Introduction to remote sensing

Karina Nielsen (karni@space.dtu.dk) and Allan Aasbjerg Nielsen

2023-08-29

Overview

- A short introduction to remote sensing
- Introduction to Google Earth Engine
- Exercises

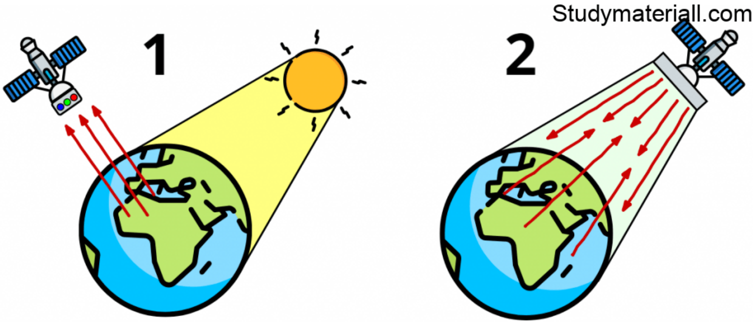
Course overview, optical part

- Module 1 Introduction, surface reflection
- Module 2 Principal component analysis (Report)
- Module 3 Supervised classification (Report)
- Module 4 Unsupervised classification (Report)
- Module 5 Change detection (Report)
- Module 6 Applications

Objectives for today

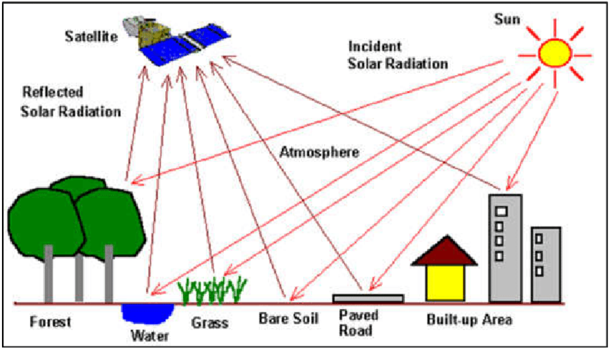
- A brief introduction to optical remote sensing
  - see/investigate how different surface types reflect at the individual bands
- Google Earth Engine:
  - be able to search for Sentinel-2/Landsat images
  - be able to visualize images with different color composites
  - be able to export an image for further analysis in R/Matlab/Python/...
  - be able to do a simple classification via normalised difference indices: NDVI, NDWI, ...
  - be able to make spectral response curves for different surface types
- Reading material regarding remote sensing https://crisp.nus.edu.sg/~research/tutorial/rsmain.htm

Principle of Remote Sensing



Optical Remote Sensing

- Capture the reflected sun light
- The various materials will reflect the sun light differently



Reflectance and wavelength

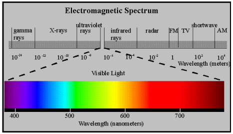


Figure 1: Source: Paul R. Baumann

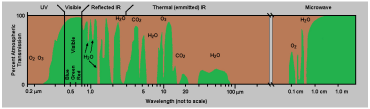


Figure 2: Source: Paul R. Baumann

- Some wavelength are absorbed in the atmosphere
- The reflectance of a material change as a function of wavelength
- The difference in reflectance of materials as a function of wavelength makes it possible to distinguish between surface materials

Examples

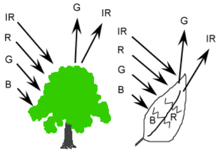


Figure 3: Health vegetation

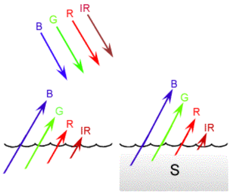


Figure 4: Water

Examples

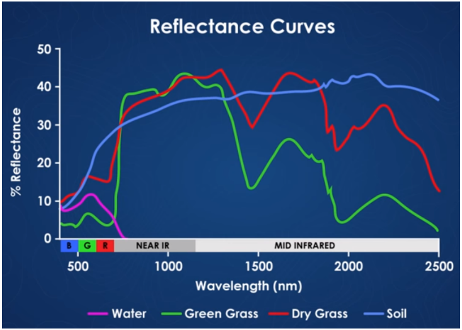
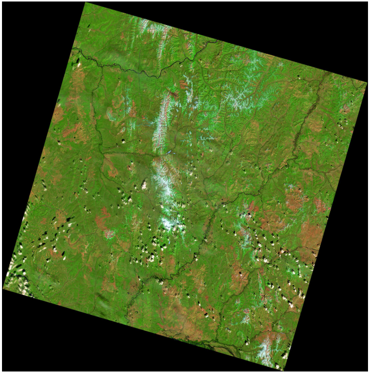


