Visualizing Fire Hot Spots & Creating Time Series Plots w/ Google Earth Engine

Google Earth Engine

Agenda

Link to GEE Fire Mapping Tutorial

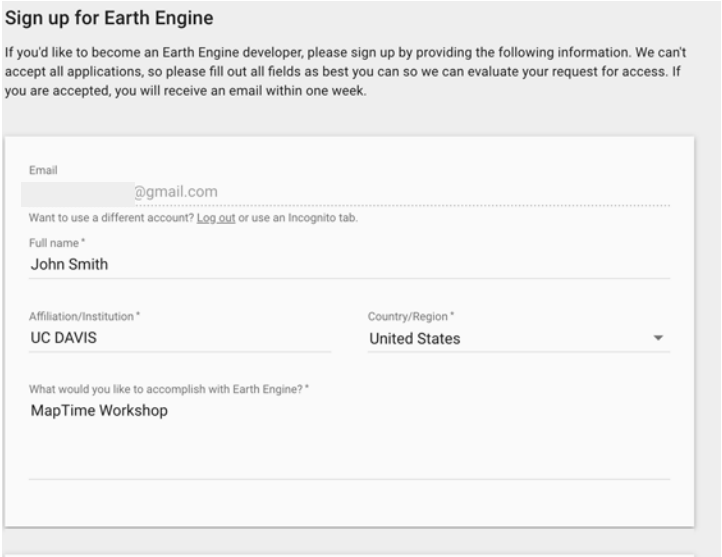
- General Overview of Google Earth Engine (GEE)
  - Accessing GEE
  - Code Editor
  - Available Datasets
- Accessing *Sierra Nevada Conservancy Region* Shapefile
  - Reading files via Code Editor
  - Fusion Table Set-Up
- Calling ImageCollection GEE Datasets
- Plotting Time Series (2012-2018)
  - MODIS Thermal Anomaly & Fire Radiative Power (FRP) Products
  - Palmer Drought Severity Index (PDSI), NDVI, NDWI
- Pre/Post Fire Animation w/ Landsat 8 & MODIS Imagery

Accessing GEE via Gmail account

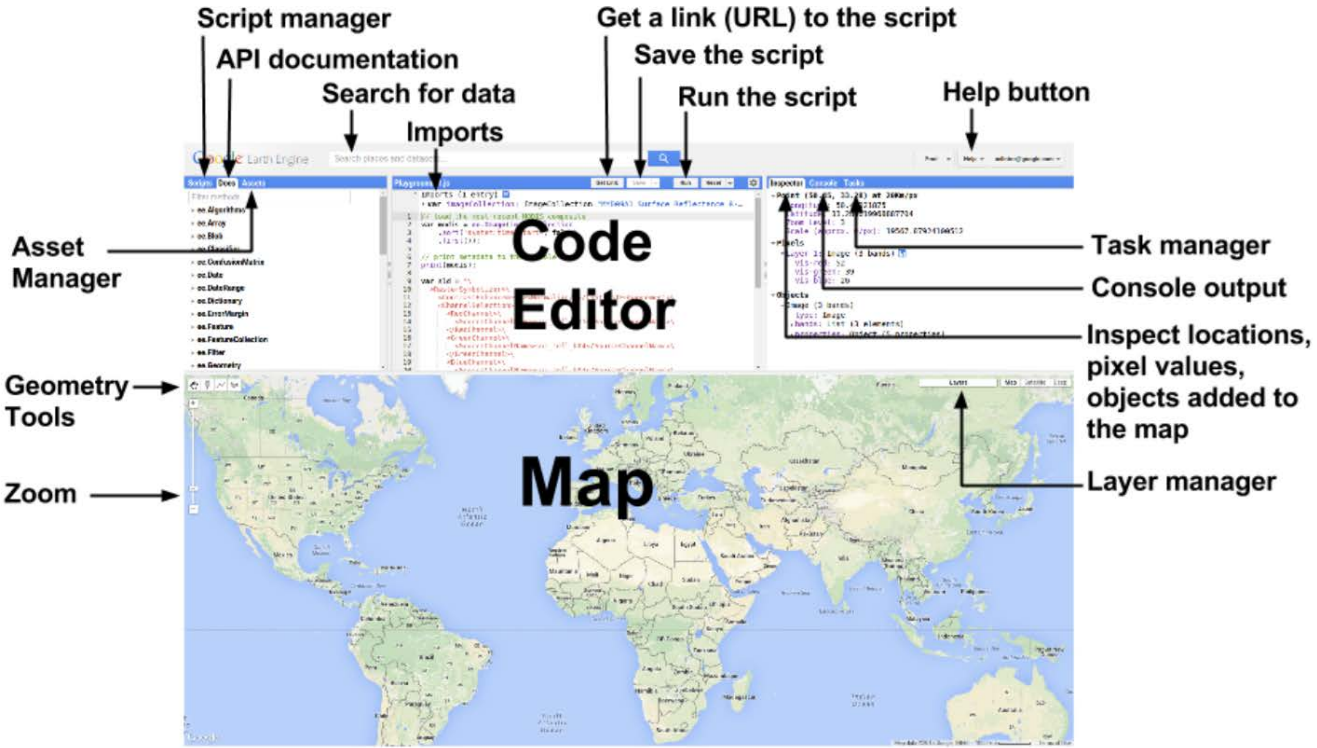
Google Earth Engine

Your account (**@gmail.com**) does not appear to be registered for Earth Engine access.
Please try one of the following:

- If you have not registered yet, you can do so here. Note that currently registration involves manual review which may take up to a few days to complete.
- If you have already registered, and believe this is in error, you can try one of the following to resolve the issue:
    1. Sign out and sign back in.
    2. Visit the Google Account Permissions page and revoke permissions for Earth Engine. Next time you attempt to access Earth Engine, you will be asked to re-authenticate, which will reset any pending issues with your credentials.
    3. If neither of these solutions work, drop us a line and we'll do our best to assist you.



