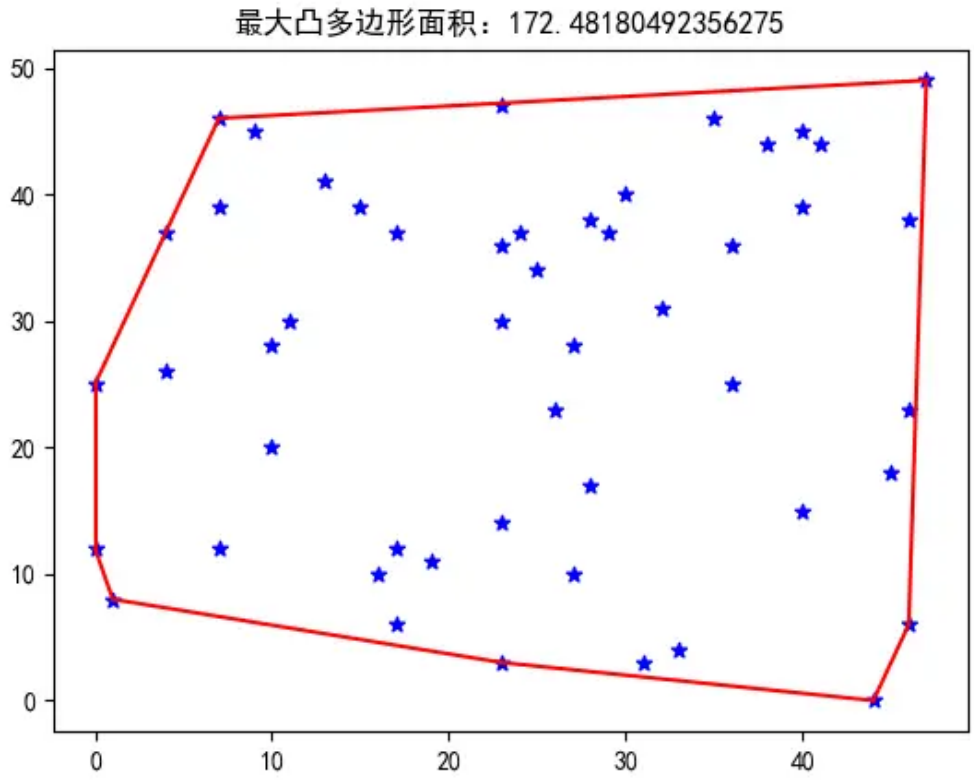
`scipy.spatial.ConvexHull`函数可以直接解决凸包问题，它的返回值有两个，分别是area和simplices。

- area：是一个值，表示最大凸多边形的面积。
- simplices：是一个尺寸为`[N,2]`的二维数组，`N`代表最大凸多边形中包含的点数，`2`表示相邻两个外围点在数组中的顺序，即他们的下标。要注意：二维数组中相邻的两个元素各自组成的连线（两点连成一线）在图中并不一定是相连的，也就是说，最大凸多边形的N-1条边，并不一定是按逆时针或顺时针生成的。

一、代码

```
import numpy as np
import random
import scipy.spatial as spt
import matplotlib.pyplot as plt
plt.rc('font', family='simhei', size=10)     # 设置中文显示，字体大小


points = np.random.randint(0, 50, (50, 2))
plt.scatter(x=points[:, 0], y=points[:, 1], marker='*', color='blue')

# 调用ConvexHull函数解决凸包问题
hull = spt.ConvexHull(points=points)


for sim in hull.simplices:
    plt.plot(points[sim, 0], points[sim, 1], 'red')
    # 数组下标用逗号隔开表示行和列分开处理。（逗号的两边还可以各加冒号限定行或列的范围）
    # 比如points[sim, 0]就表示所有点中横坐标下标为sim（两行），纵坐标下标为0（1列）的值，这些值恰好是相邻两点的x坐标，两个值，再加上points[sim, 1]也是两个值，共4个值，满足plot的参数要求。
    # plot基本参数要求：plt.plot(点1横坐标, 点2横坐标,点1纵坐标，点2纵坐标,  color='')

    plt.title("最大凸多边形面积：{}".format(hull.area)) # 图形标题

plt.show()
```

二、运行结果


最大凸多边形面积：172.48180492356275
50
40
30
20
10
0
0
10
20
30
40