最近看到 python 板块提供了不少对地图进行完美白化的方法。

比如@平流层的萝卜　python 完美白化

@非对称　利用 scipy 和 sympy 扩大三角网格形成的轮廓

也看到了 MATLAB 版对白化功能的讨论，也给出了一些方法。

比如 @斥鷃　matlab 中地图边界与掩膜的实现

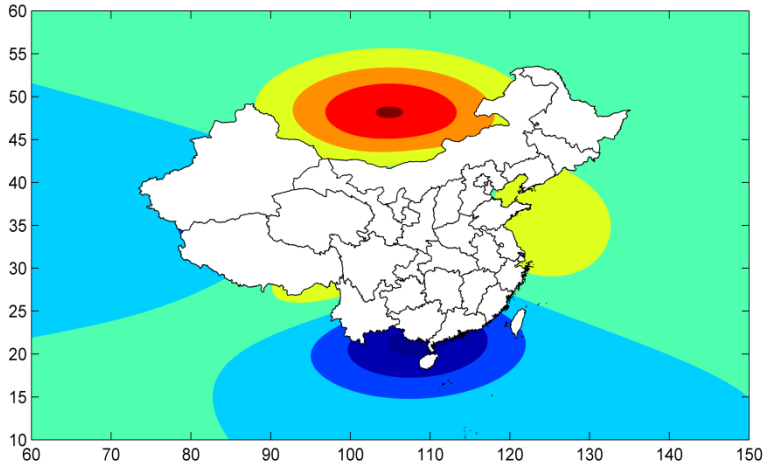
@raul928　也提供一个去地图外边界的方法

上述两种方法都能很好的对地图进行白化。

下面提供一种新的白化方法：

1) 如果需要**白化某一区域内**的话，只需要使用 **mapshow** 函数即可，以 **shapefile** 文件为例

```
z = peaks(1000);
lon = [60 150];
lat = [10 60];
[LON,LAT] = meshgrid(linspace(lon(1),lon(2),1000), linspace(lat(1),
lat(2),1000));
c = contourf(LON,LAT,z,'linestyle', 'none');
mapshow('E:\MATLAB\shp\bou2_4p.shp','displaytype','polygon',
'facecolor','w')
% 或是直接调用 maskMap 函数
% maskMap('E:\MATLAB\shp\bou2_4p.shp', false)
```



mapshow 的 displaytype 属性默认为 'line'，这里要设置为**'polygon'**。否则会出现**警告**(如果所用 shape 文件几何形状为 polygon，MATLAB 会忽视这一警告)，甚至出错。

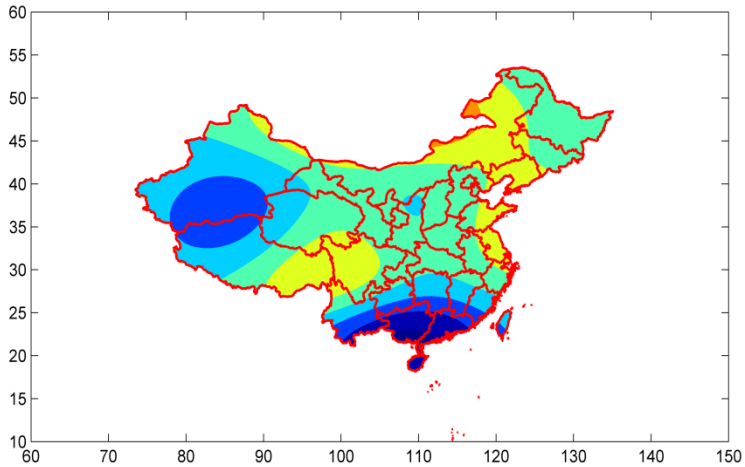
```
>> mapshow('E:\MATLAB\shp\bou2_4p.shp','displaytype','line', 'facecolor','none', 'edgecolor', 'r', 'linewidth', 1.2)
警告: Function MAPSHOW expected DisplayType 'line' to match Geometry 'Polygon'. MAPSHOW is ignoring the DisplayType value.
> In map\private\parseShowParameters at 39
  In map\private\mapstructshow at 123
  In mapshow at 228
```

