Chapter 2: Vegetation Indices

Chris Holden

03/24/2015

As seen in the previous lesson, the `Raster*` objects within the `raster` package support basic arithmetic. Now that we can read our data into the computer and we know how to do some math, let's calculate some vegetation indices.

The Normalized Difference Vegetation Index (NDVI) is so ubiquitous that it even has a Wikipedia entry. If you're here for learning how to do remote sensing image processing using GDAL and R, I suspect you don't need any introduction to this section. If you need a refresher, please visit the Wikipedia URL for NDVI.

Read in imagery

We'll use the same Landsat 7 data as in Chapter 1:

```
library(raster)
```

```
## Loading required package: sp
```

```
if (file.exists('LE70220492002106EDC00_stack.gtif') == F) {
    download.file(url='https://raw.githubusercontent.com/ceholden/open-geo-tutorial/master/example/LE70
                  destfile='LE70220492002106EDC00_stack.gtif', method='curl')
}

le7 <- brick('LE70220492002106EDC00_stack.gtif')
le7
```

```
## class       : RasterBrick
## dimensions  : 250, 250, 62500, 8  (nrow, ncol, ncell, nlayers)
## resolution  : 30, 30  (x, y)
## extent      : 462405, 469905, 1734315, 1741815  (xmin, xmax, ymin, ymax)
## coord. ref. : +proj=utm +zone=15 +datum=WGS84 +units=m +no_defs
## data source : /home/ceholden/Documents/open-geo-tutorial/R/LE70220492002106EDC00_stack.gtif
## names       : band.1.reflectance, band.2.reflectance, band.3.reflectance, band.4.reflectance, band.5
## min values  :             -32768,             -32768,             -32768,             -32768,
## max values  :              32767,              32767,              32767,              32767,
```

Let's use R's vectorized summary statistics to take a look at why NDVI (sort of) works:

```
print("Red mean:")
```

```
## [1] "Red mean:"
```

```
mean(values(le7[[3]]))
```

```
## [1] 1460.093
```

```r
print("NIR mean:")
```

```
## [1] "NIR mean:"
```

```r
mean(values(le7[[4]]))
```

```
## [1] 2700.183
```

Vegetation is very dark in the visible (e.g., red) wavelengths due to photon absorption by pigments and chlorophyll and is reflects highly in the near-infrared due to reflection within cell structural layers. Generally, as the leaf area (i.e., LAI, a measure of how much leaf area (m2) you would intercept looking upward through the canopy over a given area on the ground (m2)) of a canopy increases, the red reflectances will decrease and the near-infrared reflectances will increase. Thus, the NDVI should also increase with increasing LAI. This relationship does not hold for very high levels of LAI because the reflection dynamics causing the visible absorption and near-infrared reflection saturate. Above certain levels of LAI, the LAI ~ NDVI relationship saturates and is not informative.

To calculate NDVI, we can do simple arithmetic:

```r
ndvi <- (le7$band.4.reflectance - le7$band.3.reflectance) / (le7$band.4.reflectance + le7$band.3.reflec
ndvi
```

```
## class       : RasterLayer
## dimensions  : 250, 250, 62500  (nrow, ncol, ncell)
## resolution  : 30, 30  (x, y)
## extent      : 462405, 469905, 1734315, 1741815  (xmin, xmax, ymin, ymax)
## coord. ref. : +proj=utm +zone=15 +datum=WGS84 +units=m +no_defs
## data source : in memory
## names       : layer
## values      : -0.5382397, 0.7676477  (min, max)
```