General Overview: Code Editor



[Google Earth Engine Code Editor]

Available Datasets

MOD14A1.006: Terra Thermal Anomalies & Fire Daily Global 1km

Dataset Availability

2000-02-18T00:00:00 - Present

Dataset Provider

NASA LP DAAC at the USGS EROS Center

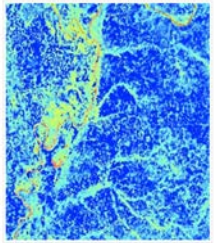
Earth Engine Snippet

```
ee.ImageCollection("MODIS/006/MOD14A1")
```

Tags

fire modis mod14a1 daily global terra

Landsat 8 Collection 1 Tier 1 8-Day NDWI Composite



Dataset Provider

Google

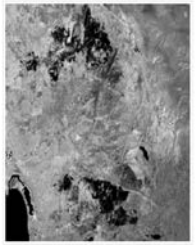
Earth Engine Snippet

```
ee.ImageCollection("LANDSAT/LC08/C01/T1_8DAY_NDWI")
```

Tags

ndwi

Landsat 8 Collection 1 Tier 1 8-Day NBRT Composite



Dataset Provider

Google

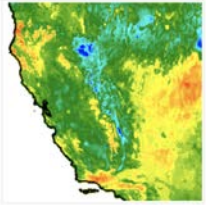
Earth Engine Snippet

```
ee.ImageCollection("LANDSAT/LC08/C01/T1_8DAY_NBRT")
```

Tags

nbrt

PDSI: University of Idaho Palmer Drought Severity Index



Dataset Availability

1979-03-01T00:00:00 - Present

Dataset Provider

University of Idaho

Earth Engine Snippet

```
ee.ImageCollection("IDAHO_EPSCOR/PDSI")
```

Tags

pdsi drought conus crop palmer geophysical climate idaho uofi

Landsat 8 Collection 1 Tier 1 8-Day NDVI Composite

Dataset Provider

Google

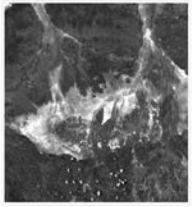
Earth Engine Snippet

```
ee.ImageCollection("LANDSAT/LC08/C01/T1_8DAY_NDVI")
```

Tags

ndvi

Landsat 8 Collection 1 Tier 1 8-Day BAI Composite



Dataset Provider

Google

Earth Engine Snippet

```
ee.ImageCollection("LANDSAT/LC08/C01/T1_8DAY_BAI")
```

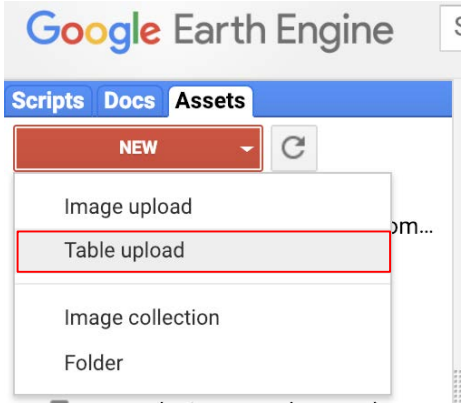
Tags

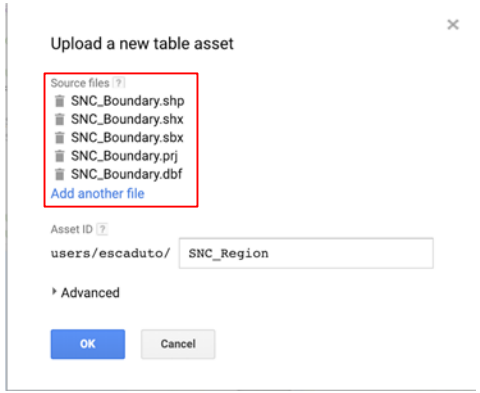
bai

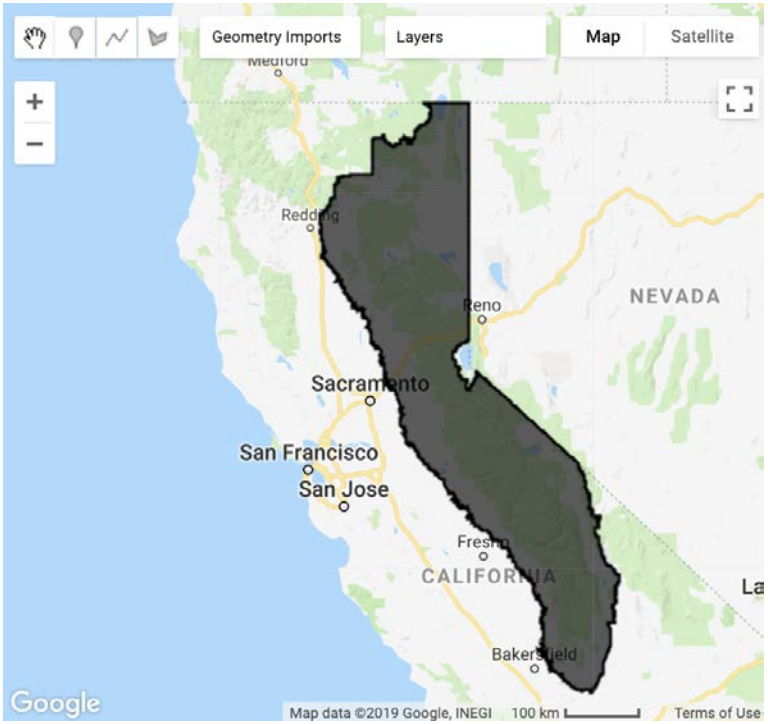
[Link to Available Datasets]