2） 白化区域外部分

```
function maskmap(shapefile, masktype, varargin)
%  对所绘制图形进行白化
%  输入参数:
%       shapefile  :  shapefile文件。 字符串型或元胞
%       masktype  : 逻辑值。
%           true  :  对区域外进行白化
%                可选参数:
%                   longitudes :  图形的经度范围，二元素向量。 数值型。
%                   latitudes  :  图形的纬度范围，二元素向量。 数值型。
%                   multiprovinces : 元胞数组。 用于指定要白化的多个区域。
%                   reversemask  :  指定要白化的是 multiprovinces 元胞中指定
%                                   的区域还是元胞中指定的区域以外的部分。 逻辑值。
%                           true  :  表示白化元胞中指定的区域。 默认值。
%                           false :  表示白化元胞中指定区域以外的部分。
%           false :  对区域内进行白化
%     共享可选参数值对:
%           facecolor :  用于指定白化区域的颜色。 默认为白色。 字符型。
%                        取值为 'none'时，不起作用。
%           edgecolor :  指定的边界线的颜色。 默认为黑色。 字符型。
%           linewidth :  边界线宽度。 默认为1. 数值型。
%%
p = inputParser;
validShape = @(x) ischar(x) || iscell(x);
validLLFcn = @(x) isvector(x) && length(x) == 2;
validFaceFcn = @(x) ischar(x) && ~strcmp(x, 'none');
defaultLon = [];  defaultLat = []; defaultProv = {}; defaultReve = true;
defaultFace = 'w';  defaultLine = 'k';  defaultWidth = 1;

addRequired(p, 'shapefile', validShape)
addRequired(p, 'masktype', @islogical);
addParameter(p, 'longitudes', defaultLon, validLLFcn);
addParameter(p, 'latitudes', defaultLat, validLLFcn);
addParameter(p, 'multiprovinces', defaultProv, @iscell);
addParameter(p, 'reversemask', defaultReve, @islogical);
```

```
addParameter(p, 'facecolor', defaultFace, validFaceFcn)
addParameter(p, 'edgecolor', defaultLine, @ischar)
addParameter(p, 'linewidth', defaultWidth, @isnumeric)

parse(p, shapefile, masktype, varargin{:});

if ischar(p.Results.shapefile)
    shapefiles{1} = p.Results.shapefile;
    shapenum = 1;
else
    shapenum = length(p.Results.shapefile);
    shapefiles = p.Results.shapefile;
end

if ~isempty(p.Results.multiprovinces)
    provinces = p.Results.multiprovinces;
    pronum = length(p.Results.multiprovinces);
    s = shaperead(p.Results.shapefile);
    provs = struct2cell(s);
    allprovs = provs(end,:)';
    allprovnum = length(allprovs);
    maskindex = zeros(allprovnum,1);
    for ii = 1:pronum
        sindex = strfind(allprovs, provinces{ii});
        for jj = 1: allprovnum
            if isempty(sindex{jj})
                sindex{jj} = 0;
            end
        end
        maskindex = maskindex | cell2mat(sindex);
    end
    if p.Results.reversemask
        maskindex = ~maskindex;
    end
end

if p.Results.masktype
    longitudes = p.Results.longitudes;
    latitudes = p.Results.latitudes;
    if isempty(longitudes) || isempty(latitudes)
        error('需要指定经纬度范围！')
    end
    loninc = longitudes(1):longitudes(end);
    latinc = latitudes(1):latitudes(end);
```

```
    lone = ones(1,length(latinc)-2)*longitudes(end);
    lons = ones(1,length(latinc)-1)*longitudes(1);
    lats = ones(1,length(loninc)-1)*latitudes(1);
    late = ones(1,length(loninc)-2)*latitudes(end);
    LON.index = [loninc lone loninc(end:-1:1) lons NaN];
    LAT.index = [lats latinc late latinc(end:-1:1) NaN];
    for i = 1:shapenum
        s = shaperead(shapefiles{i});
        LON.index = [LON.index s.X];
        LAT.index = [LAT.index s.Y];
    end
    if ~isempty(p.Results.multiprovinces)
        LON.index = [LON.index s(maskindex).X];
        LAT.index = [LAT.index s(maskindex).Y];
    end
    mapshow(LON.index, LAT.index, 'Displaytype', 'polygon', 'facecolor',
p.Results.facecolor, 'edgecolor', p.Results.edgecolor, 'linewidth',
p.Results.linewidth);
else
    mapshow(p.Results.shapefile, 'displaytype','polygon', 'facecolor',
p.Results.facecolor, 'edgecolor', p.Results.edgecolor, 'linewidth',
p.Results.linewidth)
end
end
```



