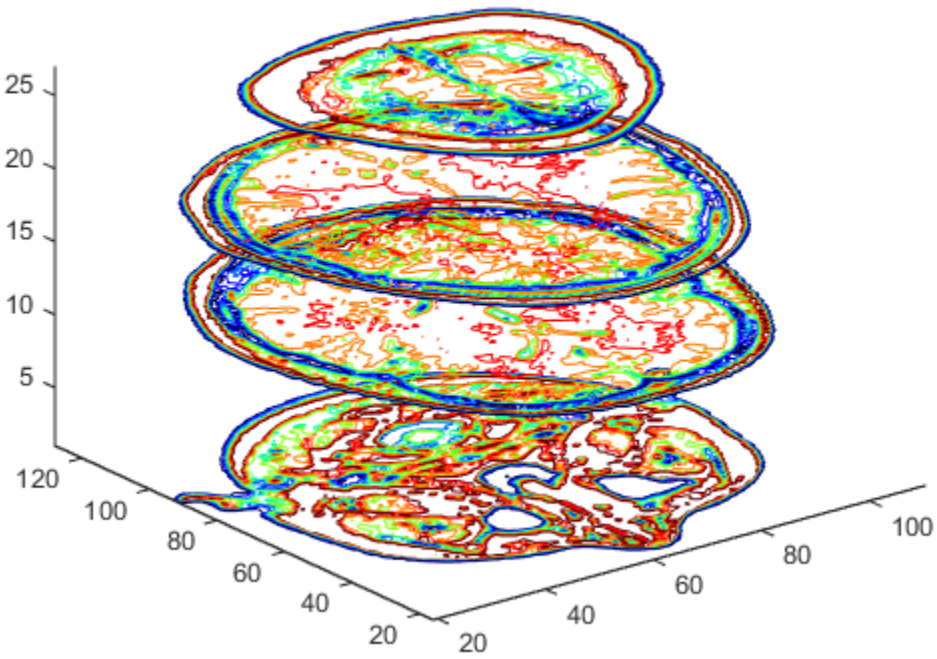
```
cm = brighten(jet(length(map)),-.5);
figure
colormap(cm)
contourslice(D,[],[],image_num)
axis ij
xlim(x)
ylim(y)
daspect([1,1,1])
```



显示三维等高线切片

与作为二维对象的图像不同，等高线切片是可以从任何方向显示的三维对象。例如，您可以在三维视图中显示四个等高线切片。

```
figure
colormap(cm)
contourslice(D,[],[],[1,12,19,27],8);
view(3);
axis tight
```



将等值面应用于 MRI 数据

您可以使用等值面显示三维体的整体结构。如果与等值顶结合使用，这种方法可以揭示等值面内部数据的相关信息。

首先，使用 **smooth3** 对数据进行平滑处理，然后使用 **isosurface** 计算等值数据。使用 **patch** 将这些数据显示在图窗中（对等值顶使用原始灰度颜色图）。

```
figure
colormap(map)
Ds = smooth3(D);
hiso = patch(isosurface(Ds,5),...
  'FaceColor',[1,.75,.65],...
  'EdgeColor','none');
  isonormals(Ds,hiso)
```

isonormals 函数使用从平滑数据获得的顶点法线来渲染等值面，从而提高等值面的质量。等值面使用一种颜色来表示其等值。

添加等值顶以显示切割曲面

使用 **isocaps** 计算另一个补片的数据，该补片与等值面以相同的等值 (**5**) 显示。使用未平滑处理的数据 (**D**) 显示内部细节。您可以将此看作是切掉顶部的头部切割图。位置较低的等值顶在最终视图中不可见。

```
hcap = patch(isocaps(D,5),...
  'FaceColor','interp',...
  'EdgeColor','none');
```

定义视图

定义视图并设置纵横比（**view**、**axis**、**daspect**）。

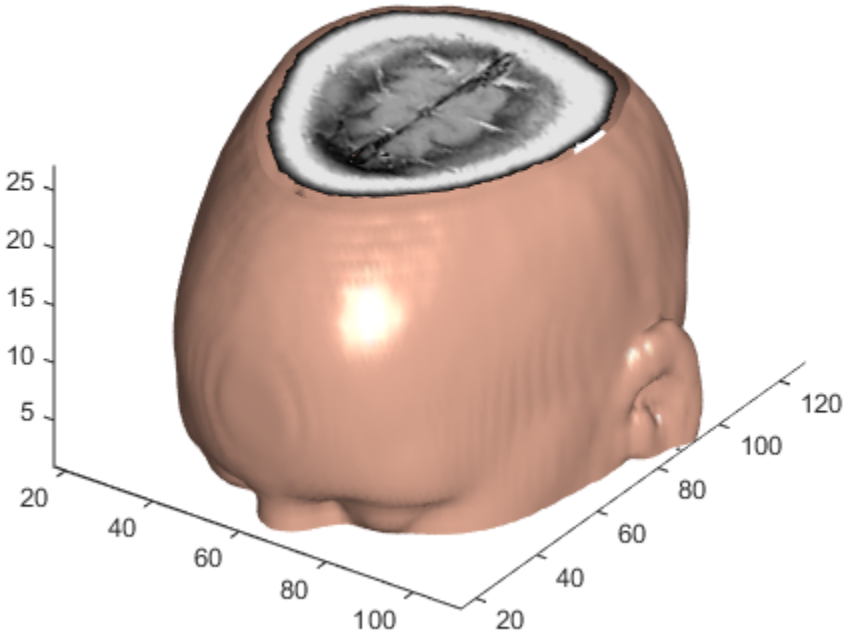
```
view(35,30)
axis tight
daspect([1,1,.4])
```

添加光照

添加光照并根据三维体数据的梯度重新计算曲面法线，从而产生更平滑的光照（**camlight**、**lighting**、**isonormals**）。增大等值顶的 **AmbientStrength** 属性，以增强颜色亮度而不影响等值面。设置等值面的 **SpecularColorReflectance**，使镜面反射光的颜色更接近等值面的颜色，然后设置 **SpecularExponent** 以减小镜面反射区的大小。

```
lightangle(45,30);
lighting gouraud
hcap.AmbientStrength = 0.6;
hiso.SpecularColorReflectance = 0;
hiso.SpecularExponent = 50;
```

将等值顶与等值面相结合，以可视方式呈现 MRI 数据。



等值顶对面使用插补着色，也就是说由图窗的颜色图决定补片的颜色。此示例使用数据附带的颜色图。

要显示其他数据值的等值顶，请尝试更改等值面的值或使用 **subvolume** 命令。有关示例，请参阅 **isocaps** 和 **subvolume** 参考页。

利用切片平面探索三维体

本节内容
“切片液体流数据” （第 3-11 页） “修改颜色映射” （第 3-13 页）

切片液体流数据

切片平面（不一定是平面）是基于切片所在区域中的三维体数据的值进行着色的曲面。切片平面可用于探测三维体数据集，以找到感兴趣的区域位置，然后可以利用其他类型的图对这些区域进行可视化（请参阅 **slice** 示例）。当同时使用其他绘图方法时，切片平面还可用于在三维体范围内添加视觉环境（有关示例，请参阅 **coneplot** 和“使用向量数据显示流线图” （第 3-27 页））。

可使用 **slice** 函数创建切片平面。本示例对 **flow** 生成的三维体进行切片。

1.调查数据

使用以下命令生成三维体数据：

```
[x,y,z,v] = flow;
```

通过查找坐标数据的最小值和最大值，确定三维体的范围。

```
xmin = min(x(:));
ymin = min(y(:));
zmin = min(z(:));

xmax = max(x(:));
ymax = max(y(:));
zmax = max(z(:));
```

2.创建与 X 轴呈一定角度的切片平面

要创建与某个坐标区平面呈一定角度的切片平面，请先定义一个曲面，然后将其旋转到所需角度。本示例使用与三维体具有相同 x 和 y 坐标范围的曲面。

```
hslice = surf(linspace(xmin,xmax,100),...
  linspace(ymin,ymax,100),...
  zeros(100));
```

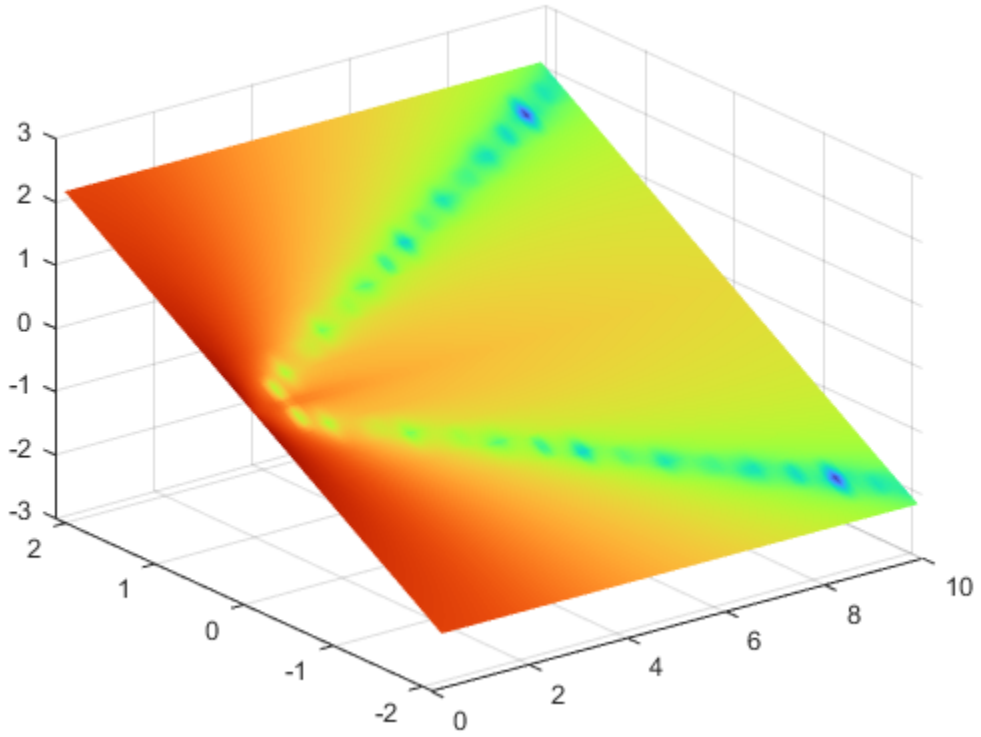
将曲面绕 x 轴旋转 -45 度，保存曲面 **XData**、**YData** 和 **ZData** 以定义切片平面，然后删除曲面。

```
rotate(hslice,[-1,0,0],-45)
xd = get(hslice,'XData');
yd = get(hslice,'YData');
zd = get(hslice,'ZData');

delete(hslice)
```

3.绘制切片平面

绘制旋转后的切片平面，将 **FaceColor** 设置为 **interp** 以便使用图窗颜色图进行着色，并将 **EdgeColor** 设置为 **none**。添加光源后，将 **DiffuseStrength** 增加到 **.8** 以使平面更亮。

```
colormap(turbo)
h = slice(x,y,z,v,xd,yd,zd);
h.FaceColor = 'interp';
h.EdgeColor = 'none';
h.DiffuseStrength = 0.8;
```



将 **hold** 设置为 **on**，并在 **xmax**、**ymax** 和 **zmin** 处再添加三个正交切片平面，形成第一个平面（按一定角度对三维体进行切片所得）的环境。

```
hold on
hx = slice(x,y,z,v,xmax,[],[]);
hx.FaceColor = 'interp';
hx.EdgeColor = 'none';

hy = slice(x,y,z,v,[],ymax,[]);
hy.FaceColor = 'interp';
hy.EdgeColor = 'none';

hz = slice(x,y,z,v,[],[],zmin);
hz.FaceColor = 'interp';
hz.EdgeColor = 'none';
```

4.定义视图

要按正确的比例显示三维体，请将数据纵横比设置为 **[1,1,1]** (**daspect**)。调整坐标轴，使之紧密切合三维体的轮廓 (**axis**)。坐标区的角度可以先行使用 **rotate3d** 进行选择，以确定最佳视图 (**view**)。

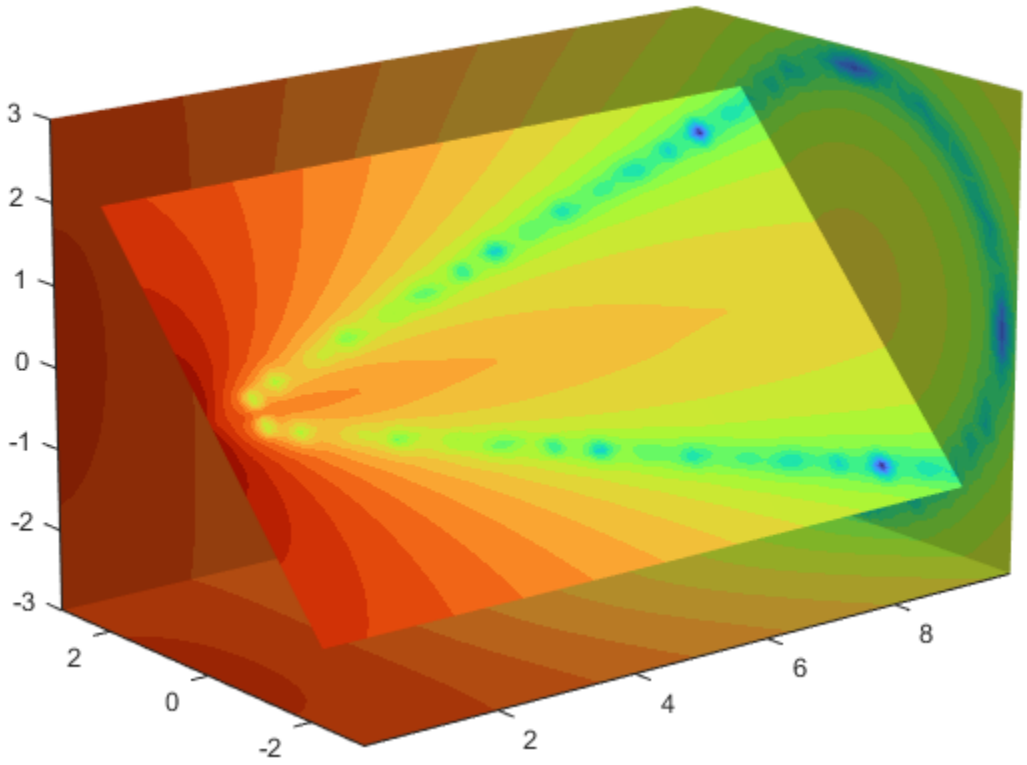
放大场景以获取更大的三维体视图 (**camzoom**)。相比默认的正交投影，选择 **perspective** 投影类型可使长方体的比例更自然 (**camproj**)。

```
daspect([1,1,1])
axis tight
view(-38.5,16)
camzoom(1.4)
camproj perspective
```

5.添加光照并指定颜色

在场景中添加光源可使四个切片平面之间的边界更清晰，因为每个平面与光源之间形成不同的角度 (**lightangle**)。选择只有 24 种颜色的颜色图（默认值为 64）可以产生明显的渐变，从而有助于表现三维体内的变化。

```
lightangle(-45,45)
colormap(turbo(24))
```