Download & Read Shapefile

- Access & download the Sierra Nevada Conservancy region **.shp** and **.KML**
- Navigate to the “Assets Manager”
- Click on “Table Upload”
- Add .shp, .shx, .sbx, .dbf, .prj files
- Specify Name and click “OK”
- Click on “Import into script” arrow

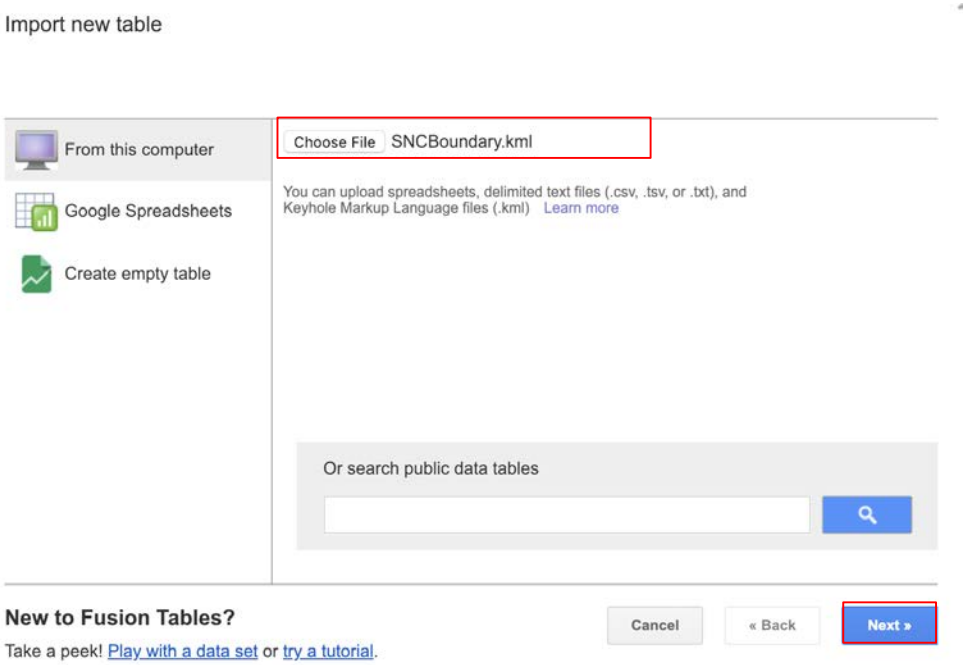






Create Fusion Table w/ KML

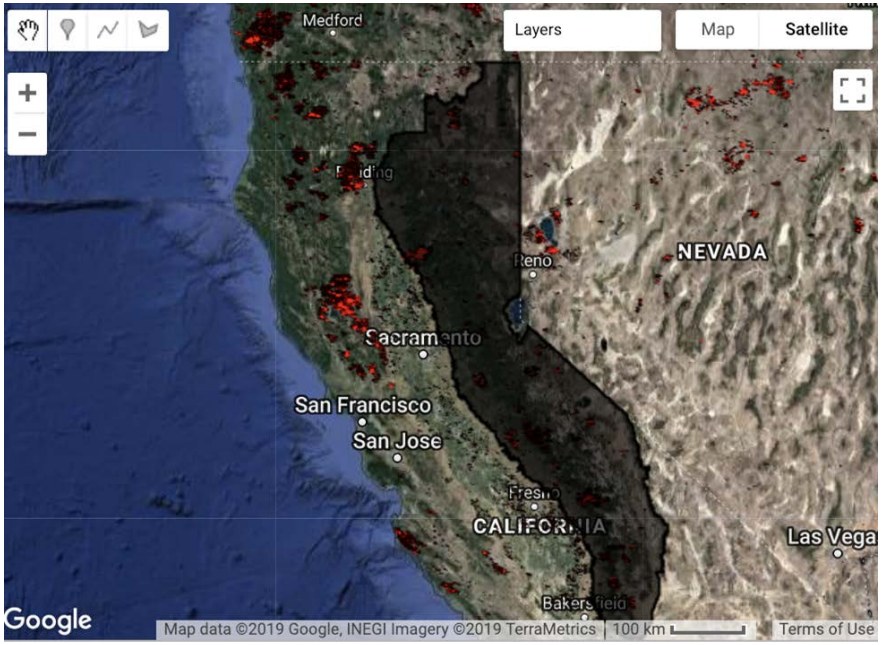
- Navigate to [Fusion Table Creator]
- Click “Next” if everything seems correct in table preview
- Fill in metadata fields if needed, and click “Finish”
- Under “File” menu, select “About this table”
- Copy ID field and paste it into Feature Collection constructor




```
// Load a Fusion Table from the ID using the FeatureCollection constructor.
var fc = ee.FeatureCollection('ft:11SfWB6oBS1iWGiQxEOqF_wUgBJL7Bux-pWU-mqd5');
```