Figure 5: Source: National Ecological Observatory Network (NEON)

Properties of a digital image in remote sensing

- Spatial resolution
- Spectral resolution
- Radiometric resolution
- Temporal resolution



Spatial resolution

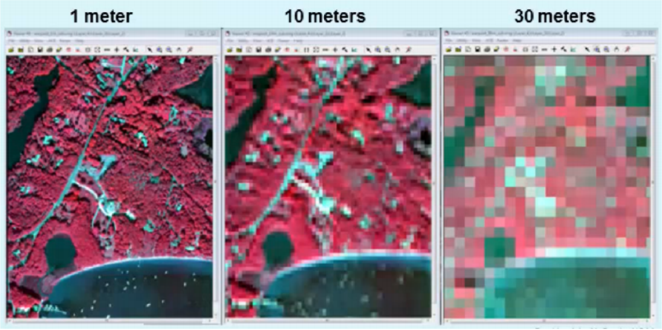
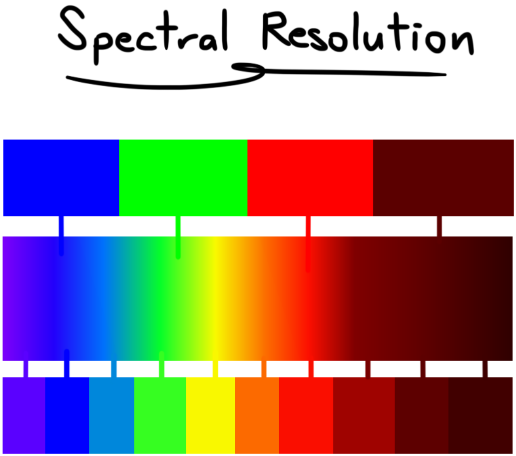


Figure 6: source: DOI: 10.13140/RG.2.2.34222.13126

Spectral resolution



Multispectral vs. Hyperspectral

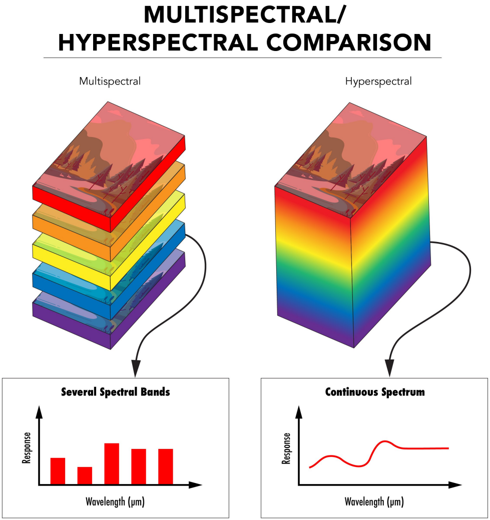


Figure 8: source: Edmund optics

Radiometric resolution

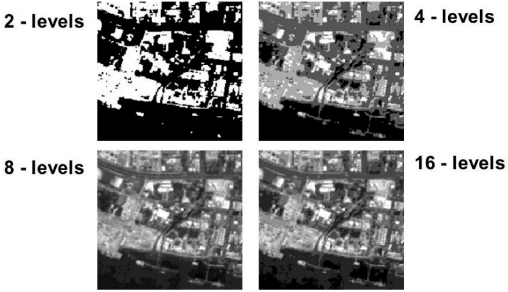


Figure 9: source: DOI: 10.13140/RG.2.2.34222.13126

- Sentinel-2 and Landsat 8: 12 bits - 0 to 4095 potential light intensity value
- Landsat 1-7: 8 bits - 0 to 255 potential light intensity values

Temporal resolution

- How frequent a region is mapped
- Important depending on what we want to map
  - Weather a high temporal resolution is needed
  - long term trends/Climate a lower temporal resolution is needed
- Landsat 8: 16 days
- Sentinel-2: 5 days (using both S2A and S2B)
- modis: 1-2 days

