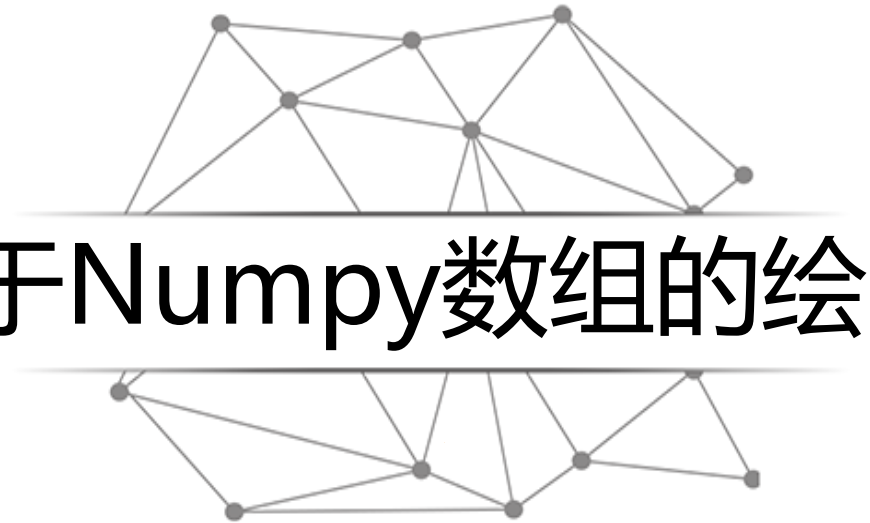


基于Numpy数组的绘图函数

mlab对Numpy建立可视化过程：

1、建立数据源

2、使用Filter（可选）

3、添加可视化模块

3D绘图函数-0D和1D数据

函数	说明
Point3d( )	基于Numpy数组x、y、z提供的三维点坐标，绘制点图形
Plot3d( )	基于1维Numpy数组x、y、z提供的三维坐标数据，绘制线图形

3D绘图函数-Points3d()

函数形式：

```
points3d(x, y, z…)
points3d(x, y, z, s, …)
points3d(x, y, z, f, …)
```

x,y,z表示numpy数组、列表或者其他形式的点三维坐标

s表示在该坐标点处的标量值

f表示通过函数f（x, y, z）返回的标量值

3D绘图函数-Points3d ()

参数	说明
color	VTK对象的颜色，定义为(0,1)的三元组
colormap	colormap的类型，例如Reds、Blues、Copper等
extent	x、y、z数组范围[xmin, xmax, ymin, ymax, zmin, zmax]
figure	画图
line_width	线的宽度，该值为float，默认为0.2
mask_points	减少/降低大规模点数据集的数量
mode	符号的模式，例如2darrow、2dcircle、arrow、cone等
name	VTK对象名字

3D绘图函数-Points3d ()

参数	说明
opcity	Vtk对象的整体透明度，该值为float型，默认为1.0
reset_zoom	对新加入场景数据的放缩进行重置。默认为True
resolution	符号的分辨率，如球体的细分数，该值为整型，默认为8
scale_factor	符号放缩的比例
scale_mode	符号的放缩模式，如vector、scalar、none
transparent	根据标量值确定actor的透明度
vmax	对colormap放缩的最大值
vmin	对colormap放缩的最小值