Remember from Chapter 1, however, that this is not necessarily a good approach for calculation because it does not take into consideration the amount of RAM required to perform the calculation. The `raster` package offers memory management automation through the `calc` and `overlay` functions:

```r
calc_ndvi <- function(x) {
    ndvi <- (x[[4]] - x[[3]]) / (x[[4]] + x[[3]])
    return(ndvi)
}
ndvi <- calc(le7, fun=calc_ndvi)
ndvi
```

```
## class       : RasterLayer
## dimensions  : 250, 250, 62500  (nrow, ncol, ncell)
## resolution  : 30, 30  (x, y)
## extent      : 462405, 469905, 1734315, 1741815  (xmin, xmax, ymin, ymax)
## coord. ref. : +proj=utm +zone=15 +datum=WGS84 +units=m +no_defs
## data source : in memory
## names       : layer
## values      : -0.5382397, 0.7676477  (min, max)
```

```
overlay_ndvi <- function(r, nir) {
    ndvi <- (nir - r) / (nir + r)
    return(ndvi)
}
ndvi <- overlay(le7[[3]], le7[[4]], fun=overlay_ndvi)
ndvi
```

```
## class       : RasterLayer
## dimensions  : 250, 250, 62500  (nrow, ncol, ncell)
## resolution  : 30, 30  (x, y)
## extent      : 462405, 469905, 1734315, 1741815  (xmin, xmax, ymin, ymax)
## coord. ref. : +proj=utm +zone=15 +datum=WGS84 +units=m +no_defs
## data source : in memory
## names       : layer
## values      : -0.5382397, 0.7676477  (min, max)
```

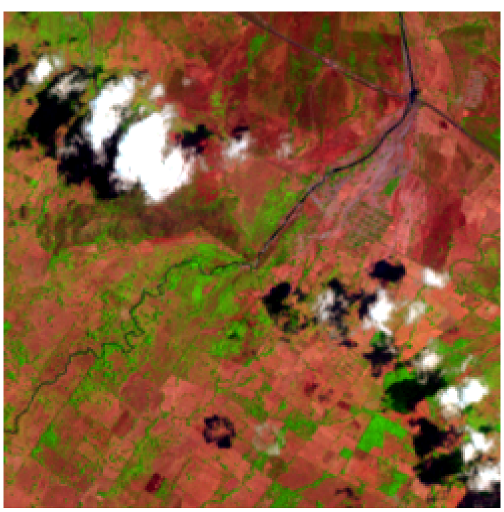
To see the value of this vegetation index, let's make a two panel plot:

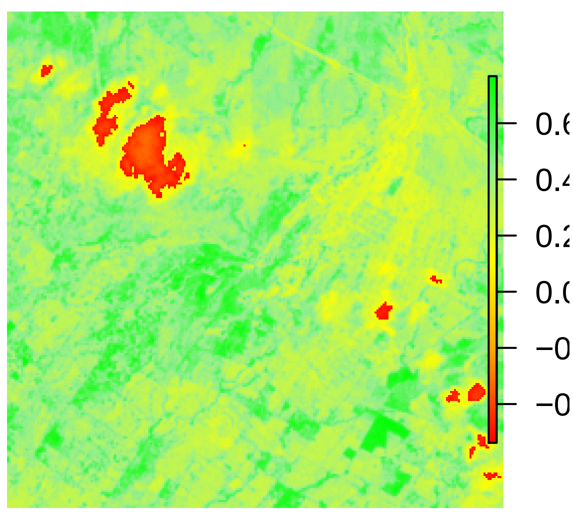
```
op <- par(mfrow = c(1, 2),
          oma = c(2, 2, 0, 0) + 0.1,
          mar = c(0, 0, 1, 1) + 0.1)

xlim <- c(xmin(ndvi), xmax(ndvi))
ylim <- c(ymin(ndvi), ymax(ndvi))

plot(xlim, ylim, type='n', axes=F, xlab='', ylab='', asp=1)
plotRGB(le7, r=5, g=4, b=3, stretch="lin", add=T)

plot(xlim, ylim, type='n', axes=F, xlab='', ylab='', asp=1)
plot(ndvi, col=colorRampPalette(c("red", "orange", "yellow", "light green", "green"))(255), 1, add=T)
```





```
dev.off()
```

```
## null device
##           1
```

I'm sure you can figure out the formatting of the axes and margins so that the plot looks more professional.

Other standard R plots, like `hist` can work on `Raster*` objects as well because the `raster` library has written specific versions for the `Raster*` classes:

```
par(mfrow=c(1, 1))
hist(ndvi, main='NDVI')
```