maskMap 函数提供了一个**逻辑值**选项用于选择区域内白化还是区域外白化。如果

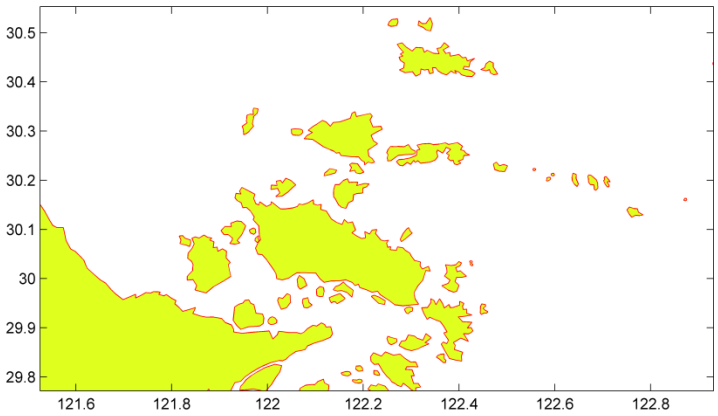
是区域内白化，只需要提供 **shapefile** 和 **masktype** 两个参数即可，如果选择区域外白化，除上述两个参数之外，还需要提供‘**longitudes**’ 和 '**latitudes**' 两个参数，调用方式和 matlab 内置函数调用方式类似。

shapefile 参数可以是 **字符串**型，也可以是 **元胞数组**，其元素为字符串。

为**字符串**时，直接使用包含绝对路径信息的 shape 文件名，
为**元胞数组**时，为包含多个省份的元胞数组，用以白化多个区域。

```
>> shapefiles = {'E:\MATLAB\shp\新疆.shp','E:\MATLAB\shp\山东.shp','E:\MATLAB\shp\台湾.shp','E:\MATLAB\shp\黑龙江.shp'}

shapefiles =

    'E:\MATLAB\shp\新疆.shp'    'E:\MATLAB\shp\山东.shp'    'E:\MATLAB\shp\台湾.shp'    'E:\MATLAB\shp\黑龙江.shp'
```

无论是白化区域内还是白化区域外，均可以通过设置 **facecolor** ， **edgecolor**, **linewidth** 等属性值设置**白化区域的颜色，边界线颜色，线宽**等。鉴于白化时通常只需要改变这几个属性，目前仅支持这几个属性。



从图中可以很明显的看到，即使一些小的岛屿也能很完美的进行白化，**没有出现任何锯齿**。

参考@平流层的萝卜 的 python 完美白化的帖子，也支持了**白化几个省份的部分**。所不同的是，此程序是通过**指定省份名称进行白化（省份名要和所读取的 shp 文件中省份名一致），而不是 shp 文件中省份的代码(**当然也能够这样实现)。