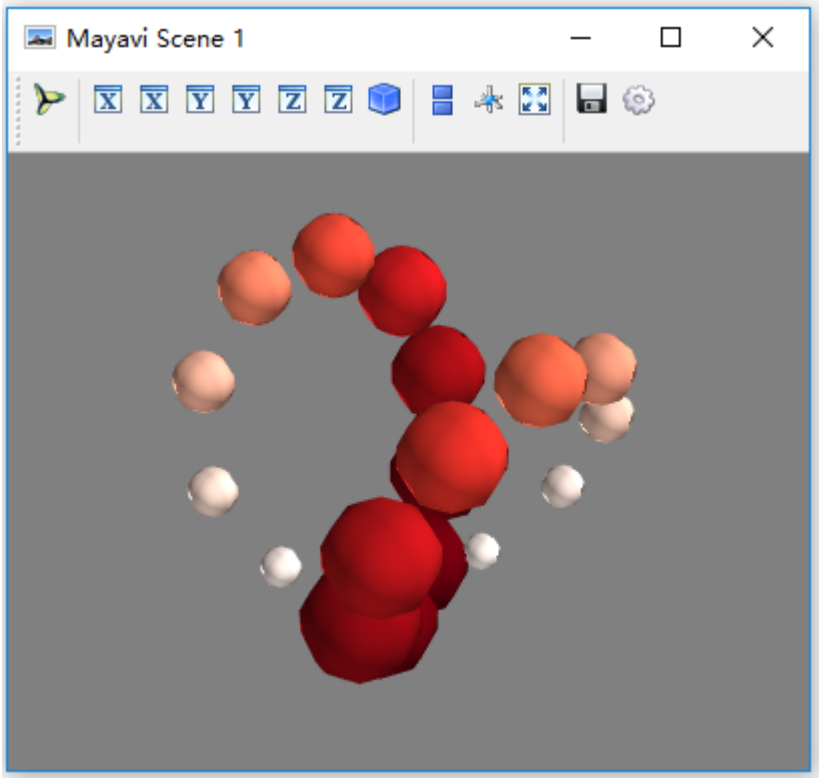
3D绘图函数-Points3d ()

```
import numpy as np
from mayavi import mlab

#建立数据
t = np.linspace(0, 4 * np.pi, 20)
x = np.sin(2 * t)
y = np.cos(t)
z = np.cos(2 * t)
s = 2 + np.sin(t)

#对数据进行可视化
points = mlab.points3d(x, y, z, s, colormap="Reds", scale_factor=.25)
mlab.show()
```

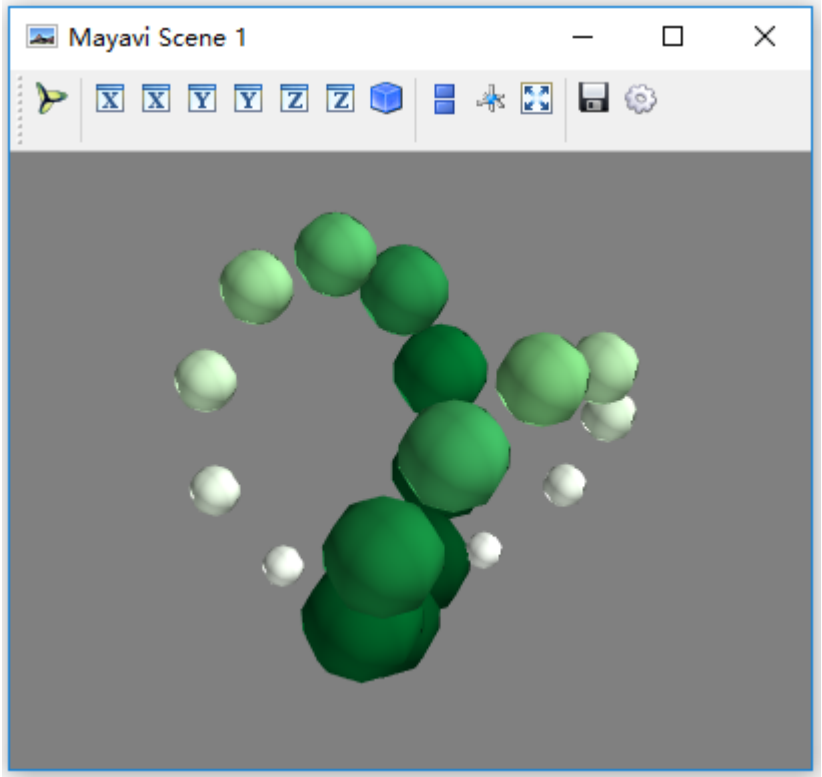
3D绘图函数-Points3d ()

```
points3d(x, y, z, s, colormap="Reds", scale_factor=.25)
```

3D绘图函数-Points3d ()



3D绘图函数-Points3d ()



mayavi.mlab.show()

```
mayavi.mlab.show(func = None, stop = False)
```

3D绘图函数-plot3d ()

函数形式：

```
plot3d(x, y, z…)
plot3d(x, y, z, s, …)
```

x,y,z表示numpy数组，或列表。给出了线上连续的点的位置

s表示在该坐标点处的标量值

3D绘图函数-plot3d ()

color、colormap、extent、figure、line_width、name、opacity、representation、reset_zoom、transparent、tube_radius、tube_sides、vmax、vmin