“修改颜色映射” （第 3-13 页）介绍了如何修改数据与颜色之间的映射。

修改颜色映射

当前颜色图决定切片平面的颜色。这使您能够通过以下方式更改切片平面的着色：

- 更改颜色图

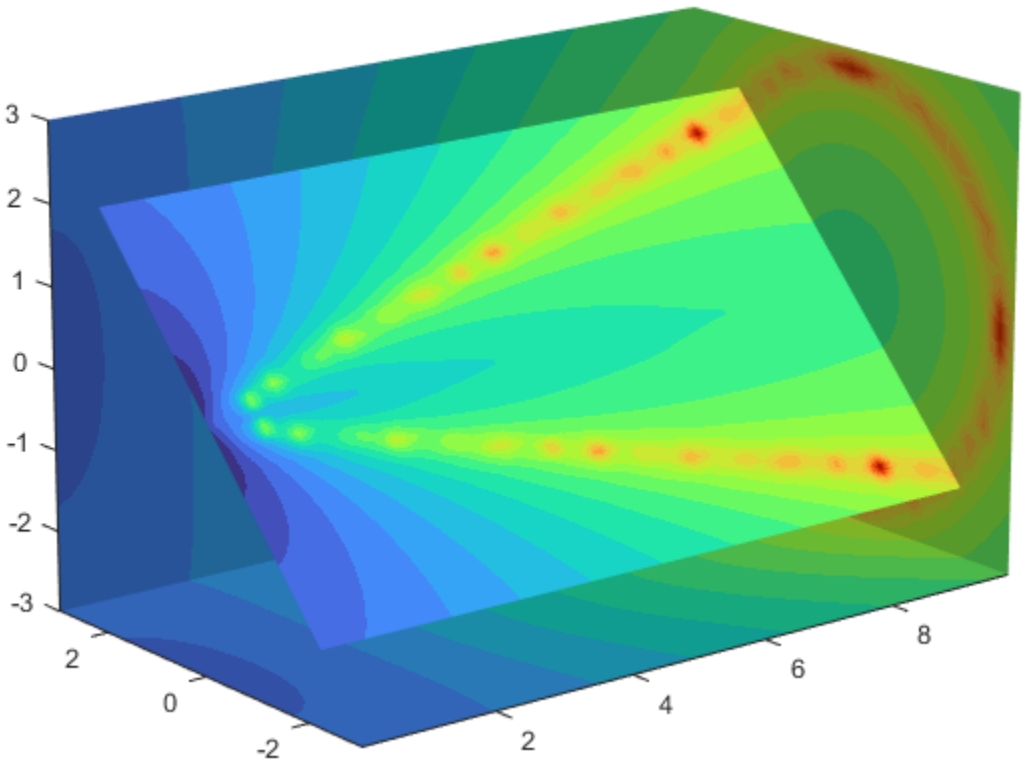
- 更改数据值与颜色之间的映射

例如，假设您只对 -5 到 2.5 之间的数据值感兴趣，并且希望所用的颜色图将低值映射到红色、将高值映射到蓝色（即与默认的 **turbo** 颜色图相反）。

1.自定义颜色图

使用 **colormap** 和 **flipud** 翻转颜色图：

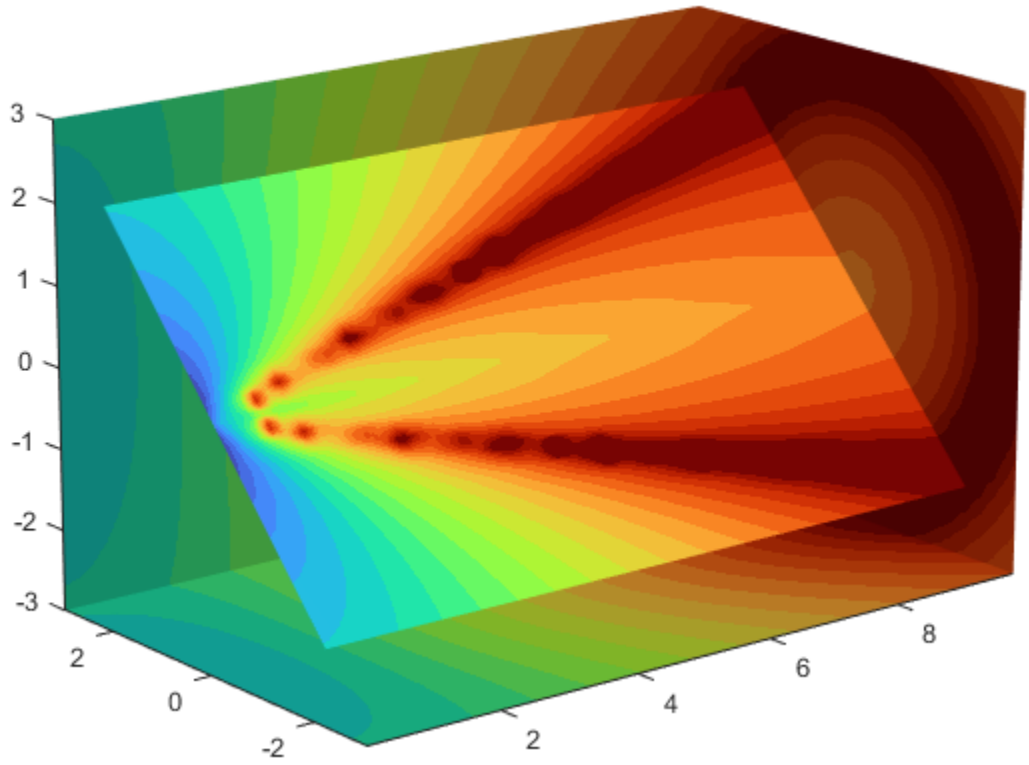
```
colormap(flipud(turbo(24)))
```



2.调整颜色范围

调整颜色范围以突出感兴趣的特定数据范围。将颜色范围调整到 -5 到 2.4832，以便将所有小于 -5 的值映射到同一种颜色（原来的数据范围是从 -11.5417 到 2.4832）。

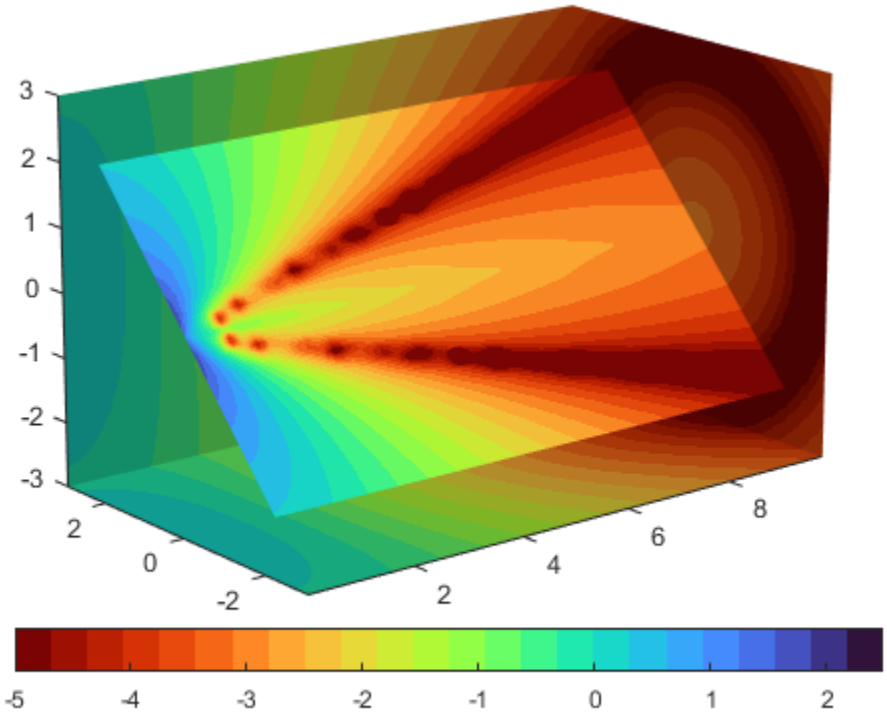
```
caxis([-5,2.4832])
```



3.添加颜色栏

添加颜色栏，以便为数据到颜色的映射提供对照标准。

```
colorbar('southoutside')
```


3
2
1
0
-1
-2
-3
2
0
-2
2
4
6
8
-5
-4
-3
-2
-1
0
1
2

使用等值面连接相等的值

液体流数据中的等值面

可以使用 **isosurface** 和 **patch** 命令创建等值面。

本示例在 **flow** 生成的三维体内创建等值面。使用以下命令生成三维体数据：

```
[x,y,z,v] = flow;
```

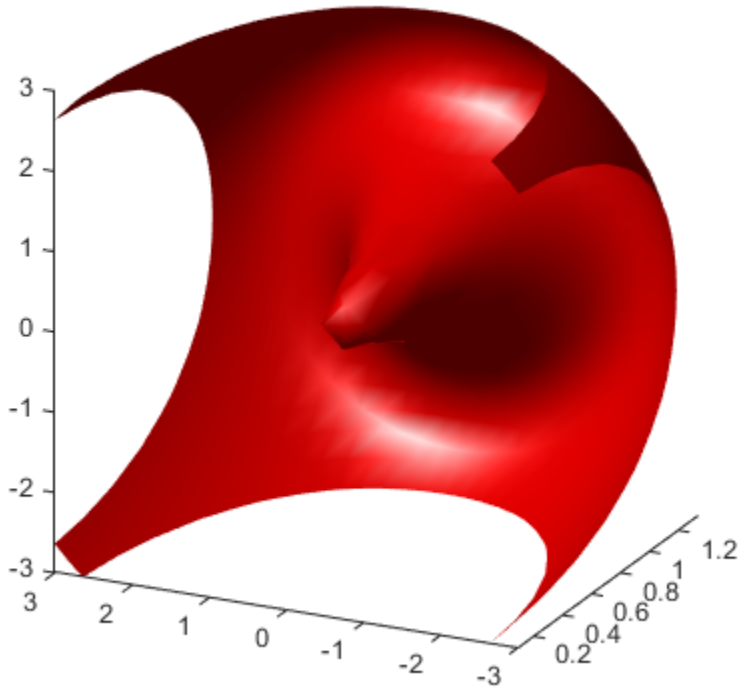
要选择等值，请确定三维体数据中值的范围。

```
min(v(:))
ans =
  -11.5417
max(v(:))
ans =
  2.4832
```

通过研究，您可以选择能够揭示有用数据信息的等值。选择等值后，使用它们创建等值面：

- 使用 **isosurface** 生成可直接传递给 **patch** 的数据。
- 根据三维体数据的梯度重新计算曲面法线，以产生更好的光照特性 (**isonormals**)。
- 将补片的 **FaceColor** 设置为红色并将 **EdgeColor** 设置为 **none**，以生成平滑、明亮的曲面。
- 调整视图并添加光照（**daspect**、**view**、**camlight**、**lighting**）。

```
hpatch = patch(isosurface(x,y,z,v,0));
isonormals(x,y,z,v,hpatch)
hpatch.FaceColor = 'red';
hpatch.EdgeColor = 'none';
daspect([1,4,4])
view([-65,20])
axis tight
camlight left;
lighting gouraud
```


3
2
1
0
-1
-2
-3
3
2
1
0
-1
-2
-3
0.2
0.4
0.6
0.8
1
1.2

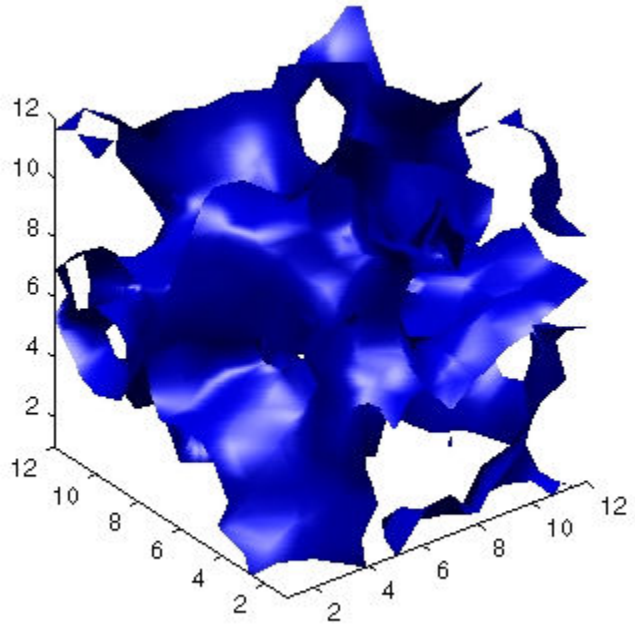
使用等值顶为可视化绘图添加环境

本节内容
“什么是等值顶？” （第 3-19 页） “等值顶的其他应用” （第 3-20 页） “定义等值顶” （第 3-20 页） “为等值面添加等值顶” （第 3-20 页）

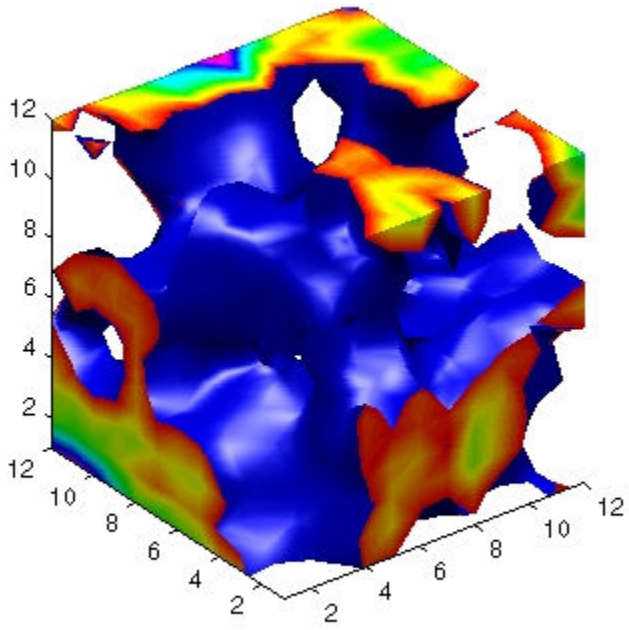
什么是等值顶?