程序中提供了两个参数用以实现此功能。两个参数分别是：

multiprovinces ： 元胞数组。用以指定要白化的区域。

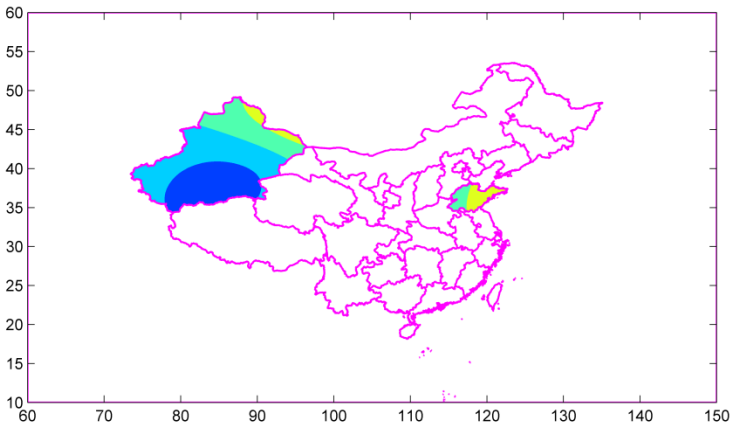
reversemask ： 逻辑值。 用以确定是否反转白化区域。

如果指定江苏省和台湾。

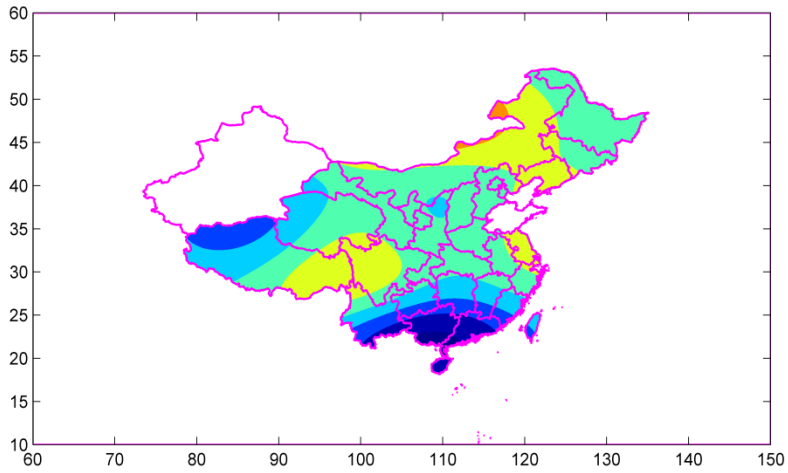
true ： 此时白化新疆维吾尔自治区和山东以外的区域

false ： 白化新疆和山东地区。

maskmap('E:\MATLAB\shp\bou2_4p.shp',true,'lon',lon,'lat',lat, 'facecolor','w', 'edgecolor', 'm', 'linewidth', 1.2, 'multiprov',{'新疆维吾尔自治区','山东省'}, 'reversemask', true)



maskmap('E:\MATLAB\shp\bou2_4p.shp',true,'lon',lon,'lat',lat, 'facecolor','w', 'edgecolor', 'm', 'linewidth', 1.2, 'multiprov',{'新疆维吾尔自治区','山东省'}, 'reversemask', false)

