

第 8 节 GEE 的数据类型 (Landsat Images, Confusion Marix)

8.1 Landsat Images

GEE 中内置了若干针对 Landsat 卫星的命令，对这些命令的合理使用能增加分析效率。下边介绍 qualityMosaic 命令，代码及执行效果如下：

```
var L8 = ee.ImageCollection("LANDSAT/LC08/C01/T1_TOA")
    .filterBounds(ee.Geometry.Point(107.193, 29.1373))
    .filterDate('2018-01-01','2018-12-31')
    .select('B[2-5]');

function add_NDVI (image){
  var NDVI = image.normalizedDifference(['B5','B4'])
  return image.addBands(NDVI)
}
var L8_NDVI = L8.map(add_NDVI)
var NDVI_Quality = L8_NDVI.qualityMosaic('nd')

Map.setCenter(106.618, 29.6653,9)
Map.addLayer(NDVI_Quality,{"bands":["B5","B4","B3"],"max":0.5,"gamma":1},'NDVI qualified')
```

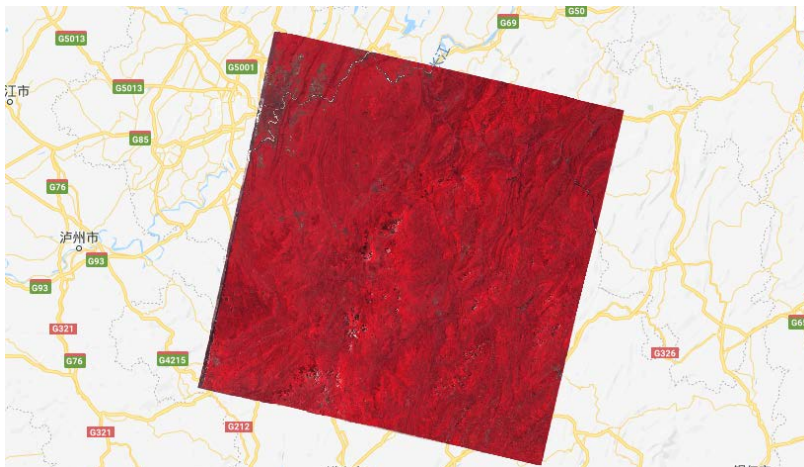


图 8.1 qualityMosaic 操作

该命令的功能是“选出最大值像素”。具体来说，本例的 Image Collection 已经通过.map 命令给其中的每个图像都增加了 NDVI(波段名为 nd)数据，当利用 qualityMosaic 命令并且选择 nd 波段为参数时，其结果就是将 Image Collection 的 NDVI 最大的像素挑选出来并组合成一副图像。

下边介绍光谱分离命令，代码及执行效果如下：

```
//加载“建设用地”，“水体”和“绿地”的 Geometry
var landsat5 = ee.ImageCollection("LANDSAT/LT05/C01/T1"),
    China_provinces = ee.FeatureCollection("users/wangjinzhalala/China_Provinces"),
    CQ_point = ee.Geometry.Point([106.54876499135139, 29.594019931841714]);
var bare = ee.Geometry.Polygon(
  [[[106.55483080849444, 29.529437948319515], [106.55139765445824, 29.52555451875115],
    [106.55654738551414, 29.52346337953001], [106.55843565976829, 29.527496216222946]]]);
var water = ee.Geometry.Polygon(
  [[[106.57311345239873, 29.569534351977307], [106.56727579226686, 29.56833967471457],
    [106.56933614290324, 29.5656515992401]]]);
var veg = ee.Geometry.Polygon(
  [[[106.77944972250918, 29.553706593461722],
    [106.7749865267084, 29.552511969146664], [106.77773310873965, 29.548330672830648]]]);
//定义时间范围
var ChongQing = China_provinces.filterBounds(CQ_point);
var start_date = '2008-01-01';
var end_date = '2008-12-31';
//合成无云图像
var filtered = landsat5.filterBounds(CQ_point).filterDate(start_date, end_date)
var composite = ee.Algorithms.Landsat.simpleComposite({collection: filtered, asFloat: true});
//确定运算波段
var unmixImage = composite.select(['B1', 'B2', 'B3', 'B4', 'B5', 'B7']);
//获取解缠的“解”
var bareMean = unmixImage.reduceRegion(ee.Reducer.mean(), bare, 30).values();
var waterMean = unmixImage.reduceRegion(ee.Reducer.mean(), water, 30).values();
var vegMean = unmixImage.reduceRegion(ee.Reducer.mean(), veg, 30).values();
var endmembers = ee.Array.cat([bareMean, vegMean, waterMean], 1);
//获取解缠的“缠”
var arrayImage = unmixImage.toArray().toArray(1);
//进行解缠
var unmixed = ee.Image(endmembers).matrixSolve(arrayImage);
var unmixedImage = unmixed.arrayProject([0]).arrayFlatten(['bare', 'veg', 'water']);
//把结果加载到底图上
Map.centerObject(CQ_point, 8)
Map.addLayer(composite, {bands: ["B4", "B3", "B2"], max: 0.3}, "Simple Composite")
Map.addLayer(unmixedImage, {}, 'Unmixed');
```