等值顶是切合等值面范围的平面，为等值面提供视觉环境。等值顶显示等值面内部的横截面视图，它为等值面提供一个端顶。

下面两张图说明了等值顶的用法。第一张图是没有等值顶的等值面。



第二张图显示了为同一个等值面添加等值顶后的效果。



等值顶的其他应用

以下示例显示了等值顶的其他一些应用：

- 使用等值顶显示剖面三维体的内部。
- 使用等值顶截出三维体原本不显示的端面。 （第 3-8 页）
- 使用等值顶增强等值面范围的可见性。 （第 3-19 页）

定义等值顶

等值顶与等值面一样，都是以 **patch** 图形对象的形式创建的。可以使用 **isocaps** 命令生成要传递给 **patch** 的数据。例如：

```
patch(isocaps(voldata,isoval),...
   'FaceColor','interp',...
...'EdgeColor','none')
```

在值 **isoval** 处为标量三维体数据 **voldata** 创建等值顶。您应该使用相同的三维体数据和等值来创建等值面，以确保等值顶的边切合等值面。

如果将补片的 **FaceColor** 属性设置为 **interp**，则着色方案会将等值顶经过的数据值映射到颜色图条目。您还可以设置其他补片属性来控制等值顶上的光照和着色效果。

为等值面添加等值顶

此示例说明如何在使用等值顶时设置着色和光照特性。有五个基本步骤：

1 生成并处理三维体数据。 （第 3-21 页）
2 创建等值面和等值顶并设置补片属性，以控制着色和光照。 （第 3-21 页）
3 创建等值顶并设置属性。 （第 3-21 页）
4 指定视图。 （第 3-21 页）
5 为场景添加光照。 （第 3-13 页）

1.准备数据

此示例使用随机 (**rand**) 数据的三维数组来定义三维体数据。然后对数据进行平滑处理 (**smooth3**)。

```
data = rand(12,12,12);
data = smooth3(data,'box',5);
```

2.创建等值面并设置属性

使用 **isosurface** 和 **patch** 创建等值面并设置着色和光照属性。减少反射光的 **AmbientStrength**、**SpecularStrength** 和 **DiffuseStrength**，以补偿为了提供更均匀的光照而使用的两个光源的亮度。

重新计算等值面的顶点法线，以产生更平滑的光照 (**isonormals**)。

```
isoval = .5;
h = patch(isosurface(data,isoval),...
  'FaceColor','blue',...
  'EdgeColor','none',...
  'AmbientStrength',.2,...
  'SpecularStrength',.7,...
  'DiffuseStrength',.4);
isonormals(data,h)
```

3.创建等值顶并设置属性

使用与等值面相同的数据和等值定义 **isocaps**。指定插补着色并选择合适的颜色图，从而为蓝色等值面提供较默认颜色图 (**colormap**) 更好的对比色。

```
patch(isocaps(data,isoval),...
  'FaceColor','interp',...
  'EdgeColor','none')
colormap hsv
```

4.定义视图

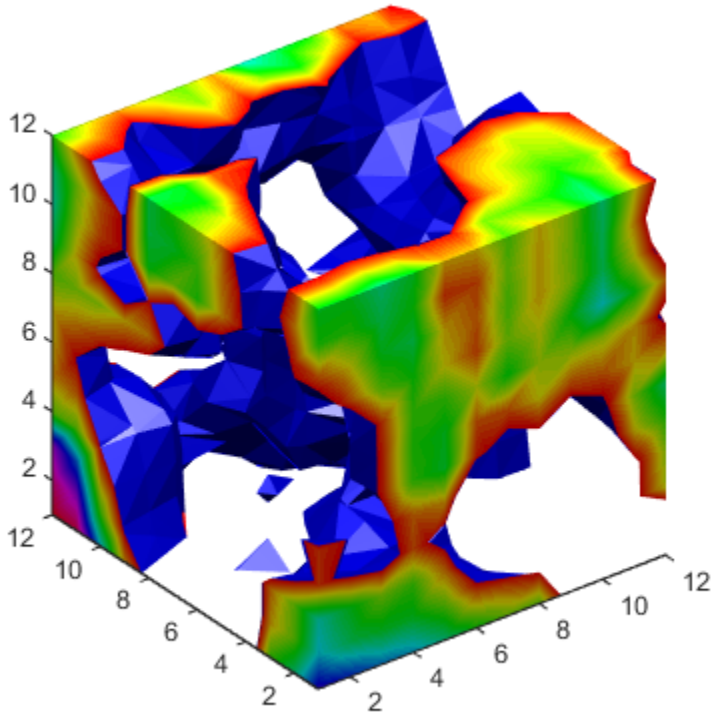
将数据纵横比设置为 **[1,1,1]**，以便按正确的比例显示 (**daspect**)。消除坐标区内的空白并将视图设置为三维（**axis tight**、**view**）。

```
daspect([1,1,1])
axis tight
view(3)
```

5.添加光照

为了添加非常均匀的光照，同时又能利用光源在形状上制造可见的细微变化的能力，本示例使用了两个光源，分别位于相机的左右两侧 (**camlight**)。使用 Gouraud 光照产生最平滑的颜色变化 (**lighting**)。

```
camlight right
camlight left
```


12
10
8
6
4
2
12
10
8
6
4
2
2
4
6
8
10
12

可视化向量三维体数据

本节内容
“流线、流粒子、流带、流、流管和圆锥” （第 3-23 页） “对向量数据使用标量方法” （第 3-23 页） “指定流线图的起点” （第 3-23 页） “访问三维体数据的子区域” （第 3-26 页）

流线、流粒子、流带、流、流管和圆锥

向量三维体数据比标量数据包含更多的信息，因为数据集中的每个坐标点都有三个关联的值。这些值定义了表示模和方向的向量。液体流速就是向量数据的一个示例。

有许多方法可用于可视化向量数据：

- 流线跟踪沉浸在向量场中的无质量粒子的轨迹。
- 流粒子是跟踪流线的标记，可用于制作流线动画。
- 流带类似于流线，只不过它们可以借助带的宽度指示扭曲。流带可用于指示旋度角速度。
- 流管类似于流线，但您也可以控制管的宽度。流管可用于显示向量场的散度。
- 圆锥图通过显示圆锥形箭尖或普通箭尖来表示每个点处数据的模和方向。

通常，这些函数要与其他可视化方法（如等高线、切片平面和等值面）结合使用，才能最好地表现数据。本节中的示例说明了其中一些方法。

对向量数据使用标量方法

等高线切片、切片平面和等值面等可视化方法需要标量三维体数据。通过获取向量的模，您可以将这些方法用于向量数据。例如，**wind** 数据集返回三个坐标数组和三个向量分量数组 **u**、**v**、**w**。在这种情况下，速度向量的模等于三维体中每个对应坐标点处的风速。

```
wind_speed = sqrt(u.^2 + v.^2 + w.^2);
```

数组 **wind_speed** 包含三维体数据的标量值。但是，这种方法产生的信息是否有用取决于向量数据的模表示什么样的物理现象。

指定流线图的起点

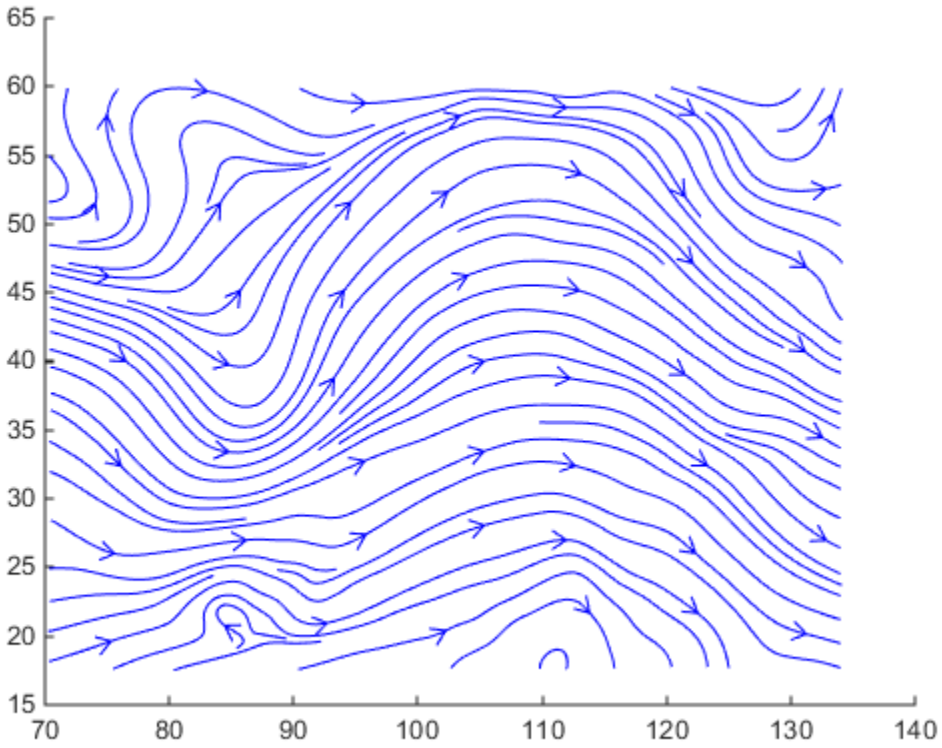
流线图（流线、流带、流管以及圆锥体或箭头）描绘三维向量场的流动。MATLAB 流绘图函数（**streamline**、**streamribbon**、**streamtube**、**coneplot**、**stream2**、**stream3**）都需要您指定每个流跟踪的起点。

确定起点

一般而言，了解数据的特性有助于您选择起点。主要流向以及数据坐标范围等信息可以帮助您决定在何处计算数据。

streamslice 函数对于研究数据非常有用。例如，下面这些语句绘制了向量场范围中间某个 z 值处的切片。

```
load wind
zmax = max(z(:)); zmin = min(z(:));
streamslice(x,y,z,u,v,w,[],[],(zmax-zmin)/2)
```



这个流切片图指明流的方向为正 x，并且您可以在 x 和 y 方向选择起点。您可以在 x-z 平面或 y-z 平面创建类似的三维体切片图，以便深入了解数据的范围和方向。

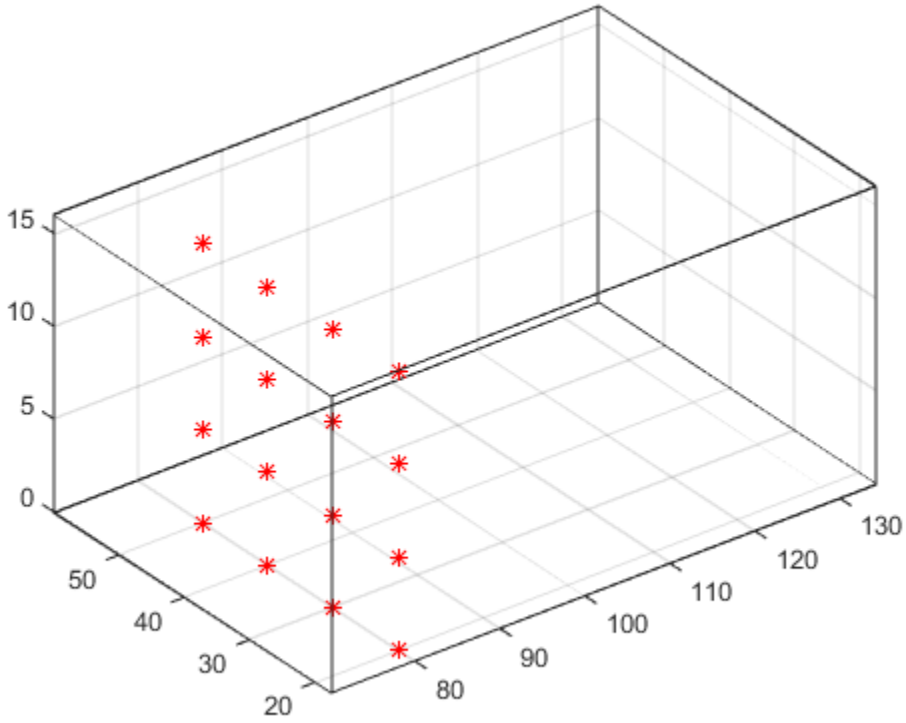
指定起点坐标数组

要指定一条流线的起点，您需要知道该点的 x、y 和 z 坐标。**meshgrid** 命令为创建起点数组提供了便捷的途径。例如，您可以从上一个流切片图显示的风数据中选择以下起点。

```
[sx,sy,sz] = meshgrid(80,20:10:50,0:5:15);
```

此语句将起点定义为全部位于 x = 80 处，y 的范围从 20 到 50，z 的范围从 0 到 15。您可以使用 **plot3** 显示这些位置。

```
plot3(sx(:),sy(:),sz(:),'*r');
axis(volumebounds(x,y,z,u,v,w))
grid on
set(gca,'BoxStyle','full','Box','on')
daspect([2 2 1])
```



您不需要使用三维数组（比如 **meshgrid** 返回的数组），但每个数组的大小必须相同，而 **meshgrid** 的便捷之处在于它可以在各个坐标的唯一值数量不同时生成数组。您还可以将起点数组定义为列向量。例如，**meshgrid** 将返回三维数组：

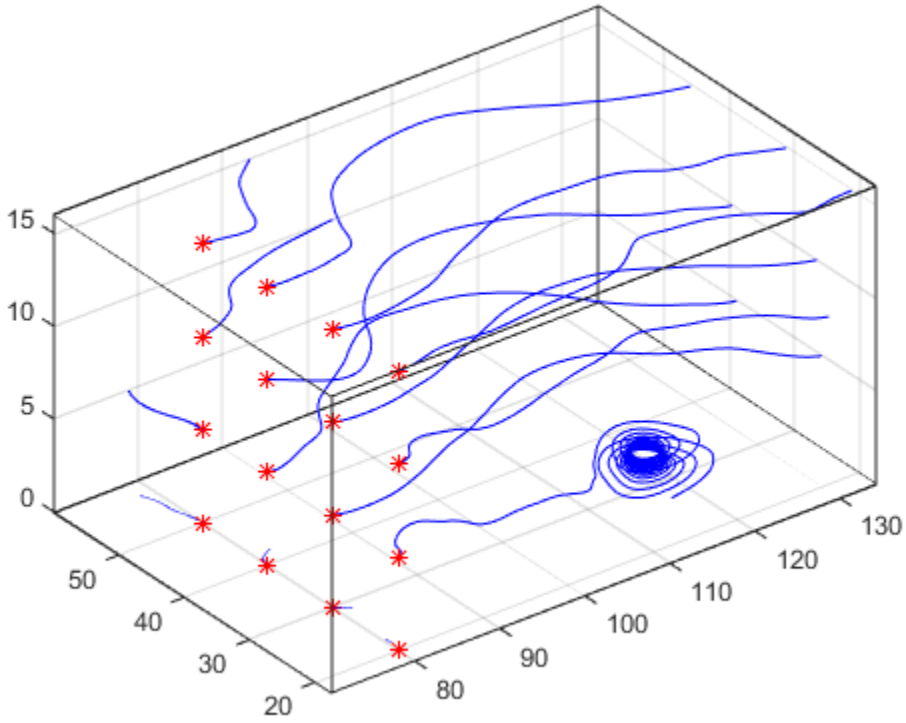
```
[sx,sy,sz] = meshgrid(80,20:10:50,0:5:15);
whos
```

```
  Name      Size             Bytes  Class     Attributes

  sx        4x1x4              128  double
  sy        4x1x4              128  double
  sz        4x1x4              128  double
```

此外，您还可以使用 16×1 列向量，由三个数组的相应元素构成每个起点的坐标。（相当于将 **meshgrid** 返回的值索引为 **sx(:)**、**sy(:)** 和 **sz(:)**。）

例如，在起点后面添加流线，将生成：

```
streamline(x,y,z,u,v,w,sx(:),sy(:),sz(:))
```



访问三维体数据的子区域

subvolume 函数为访问三维体数据集的子区域提供了简便的方法。**subvolume** 允许您根据范围选择感兴趣的区域，而不用使用冒号运算符对定义三维体的三维数组进行索引。请考虑以下两种创建子三维体数据的方法 - 使用冒号运算符进行索引，以及使用 **subvolume**。

使用冒号运算符进行索引

当您对数组进行索引时，您处理的是值，它们指定数组每个维度的各个元素。

```
load wind
xsub = x(1:10,20:30,1:7);
ysub = y(1:10,20:30,1:7);
zsub = z(1:10,20:30,1:7);
usub = u(1:10,20:30,1:7);
vsub = v(1:10,20:30,1:7);
wsub = w(1:10,20:30,1:7);
```

使用 subvolume 函数

subvolume 允许您使用可从坐标区读取的坐标值。例如：

```
lims = [100.64 116.67 17.25 28.75 -0.02 6.86];
[xsub,ysub,zsub,usub,vsub,wsub] = subvolume(x,y,z,u,v,w,lims);
```

