Регрессионная нормализация данных дистанционного зондирования в R

Описание метода и реализации

Обсудить в форуме Комментариев — 4

При совместном анализе нескольких изображений часто необходимо их привести одно к другому статистически, чтобы использовать единую методологию классификации или другого численного анализа. Рассмотрим на примере двух изображений процедуру приведения их друг к другу с помощью регрессионной нормализации. Данный способ является одним из наиболее часто используемых в дистанционном зондировании и описан в классической работе: Collins, J.B., & Woodcock, C.E. (1996). An assessment of several linear change detection techniques for mapping forest mortality using multitemporal Landsat TM data. Remote Sensing of Environment, 56, 66-77

Согласно этой методологии на двух снимках находятся относительно неизменившиеся области (pseudo invariant targets), обычно в двух противоположных концах спектра (например вода и выходы скальных обнажений или чистый песок). Значения спектральных яркостей регрессируются друг другу поканально. Затем коэффициенты регрессии используются чтобы пересчитать одно из изображений, приблизив его таким образом численно к другому.

Итак:

Данные: два снимка, один - исходный, тот который мы хотим приблизить к второму - опорному.

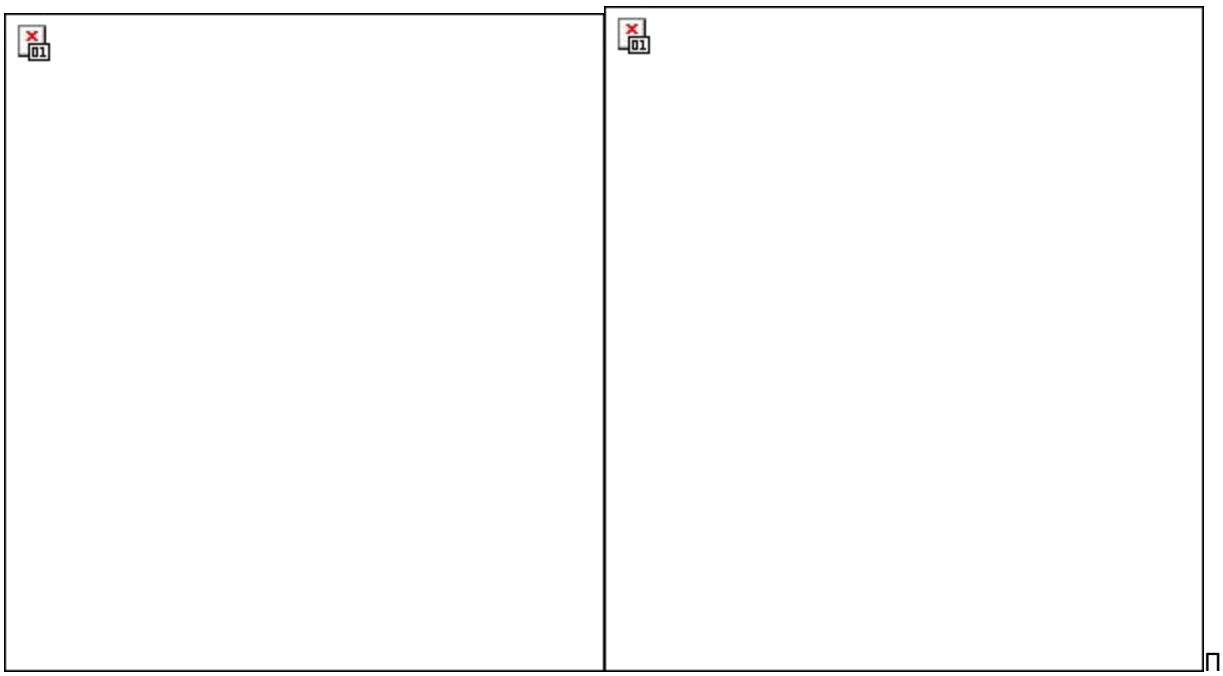
Программное обеспечение: R и одно из средств профилирования (по выбору), разумеется помимо R может быть использованы и другие пакеты, например полученные коэффициенты регрессии могут быть использованы в Modeller ERDAS IMAGINE и другом ПО.

Получаем набор значений

Для начала работы, необходимо получить набор значений, для которых будет получены коэффициенты регрессии. Получить эти значения можно множеством разных способов, один из них - идентифицировать на паре снимков 20-30 точек в областях не подвергшихся каким либо изменениям между временем получения первого и второго снимка. Разумеется, это не касается фенологических изменений.

Успешность данного метода нормализации целиком зависит от правильность выбора данных для получения коэффициентов регрессии. Данные не должны собираться в районе сильных изменений. Следует также учитывать сложно привязки отдельных пикселей, поэтому точки должны ставиться сериями 2x2 или 3x3 с последующим удалением выбросов или взятием среднего/медианы. Так же следует учитывать возможное смещение снимков относительно друг друга, это особенно актуально для данных низкого разрешения.

Пример графика распределения яркостей в первом канале тестового изображения


Пример расставления нескольких серий точек

Точки могут создаваться в векторном слое. Дальше этот слой может быть использован в одном из инструментов профилирования для экстракции поканальных значений. Например можно использовать наше расширение для Arcview.