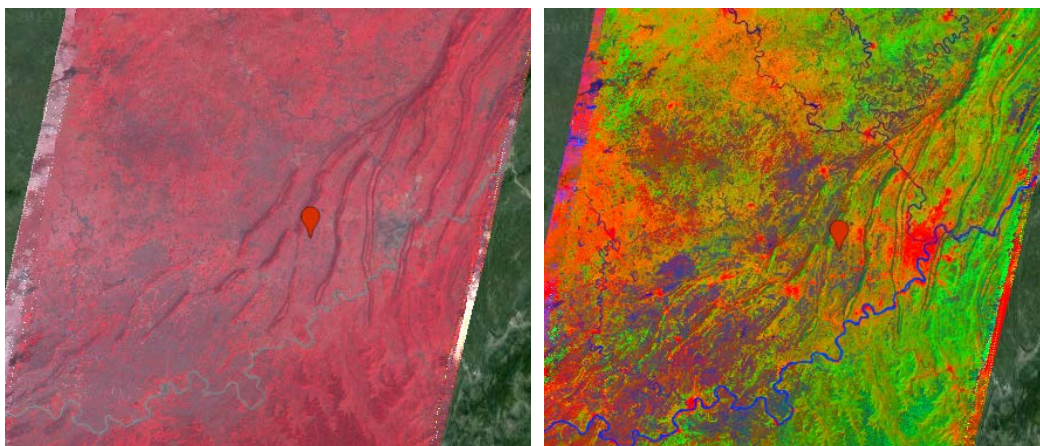


图 8.2 光谱解缠操作

光谱分离可以看做一个解方程的过程。简单的来说，对于 30m×30m 的像素来说，其占地面积为 1.34 亩，在这样大的面积上，如果有一半的水和一半的绿地，那么理论上它的光谱里应该有一半来自水，一半来自绿地。对像素求取地物光谱的比例的过程就是光谱分离，其概念方程如下：

$$\text{像素} \times \text{比例} = \text{终端像元}$$

其中，终端像元是已知的，像素在图像的不同位置有着不同的值，那么我们就可以求出不同位置上不同地物的光谱比例。本例中将地物分成了“建设用地”，“水体”和“绿地”三类，那么就可以求出不同位置这三种地物的光谱比例。配合显色设置，图上越红的地区建设用地的比例就越大，越蓝的地区水体的比例越大，越绿的地区绿地的比例越大。

下边介绍标归一化差值的计算命令，代码如下：

```
var l8 = ee.ImageCollection("LANDSAT/LC08/C01/T1");
var l8_one = l8.filterBounds(ee.Geometry.Point(106.705, 31.881))
               .filterDate('2016-01-01','2018-12-31')
               .sort('CLOUD_COVER').first()

var NDVI_1 = l8_one.normalizedDifference(['B5','B4'])
var NDVI_2 = l8_one.expression('(B4+B5)/(B4-B5)',{
  'B5': l8_one.select('B5'),
  'B4': l8_one.select('B4')
})

print(NDVI_1,NDVI_2)
Map.addLayer(NDVI_1,{"min":-0.1,"max":0.4,"palette":["0f0bff","ff3305","fbff1b","0bff1e"]})
```

本例中，通过将.normalizedDifference(['B5','B4']) 与 .expression('(B4+B5)/(B4-B5)')对比可以发现，归一化差值计算本质上是将两个参数进行固定数学运算的命令模板，即“(前+后)/(前-后)”。因此可以合理运用这个命令，以计算 NDVI 或者 NDWI 等归一化参数，以提高效率。