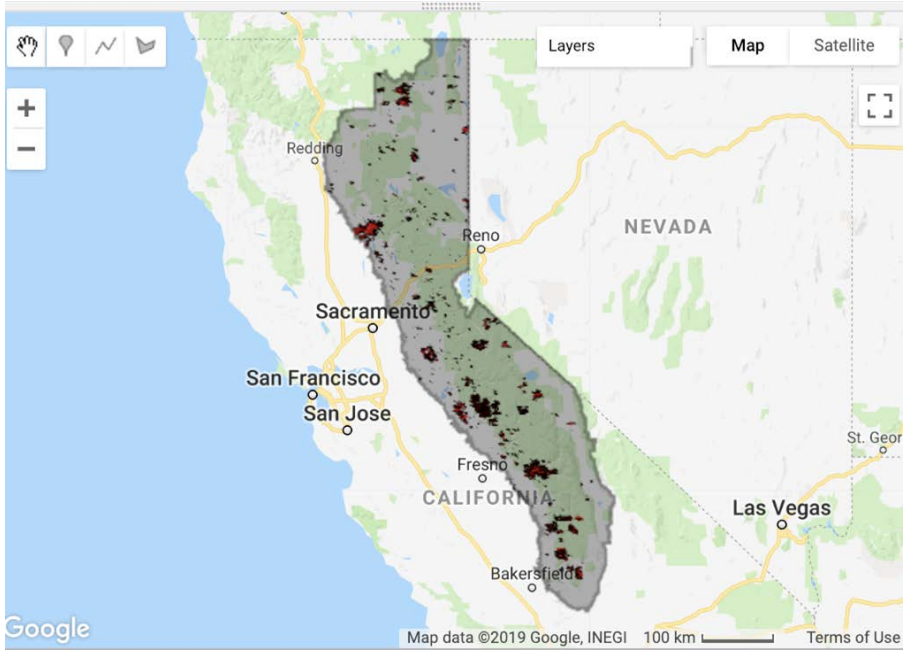
Active Fire Product: MOD14A1

```
Imports (1 entry)
  var table: Table users/escaduto/SNC_Boundary

// Call MODIS active fire product and filter by date
var MODIS_AFP = ee.ImageCollection('MODIS/006/MOD14A1')
                  .filterDate('2015-01-01', '2018-12-31');

// Set min and max FRP values to visualize
var fireMaskVis = {
  min: 0.0,
  max: 6000.0,
  bands: ['MaxFRP', 'FireMask'],
};

// Set Map Center
Map.setCenter(-121.2673,39.0308, 6);

// Add layers to map
Map.addLayer(MODIS_AFP, fireMaskVis, 'Fire Mask');
Map.addLayer(table, {}, 'SNC_Region');
```



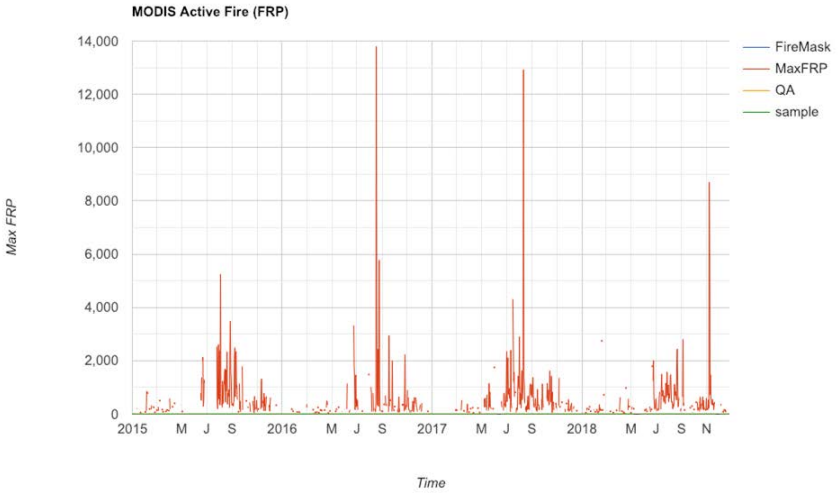
Extract ImageCollection w/ AOI

```
1
2  // Call MODIS active fire product and filter by date
3  var MODIS_AFP = ee.ImageCollection('MODIS/006/MOD14A1')
4                    .filterDate('2015-01-01', '2018-12-31');
5
6  // Set min and max FRP values to visualize
7  var fireMaskVis = {
8    min: 0.0,
9    max: 6000.0,
10   bands: ['MaxFRP', 'FireMask'],
11 };
12
13 // run function where it clips each date by Sierra polygon
14 var MOD_monte = MODIS_AFP.map(function(img) {return img.clip(table)})
15
16 // Set Map Center
17 Map.setCenter(-121.2673,39.0308, 6);
18
19 // Add layers to map
20 Map.addLayer(MOD_monte, fireMaskVis, 'Fire Mask');
21 Map.addLayer(table, {}, 'SNC_Region');
22
```