然后您可以使用子三维体数据作为需要向量三维体数据的任何函数的输入。

使用向量数据显示流线图

本节内容
“风的映射数据” （第 3-27 页） “1.确定坐标的范围” （第 3-27 页） “2.添加切片平面以提供视觉环境” （第 3-27 页） “3.在切片平面上添加等高线” （第 3-27 页） “4.定义流线的起点” （第 3-28 页） “5.定义视图” （第 3-28 页）

风的映射数据

MATLAB 向量数据集 **wind** 代表北美地区的气流。本示例结合使用了几种方法：

- 利用流线跟踪风速
- 利用切片平面显示数据的横截面视图
- 利用切片平面上的等高线提高切片平面着色的可见性

1.确定坐标的范围

加载数据并确定用来定位切片平面和等高线图的最小值和最大值（**load**、**min**、**max**）。

```
load wind
xmin = min(x(:));
xmax = max(x(:));
ymax = max(y(:));
zmin = min(z(:));
```

2.添加切片平面以提供视觉环境

计算向量场的模（代表风速），以生成用于 **slice** 命令的标量数据。沿 x 轴在 **xmin**、**100** 和 **xmax** 处、沿 y 轴在 **ymax** 处以及沿 z 轴在 **zmin** 处创建切片平面。指定插值面着色以使切片着色指示风速，但不绘制边（**sqrt**、**slice**、**FaceColor**、**EdgeColor**）。

```
wind_speed = sqrt(u.^2 + v.^2 + w.^2);
hsurfaces = slice(x,y,z,wind_speed,[xmin,100,xmax],ymax,zmin);
set(hsurfaces,'FaceColor','interp','EdgeColor','none')
colormap turbo
```

3.在切片平面上添加等高线

在切片平面上绘制浅灰色等高线以帮助量化颜色映射（**contourslice**、**EdgeColor**、**LineWidth**）。

```
hcont = ...
contourslice(x,y,z,wind_speed,[xmin,100,xmax],ymax,zmin);
set(hcont,'EdgeColor',[0.7 0.7 0.7],'LineWidth',0.5)
```

4.定义流线的起点

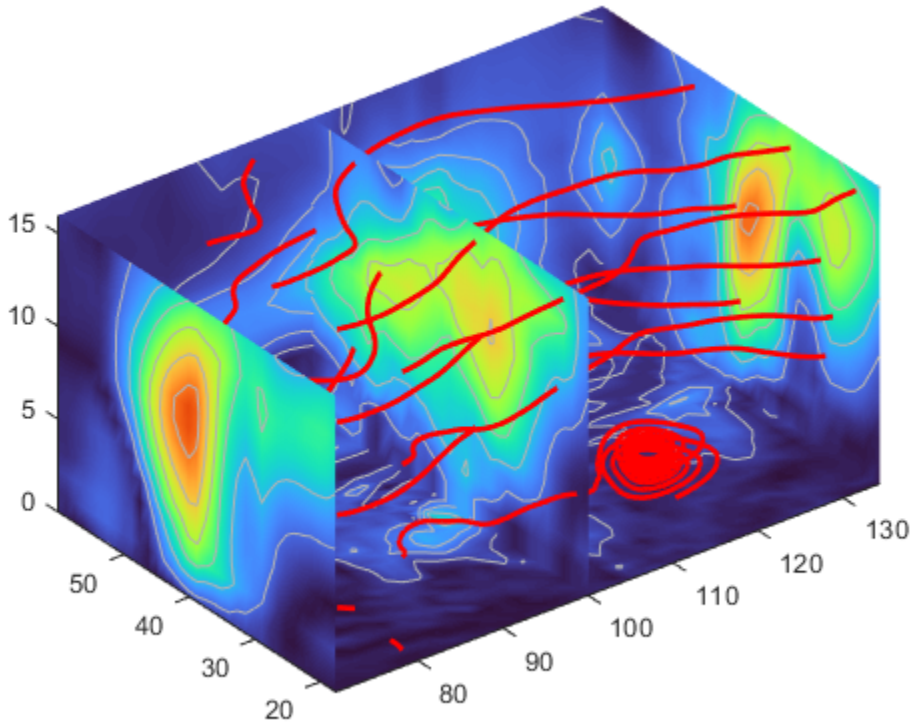
在本示例中，所有流线都从 x 轴上的值 80 处开始，在 y 方向上的范围为 20 到 50，在 z 方向上的范围为 0 到 15。保存流线的句柄并设置线宽和颜色（**meshgrid**、**streamline**、**LineWidth**、**Color**）。

```
[sx,sy,sz] = meshgrid(80,20:10:50,0:5:15);
hlines = streamline(x,y,z,u,v,w,sx,sy,sz);
set(hlines,'LineWidth',2,'Color','r')
```

5.定义视图

设置视图，扩展 z 轴以便于观察图形（**view**、**daspect**、**axis**）。

```
view(3)
daspect([2,2,1])
axis tight
```



有关使用圆锥体绘制相同数据的示例，请参阅 **coneplot**。

利用流带显示旋度

本节内容
“流带可以显示哪些信息” （第 3-29 页） “1.选择要绘制的数据子集” （第 3-29 页） “2.计算旋度角速度和风速” （第 3-29 页） “3.创建流带” （第 3-29 页） “4.定义视图并添加光照” （第 3-30 页）

流带可以显示哪些信息

与流线类似，流带可以表明流的方向，但通过扭曲带状流线，流带还可以显示围绕流坐标轴的旋转。**streamribbon** 函数允许您为流带中的每个顶点指定扭曲角度（以弧度为单位）。

与 **curl** 函数结合使用时，**streamribbon** 可用于显示向量场的旋度角速度。下例演示了这一技术。

1.选择要绘制的数据子集

加载 **wind** 数据集并使用 **subvolume** 选择关注区域。先绘制完整数据集可以帮助您选择关注区域。

```
load wind
lims = [100.64 116.67 17.25 28.75 -0.02 6.86];
[x,y,z,u,v,w] = subvolume(x,y,z,u,v,w,lims);
```

2.计算旋度角速度和风速

计算旋度角速度和风速。

```
cav = curl(x,y,z,u,v,w);
wind_speed = sqrt(u.^2 + v.^2 + w.^2);
```

3.创建流带

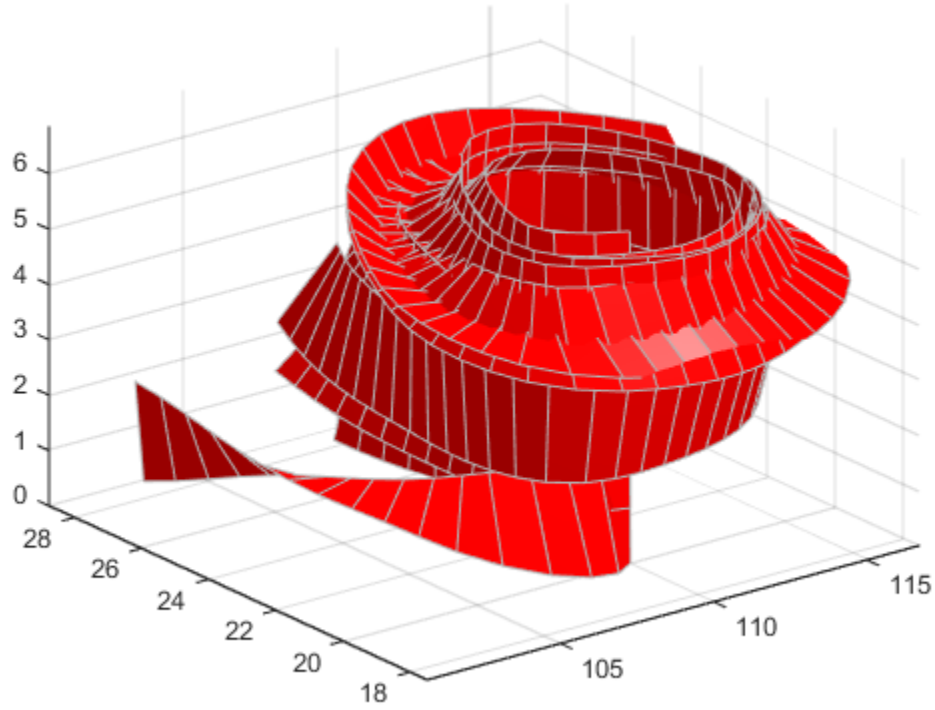
- 使用 **meshgrid** 为流带创建起点数组。有关指定起点数组的信息，请参阅“指定流线图的起点” （第 3-23 页）。
- **stream3** 以 **.5** 为步长计算流线顶点。
- **streamribbon** 按因子 **2** 缩放流带宽度，以提高扭曲（表明旋度角速度）的可见性。
- **streamribbon** 返回它创建的曲面对象的句柄，然后使用它们将曲面颜色设置为红色 (**FaceColor**)、将曲面边的颜色设置为浅灰色 (**EdgeColor**)，并稍微提高应用光照后反射的环境光的亮度 (**AmbientStrength**)。

  ```
  [sx sy sz] = meshgrid(110,20:5:30,1:5);
  verts = stream3(x,y,z,u,v,w,sx,sy,sz,.5);
  h = streamribbon(verts,x,y,z,cav,wind_speed,2);
  set(h,'FaceColor','r',...
    'EdgeColor',[.7 .7 .7],...
    'AmbientStrength',.6)
  ```

4.定义视图并添加光照

- **volumebounds** 命令为设置 **axis** 和颜色范围提供了便捷的途径。
- 添加 **grid** 并将 **view** 设置为三维（**streamribbon** 不会更改当前视图）。
- **camlight** 在视点右侧创建光源，**lighting** 将光照方法设置为 Gouraud。

```
axis(volumebounds(x,y,z,wind_speed))
grid on
view(3)
camlight right;
```



利用流管显示散度

本节内容
“流管可以显示哪些信息” （第 3-31 页） “1.加载数据并计算所需的值” （第 3-31 页） “2.绘制切片平面” （第 3-31 页） “3.在切片平面上添加等高线” （第 3-32 页） “4.创建流管” （第 3-32 页） “5.定义视图” （第 3-32 页）

流管可以显示哪些信息

流管类似于流线，只不过流管具有宽度，为表示信息提供了另外一个维度。

默认情况下，MATLAB 图形通过流管的宽度显示向量场的散度。您还可以为每个流管顶点定义宽度，从而将其他数据映射到宽度。

本示例使用以下方法：

- 利用流管指示 **wind** 数据集中向量场的流向和散度
- 利用着色的切片平面指示叠加的风流的速度，并利用等高线提高可见性

输入项包括三维体坐标、向量场分量以及流管的起点位置。

1.加载数据并计算所需的值

加载数据并计算绘图所需的值。这些值包括：

- 切片平面的位置（最大 **x** 值、最小 **y** 值和海拔值）
- 流管起点的最小 **x** 值
- 风速（向量场的模）

```
load wind
xmin = min(x(:));
xmax = max(x(:));
ymin = min(y(:));
alt = 7.356; % z value for slice and streamtube plane
wind_speed = sqrt(u.^2 + v.^2 + w.^2);
```

2.绘制切片平面

绘制切片平面 (**slice**) 并设置 **surface** 属性以创建平滑着色的切片。使用 **hsv colormap** 中的 16 种颜色。

```
hslice = slice(x,y,z,wind_speed,xmax,ymin,alt);
set(hslice,'FaceColor','interp','EdgeColor','none')
colormap hsv(16)
```

3.在切片平面上添加等高线

在切片平面上添加等高线 (**contourslice**)。调整等高线间隔，使线条与切片平面上的颜色边界匹配：

- 调用 **caxis** 以获取当前颜色范围。
- 将 **contourslice** 使用的插值方法设置为 **linear**，以便与 **slice** 使用的默认值匹配。

```
color_lim = caxis;
cont_intervals = linspace(color_lim(1),color_lim(2),17);
hcont = contourslice(x,y,z,wind_speed,xmax,ymin,...
  alt,cont_intervals,'linear');
set(hcont,'EdgeColor',[.4 .4 .4],'LineWidth',1)
```

4.创建流管

使用 **meshgrid** 创建流管起点数组，起点从最小 **x** 值开始，在 **y** 方向上的范围为 20 到 50，并位于 **z** 方向上的单个平面中（对应于其中一个切片平面）。

流管 (**streamtube**) 绘制在指定的位置，并放大为默认宽度的 1.25 倍，以突出散度（宽度）的变化。向量 [1.25 30] 中的第二个元素指定流管周长上的点数（默认值为 20）。随着流管大小的增加，您可能需要增加此值的大小，以保持光滑的流管外观。

在调用 **streamtube** 之前设置数据纵横比 (**daspect**)。

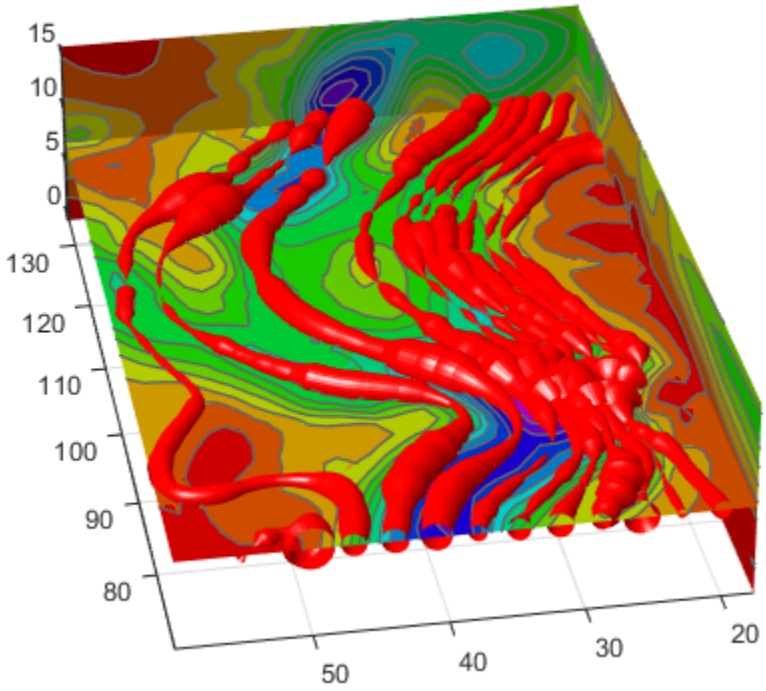
流管是曲面对象，因此您可以通过设置曲面属性来控制其外观。本示例通过设置曲面属性获得明亮的红色曲面。

```
[sx,sy,sz] = meshgrid(xmin,20:3:50,alt);
daspect([1,1,1]) % set DAR before calling streamtube
htubes = streamtube(x,y,z,u,v,w,sx,sy,sz,[1.25 30]);
set(htubes,'EdgeColor','none','FaceColor','r',...
  'AmbientStrength',.5)
```

5.定义视图

定义视图并添加光照（**view**、**axis volumebounds**、**Projection**、**camlight**）。

```
view(-100,30)
axis(volumebounds(x,y,z,wind_speed))
set(gca,'Projection','perspective')
camlight left
```


15
10
5
0
130
120
110
100
90
80
50
40
30
20

创建流粒子动画

本节内容
“随时间变化的抛射体路径” （第 3-34 页） “粒子动画可以显示哪些信息” （第 3-35 页） “1.指定数据范围的起点” （第 3-35 页） “2.创建流线以指示粒子路径” （第 3-35 页） “3.定义视图” （第 3-35 页） “4.计算流粒子顶点” （第 3-36 页）