本例代码的执行效果如下：

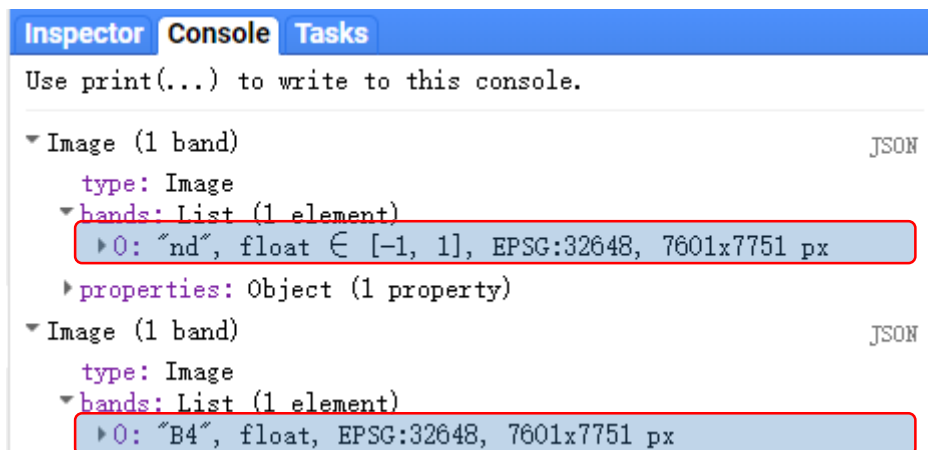


图 8.3 归一化差值计算

下边介绍标 Landsat 的云量指数计算命令，代码及运行效果如下：

```
var l8_TOA = ee.ImageCollection("LANDSAT/LC08/C01/T1_TOA")
    .filterBounds(ee.Geometry.Point(106.2608, 29.5755))
    .first()
var Cloud_Rate = ee.Algorithms.Landsat.simpleCloudScore(l8_TOA)

print(Cloud_Rate)

Map.setCenter(106.14, 30.3488, 8)
Map.addLayer(Cloud_Rate, {bands: ['B5', 'B4', 'B3'], max: 0.3})
Map.addLayer(Cloud_Rate.select('cloud'))
```

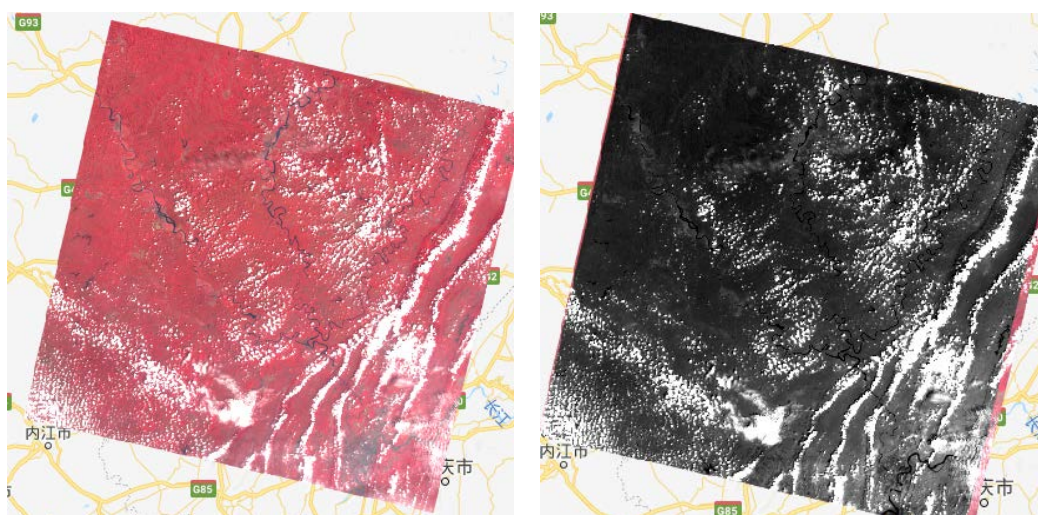


图 8.4 Landsat 的云量指数

下边介绍标 Landsat 的简单去云合成命令，代码及运行效果如下：

```
var L8_RAW = ee.ImageCollection("LANDSAT/LC08/C01/T1")
    .filterBounds(ee.Geometry.Point(106.1125, 30.3772))
    .filterDate('2018-01-01','2018-12-31')
var L8_Siple_Composite =
ee.Algorithms.Landsat.simpleComposite({collection:L8_RAW,asFloat:true})

print(L8_RAW,L8_Siple_Composite)
Map.setCenter(106.1125, 30.3772,8)
Map.addLayer(L8_Siple_Composite,{bands:['B5','B4','B3'],max:0.3})
```