Sentinel-2 band information

Sentinel-2 MSI: MultiSpectral Instrument, Level-1C

Dataset Availability

2015-06-23T00:00:00 - 2021-08-21T00:00:00

Dataset Provider

European Union/ESA/Copernicus

Collection Snippet

```
ee.ImageCollection("COPERNICUS/S2")
```

See example

Tags

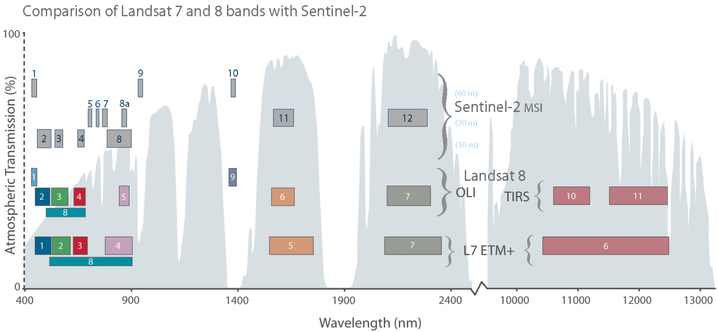
msi radiance eu esa copernicus sentinel

DESCRIPTION BANDS IMAGE PROPERTIES TERMS OF USE

Name	Description	Resolution	Wavelength	Scale
B1	Aerosols	60 meters	443.9nm (S2A) / 442.3nm (S2B)	0.0001
B2	Blue	10 meters	496.6nm (S2A) / 492.1nm (S2B)	0.0001
B3	Green	10 meters	560nm (S2A) / 559nm (S2B)	0.0001
B4	Red	10 meters	664.5nm (S2A) / 665nm (S2B)	0.0001
B5	Red Edge 1	20 meters	703.9nm (S2A) / 703.8nm (S2B)	0.0001
B6	Red Edge 2	20 meters	740.2nm (S2A) / 739.1nm (S2B)	0.0001
B7	Red Edge 3	20 meters	782.5nm (S2A) / 779.7nm (S2B)	0.0001
B8	NIR	10 meters	835.1nm (S2A) / 833nm (S2B)	0.0001

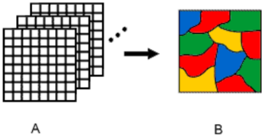
CLOSE IMPORT

Sentinel-2

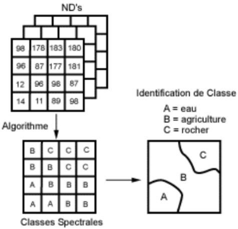


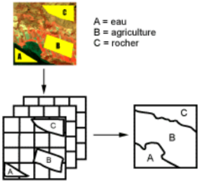
Comparison of Landsat 7 and 8 bands with Sentinel-2
Atmospheric Transmission (%)
100
0
1
9
10
5 6 7
8a
2 3 4 8
11
12
(60 m)
Sentinel-2 MSI
(20 m)
(10 m)
1
9
2 3 4
5
6
7
8
Landsat 8
OLI
TIRS
10
11
1 2 3
4
5
7
6
8
L7 ETM+
400
900
1400
1900
2400
10000
11000
12000
13000
Wavelength (nm)

Classification



Unsupervised (clustering) | Supervised





Remote Sensing Applications

- Agriculture
  - Crop Type Mapping
  - Crop Monitoring and Damage Assessment
- Forestry
  - Clear Cut Mapping and Deforestation
  - Species Identification and Typing
  - Burn Mapping
- Geology
  - Structural Mapping and Terrain Analysis
  - Geologic Unit Mapping
- Hydrology
  - Surface water extent mapping
  - Flood Delineation and Mapping
  - Soil Moisture

Remote Sensing Applications continued

- Sea Ice
  - Ice Type and Concentration
  - Ice Motion
- Land Cover and Land Use
  - Land Use Change (Rural / Urban)
  - Land Cover / Biomass Mapping
- Mapping
  - Planimetry
  - Digital Elevation Models
  - Topographic and Baseline Thematic Mapping
- Oceans and Coastal Monitoring
  - Ocean Features
  - Ocean Colour and Phytoplankton Concentration
  - Oil Spill Detection