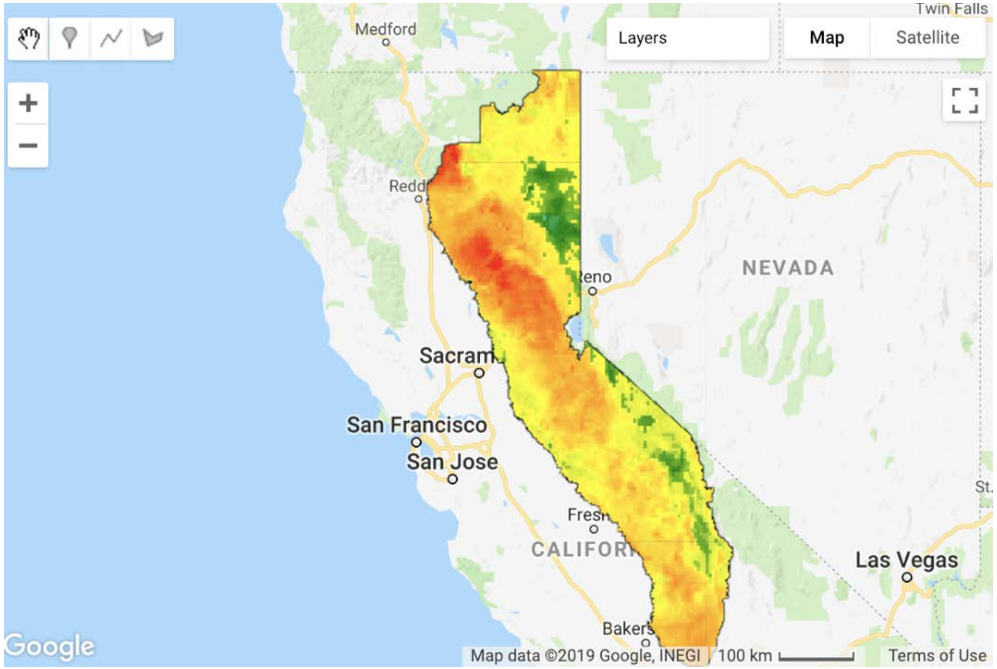
Create Time Series Plot - FRP[2015-2018]

```
//Part 1:
// Call MODIS active fire product and filter by date
var MODIS_AFP = ee.ImageCollection('MODIS/006/MOD14A1')
                  .filterDate('2015-01-01', '2018-12-31');

// Set min and max FRP values to visualize
var fireMaskVis = {
  min: 0.0,
  max: 6000.0,
  bands: ['MaxFRP', 'FireMask'],
};

// Run function where it clips each date by Sierra polygon
var MOD_monte = MODIS_AFP.map(function(img) {return img.clip(table)})

// Set Map Center
Map.setCenter(-121.2673,39.0308, 6);

// Add layers to map
Map.addLayer(MOD_monte, fireMaskVis, 'Fire Mask');
Map.addLayer(table, {}, 'SNC_Region');

// Set chart labels, type, etc.
var MODOptions = {
  title: 'MODIS Active Fire (FRP)',
  hAxis: {title: 'Time'},
  vAxis: {title: 'Max FRP'},
  lineWidth: 1
};

// Call timeseries plot function, add .setOptions()
var MODAFChart = ui.Chart.image.series(MODIS_AFP, table, ee.Reducer.mean(), 2000)
                  .setOptions(MODOptions);
print(MODAFChart)
```



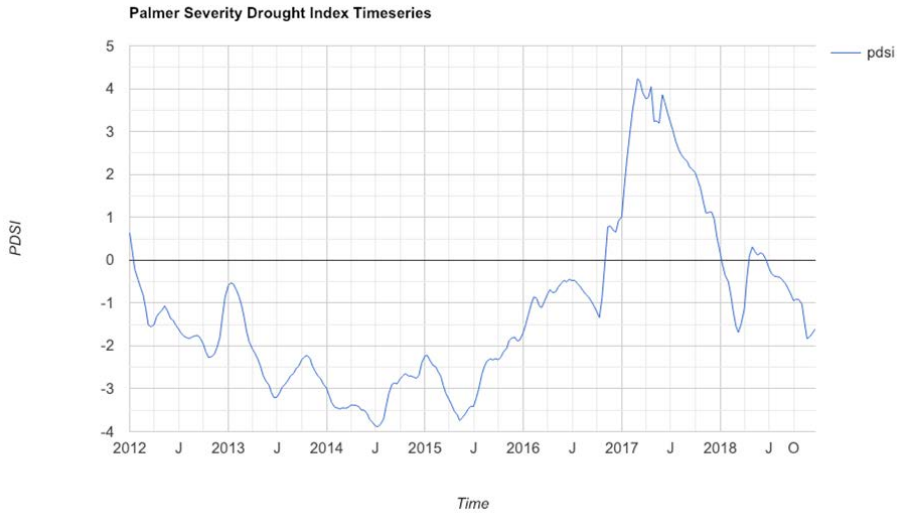
Palmer Drought Severity Index (PDSI)

```
// Part 2:
// Call Drought Indices, specity date
var RAW_PDSI = ee.ImageCollection('IDAHO_EPSCOR/PDSI')
                 .filter(ee.Filter.date('2012-01-01', '2018-12-31'))
                 .select('pdsi');

// Run function where PDSI layers are clipped by Sierra polygon
var PDSI = RAW_PDSI.map(function(img) {return img.clip(table)})

// Set range and specify color gradient
var pdsiVis = {
  min: -5.0,
  max: 10.0,
  palette: ['red', 'yellow', 'green', 'cyan', 'blue'],
};

Map.addLayer(PDSI, pdsiVis, 'PDSI');
```