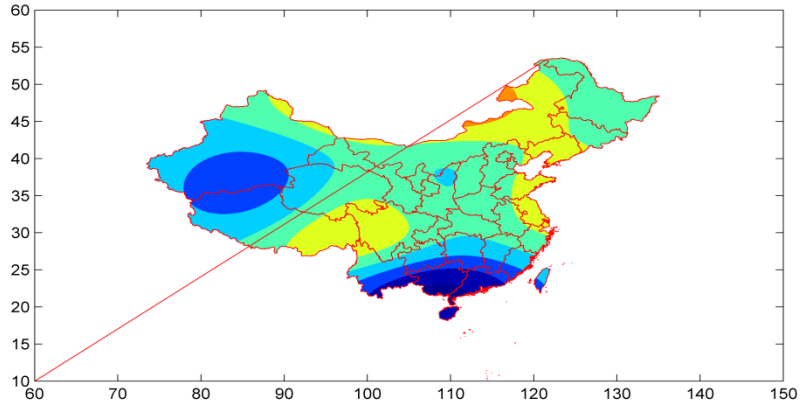


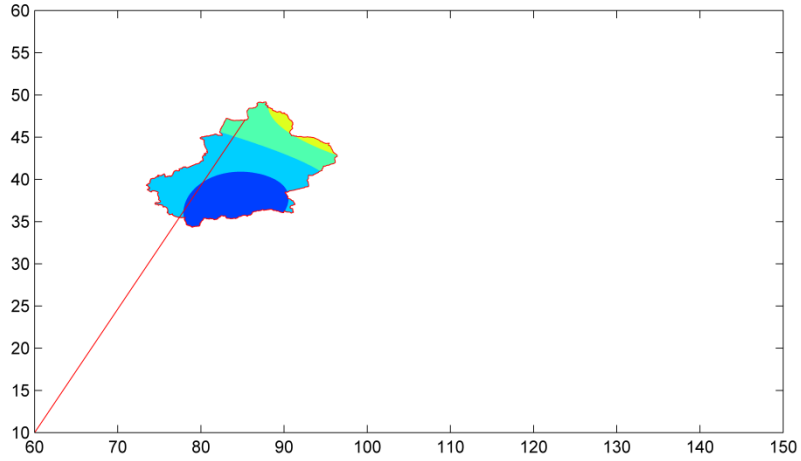
函数中的参数均可进行简写，MATLAB 会自动对应，但是简写时不要出现相同的简写，要不然会出错。至于为什么会出错将在另一个主题中进行解释。

====如果你对实现算法不感兴趣的话，下面的内容可直接忽略==========

大家都知道，绘制好图形后，MATLAB 是以一个窗口的形式显示的，而这个图形的 **X 和 Y 轴均是要设置上下限**的，去掉图中的地图边界线，这个窗口就是一个多边形。加上地图之后，在这个多边形内就会出现又一个多边形，这就相当于**嵌套图形**了。既然可以把地图内白化掉，那么只要把两个多边形之间的部分设置为同一个颜色（白色）不就可以了吗？也就是**重新制作一个类似省份的地形轮廓数据**即可，把这个当成一个具体地方。只是这个地方是我们创造的而已。