Global surface water

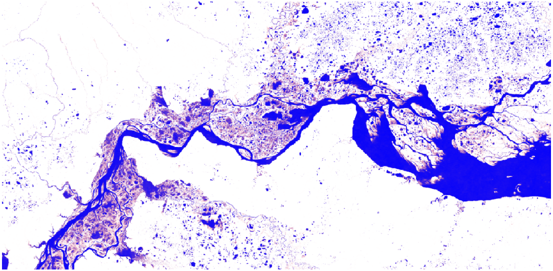


Figure 10: source: https://global-surface-water.appspot.com/

- Water occurrence based on landsat images
- Pixel value indicates the likelihood in percentage of water over the Landsat period

Data sources

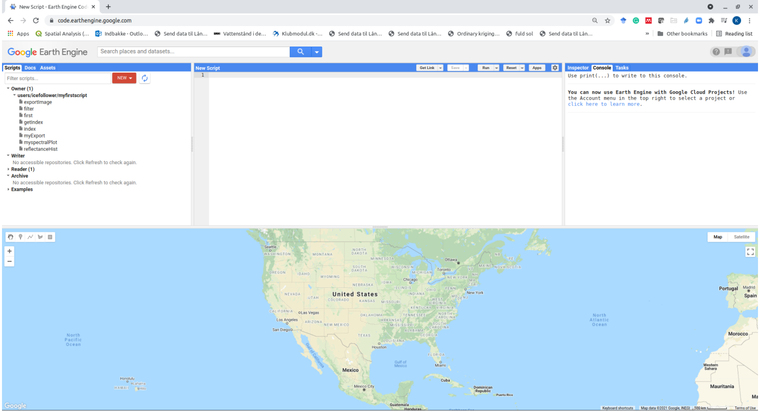
- http://earthexplorer.usgs.gov/ (Many different data sets)
- https://scihub.copernicus.eu/ (Sentinel data sets)
- https://developers.google.com/earth-engine/datasets/

Exercise Google Earth Engine

- Importing and visualizing an image
- Searching for an image by various filters
- Exporting a single image
- Simple classification via Normalised Indices
- Creating charts in GEE: Spectral profiles for different surface types
- Application: Detecting wild fires in the Western US

What is Google Earth engine

- Cloud platform that contains a large collection of satellite imagery and other derived products
- You can write and run your own code (Java script)
- You do not need to download the data
- You get access to large amounts of computer power



Let us have a look at GEE

- Now go to your GEE account https://code.earthengine.google.com/
- To get help and tons of examples consult https://developers.google.com/earth-engine/guides

Example 1: Follow along, Loading a Sentinel-2 image

- For simplicity we will start by importing a single Sentinel-2 image
- The image that we will look at is around Genoa, Italy (write this in the search bar to center the map)
- With the function `ee.Image` we can load an image
- To run the code click on the 'Run' bottom
- To see that something happened we can print out the name of the file, which will appear in the console tab

```
// to import an image
var sent2 = ee.Image('COPERNICUS/S2/20160718T101032_20160718T101028_T32TNQ');
print(sent2);
```

- To see the image we will use the function `Map.addLayer(eeObject, visParams, name, shown, opacity)`
- Investigate `visParams`

```
Map.addLayer(sent2,{bands:['B4','B3','B2']}); // RGB color composite
```

- Now you should see an image that appears black

Example 1: Follow along, true color composite

- Try and use the inspector function to get a sense of the range of the reflectence values
- Between 0-3000 seems like a first good choice
- To get a better representation let us insert some min and max values

```
Map.addLayer(sent2,{bands:['B4','B3','B2'],min:0,max:3000},'True color composite'); //RGB color composite
```

- The last argument is the name of the layer

Example 1: Follow along, False color composite

- Try to make a false color composite: NIR goes in the red channel, red goes in the green channel and green goes in the blue channel

```
Map.addLayer(sent2,{bands:['B8','B4','B3'], min:0, max:3000}, "False-colour"); //NIR,R,G
```

- Exercise
  - What do you see compared to the true color composite?
  - Try and make other false color composites (not all combinations may apply for this particular image)
    - See for example https://www.l3harrisgeospatial.com/Support/Self-Help-Tools/Help-Articles/Help-Articles-Detail/ArtMID/10220/ArticleID/15691/The-Many-Band-Combinations-of-Landsat-8 (remember to translate into the right Sentinel-2 bands)
    - or examples using Sentinel-2 https://www.gears-lab.com/intro_rs_lab2/#practical-exercise
  - Try also to load the individual bands and investigate how different surface types reflects. Use the inspector tab to get reflectence values of individual pixels