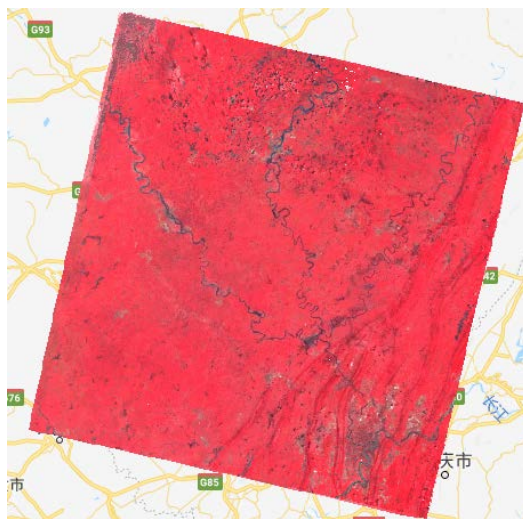


图 8.5 Landsat 的简单去云合成

本节介绍的 Landsat 系列命令如下，尝试回忆其格式与功能。

```
LansatCollection.qualityMosaic()
LansatCollection.unmixing()
LansatCollection.normalizedDifference()
ee.Algorithms.Landsat.simpleCloudScore()
ee.Algorithms.Landsat.simpleComposite()
```

8.2 Confusion Matrix

混淆矩阵常用于分类图像的分类质量验证，本节以下边的矩阵为例对混淆矩阵进行介绍。

表 8.1 混淆矩阵举例

	0	1	2	3	4
0	1817491	1038523	5543	240068	1240
1	1016690	2838574	745	137438	330
2	12611	2055	115033	142	84
3	150432	228089	24	94365	78
4	5104	2466	745	283	50583

下边介绍标混淆矩阵的创建，代码及运行效果如下：

```
var Array_Transform = ee.Array( [ [1817491,1016690,12611 ,150432,5104 ],  
                                  [1038523,2838574,2055 ,228089,2466 ],  
                                  [5543 ,745 ,115033,24 ,745 ],  
                                  [240068 ,137438 ,142 ,94365 ,283 ],  
                                  [1240 ,330 ,84 ,78 ,50583]]);  
  
var Confusion_Matrix = ee.ConfusionMatrix( Array_Transform );  
print( Confusion_Matrix );
```

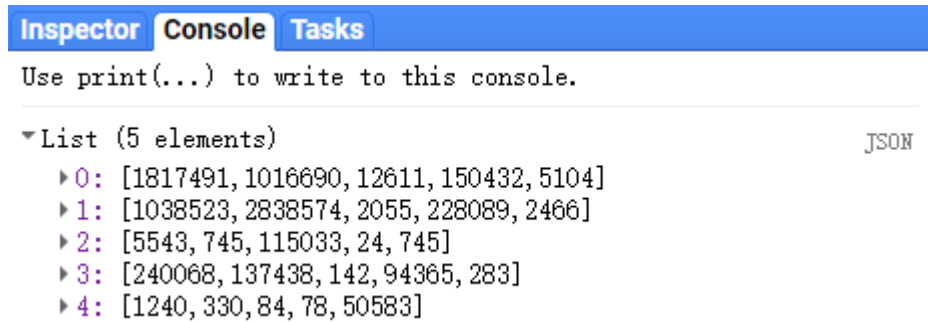


图 8.6 混淆矩阵的创建

下边介绍标混淆矩阵的质量结果计算，代码及运行效果如下：

```
var Array_Transform = ee.Array( [ [1817491,1016690,12611 ,150432,5104 ],  
                                  [1038523,2838574,2055 ,228089,2466 ],  
                                  [5543 ,745 ,115033,24 ,745 ],  
                                  [240068 ,137438 ,142 ,94365 ,283 ],  
                                  [1240 ,330 ,84 ,78 ,50583]]);  
  
var Confusion_Matrix = ee.ConfusionMatrix( Array_Transform );  
var Matrix_Kappa = Confusion_Matrix.kappa()  
var Matrix_Accuracy = Confusion_Matrix.accuracy()  
var Matrix_ProducersAccuracy = Confusion_Matrix.producersAccuracy()  
var Matrix_ConsumersAccuracy = Confusion_Matrix.consumersAccuracy()  
var Matrix_Order = Confusion_Matrix.order()  
  
print(Matrix_Kappa);  
print(Matrix_Accuracy );  
print(Matrix_ProducersAccuracy);  
print(Matrix_ConsumersAccuracy);  
print(Matrix_Order);
```

Inspector	Console	Tasks
Use print(...) to write to this console.		
0.35559755613519634		
0.6336142897502892		
▸ List (5 elements)		JSON
▸ List (1 element)		JSON
▸ [0, 1, 2, 3, 4]		JSON

图 8.7 质量结果计算

对混淆矩阵的质量参数含义的理解还需要结合其数学含义进一步的学习。



This work is licensed under a [Creative Commons Attribution-NonCommercial 4.0 International License](https://creativecommons.org/licenses/by-nc/4.0/).

本材料由王金柱（西南大学&迪肯大学）创作。如有需要请与我联系。
This doc contributed by Jinzhu Wang of Southwest University & Deakin University.
Email: wangjinzhu1a@gmail.com