但是在实现过程中遇到了一些问题，比如：



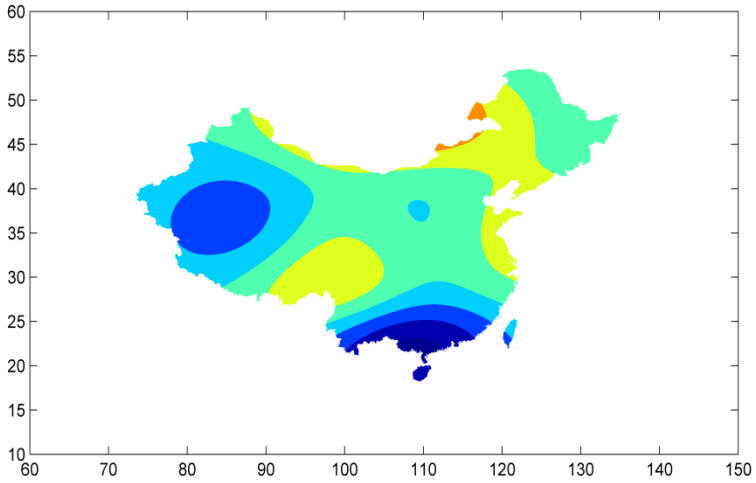


可以很明显的看出，图中**多了一条和边界线颜色相同的直线**，而这一条线就是**地形边界的起始点和矩形的结束点相连接所形成**的。

可以通过两种方法解决上述问题。

1） 通过设置 mapshow 函数的 **edgecolor** 属性值为 **‘none’**。

这是一开始所使用的方法，但是如果这样设置的话就会出现一个新的问题。

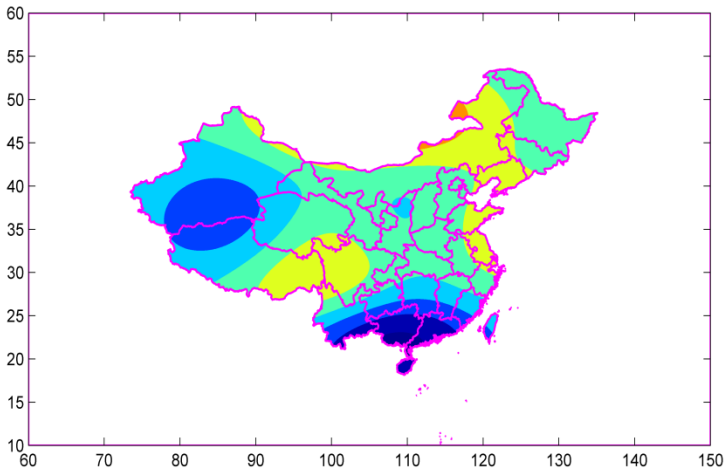


边界线会消失不见，如果要添加边界线的话，**只能在调用 maskMap 白化之前或之后**执行以下命令

mapshow('E:\MATLAB\shp\bou2_4p.shp', 'displaytype', 'polygon','facecolor', 'none')

2） 在构成矩形的点阵中**添加 NaN** 作为最后一个点即可完美解决。这也是目前函数所使用的方法。大家可以测试一下。

```
maskMap('E:\MATLAB\shp\bou2_4p.shp',true,'lon',lon,'lat', lat,
'facecolor','w', 'edgecolor', 'm', 'linewidth', 1.2)
print('-dpng', '-r600', 'new.png') %  用于输出高质量的图片
```



直接调用上述函数即可。示例代码如下：

```
z = peaks(1000);
lon = [60 135];
lat = [10 60];
[LON,LAT] = meshgrid(linspace(lon(1),lon(2),1000), linspace(lat(1),
lat(2),1000));
c = contourf(LON,LAT,z,'linestyle', 'none');
maskMap('E:\MATLAB\shp\bou2_4p.shp', true, 'lon',lon,'lat',lat,
'facecolor', 'w', 'edgecolor', 'b', 'linewidth', 1.5)
```

由于我这**没有县级的 shp 文件**，因此无法测试能否添加县级部分。考虑到添加省级 shp 文件时没有出现问题，如果使用包含县级信息的 shp 文件的话，应该也是可以的。

注意：使用此函数进行白化时，请使用几何形状为'polygon'的 shp 文件。比如 bou1_4p.shp，bou2_4p.shp 等文件，其中的'p'即表示几何形状为'polygon'，如果是'l'，则表示几何形状为'line'。

使 用 过 程 中 如 遇 到 问 题 欢 迎 反 馈 ！

==========

2016.9.8 解决调用 **maskMap** 函数白化区域内出错的问题

2016.9.9 使用方案 **2** 替换方案 **1**，解决白化时出现一条直线的问题

2016.9.11 解决白化多个省份时需额外添加中国边界的问题

后话：解决了白化多个省份时需要额外添加中国边界的问题之后，程序中有一部分功能重复了。

shapefile 参数的元胞数组可用于提供多个省份的 **shp** 文件进行白化；

multiprovinces 参数也提供了白化多个省份的功能

出现这种问题也是由于一开始没有好的办法解决白化多个省份需额外添加中国地图边界的问题。目前解决了这一问题，本想直接去除 **shapefile** 参数值元胞数组类型的功能，但是鉴于两者并不冲突，因此暂时搁置一旁。

白化多个省份时建议使用 **multiprovinces** 参数进行，可同时配合 **reversemask** 参数。