参数	说明
tube_radius	线管的半径，用于描述线的粗细
tube_sides	表示线的分段数，该值为整数，默认为6

3D绘图函数-plot3d ()

```
import numpy as np
from mayavi import mlab

#建立数据
n_mer, n_long = 6, 11
dphi = np.pi / 1000.0
phi = np.arange(0.0, 2 * np.pi + 0.5 * dphi, dphi)
mu = phi * n_mer
x = np.cos(mu) * (1 + np.cos(n_long * mu / n_mer) * 0.5)
y = np.sin(mu) * (1 + np.cos(n_long * mu / n_mer) * 0.5)
z = np.sin(n_long * mu / n_mer) * 0.5

#对数据进行可视化
l = mlab.plot3d(x, y, z, np.sin(mu), tube_radius=0.025, colormap='Spectral')
mlab.show()
```

3D绘图函数-plot3d ()

```
plot3d(x, y, z, np.sin(mu), tube_radius=0.025,
                colormap='Spectral')
```

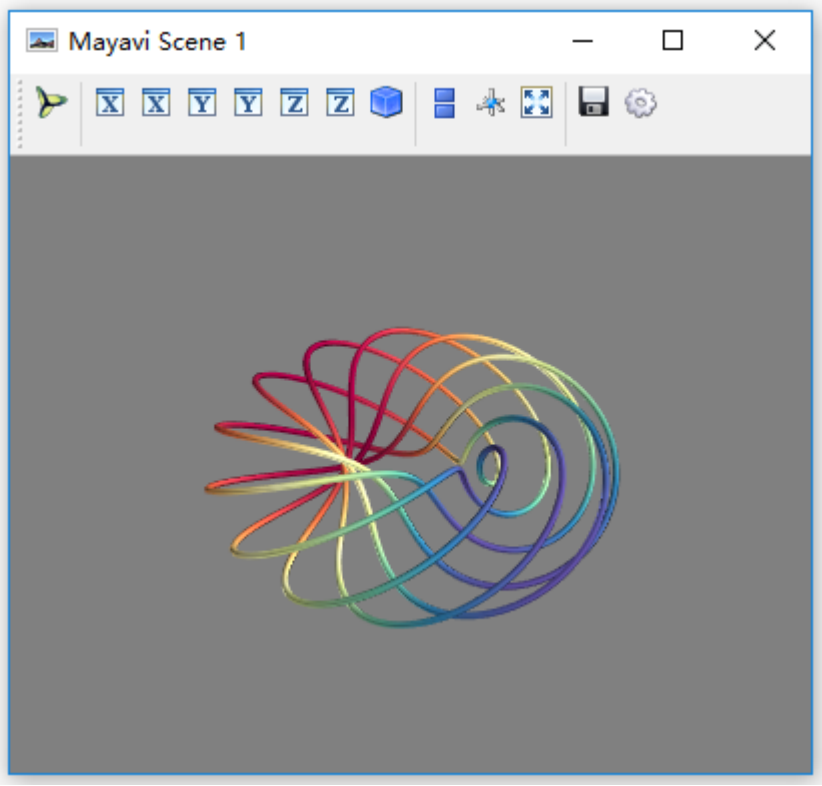
x,y,z表示numpy数组，给出了线上连续的点的位置

np.sin(mu)表示在该坐标点处的标量值

tube_radius绘制线的半径为0.025

colormap采用Spectral颜色模式。

3D绘图函数-plot3d ()



3D绘图函数-2D数据

函数	说明
imshow()	将二维数组可视化为一张图像
surf()	将二维数组可视化为一个平面，z轴描述了数组点的高度
contour_surf()	将二位数组可视化为等高线，高度值由数组点的值来确定
mesh()	绘制由三个二维数组x、y、z描述坐标点的网格平面
barchart()	根据二维、三维或者点云数据绘制的三维柱状图
triangular_mesh()	绘制由x、y、z坐标点描述的三角网格面

3D绘图函数-imshow ()