随时间变化的抛射体路径

此示例演示如何使用三维箭头图将抛射体路径显示为时间函数。

使用速度和加速度常量 **vz** 与 **a** 显示如下抛射体路径。计算时间从 0 到 1 的变化过程中高度 **z** 的值。

$$z(t) = v_z t + \frac{at^2}{2}$$

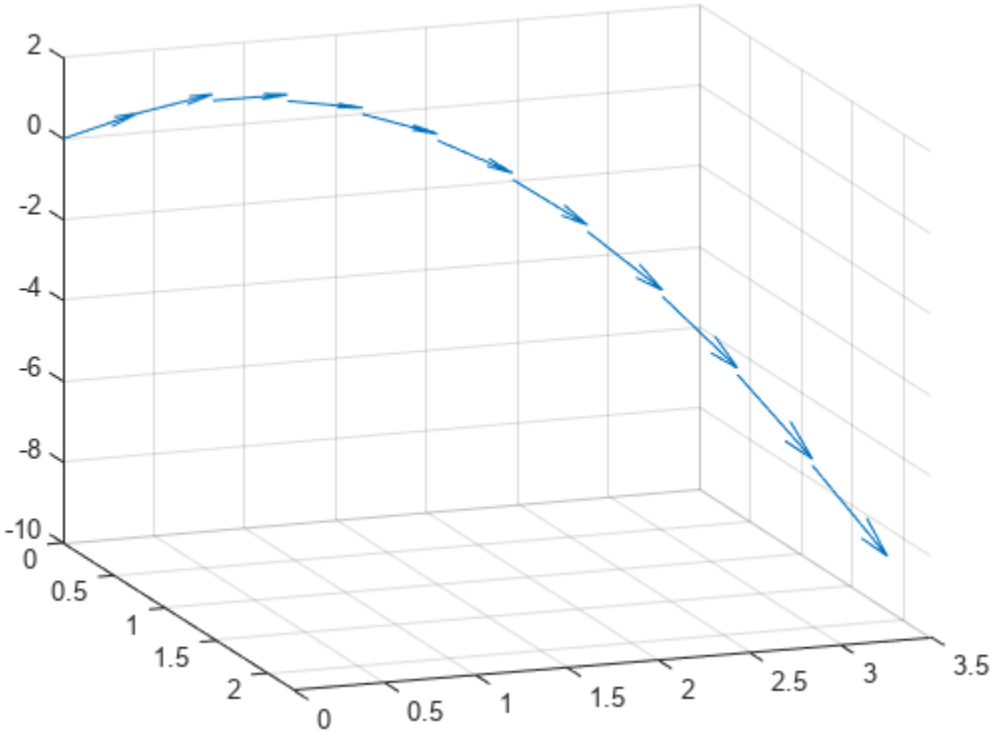
```
vz = 10; % velocity constant
a = -32; % acceleration constant
t = 0:.1:1;
z = vz*t + 1/2*a*t.^2;
```

计算 x 方向和 y 方向上的位置。

```
vx = 2;
x = vx*t;

vy = 3;
y = vy*t;
```

计算速度向量的分量并使用三维箭头图显示该向量。将坐标区的视点改为 **[70,18]**。

```
u = gradient(x);
v = gradient(y);
w = gradient(z);
scale = 0;

figure
quiver3(x,y,z,u,v,w,scale)
view([70,18])
```



粒子动画可以显示哪些信息

流粒子动画可用于可视化向量场的流向和速度。“粒子”（由线标记表示）用于跟踪沿特定流线的流动。动画中每个粒子的速度与流线上任何给定点处的向量场的模成正比。

1.指定数据范围的起点

本示例通过指定适当的起点来确定要绘制的体区域。在本例中，流线图的起点为 x = 100，y 方向上的范围为 20 到 50，并位于 z = 5 的平面上，这并不是完整的体范围。

```
load wind
[sx sy sz] = meshgrid(100,20:2:50,5);
```

2.创建流线以指示粒子路径

本示例使用流线（**stream3**、**streamline**）跟踪动画粒子的路径，为动画添加视觉环境。

```
verts = stream3(x,y,z,u,v,w,sx,sy,sz);
sl = streamline(verts);
```

3.定义视图

虽然所有流线都开始于 z = 5 的平面，但某些螺旋线的值更低。以下设置提供了清晰的动画视图：

- 选择的视点 (**view**) 既能显示包含大部分流线的平面，又能显示螺旋线。
- 将数据的纵横比 (**daspect**) 设为 **[2 2 0.125]** 可在 z 方向提供更高的分辨率，使螺旋线中的流粒子更容易看清。
- 将坐标区范围设置为与数据范围匹配 (**axis**)，然后绘制轴框 (**box**)。

```
view(-10.5,18)
daspect([2 2 0.125])
axis tight;
set(gca,'BoxStyle','full','Box','on')
```

4.计算流粒子顶点

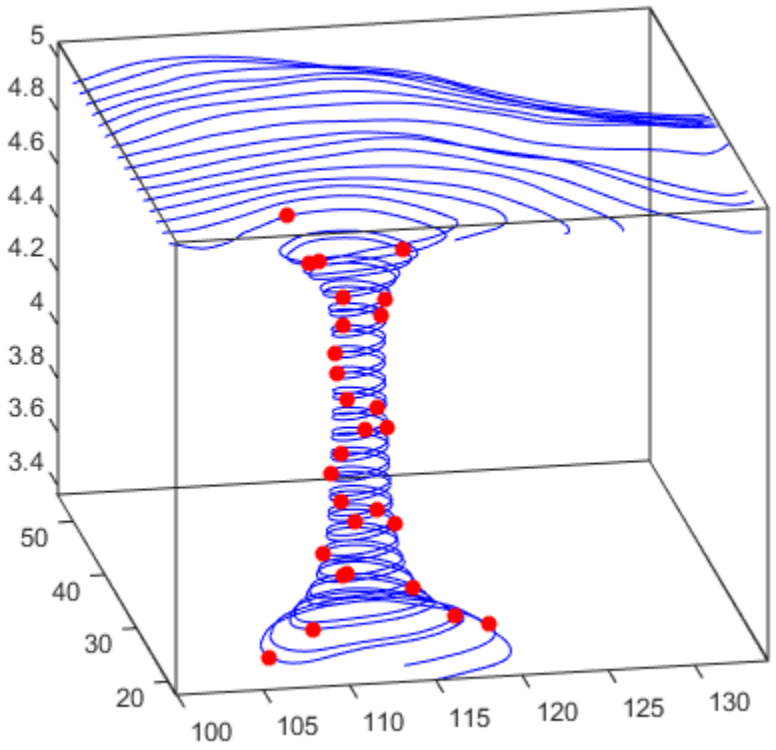
确定流线上要绘制粒子的顶点。**interpstreamspeed** 函数基于流线顶点和向量数据的速度返回此数据。本示例将速度缩放 0.05，以增加插值顶点的数量。

将坐标区的 **SortMethod** 属性设置为 **childorder**，使动画运行速度更快。

streamparticles 函数设置以下属性：

- 将 **Animate** 设置为 **10**，使动画运行 10 次。
- 将 **ParticleAlignment** 设置为 **on**，以便同时开始所有粒子跟踪。
- 将 **MarkerEdgeColor** 设置为 **none**，以便只绘制圆形标记的面。如果不绘制标记的边，动画通常会运行得更快。
- 将 **MarkerFaceColor** 设置为 **red**。
- 将 **Marker** 设置为 **o**，以绘制圆形标记。您也可以使用其他线标记。

```
iverts = interpstreamspeed(x,y,z,u,v,w,verts,0.01);
set(gca,'SortMethod','childorder');
streamparticles(iverts,15,...
    'Animate',10,...
    'ParticleAlignment','on',...
    'MarkerEdgeColor','none',...
    'MarkerFaceColor','red',...
    'Marker','o');
```


5
4.8
4.6
4.4
4.2
4
3.8
3.6
3.4
50
40
30
20
100
105
110
115
120
125
130

利用圆锥图显示向量场

本节内容
“圆锥图可以显示哪些信息” （第 3-38 页） “1.创建等值面” （第 3-38 页） “2.为等值面添加等值顶” （第 3-38 页） “3.创建第一组圆锥体” （第 3-39 页） “4.创建第二组圆锥体” （第 3-39 页） “5.定义视图” （第 3-39 页） “6.添加光照” （第 3-39 页）

圆锥图可以显示哪些信息

本示例绘制 **wind** 数据的速度向量圆锥图。生成的圆锥图利用了好几种可视化方法：

- 通过等值面为圆锥图提供视觉环境，并提供为一组圆锥体选择特定数据值的方法。
- 通过光照使等值面的形状清晰可见。
- 通过透视投影、相机定位和视角调整合成最终视图。

1.创建等值面

在矩形数据空间显示等值面可为圆锥图提供视觉环境。创建等值面需要多个步骤：

1. 计算代表风速的向量场的模。
2. 使用 **isosurface** 和 **patch** 绘制等值面，以说明矩形空间中哪些位置的风速等于特定值。等值面内的区域风速较高，等值面外的区域风速较低。
3. 使用 **isonormals** 根据体数据计算等值面的顶点法线，而不是根据用来渲染等值面的三角形来计算顶点法线。这样的法线通常可以生成更准确的结果。
4. 设置等值面的视觉属性，使其为红色并且不绘制边（**FaceColor**、**EdgeColor**）。

```
load wind
wind_speed = sqrt(u.^2 + v.^2 + w.^2);
hiso = patch(isosurface(x,y,z,wind_speed,40));
isonormals(x,y,z,wind_speed,hiso)
hiso.FaceColor = 'red';
hiso.EdgeColor = 'none';
```

2.为等值面添加等值顶

等值顶与切片平面的相似之处在于它们都显示体的横截面。它们被设计成等值面的端顶。在等值顶上使用插值面颜色将导致数据值映射到当前颜色图中的颜色。要为等值面创建等值顶，请按相同的等值定义它们（**isocaps**、**patch**、**colormap**）。

```
hcap = patch(isocaps(x,y,z,wind_speed,40),...
  'FaceColor','interp',...
  'EdgeColor','none');
colormap hsv
```

3.创建第一组圆锥体

- 在调用 **coneplot** 之前，请使用 **daspect** 设置坐标区的数据纵横比，以便函数确定圆锥体的正确大小。
- 通过计算具有较小等值的另一个等值面来确定放置圆锥体的点（使圆锥体显示在第一个等值面之外），并使用 **reducepatch** 减少面和顶点的数量（使图中的圆锥体不会过多）。
- 绘制圆锥体，并将面颜色设置为 **blue**，将边颜色设置为 **none**。

```
daspect([1 1 1]);
[f,verts] = reducepatch(isosurface(x,y,z,wind_speed,30),0.07);
h1 = coneplot(x,y,z,u,v,w,verts(:,1),verts(:,2),verts(:,3),3);
h1.FaceColor = 'blue';
h1.EdgeColor = 'none';
```

4.创建第二组圆锥体

1 使用数据范围内的值创建第二组点（**linspace**、**meshgrid**）。

2 绘制第二组圆锥体，并将面颜色设置为 green，将边颜色设置为 none。

```
xrange = linspace(min(x(:)),max(x(:)),10);
yrange = linspace(min(y(:)),max(y(:)),10);
zrange = 3:4:15;
[cx,cy,cz] = meshgrid(xrange,yrange,zrange);
h2 = coneplot(x,y,z,u,v,w,cx,cy,cz,2);
h2.FaceColor = 'green';
h2.EdgeColor = 'none';
```

5.定义视图

1 使用 **axis** 命令将坐标轴范围设置为等于数据的最小值和最大值，并将图包含在框中以增强立体感 (**box**)。

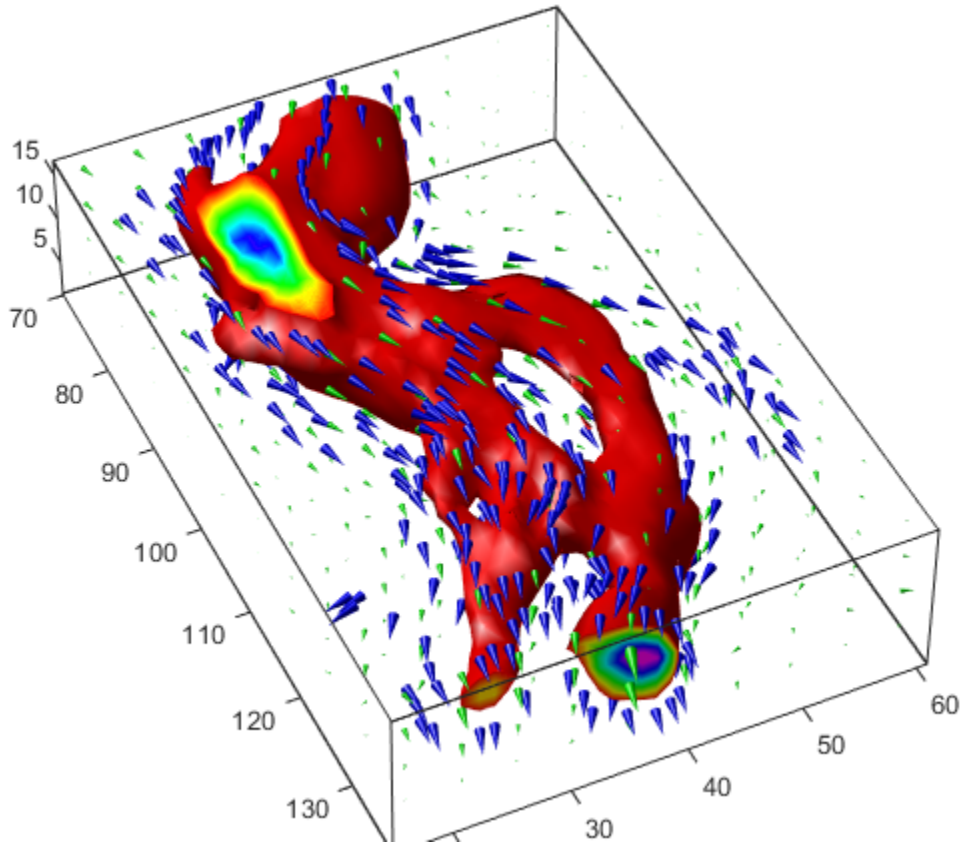
2 将投影类型设置为透视，以生成更自然的体视图。设置视点并放大，以使场景变大（**camproj**、**camzoom**、**view**）。

```
axis tight
set(gca,'BoxStyle','full','Box','on')
camproj perspective
camzoom(1.25)
view(65,45)
```

6.添加光照

添加光源并使用 Gouraud 光照，使等值面具有最平滑的光照效果。提高等值顶上的背景光强度，使等值顶更亮（**camlight**、**lighting**、**AmbientStrength**）。

```
camlight(-45,45)
hcap.AmbientStrength = 0.6;
lighting gouraud
```


15
10
5
70
80
90
100
110
120
130
60
50
40
30

可视化体数据

以下示例演示在 MATLAB® 中以可视方式呈现体数据的几种方法。

显示等值面

等值面指空间体内的所有点都具有一个常量值的曲面。使用 **isosurface** 函数生成曲面外的面和顶点，并使用 **isocaps** 函数生成体端顶的面和顶点。使用 **patch** 命令绘制空间体及其端顶。

```
load mri D
D = squeeze(D);
limits = [NaN NaN NaN NaN NaN 10]
```

```
limits = 1×6

   NaN   NaN   NaN   NaN   NaN    10
```

```
[x, y, z, D] = subvolume(D, limits);
[fo,vo] = isosurface(x,y,z,D,5);
[fe,ve,ce] = isocaps(x,y,z,D,5);
figure
p1 = patch('Faces', fo, 'Vertices', vo);
p1.FaceColor = 'red'
```

```
p1 =
  Patch with properties:

    FaceColor: [1 0 0]
    FaceAlpha: 1
    EdgeColor: [0 0 0]
    LineStyle: '-'
        Faces: [23351x3 double]
     Vertices: [12406x3 double]

  Show all properties
```

```
p1.EdgeColor = 'none'
```