Результатом работы инструмента профилирования является таблица, где каждой точке соответствует запись с набором значений некотором количестве полей. Это количество равно количеству каналов исходного и опорного изображения.

Получаем коэффициенты регрессии

После получения данных для работы перейдем в статистический пакет для вычисления регрессии по каждому каналу.

Мы работаем с векторными данными хранящимися в shape-формате, в свою очередь хранящем атрибутику в формате dbf. Будем считывать атрибутику прямо из него, для этого понадобится библиотека foreign, которая может считывать dbf. Загрузим данные в переменную data:

```
library(foreign)
samplename = "D:\\Work\\profile.dbf"
data = read.dbf(samplename)
```

Создадим два вектора, которые будут хранить значения для intercept и slope:

```
intercepts = numeric()
slopes = numeric()
```

И в цикле, по количеству каналов (в нашем случае их 5) расчитаем коэффициенты регрессии:

```
numbands=5
for (i in 1:numbands) {
        model = lm(data,i+numbands~data,i)
        intercept = model$coefficients1
        intercepts = cbind(intercepts,intercept)
        slope = model$coefficients2
        slopes = cbind(slopes,slope)
}
```

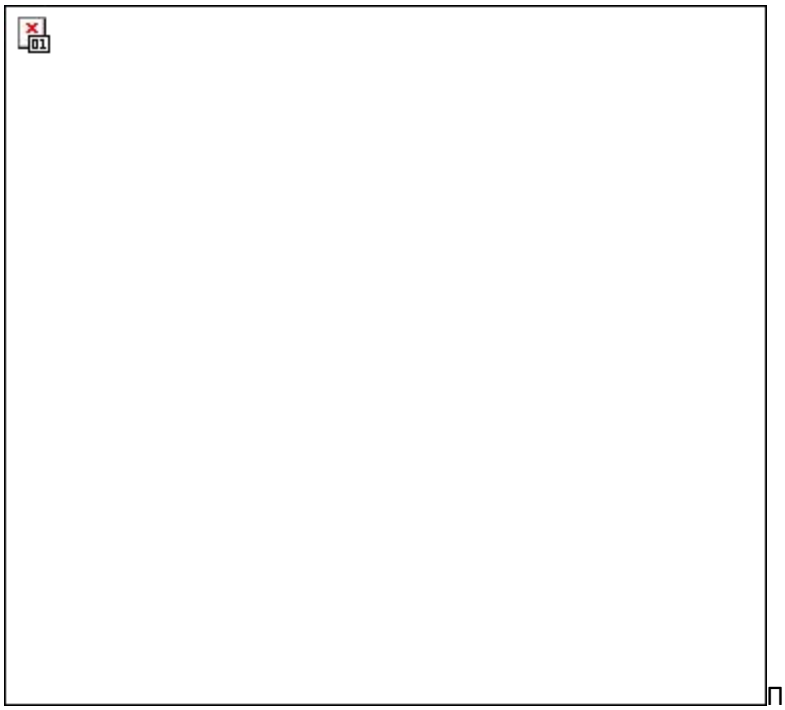
Сохраним полученные коэффициенты для дальнейшего использования в текстовый файл.

```
res = rbind(intercepts,slopes)
```

```
write.table(res,file=outputfile,sep=",",row.names=F,col.names=F)
```

Результат регрессии можно оценить визуально на предмет выбросов, облако точек должно стремиться к эллипсу, чем уже, тем лучше. По возможности от атипичных выбросов необходимо избавиться, либо убиранием самих точек и перерасчетом, либо редактирования набора данных непосредственно в R.

```
plot(data,4,data,4+numbands)
abline(lm(data,4+numbands~(data,4)))
```



Пример распределения данных по одному из каналов и линейная регрессия (x - исходный снимок, y - опорный снимок).

Нормализуем один снимок относительно другого

После того как коэффициенты регрессии для каждого канала получены, можно приступать к изменению исходного снимка. Средства R позволяют выполнять модификацию растров непосредственно в среде, без привлечения дополнительных средств. Работа с растровыми данными подробно описана в соответствующей статье. Продемонстрируем наши действия еще раз (fullimagename хранит путь к исходному изображению, height и width будут хранить высоту и ширину изображения для использования далее):

```
library(rgdal)
image = new("GDALReadOnlyDataset", fullimagename)
imagedata = data.frame(getRasterTable(image)3:numbands+2)
width = dim(image)2
height = dim(image)1
```

Сохраним его привязку, чтобы пересоздать ее для нового файла:

```
pixsize = as.numeric(GDALinfo(fullimagename)6)
originx = as.numeric(GDALinfo(fullimagename)4) + pixsize/2
originy = as.numeric(GDALinfo(fullimagename)5) + pixsize*height - pixsize/2
wldinfo = rbind(pixsize,0,0,-1*pixsize,originx,originy)
```

Применим регрессионные коэффициенты:

```
res = 0
for (i in 1:numbands) {
        newdata = as.numeric(interceptsi) + as.numeric(slopesi)*imagedata,i
        res = cbind(res,newdata)
}
res = round(res,-1)
```