函数形式：

```
imshow(s, …)
```

s是一个二维数组,s的值使用colormap被映射为颜色

3D绘图函数-imshow ()

color、colormap、extent、figure、interpolate、

line_width、name、opacity、reset_zoom、transparent、

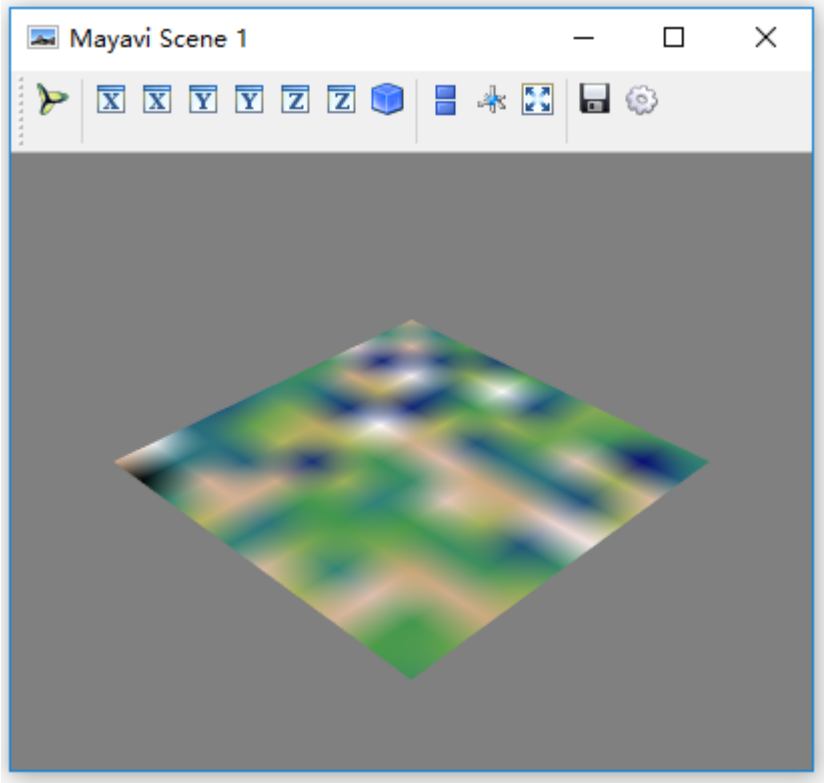
vmax、vmin

参数	说明
interpolate	图像中的像素是否被插值，该值为布尔型，默认为True

3D绘图函数-imshow ()

```
import numpy
from mayavi import mlab
#建立数据
s = numpy.random.random((10, 10))

#对数据进行可视化
img = mlab.imshow(s, colormap='gist_earth')
mlab.show()
```

3D绘图函数-imshow ()



3D绘图函数-surf()

函数形式：

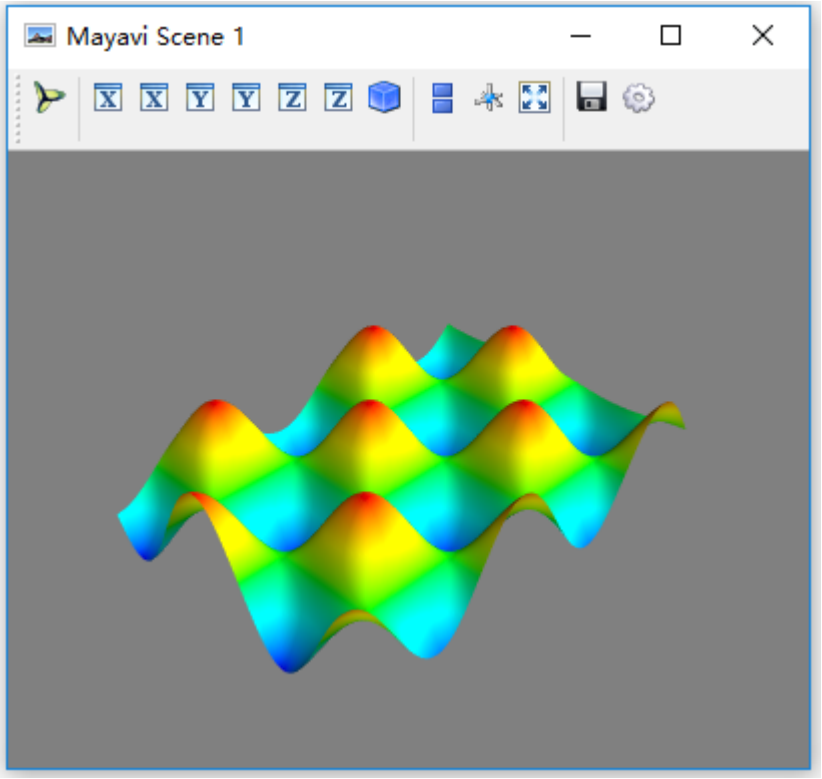
```
surf(s, …)
surf(x, y, s,…)
surf(x, y, f,…)
```