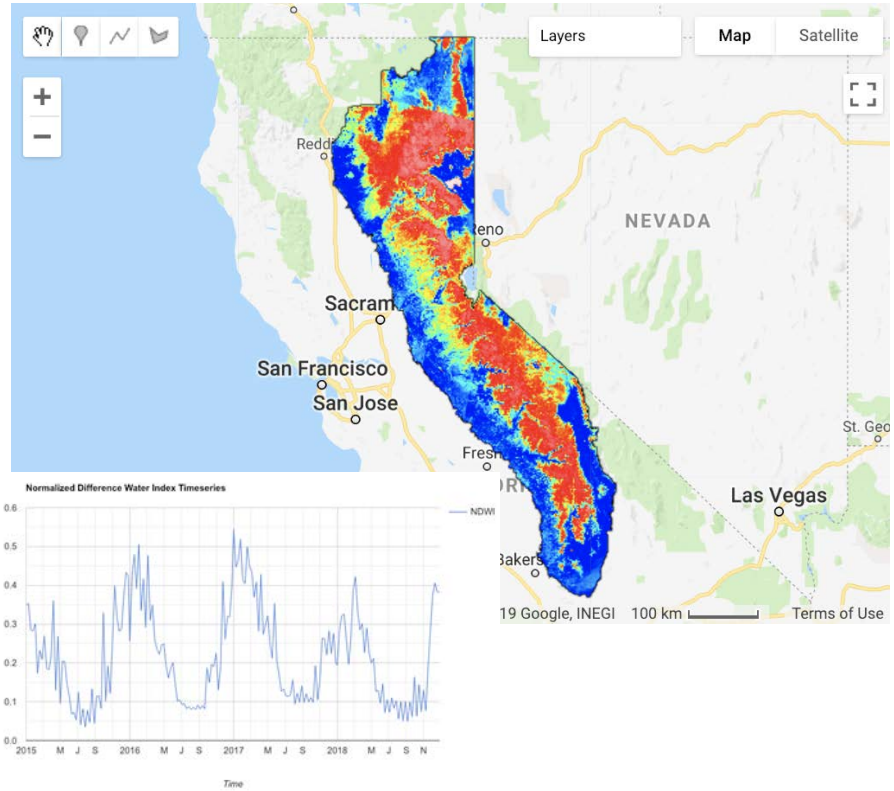
Create Time Series Plot - PDSI[2012-2018]

```
// Part 2:
// Call Drought Indices, specity date
var RAW_PDSI = ee.ImageCollection('IDAHO_EPSCOR/PDSI')
                 .filter(ee.Filter.date('2012-01-01', '2018-12-31'))
                 .select('pdsi');

// Run function where PDSI layers are clipped by Sierra polygon
var PDSI = RAW_PDSI.map(function(img) {return img.clip(table)})

// Set range and specify color gradient
var pdsiVis = {
  min: -5.0,
  max: 10.0,
  palette: ['red', 'yellow', 'green', 'cyan', 'blue'],
};

Map.addLayer(PDSI, pdsiVis, 'PDSI');

// Set chart labels, type, etc.
var PDSIoptions = {
  title: 'Palmer Severity Drought Index Timeseries',
  hAxis: {title: 'Time'},
  vAxis: {title: 'PDSI'},
  lineWidth: 1
};

var PDSIChart = ui.Chart.image.series(RAW_PDSI, table, ee.Reducer.mean(), 200)
                  .setOptions(PDSIoptions);
print(PDSIChart);
```



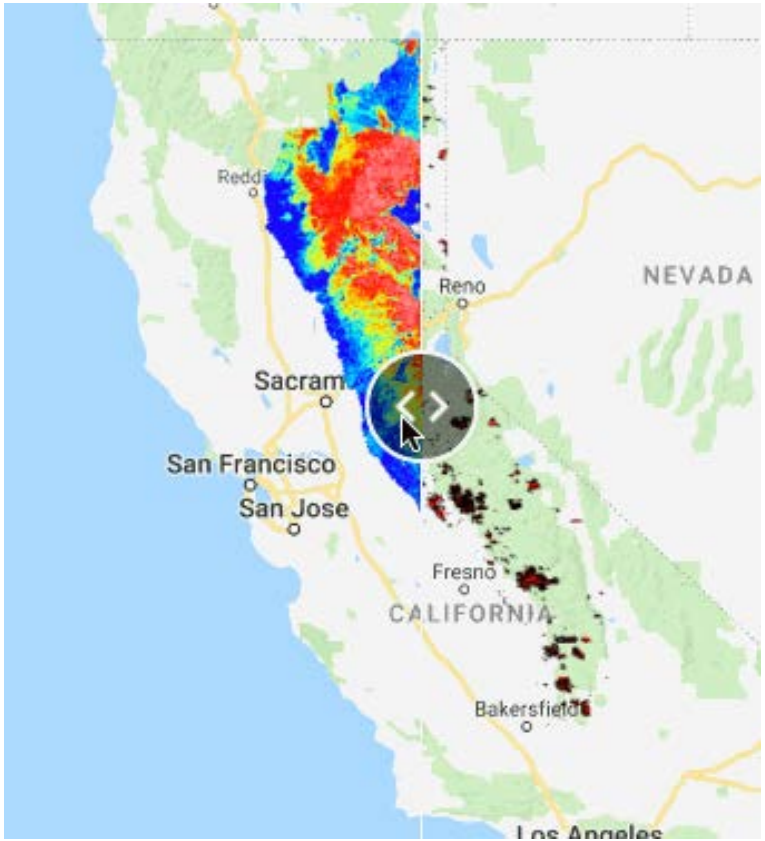
Normalized Difference Water Index (NDWI)

```
// Part 3:
// Normalized Difference Water Index
var RAW_NDWI = ee.ImageCollection('LANDSAT/LC08/C01/T1_8DAY_NDWI')
                 .filterDate('2015-01-01', '2018-12-31')
                 .select('NDWI');

// Run function where NDWI layers are clipped by Sierra polygon
var NDWI_bounded = RAW_NDWI.map(function(img) {return img.clip(table)})

// Set range and specify color gradient
var ndwiVis = {
    min: 0.0,
  max: 1.0,
  palette: ['0000ff', '00ffff', 'ffff00', 'ff0000', 'ffffff']
};

// Add NDWI to map layer
Map.addLayer(NDWI_bounded, ndwiVis, 'NDWI');

// Set chart labels, type, etc.
var NDWIOptions = {
  title: 'Normalized Difference Water Index Timeseries',
  hAxis: {title: 'Time'},
  vAxis: {title: 'NDWI'},
  lineWidth: 1
};

var NDWIChart = ui.Chart.image.series(RAW_NDWI, table, ee.Reducer.mean(),
                 .setOptions(NDWIOptions);

print(NDWIChart);
```