Создадим новый растр в формате ERDAS IMAGINE:

```
img_driver <- new("GDALDriver", "HFA")
imgraster <- new("GDALTransientDataset", img_driver, height, width, numbands, 'Int16')
```

Для каждого канала запишем результат пересчета с использованием регрессионных коэффициентов:

```
for (i in 1:numbands) {
        resmtx = matrix(res,i,ncol=height,nrow=width)
        resmtxis.na(resmtx) <- 0
        putRasterData(imgraster, resmtx, band=i)
}
```

Наконец сохраним результаты:

```
img_file = paste(inputdir,imagename,"-reg",".img",sep="")
wld_file = img_file
substr(wld_file, nchar(wld_file)-2, nchar(wld_file)) <- "igw"
write(wldinfo,wld_file,sep="\n")
saveDataset(imgraster, img_file)
```

и подчистим сессию GDAL:

```
GDAL.close(imgraster)
GDAL.close(img_driver)
```

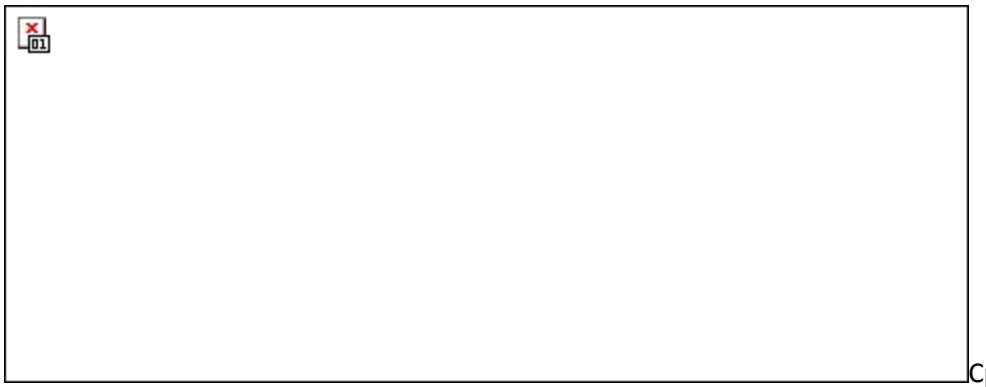
Результаты

После регрессионной нормализации обычно проверяют насколько изменилась статистика изображения. Для этого используют еще один набор данных, содержащий некоторые области, обычно изменившиеся и не изменившиеся. Для визуализации удобно применять boxplot'ы.

```
boxplot(class1data1,i,class1data2,i,class2data1,i,class2data2,i)
axis(1,labels=c("2006","2007","2006","2007"),at=c(1,2,3,4))
```

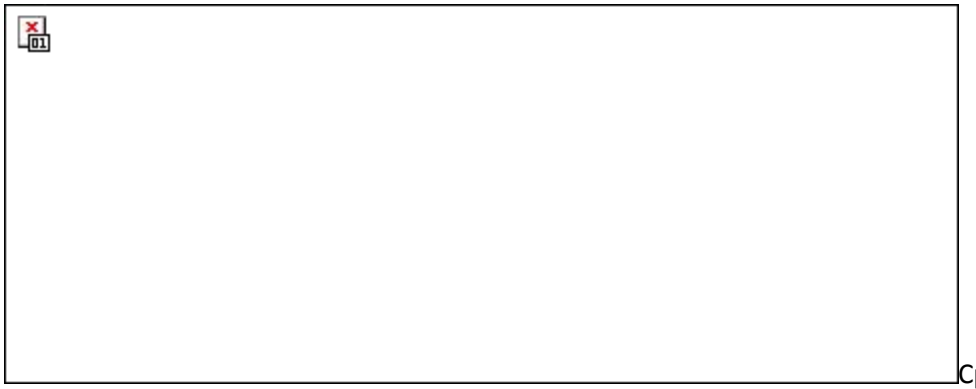
Приведем пример.

До нормализации:



Сравнение двух наборов по сгоревшим (первая пара) и несгоревшим территориям (вторая пара) в одном из каналов

И те же данные после нормализации:


Сравнение двух наборов по сгоревшим (первая пара) и несгоревшим территориям (вторая пара) в том же канале

Заключение

Мы продемонстрировали простой механизм нормализации двух изображений с помощью регрессии в R. Если у вас есть дополнения к этой статье и/или вопросы, пожалуйста оставляйте их на форуме или отправляйте с помощью формы отправки сообщения.

Полные версии скриптов для R:

- Получение регрессионных коэффициентов
- Применение регрессионных коэффициентов

Обсудить в форуме Комментариев — 4

Ссылки по теме

- Работа с растровыми данными в R: rgdal
- Collins, J.B., & Woodcock, C.E. (1996). An assessment of several linear change detection techniques for mapping forest mortality using multitemporal Landsat TM data. Remote Sensing of Environment, 56, 66-77

Последнее обновление: March 14 2011

Дата создания: 27.05.2009
Автор(ы): Максим Дубинин