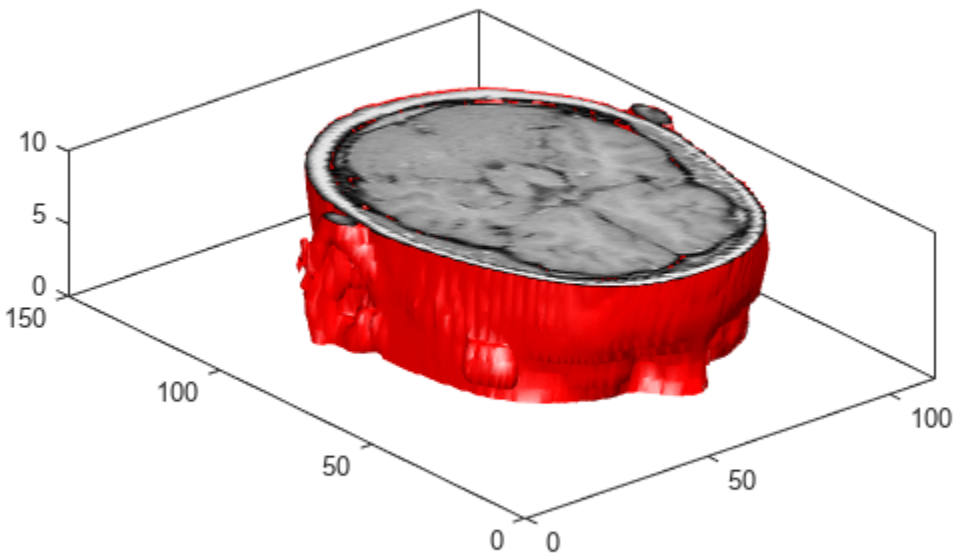
```
p1 =
  Patch with properties:

    FaceColor: [1 0 0]
    FaceAlpha: 1
    EdgeColor: 'none'
    LineStyle: '-'
        Faces: [23351x3 double]
     Vertices: [12406x3 double]

  Show all properties
```

```
p2 = patch('Faces', fe, 'Vertices', ve, ...
   'FaceVertexCData', ce)
```

```
p2 =
  Patch with properties:
```

```
  FaceColor: [0 0 0]
  FaceAlpha: 1
  EdgeColor: [0 0 0]
  LineStyle: '-'
      Faces: [27265x3 double]
   Vertices: [14250x3 double]

  Show all properties
```

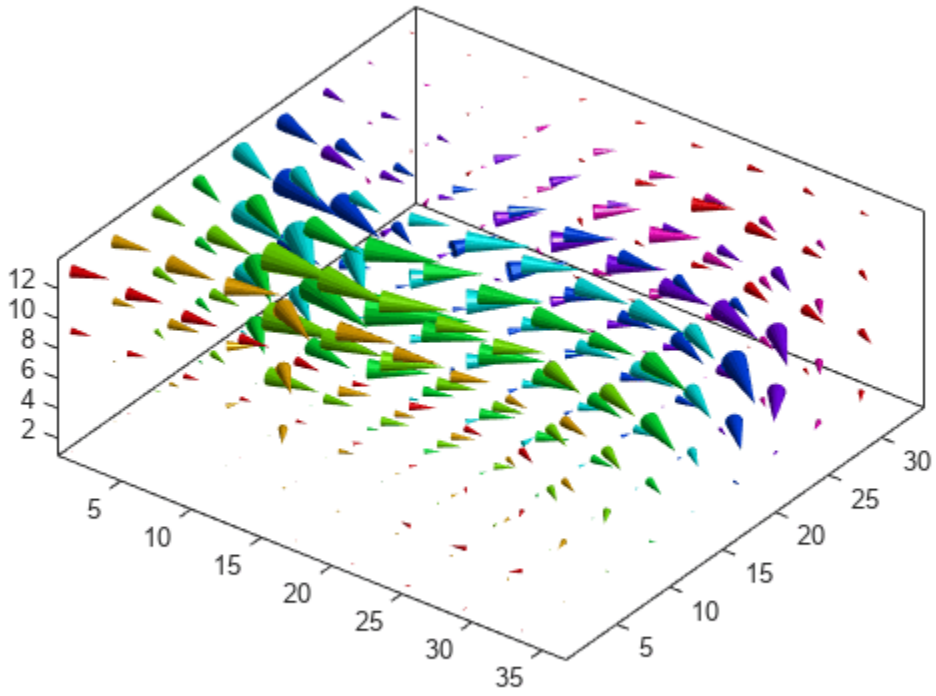
```
p2.FaceColor = 'interp';
p2.EdgeColor = 'none';

view(-40,24)
daspect([1 1 0.3])
colormap(gray(100))
box on

camlight(40,40)
camlight(-20,-10)
lighting gouraud
```



创建锥体绘图

coneplot 命令在空间体的 x、y、z 点处将速度向量绘制为锥体。锥体表示在每个点处向量场的幅值和方向。

```
cla
load wind u v w x y z
[m,n,p] = size(u)

m = 35

n = 41

p = 15

[Cx, Cy, Cz] = meshgrid(1:4:m,1:4:n,1:4:p);
h = coneplot(u,v,w,Cx,Cy,Cz,y,4);
set(h,'EdgeColor', 'none')

axis tight equal
view(37,32)
box on
colormap(hsv)
light
```

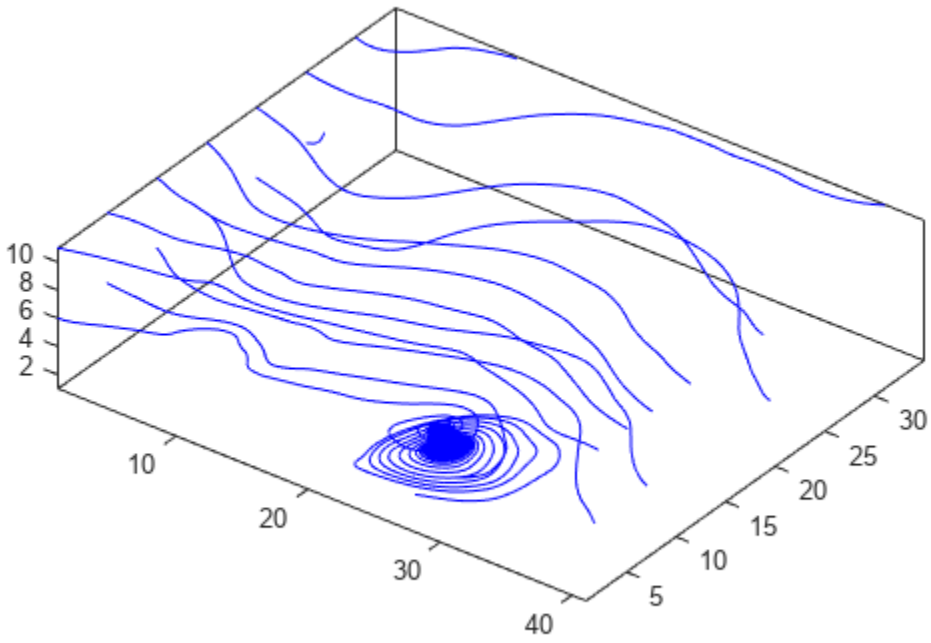


绘制流线图

streamline 函数在空间体的 x、y、z 点处绘制速度向量的流线图，以表示三维向量场的流向。

```
cla

[m,n,p] = size(u);
[Sx, Sy, Sz] = meshgrid(1,1:5:n,1:5:p);
```

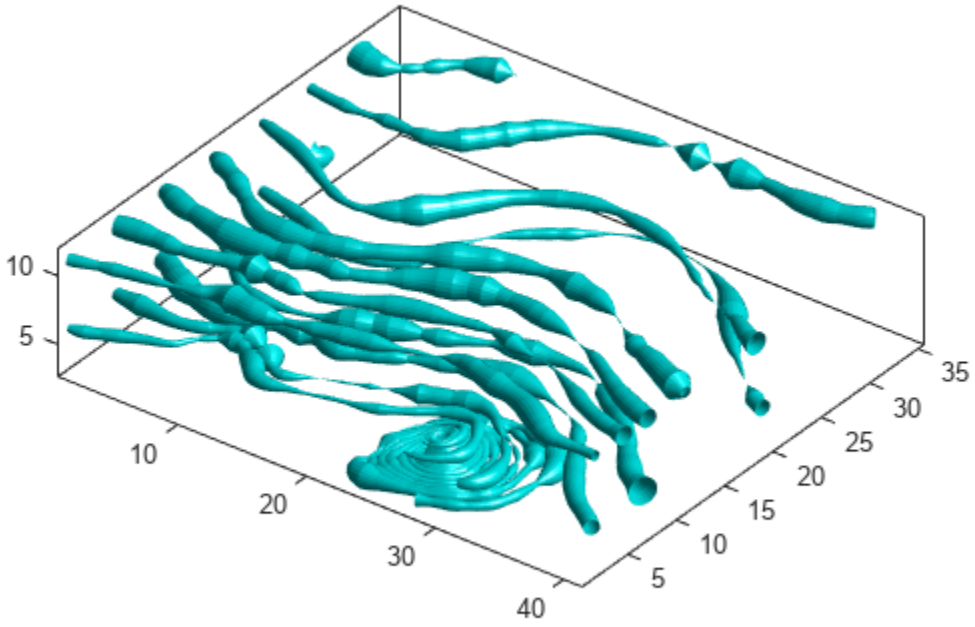
```
streamline(u,v,w,Sx,Sy,Sz)

axis tight equal
view(37,32)
box on
```



绘制流管图

streamtube 函数在空间体的 x、y、z 点处将速度向量绘制为流管图。管的宽度与每个点处向量场的归一化发散成比例。

```
cla

[~,n,p] = size(u);
[Sx, Sy, Sz] = meshgrid(1,1:5:n,1:5:p);
h = streamtube(u,v,w,Sx,Sy,Sz);
set(h, 'FaceColor', 'cyan')
set(h, 'EdgeColor', 'none')

axis tight equal
view(37,32)
box on
light
```



合并不同空间体的可视化绘图

在单个绘图中合并空间体的可视化绘图以生成更为全面的体内速度场全貌。

```
cla
spd = sqrt(u.*u + v.*v + w.*w);
[fo,vo] = isosurface(x,y,z,spd,40);
[fe,ve,ce] = isocaps(x,y,z,spd,40);
p1 = patch('Faces', fo, 'Vertices', vo);
p1.FaceColor = 'red'
```

```
p1 =
  Patch with properties:

    FaceColor: [1 0 0]
    FaceAlpha: 1
    EdgeColor: [0 0 0]
    LineStyle: '-'
        Faces: [5340x3 double]
     Vertices: [2727x3 double]

  Show all properties
```

```
p1.EdgeColor = 'none'
```

```
p1 =
  Patch with properties:
```

```
    FaceColor: [1 0 0]
    FaceAlpha: 1
    EdgeColor: 'none'
    LineStyle: '-'
        Faces: [5340x3 double]
     Vertices: [2727x3 double]

  Show all properties
```

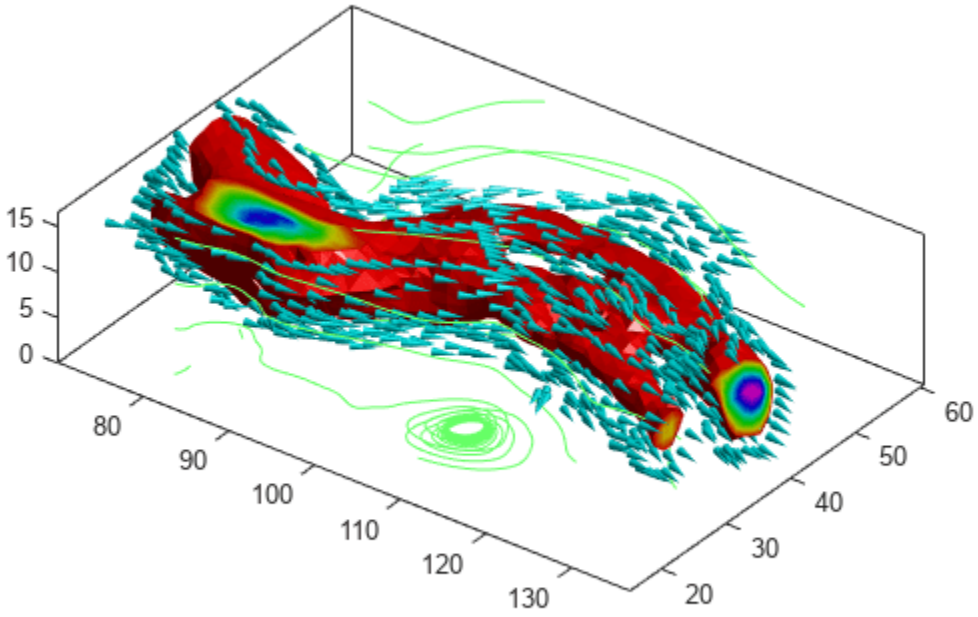
```
p2 = patch('Faces', fe, 'Vertices', ve, ...
  'FaceVertexCData', ce)

p2 =
  Patch with properties:

    FaceColor: [0 0 0]
    FaceAlpha: 1
    EdgeColor: [0 0 0]
    LineStyle: '-'
        Faces: [464x3 double]
     Vertices: [301x3 double]

  Show all properties
```

```
p2.FaceColor = 'interp'

p2 =
  Patch with properties:

    FaceColor: 'interp'
    FaceAlpha: 1
    EdgeColor: [0 0 0]
    LineStyle: '-'
        Faces: [464x3 double]
     Vertices: [301x3 double]

  Show all properties
```

```
p2.EdgeColor = 'none'

p2 =
  Patch with properties:

    FaceColor: 'interp'
    FaceAlpha: 1
    EdgeColor: 'none'
    LineStyle: '-'
        Faces: [464x3 double]
     Vertices: [301x3 double]

  Show all properties
```

```
[fc, vc] = isosurface(x, y, z, spd, 30);
[fc, vc] = reducepatch(fc, vc, 0.2);
```

```
h1 = coneplot(x,y,z,u,v,w,vc(:,1),vc(:,2),vc(:,3),3);
h1.FaceColor = 'cyan';
h1.EdgeColor = 'none';

[sx, sy, sz] = meshgrid(80, 20:10:50, 0:5:15);
h2 = streamline(x,y,z,u,v,w,sx,sy,sz);
set(h2, 'Color', [.4 1 .4])

axis tight equal
view(37,32)
box on
light
```