s是一个高程矩阵，用二维数组表示。

3D绘图函数-surf()

```
import numpy as np
from mayavi import mlab

def f(x, y):
    return np.sin(x - y)+np.cos(x + y)

x, y = np.mgrid[-7.:7.05:0.1, -5.:5.05:0.05]
s = mlab.surf(x, y, f)
mlab.show()
```

3D绘图函数-surf()

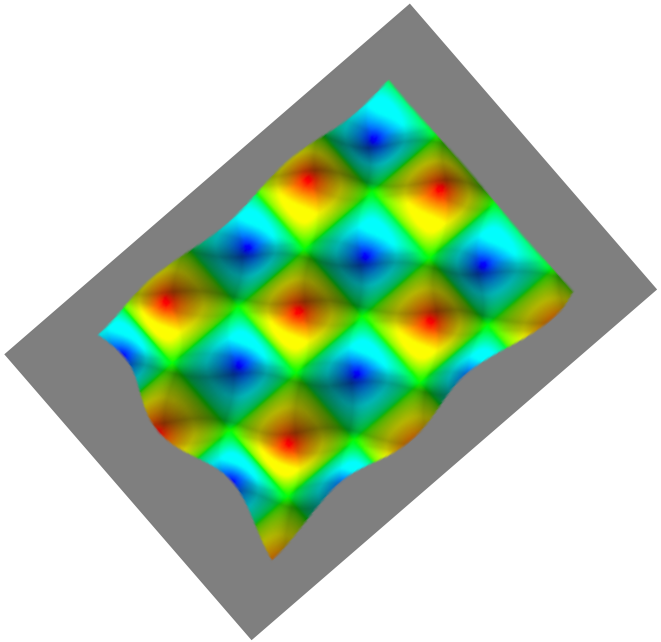


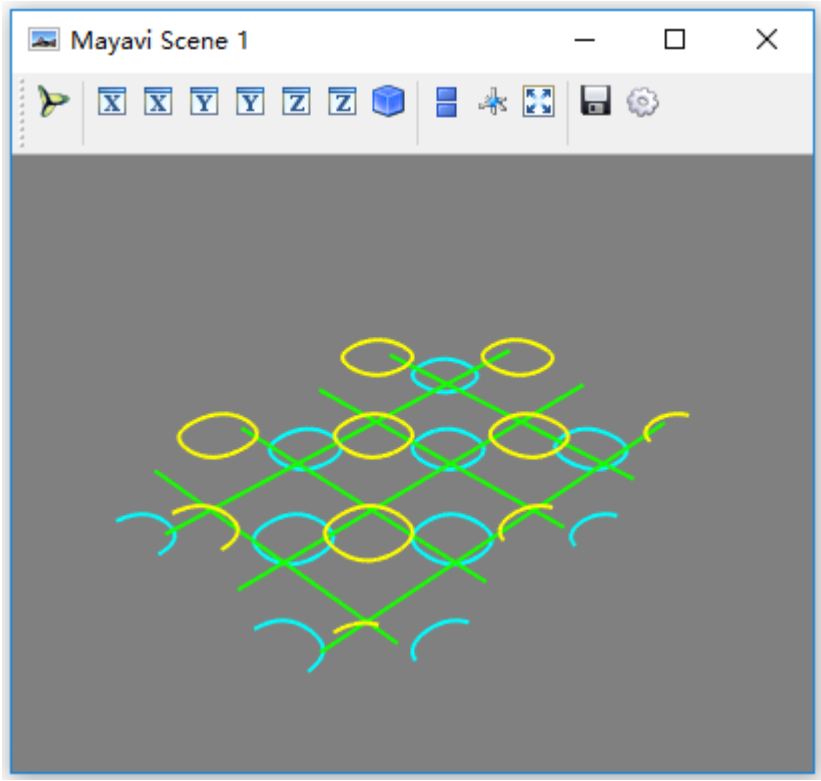
3D绘图函数-contour_surf()