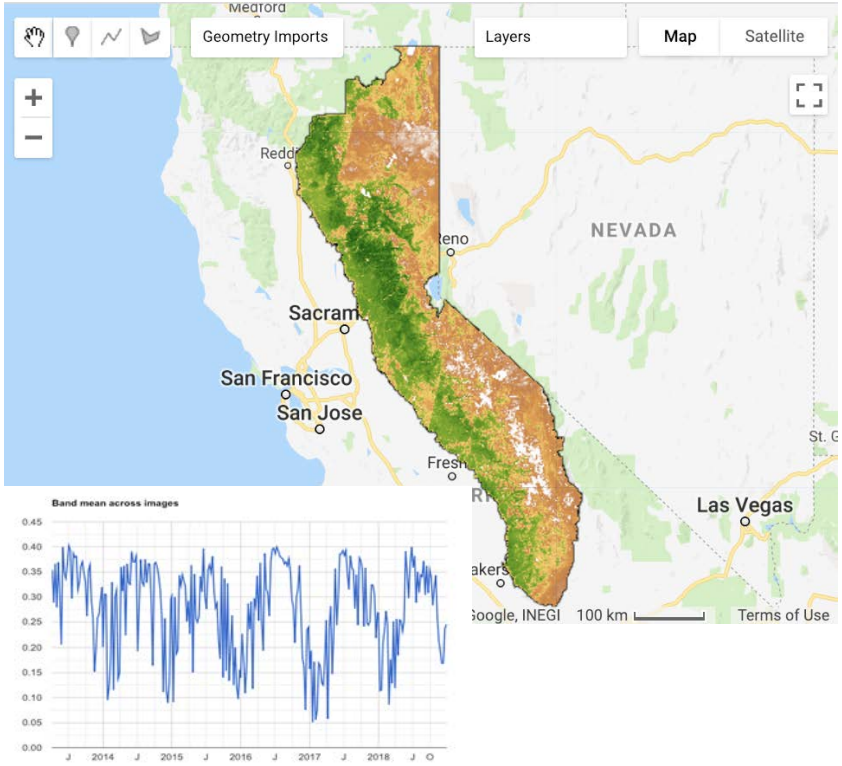
Split Screen Widget

```
// Create Split Map Slider
// Specify left and right maps, add appripriate layers
var leftMap = ui.Map();
var rightMap = ui.Map();
leftMap.addLayer(NDWI_bounded, ndwiVis, 'NDWI');
rightMap.addLayer(MOD_monte, fireMaskVis, 'MODIS AFP');

// Create a SplitPanel to hold the adjacent, linked maps.
var splitPanel = ui.SplitPanel({
  firstPanel: leftMap,
  secondPanel: rightMap,
  wipe: true
});
var linker = ui.Map.Linker([leftMap, rightMap]);

// Display only the split panel in ui root
//**(Uncomment 2 lines of Code Below)**
ui.root.widgets().reset([splitPanel]);
leftMap.centerObject(table);
```



Normalized Difference Vegetation Index (NDVI)