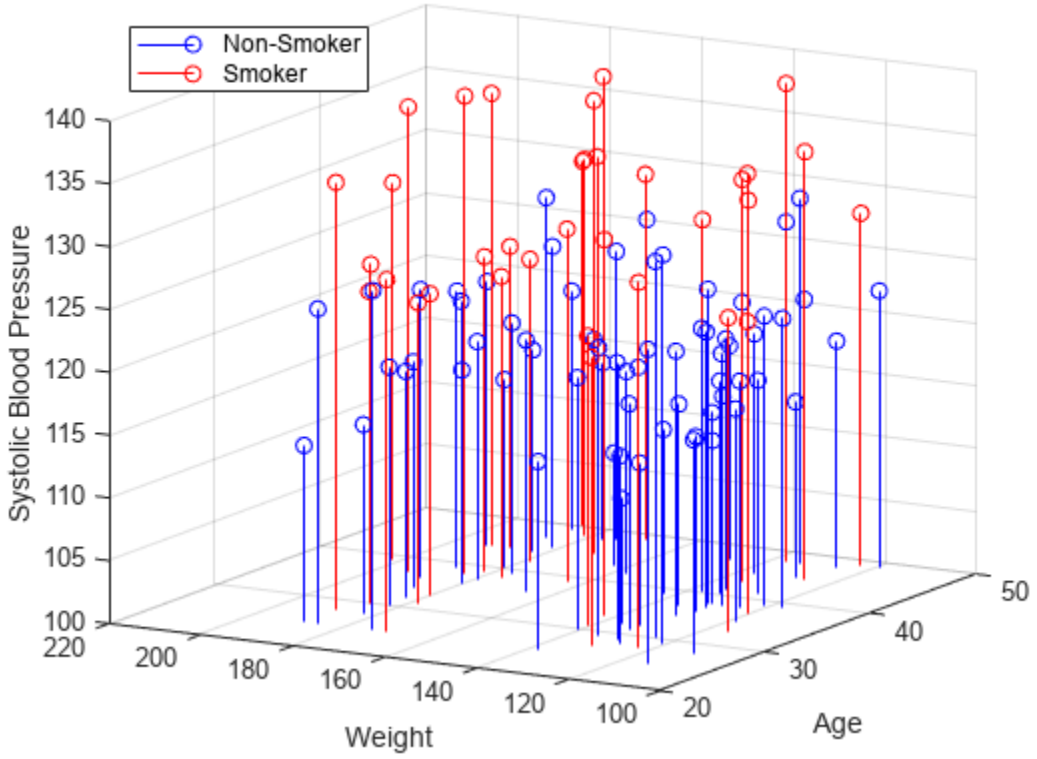
可视化四维数据

以下示例演示在 MATLAB® 中以可视方式呈现四维 (4-D) 数据的几种方法。

以可视化形式呈现具有离散变量的四维数据

有时数据含有一个离散变量，即该变量仅有几个可能的值。您可以对每个离散组中的数据创建多个具有相同类型的绘图。例如，使用 **stem3** 函数查看三个变量之间的关系，第四个变量则将总体数据划分为若干个离散组。

```
load patients Smoker Age Weight Systolic                    % load data

nsIdx = Smoker == 0;
smIdx = Smoker == 1;

figure
stem3(Age(nsIdx), Weight(nsIdx), Systolic(nsIdx), 'Color', 'b')    % stem plot for non-smokers
hold on
stem3(Age(smIdx), Weight(smIdx), Systolic(smIdx), 'Color', 'r')    % stem plot for smokers
hold off

view(-60,15)
zlim([100 140])

xlabel('Age')                                          % add labels and a legend
ylabel('Weight')
zlabel('Systolic Blood Pressure')
legend('Non-Smoker', 'Smoker', 'Location', 'NorthWest')
```

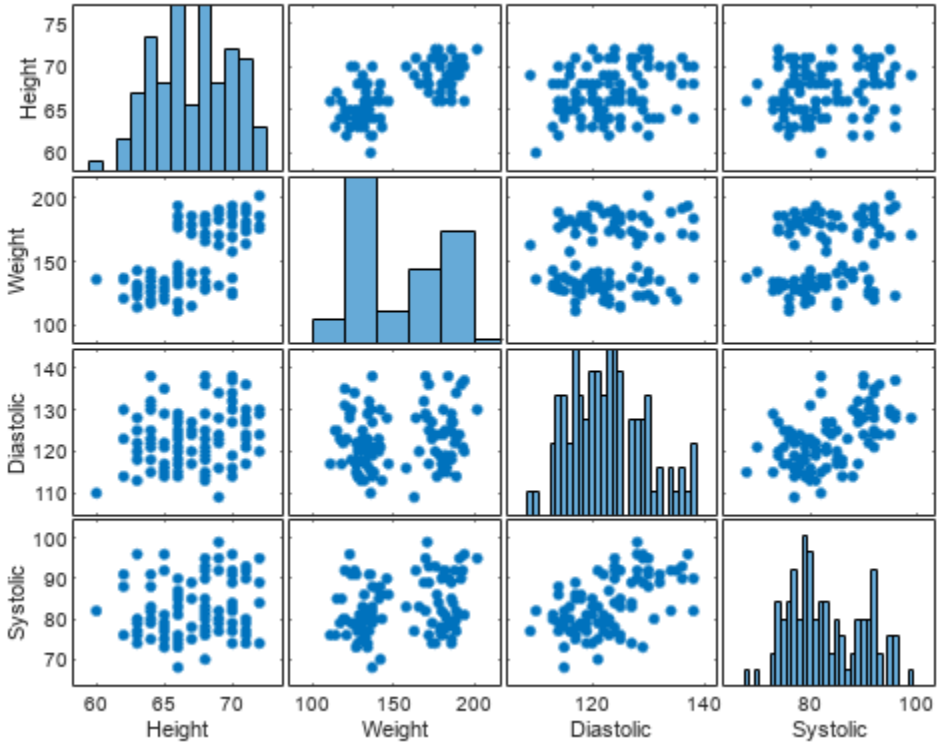


用多个绘图可视化四维数据

有了大型数据集，您可能想要查看各个变量是否相互关联。您可以使用 **plotmatrix** 函数创建绘图的 n x n 矩阵，以查看变量之间的成对关系。**plotmatrix** 函数返回两个输出。第一个输出是散点图中使用的线条对象的矩阵。第二个输出是所创建的坐标区对象的矩阵。

plotmatrix 函数还可用于更高阶数据集。

```
load patients Height Weight Diastolic Systolic    % load data

labels = {'Height' 'Weight' 'Diastolic' 'Systolic'};
data = [Height Weight Systolic Diastolic];

[h,ax] = plotmatrix(data);                  % create a 4 x 4 matrix of plots
for i = 1:4                                 % label the plots
  xlabel(ax(4,i), labels{i})
  ylabel(ax(i,1), labels{i})
end
```

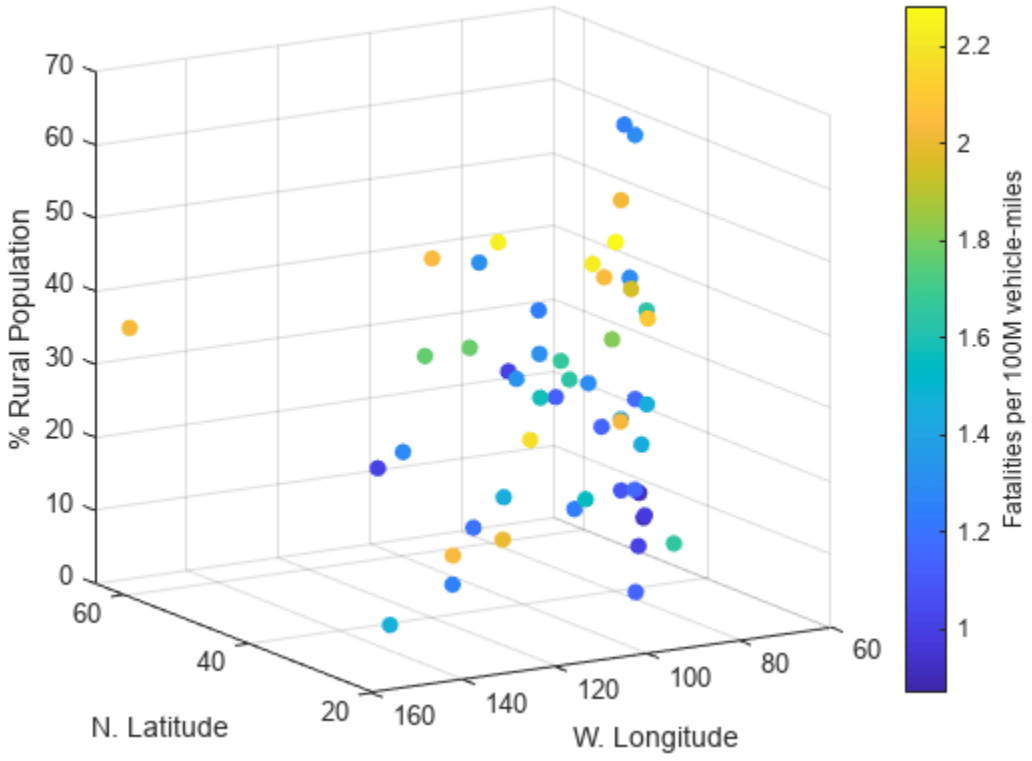


以可视化形式呈现包含三个变量的函数

对于许多类型的四维数据，您可以使用颜色来表示第四维度。如果您有一个三变量函数，这通常会很有效。

例如，将美国高速公路死亡数据表示为经度、纬度以及位置是在农村还是城市的函数。绘图中的 x、y 和 z 值表示这三个变量。颜色表示高速公路死亡人数。

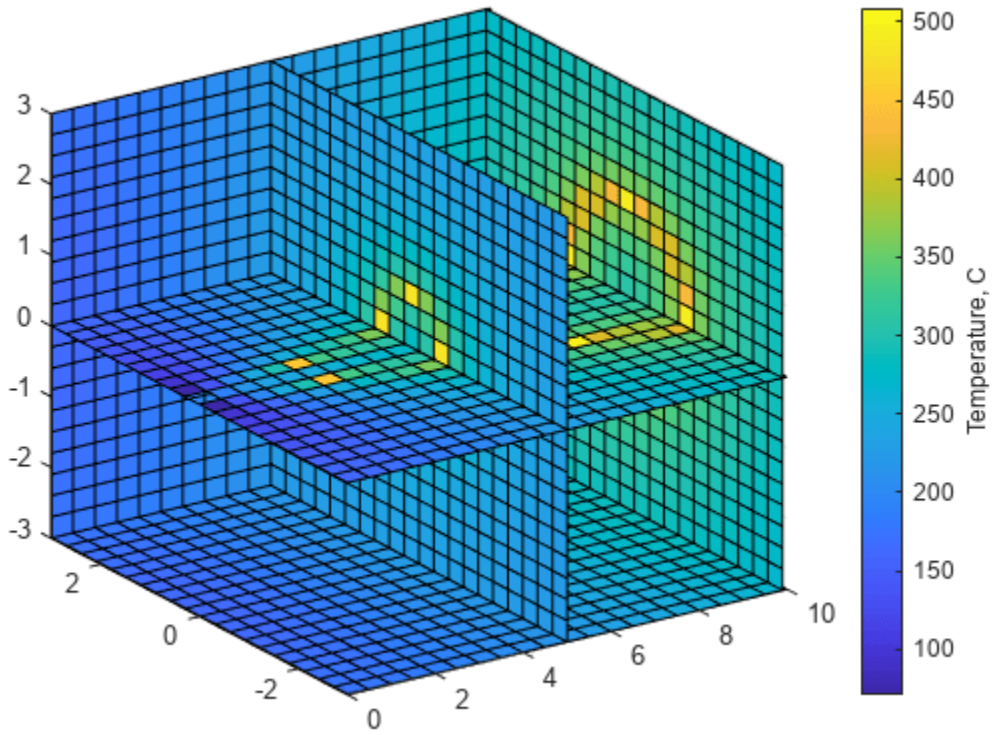
```
cla
load accidents hwydata                    % load data

long = -hwydata(:,2);                     % longitude data
lat = hwydata(:,3);                       % latitude data
rural = 100 - hwydata(:,17);              % percent rural data
fatalities = hwydata(:,11);               % fatalities data

scatter3(long,lat,rural,40,fatalities,'filled')    % draw the scatter plot
ax = gca;
ax.XDir = 'reverse';
view(-31,14)
xlabel('W. Longitude')
ylabel('N. Latitude')
zlabel('% Rural Population')

cb = colorbar;                            % create and label the colorbar
cb.Label.String = 'Fatalities per 100M vehicle-miles';
```



可视化空间体中的数据

您的数据可能包含测量物理对象所得的值，如管道内的温度。在此情况下，物理维度可以表示为一个空间体，用颜色表示测量的幅值。例如，使用 **slice** 函数显示在空间体横截面处所测得的变量的值。

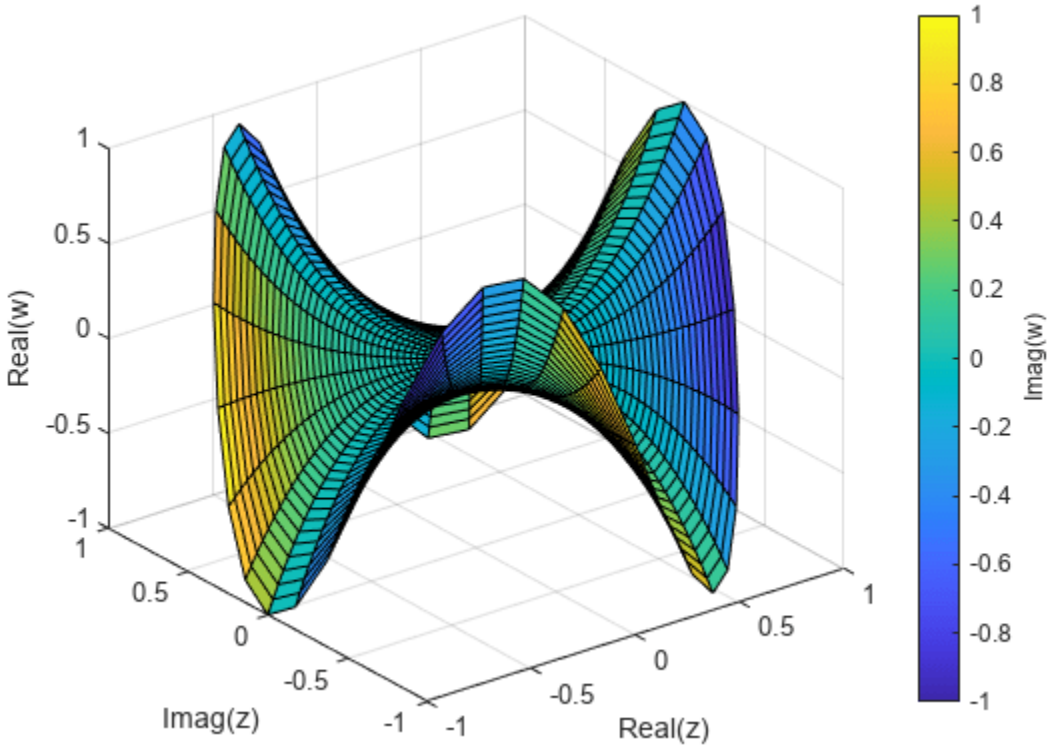
```
load fluidtemp x y z temp                % load data

xslice = [5 9.9];                        % define the cross sections to view
yslice = 3;
zslice = ([-3 0]);

slice(x, y, z, temp, xslice, yslice, zslice)    % display the slices
ylim([-3 3])
view(-34,24)

cb = colorbar;                           % create and label the colorbar
cb.Label.String = 'Temperature, C';
```