Example 1 Summary of code

```
//Example 1
//load image
var sent2 = ee.Image('COPERNICUS/S2/20160718T101032_20160718T101028_T32TNQ');

//print name of image
print(sent2);

//add image to map
Map.addLayer(sent2,{bands:['B4','B3','B2'],min:0,max:3000},'True color composite',false); //RGB color composite
Map.addLayer(sent2,{bands:['B8','B4','B3'], min:0, max:3000}, "False-colour",false); //NIR,B,G
//map layers of individual bands
Map.addLayer(sent2,{bands:['B1'], min:0, max:3000}, "ultra blue");
Map.addLayer(sent2,{bands:['B2'], min:0, max:3000}, "blue");
Map.addLayer(sent2,{bands:['B3'], min:0, max:3000}, "green");
Map.addLayer(sent2,{bands:['B4'], min:0, max:3000}, "red");
Map.addLayer(sent2,{bands:['B8'], min:0, max:3000}, "NIR");
```

Follow along import an image collection

- Typically we want to find an image for a specific region and time period
- Below we have added the entire collection of Sentinel-2 images
- When displaying an image collection it can be useful to do some filtering otherwise we will load all images in the map window, which may take some time.

```
var S2 = ee.ImageCollection("COPERNICUS/S2");

var S2_filter = S2.filterDate('2016-01-01', '2016-1-05');
var visParams = {bands: ['B4', 'B3', 'B2'], max: 3000};
Map.addLayer(S2_filter, visParams, 'S2 collection');
```

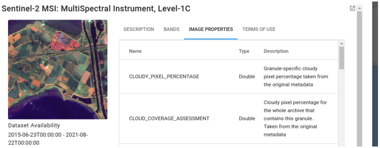
- Here we have filtered the collection by start and end date
- Exercise
    - Try and change the date interval and zoom in and out
    - Try and make a false color composite of the image collection
    - You can also try to add another image collection e.g. Landsat 8

Further filtering

- Let us apply a spatial filter. Click on the 'tear drop' symbol and the click on the map to add the symbol 
- You will now see that a 'geometry' have been added to the code, rename this to 'roi'.
- We can now use `filterBounds` to find all images that intersects with the geometry 'roi'
- The newest is shown on the top. Use the `print()` command to get a list of the images

```
var S2_filter = S2.filterDate('2016-01-01', '2016-05-01').filterBounds(roi);
```

- To get the least cloudy image, we can sort the selected images based on cloud cover



Continued ..

```
var S2_filter = S2.filterDate('2016-01-01', '2016-05-01').filterBounds(roi).sort('CLOUD_COVERAGE_ASSESSMENT').first();
```

- This will give us the least cloudy image in the given region and time span
- Exercises: From the Sentinel-2 image collection, find the least cloudy image in an area of your interest
  - Make different color composites
  - Find an image from another season e.g. wet/dry or summer/Winter
  - Investigate changes in the vegetation and other surfaces
- Exercise Landsat thermal band: Use the Landsat8/Landsat9 image collection
  - Can you identify the current volcanic eruption in the Kilauea crater, Hawaii or the August 2022 volcanic eruption on Island (Hint use Landsat9).
  - Have a look over a city using the thermal band

Normalised difference indices, simple classifiers

- The vegetation index *NDVI* is given by

$$NDVI = \frac{NIR - RED}{NIR + RED}$$

- The NDVI will be a number between -1 and 1, where a high number will correspond healthy green vegetation

```
//Define variable NDVI from equation
    var NDVI = S2_filter.expression(
        "(NIR - RED) / (NIR + RED)",
        {
          RED: S2_filter.select("B4"),    //  RED
          NIR: S2_filter.select("B8"),    // NIR
        });
// add the NDVI to the map
Map.addLayer(NDVI, {min: 0, max: 1}, "NDVI");
```

- Or simply use the build in function

```
var NDVI2 = S2_filter.normalizedDifference(['B8', 'B4']).rename(['ndvi']); //build in function

//display NDVI with a color palette
Map.addLayer(NDVI2, {min: 0, max: 1,palette:['blue','white','green']}, "NDVI2");
```

Exercise: NDVI and other indices

- Try and calculate the NDVI for another season e.g. wet/dry spring-summer/fall
- Try and calculate other normalised difference indices
  - Water index, gives an indication of soil moister

$$NDWI = \frac{Green - NIR}{Green + NIR}$$

  - Build up index

$$NDBI = \frac{SWIR - NIR}{SWIR + NIR}$$

- What do see?