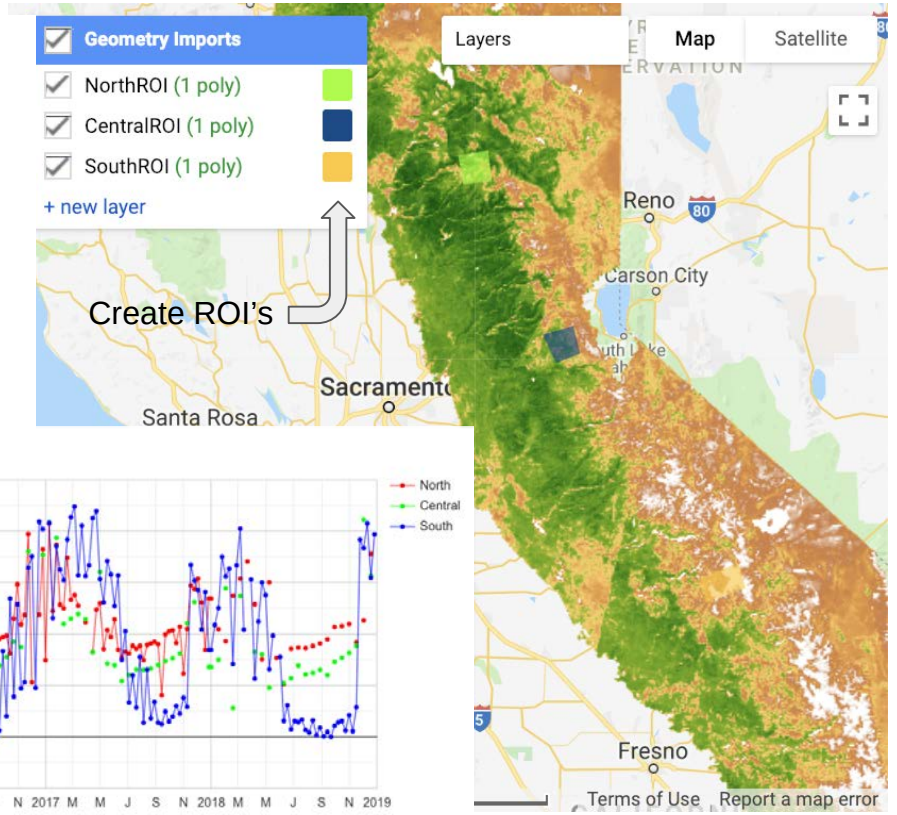
```
// Part 4:
// Normalized Difference Vegetation Index
var RAW_NDVI = ee.ImageCollection('LANDSAT/LC08/C01/T1_8DAY_NDVI')
                 .filterDate('2015-01-01', '2018-12-31')
                 .select('NDVI');

// Run function where NDVI layers are clipped by Sierra polygon
var NDVI_bounded = RAW_NDVI.map(function(img) {return img.clip(table)})

// Set range and specify color gradient
var ndviVis = {
  min: 0.0,
  max: 1.0,
  palette: [
    'FFFFFF', 'CE7E45', 'DF923D', 'F1B555', 'FCD163', '99B718', '74A901',
    '66A000', '529400', '3E8601', '207401', '056201', '004C00', '023B01',
    '012E01', '011D01', '011301'
  ],
};

// Add NDWI to map layer
Map.addLayer(NDVI_bounded, ndviVis, 'NDVI');

// Set chart labels, type, etc.
var NDVIOptions = {
  title: 'Normalized Difference Vegetation Index Timeseries',
  hAxis: {title: 'Time'},
  vAxis: {title: 'NDVI'},
  lineWidth: 1
};

// Call timeseries plot function, add .setOptions()
var NDVIChart = ui.Chart.image.series(RAW_NDVI, table, ee.Reducer.mean(), 2000)
                 .setOptions(NDVIOptions);

print(NDVIChart);
```



Temporal Comparison by Region

```
// Analysis by Region
var regions = ee.FeatureCollection([
  ee.Feature(    // North.
    // North.
NorthROI, {label: 'North'}),
  ee.Feature(  // Central.
    // Central.
CentralROI, {label: 'Central'}),
  ee.Feature(  // South.
    // South.
SouthROI, {label: 'South'})
]);
```

```
// Call seriesByRegion plot function, set chart type, and add .setOptions()
var NDWITimeSeries = ui.Chart.image.seriesByRegion(
    RAW_NDWI, regions, ee.Reducer.mean(), 'NDWI', 2000,'system:time_start', 'label')
        .setChartType('ScatterChart')
        .setOptions({
          title: 'NDWI Regional Timeseries',
          vAxis: {title: 'NDWI'},
          lineWidth: 1,
          pointSize: 4,
          series: {
            0: {color: 'FF0000'}, // north
            1: {color: '00FF00'}, // central
            2: {color: '0000FF'}  // south
}});

// Display NDWI Time Series
print(NDWITimeSeries);
```



Animation1: Landsat 8 30-m TOA

```
// Load a Landsat 5 image collection.
var collection = ee.ImageCollection('LANDSAT/LC08/C01/T1_TOA')
  .filter(ee.Filter.eq('WRS_PATH', 44))
  .filter(ee.Filter.eq('WRS_ROW', 32))
  // Filter cloudy scenes.
  .filter(ee.Filter.lt('CLOUD_COVER', 30))
  .filterDate('2018-08-01','2018-12-30')
  // Need to have 3-band imagery for the video.
  .select(['B5', 'B4', 'B3'])
  // Need to make the data 8-bit.
  .map(function(image) {
    return image.multiply(512).uint8();
  });

// Define an area to export.
var polygon = ee.Geometry.Rectangle([-121.8248,39.8514, -121.3524,39.6508]);

// Export (change dimensions or scale for higher quality).
Export.video.toDrive({
  collection: collection,
  description: 'Landsat_Camp',
  dimensions: 720,
  framesPerSecond: 1,
  region: polygon
});
```

Animation 2: MODIS 500-m Surface Reflectance

```
//Part 5B
// Load a MODIS image collection.
var collection = ee.ImageCollection('MODIS/006/MOD09GA')
  .filterBounds(table)
  .filterDate('2018-11-01','2018-12-30')
  // Need to have 3-band imagery for the video.
  .select(['sur_refl_b01', 'sur_refl_b04', 'sur_refl_b03'])
  // Need to make the data 8-bit.
  .map(function(image) {
    return image.multiply(0.0256).uint8();
  });
print(collection)
// Define an area to export.
var polygon = ee.Geometry.Rectangle([-121.8248,39.8514, -121.3524,39.6508]);

// Export (change dimensions or scale for higher quality).
Export.video.toDrive({
  collection: collection,
  description: 'MODIS_Camp',
  dimensions: 720,
  framesPerSecond: 1,
  region: box
});
```

Additional Resources

- Google Earth Engine Developer's Guide
- Using GEE API w/ Python (Jupyter Notebook)
- GEE Developer Forum
- GeoHackWeek Tutorials