绘制包含复变量的函数

复函数的输入和输出都含有实部和虚部。您可以使用带有颜色的三维绘图表示复函数。在此情况下，x 和 y 轴表示输入的实部和虚部。z 轴表示输出的实部，颜色表示输出的虚部。

```
r = (0:0.025:1)';                 % create a matrix of complex inputs
theta = pi*(-1:0.05:1);
z = r*exp(1i*theta);
w = z.^3;                         % calculate the complex outputs

surf(real(z),imag(z),real(w),imag(w))    % visualize the complex function using surf
xlabel('Real(z)')
ylabel('Imag(z)')
zlabel('Real(w)')
cb = colorbar;
cb.Label.String = 'Imag(w)';
```


Real(w)
1
0.5
0
-0.5
-1
Imag(z)
1
0.5
0
-0.5
-1
Real(z)
1
0.5
0
-0.5
-1
Imag(w)
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1

显示复杂三维对象

此示例说明如何创建并显示复杂三维对象以及控制其外观。

获取对象的几何图

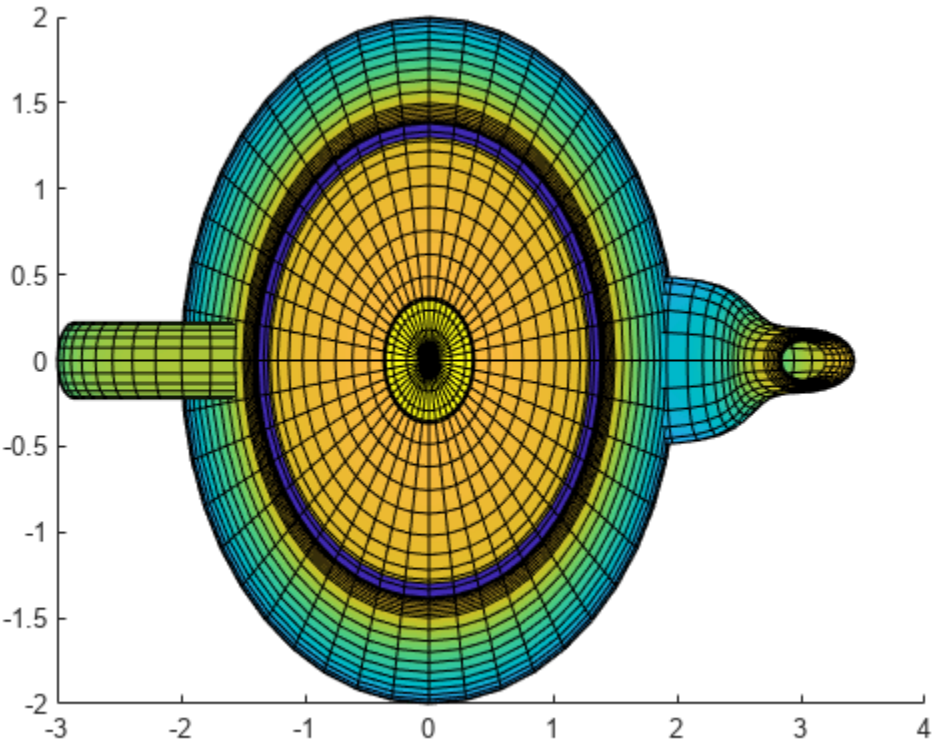
此示例使用一个称为 Newell 茶壶的图形对象。茶壶的顶点、面和颜色索引数据由 **teapotData** 函数计算得出。由于茶壶是一个复杂的几何形状，函数因而返回大量的顶点（4608 个）和面（3872 个）。

```
[verts, faces, cindex] = teapotGeometry;
```

创建茶壶补片对象

使用几何数据，用 **patch** 命令绘制茶壶。**patch** 命令创建补片对象。

```
figure
p = patch('Faces',faces,'Vertices',verts,'FaceVertexCData',cindex,'FaceColor','interp')
```

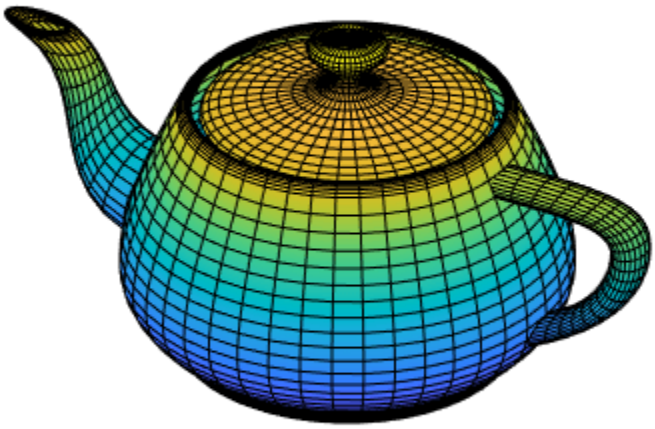



```
p =
  Patch with properties:

    FaceColor: 'interp'
    FaceAlpha: 1
    EdgeColor: [0 0 0]
    LineStyle: '-'
        Faces: [3872x4 double]
```

```
  Vertices: [4608x3 double]

  Show all properties
```

使用 **view** 命令更改对象的方向。

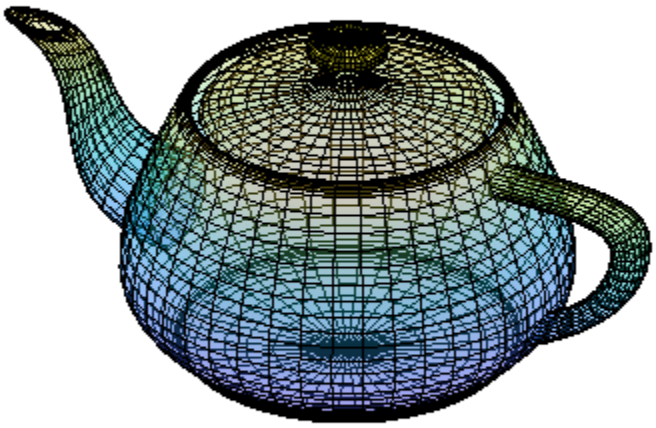
```
view(-151,30)     % change the orientation
axis equal off    % make the axes equal and invisible
```



调整透明度

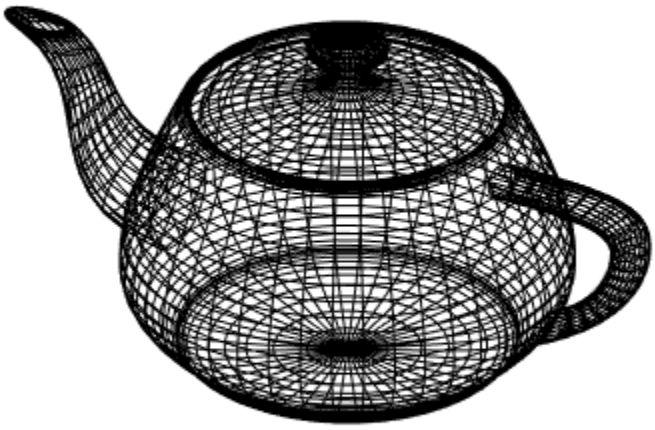
使用补片对象的 **FaceAlpha** 属性使对象变得透明。

```
p.FaceAlpha = 0.3;   % make the object semi-transparent
```



如果 **FaceColor** 属性设置为“none”，则该对象会作为线框图显示。

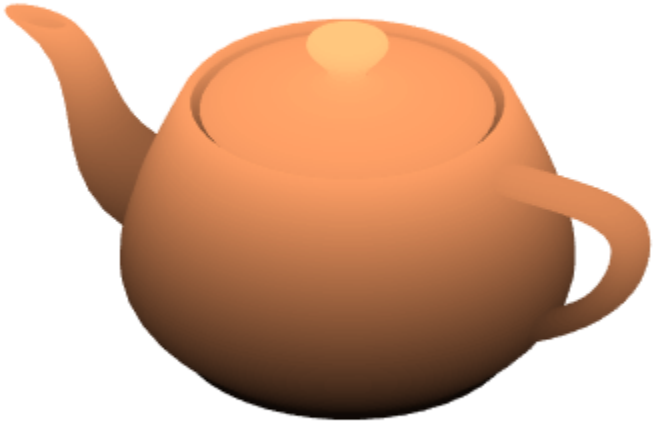
```
p.FaceColor = 'none';    % turn off the colors
```



更改颜色图

使用 **colormap** 函数更改对象的颜色。

```
p.FaceAlpha = 1;          % remove the transparency
p.FaceColor = 'interp';   % set the face colors to be interpolated
p.LineStyle = 'none';     % remove the lines
colormap(copper)          % change the colormap
```



用光源照射对象

添加一个光源，使对象看起来更加逼真。

```
l = light('Position',[-0.4 0.2 0.9],'Style','infinite')
```

```
l =
  Light with properties:

      Color: [1 1 1]
      Style: 'infinite'
   Position: [-0.4000 0.2000 0.9000]
    Visible: on

  Show all properties
```

```
lighting gouraud
```

补片对象的以下属性会影响光照强度和对象的反光属性。

- **AmbientStrength** - 控制环境光的强度
- **DiffuseStrength** - 控制散射光的强度
- **SpecularStrength** - 控制反射光的强度
- **SpecularExponent** - 控制反射光的粗糙度
- **SpecularColorReflectance** - 控制反射颜色的计算方式。

您可以分别设置这些属性。若要将这些属性设置为一组预先确定的值来获得近似金属材料、闪光材料或哑光材料的外观，请使用 **material** 命令。

material shiny

使用光源的 **Position** 属性调整其位置。位置以 x、y、z 坐标表示。

```
l.Position = [-0.1 0.6 0.8]
```

```
l =
  Light with properties:

      Color: [1 1 1]
      Style: 'infinite'
   Position: [-0.1000 0.6000 0.8000]
    Visible: on

  Show all properties
```

显示地貌数据

下面的示例说明了多种表示地球地貌的方法。此示例中的数据取自美国商务部海洋及大气管理局 (NOAA) 国家地理数据中心，数据通告编号为 88-MGG-02。

关于地貌数据

数据文件 **topo.mat** 包含地貌数据。**topo** 是海拔高度数据，**topomap1** 是海拔高度的颜色图。

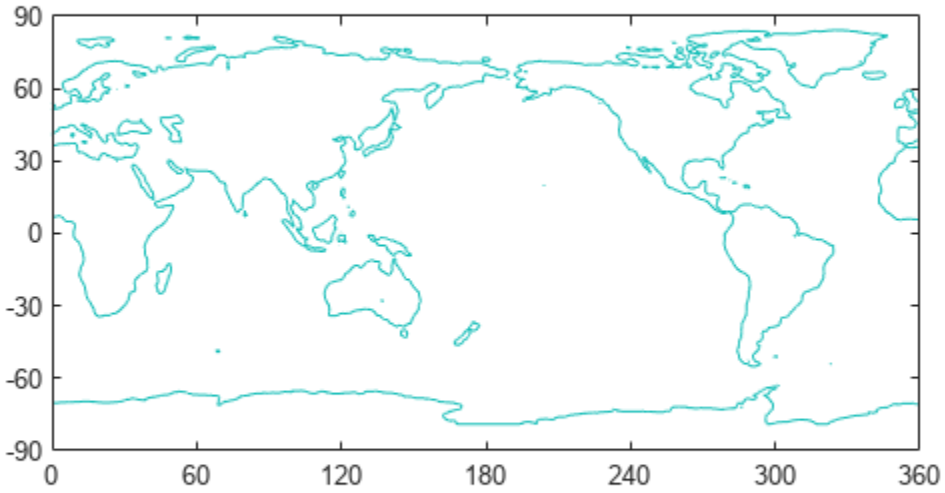
```
load topo topo topomap1    % load data
whos('topo','topomap1')
```

```
  Name         Size            Bytes  Class    Attributes

  topo       180x360          518400  double
  topomap1    64x3              1536  double
```

创建等高线图

以可视化形式呈现地貌数据的一种方法是创建等高线图。若要显示地球上各大洲的轮廓，请绘制海拔高度为零的点。**contour** 中的前三个输入参数指定等高线图上的 X、Y 和 Z 值。第四个参数指定要绘制的等高线层级。

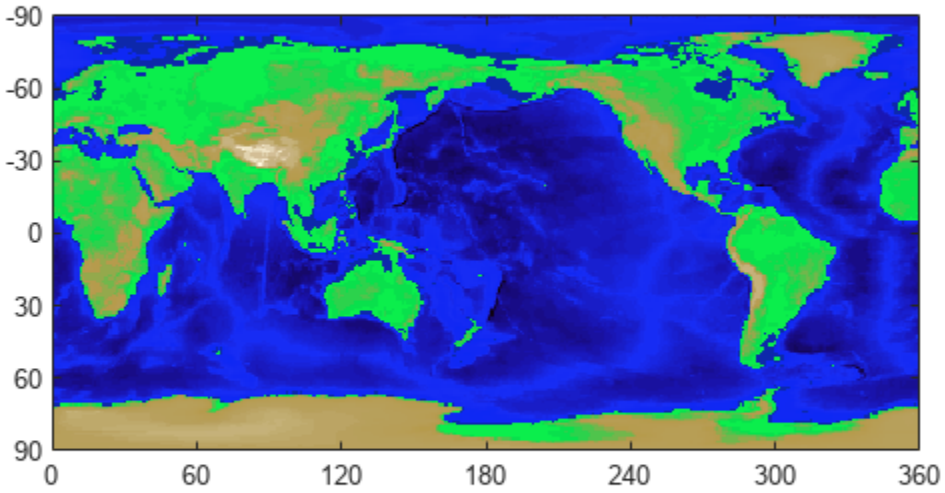
```
x = 0:359;                      % longitude
y = -89:90;                     % latitude

figure
contour(x,y,topo,[0 0])

axis equal                       % set axis units to be the same size
box on                          % display bounding box

ax = gca;                       % get current axis
ax.XLim = [0 360];                  % set x limits
ax.YLim = [-90 90];                 % set y limits
ax.XTick = [0 60 120 180 240 300 360];   % define x ticks
ax.YTick = [-90 -60 -30 0 30 60 90];     % define y ticks
```



以图像形式查看数据

您可以使用高程数据和自定义颜色图创建地貌图像。地貌数据被视为自定义颜色图的索引。将图像的 **CDataMapping** 设置为 **'scaled'** 以将数据值线性缩放至颜色图的范围。在颜色图上，不同深浅的绿色表示海拔高度数据，不同深浅的蓝色表示海平面下的深度。

```
image([0 360],[-90 90], flip(topo), 'CDataMapping', 'scaled')
colormap(topomap1)

axis equal                          % set axis units to be the same size

ax = gca;                           % get current axis
ax.XLim = [0 360];                      % set x limits
ax.YLim = [-90 90];                     % set y limits
ax.XTick = [0 60 120 180 240 300 360];    % define x ticks
ax.YTick = [-90 -60 -30 0 30 60 90];      % define y ticks
```



使用纹理映射

纹理映射将二维图像映射到三维曲面上。若要将地貌映射到球形曲面，请将由 **CData** 属性指定的曲面颜色设置为地貌数据并将 **FaceColor** 属性设置为 **'texturemap'**。

```
clf
[x,y,z] = sphere(50);          % create a sphere
s = surface(x,y,z);            % plot spherical surface

s.FaceColor = 'texturemap';    % use texture mapping
s.CData = topo;                % set color data to topographic data
s.EdgeColor = 'none';          % remove edges
s.FaceLighting = 'gouraud';    % preferred lighting for curved surfaces
s.SpecularStrength = 0.4;      % change the strength of the reflected light

light('Position',[-1 0 1])     % add a light

axis square off                % set axis to square and remove axis
view([-30,30])                 % set the viewing angle
```