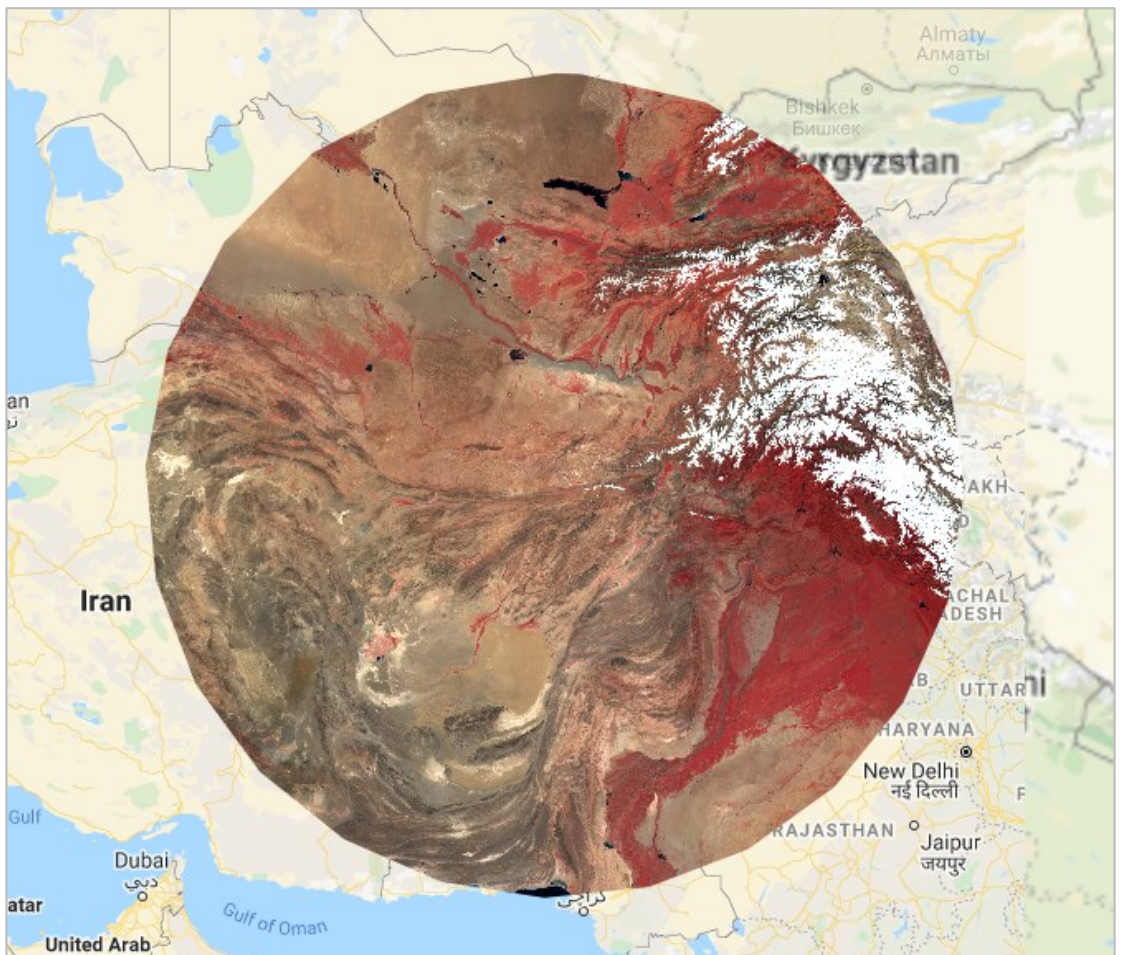
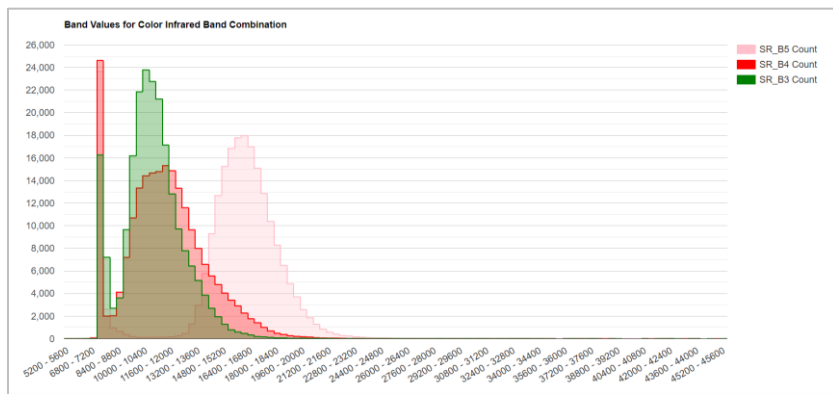


Outlook for Today:

- Result: Create histograms and a few band-combinations for a huge AOI
 - Get to know GEE
 - Get a feeling for cloud processing and why its so powerful
 - See why band combinations are useful
 - Enjoy learning about GEE!