Exporting an image

- The Sentinel-2 and Landsat scenes are large in terms of storage. It can be useful to
  - only save a selected number of bands
  - only save a part of an image
- To select bands

```
var imageExport = S2_filter.select(['B2','B3','B4','B8']);
```

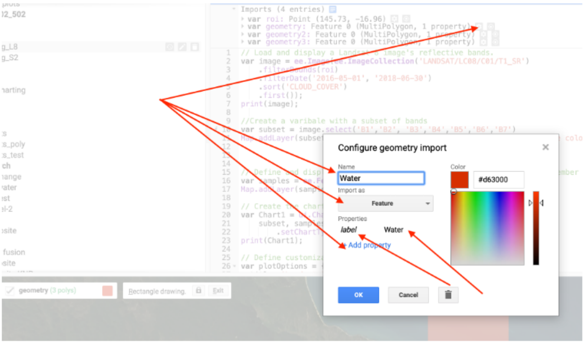
- To select a region
  - click on the symbol to draw a rectangle
  - Create a new geometry layer under the bar “Geometry” and rename to ‘poly’
  - Draw a rectangle that defines the area to be exported
- Let us export the image to out google drive

```
Export.image.toDrive({
  image: imageExport,
  description: 'imageToDrive',
  region: poly,
  scale: 10
});
```

- The Tasks tab becomes active, now press run
- When the task is done, go to your google drive where you can download the image

Plotting spectral response curves

- With the inspector tab we can get the spectral response for one pixel
    - To get a better representation we can calculate the mean of pixels in a box
- Find a cloud free image over an interesting area
- Create e.g. 3 or more rectangles depending on the number of surface types
- The geometry must be changed to a feature see image below



Plotting spectral response curves continued

- Here a Sentinel-2 image is selected

```
var S2 = ee.ImageCollection("COPERNICUS/S2");
var image = S2.filterDate('2020-05-01', '2020-08-01').filterBounds(roi).sort('CLOUD_COVERAGE_ASSESSMENT').first();
```

- Then we need to select the bands to be used and create a feature collection. Here the features ‘Water’, ‘fields’ and ‘city’ were used

```
var subset = image.select('B1','B2','B3','B4','B8');
var samples = ee.FeatureCollection([water,fields,city]);
```

- To create the scatter chart use

```
var Chart1 = ui.Chart.image.regions(
    subset, samples, ee.Reducer.mean(), 10, 'label')
        .setChartType('ScatterChart');
print(Chart1);
```

Plotting spectral response curves continued

- We can improve the chart by specifying plotting options

```
// wavelength corresponding to the selected bands
var wavelengths = [442, 492, 559,665 , 833];

// Define customization options.
var plotOptions = {
  title: 'S2 TOA reflectance spectra',
  hAxis: {title: 'Wavelength (nanometers)'},
  vAxis: {title: 'Reflectance'},
  lineWidth: 1,
  pointSize: 4,
  series: {
    0: {color: 'blue'}, // Water
    1: {color: 'green'}, // Fields
    2: {color: 'red'}, // City
}};

// Create the scatter chart
var Chart2 = ui.Chart.image.regions(
    subset, samples, ee.Reducer.mean(), 10, 'label',wavelengths)
        .setChartType('ScatterChart')
        .setOptions(plotOptions);
print(Chart2);
```

Plotting spectral response curves continued

- The rectangles/polygons can subsequently be resized of moved to another location
- Simply run the script again to get the response curve for the new regions

Exercise:

- Add more bands from the Sentinel-2 image
- Add more surface types
- Try to investigate response curves for different seasons: wet/dry spring/fall
- Try and do the same for a Landsat image

Exercise: Wild fire detection in the US or Canada

- Use Landsat 8 or Sentinel-2 images to detect wild fires in the Western US, Canada[1], ... the Normalized Burn Ration (NBR)
- To see where there currently is wild fires consult https://fire.airnow.gov
- Consult the following web-page to learn and understand the NBR https://un-spider.org/advisory-support/recommended-practices/recommended-practice-burn-severity/in-detail/normalized-burn-ratio
- Hint: to subtract two images you can use the following command

```
var mydiff = image1.subtract(image2); // (image1-image2)
```

[1] A clear example is a fire is seen near Hay River, Canada (look for a recent image)