与Surf()类似，但求解的是等值线

```
import numpy as np
from mayavi import mlab

def f(x, y):
    return np.sin(x - y)+np.cos(x + y)

x, y = np.mgrid[-7.:7.05:0.1, -5.:5.05:0.05]
con_s = mlab.contour_surf(x, y, f)
mlab.show()
```

3D绘图函数-contour_surf()





3D绘图函数-3D数据

函数	说明
contour3d()	三维数组定义的体数据的等值面可视化
quiver3d()	三维矢量数据的可视化，箭头表示在该点的矢量数据
flow()	绘制3维数组描述的向量场的粒子轨迹

3D绘图函数-contour3d()

函数形式：

```
contour3d(scalars, …)
contour3d(x, y, z, scalars,…)
```

scalars网格上的数据，用三维numpy数组表示。

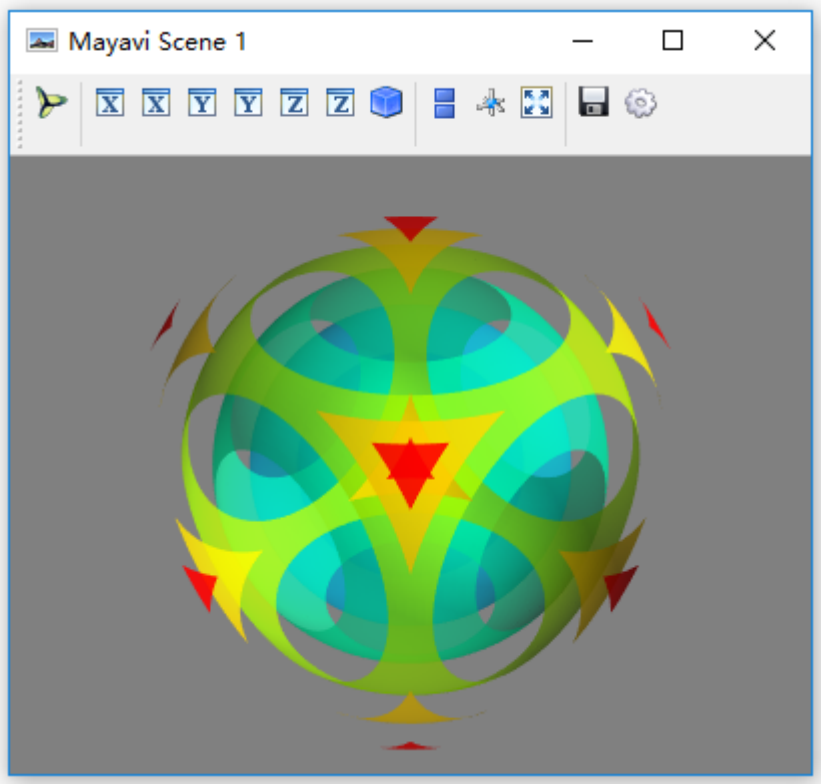
x，y，z三维空间坐标

参数	说明
contours	定义等值面的数量

3D绘图函数-contour3d()

```
import numpy
from mayavi import mlab

x, y, z = numpy.ogrid[-5:5:64j, -5:5:64j, -5:5:64j]
scalars = x * x + y * y + z * z
obj = mlab.contour3d(scalars, contours=8, transparent=True)
mlab.show()
```

3D绘图函数-contour3d()



3D绘图函数-quiver3d()

函数形式：

```
quiver3d(u, v, w …)
quiver3d(x, y, z, u, v, w …)
quiver3d(x, y, z, f, …)
```

u，v，w用numpy数组表示的向量

x，y，z表示箭头的位置，u，v，w矢量元素

f需要返回在给定位置（x，y，z）的（u，v，w）矢量

3D绘图函数-quiver3d()

```
import numpy as np
from mayavi import mlab

x, y, z = np.mgrid[-2:3, -2:3, -2:3]
r = np.sqrt(x ** 2 + y ** 2 + z ** 4)
u = y * np.sin(r) / (r + 0.001)
v = -x * np.sin(r) / (r + 0.001)
w = np.zeros_like(z)

obj = mlab.quiver3d(x, y, z, u, v, w, line_width=3, scale_factor=1)
mlab.show()
```

3D绘图函数-quiver3d()