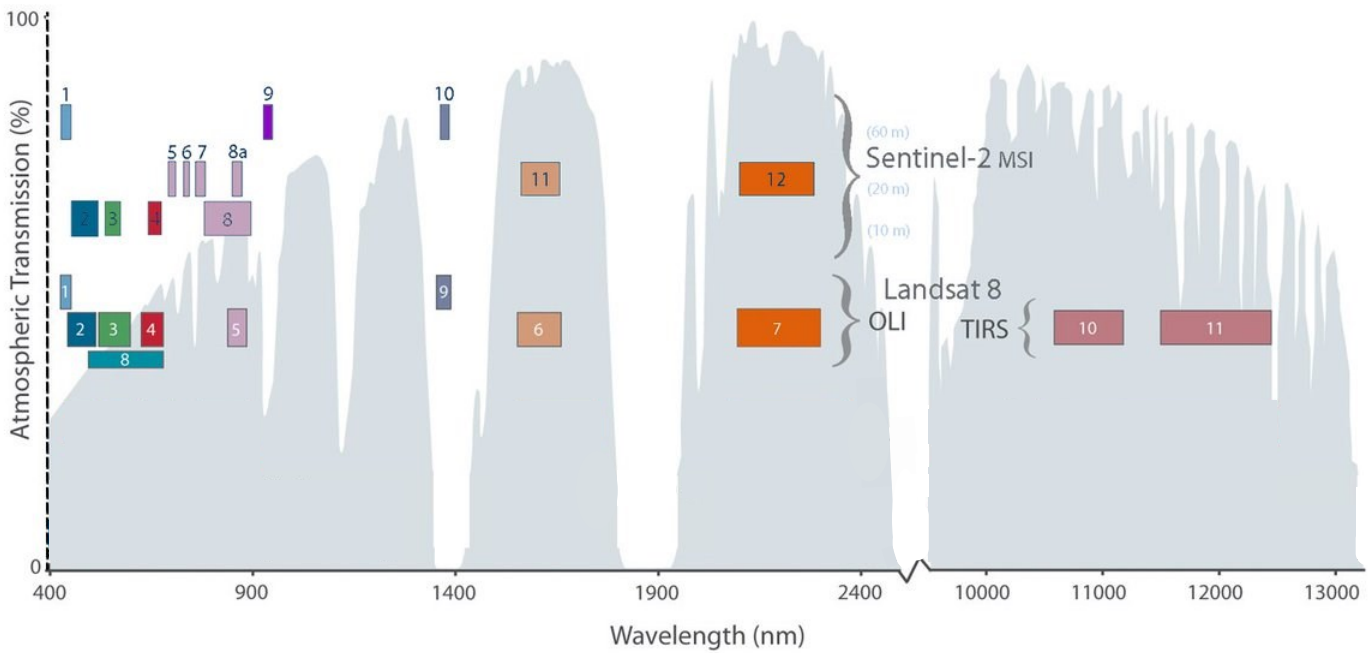


Today's topic: Band combinations

Knowing how to use them can be really useful!

Let's first take a look at what bands are actually captured by some satellite sensors.

Comparison of Landsat 7 and 8 bands with Sentinel-2



Practical: Image Data and Band Combinations in Google Earth Engine

■ Landsat 8 OLI (Operational Land Imager) and TIRS (Thermal Infrared Sensor)

Band	Resolution	Wavelength	Description
B1	30 m	0.433 to 0.453 μm	Coastal / Aerosol
B2	30 m	0.450 to 0.515 μm	Visible blue
B3	30 m	0.525 to 0.600 μm	Visible green
B4	30 m	0.630 to 0.680 μm	Visible red
B5	30 m	0.845 to 0.885 μm	Near-infrared
B6	30 m	1.56 to 1.66 μm	Short wavelength infrared
B7	60 m	2.10 to 2.30 μm	Short wavelength infrared
B8	15 m	0.50 to 0.68 μm	Panchromatic
B9	30 m	1.36 to 1.39 μm	Cirrus
B10	100 m	10.3 to 11.3 μm	Long wavelength infrared
B11	100 m	11.5 to 12.5 μm	Long wavelength infrared

