DATA VISUALIZATION

Created by: Megs Seeley

This tutorial requires familiarity with GEE.

Complete code can be found here: https://code.earthengine.google.com/ac46b4bf03604b39311f89ca8a337da5

**Section 0 – Import Data & Add NDVI Bands**

This lesson will explore the phenology of Northern Wisconsin using NDVI derived from Landsat 8 data. While NDVI is a relatively simple metric and is easy to calculate, we will see how different methods of manipulating our data can convey a large amount of information.

Google Earth Engine has documentation on creating charts from your remote sensing imagery, but choosing which charts and which data to use is the art of data visualization.

**Import Data and Draw ROI**

**var** l8 = ee.ImageCollection(‘LANDSAT/LC08/C01/T1\_RT’)

Next, draw a polygon in northern Wisconsin.

IMPORTANT NOTE: If your polygon is too large, you will overload the code editor and get an error.

**Filter Data**

**var** l8\_wisc = l8

 .filterBounds(geometry);

**Create NDVI Band**

**var** addNDVI = **function**(image){

 **var** ndvi = image.normalizedDifference(['B5', 'B4']).rename('ndvi');

 **return** image.addBands(ndvi);

};

l8\_wisc = l8\_wisc.map(addNDVI).select('ndvi');

Now that we have our data, we can get into the visualization!

**Section 1 – Time Series**

Creating charts in GEE is fairly straightforward. The built-in functions allow us to easily plot our data without extra processing.

We will first plot mean NDVI over time for our study region.

print(ui.Chart.image.series(l8\_wisc,

 geometry,

 ee.Reducer.mean(),

 200);

Now, we probably want to customize our chart. Since we are plotting NDVI, let’s turn our trendline green, provide some labels, and specify the dimensions.

var customizeChart = {

 colors:['green'],

 width:500,

 height:250,

 title: 'Northern Wisconsin NDVI',

};

We can add these customizations to our chart using ‘.setOptions’

print(ui.Chart.image.series(l8\_wisc,

 geometry,

 ee.Reducer.mean(),

 200)

 .setOptions(customizeChart);

While this chart is interesting, it is a bit difficult to compare the phenology from year to year. Let’s plot mean NDVI for each individual year.

print(ui.Chart.image.doySeriesByYear(l8\_wisc,

 ‘ndvi’,

 geometry,

 ee.Reducer.mean(),

 500);

**Section 2 – ANOMOLIES BAR CHARTS & DATA MANIPULATION**

As you can see, some charts require very little data preparation. Visualizing your data can be more difficult when it is not in the correct format. Let’s look at an example where we want to visualize the difference from the mean, or anomalies, in our NDVI data.

**Calculate mean NDVI for each month from 2014-2018**

**var** months = ee.List.sequence(1,12);

**var** years = ee.List.sequence(2014,2018);

**var** ndviReduced = months.map(function (m) {

 **return** years.map(function (y) {

 **var** l8\_mean = l8\_wisc.filter(ee.Filter.calendarRange(y, y, 'year'))

 .filter(ee.Filter.calendarRange(m, m, 'month'))

 .reduce(ee.Reducer.median())

 .reduceRegion({

 reducer: ee.Reducer.mean(),

 geometry: geometry,

 scale: 500,

 maxPixels: 1000000000

 });

 **return** l8\_mean.get('ndvi\_median');

 });

});

print(ndviReduced);

The above code creates two functions that iterates through the years and through the months and creates an average NDVI in our study area for each month. Look closely at the output. We get a list of lists. Each of the 12 lists are separated by month (e.g. January, February, ect.) and contain the mean NDVI for that month in each year.

**Calculate Monthly Anomalies**

The next challenge is to calculate an average across all years for each month and subtract all the means within that month.

**var** getAnom = **function**(collection, arr){

 **var** month = collection.get(arr);

 month = ee.List(month);

 **var** monthAvg = month.reduce(ee.Reducer.mean());

 **var** anom = month.map(function(a){

 **var** value = ee.Number(a);

 **return** value.subtract(monthAvg);

 });

 **return** anom;

};

Next, we will apply this function to get the anomalies for each month. We can do this by simply running the function 12 times.

**var** jan = getAnom(ndviReduced,0);

**var** feb = getAnom(ndviReduced,1);

**var** mar = getAnom(ndviReduced,2);

**var** apr = getAnom(ndviReduced,3);

**var** may = getAnom(ndviReduced,4);

**var** jun = getAnom(ndviReduced,5);

**var** jul = getAnom(ndviReduced,6);

**var** aug = getAnom(ndviReduced,7);

**var** sep = getAnom(ndviReduced,8);

**var** oct = getAnom(ndviReduced,9);

**var** nov = getAnom(ndviReduced,10);

**var** dec = getAnom(ndviReduced,11);

BONUS: Create a loop that iterates through a list of months and outputs the anomaly for each month.

Our data is *almost* in the correct format. In order to insert our data into a bar chart, we need to convert the lists into a data frame. We will create a loop that both creates the data frames then prints a bar chart with the data.

**var** monthName = ee.List(['January', 'February', 'March', 'April', 'May', 'June', 'July', 'August', 'September', 'October', 'November', 'December']);

**var** monthCall = ee.List([jan,feb,mar,apr,may,jun,jul,aug,sep,oct,nov,dec]);

**for** (**var** i = 0; i<12; i++){

 var nam = monthName.get(i)

 print(nam)

 // Chart customization

 **var** customizeBar = {

 colors:['green'],

 width:500,

 height:250,

 title: 'Northern Wisconsin NDVI Anomolies',

 vAxis:{minValue:-0.1, maxValue:0.1}

 };

 //Add data to a data table

 **var** monthCall1 = ee.List(monthCall.get(i));

 print(monthCall1)

 **var** datTable = {

 cols: [{id:'year', label: 'Year', type:'string'},

 {id: 'ndvi', lable:'Mean NDVI', type:'number'}],

 rows: [{c:[{v: "2014"}, {v: monthCall1.get(0)**.getInfo**()}]}, // Cast number out of ee object using getInfo

 {c:[{v:"2015"}, {v:monthCall1.get(1)**.getInfo**()}]},

 {c:[{v:"2016"}, {v:monthCall1.get(2)**.getInfo**()}]},

 {c:[{v:"2017"}, {v:monthCall1.get(3)**.getInfo**()}]},

 {c:[{v:"2018"}, {v:monthCall1.get(4)**.getInfo**()}]}]

 };

 // Print anomoly charts

 print(**ui.Chart**(datTable, 'ColumnChart', customizeBar));

}

As you explore other option for plotting your data in GEE, you can refer to the GEE chart documents: <https://developers.google.com/earth-engine/charts>

GEE creates charts using Google Charts. You can refer to the Google chart documents for information on customizing: <https://developers.google.com/chart/interactive/docs/customizing_charts>