■ Sentinel-2 MSI (Multispectral Imager)

Band	Resolution	Central Wavelength	Description
B1	60 m	443 nm	Ultra blue (Coastal and Aerosol)
B2	10 m	490 nm	Blue
B3	10 m	560 nm	Green
B4	10 m	665 nm	Red
B5	20 m	705 nm	Visible and Near Infrared (VNIR)
B6	20 m	740 nm	Visible and Near Infrared (VNIR)
B7	20 m	783 nm	Visible and Near Infrared (VNIR)
B8	10 m	842 nm	Visible and Near Infrared (VNIR)
B8a	20 m	865 nm	Visible and Near Infrared (VNIR)
B9	60 m	940 nm	Short Wave Infrared (SWIR)
B10	60 m	1375 nm	Short Wave Infrared (SWIR)
B11	20 m	1610 nm	Short Wave Infrared (SWIR)
B12	20 m	2190 nm	Short Wave Infrared (SWIR)

Practical: Image Data and Band Combinations in Google Earth Engine

Some commonly used band combinations:



Natural Color. Used for visual interpretation.



The color infrared composite is used for vegetation analyses. In this case, plants reflect near infrared and green light, while absorbing red. Since they reflect more near infrared than green, plant-covered land appears deep red. Denser plant growth is darker red. This band combination is valuable for analysing plant health.

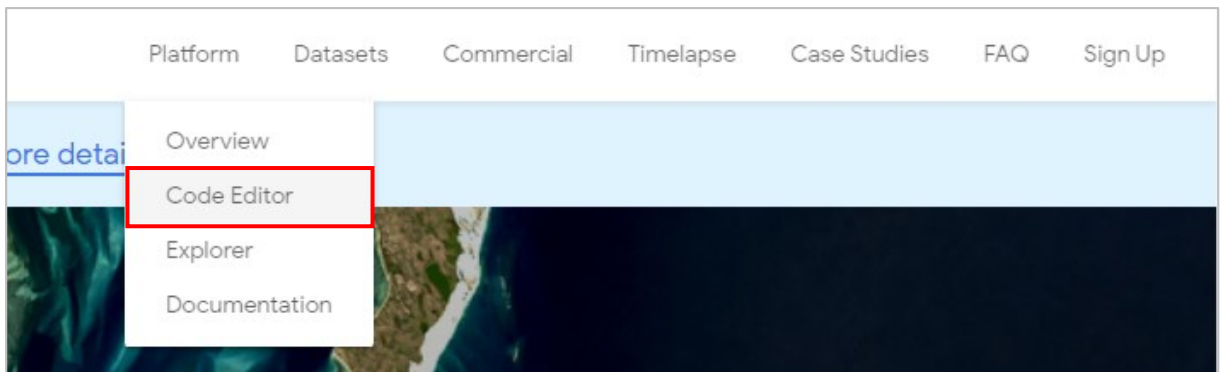


Portrays a landscape in false colors, but in a way that resembles the natural appearance. This combination is often used for geologic analyses (especially in desert landscapes since different surface soils appear in different colors), or agriculture, forestry, or fire management and post-fire analysis.

Practical: Image Data and Band Combinations in Google Earth Engine

Getting Started with Google Earth Engine

- For information signing up for GEE, see Blackboard > Information > Software and Data



The GEE Interface in plain words:

A screenshot of the Google Earth Engine web interface. The interface is divided into several sections: a left sidebar with 'Scripts', 'Docs', and 'Assets' panels; a central 'Code Editor' panel with a 'New Script' button and a 'Run' button; an 'Inspector' panel on the right showing 'Console' and 'Tasks' tabs; and a bottom map panel showing a satellite view of a region in the Northeastern United States. Red text annotations are overlaid on the interface to explain the components.

Scripts: Where you can save scripts to use them again later.

Docs: Short documentation of all algorithms that you can use.

Assets: Where you can upload or import additional data. For example an AOI.

Code Editor: The place where you tell GEE what you want to see and do and how you want to edit, visualise or analyse your images.

Inspector: Gives you information about things in the map. When you click on a pixel it will tell you the coordinates and band values, etc.

Console: Prints out anything that you told GEE to print in the Code Editor.

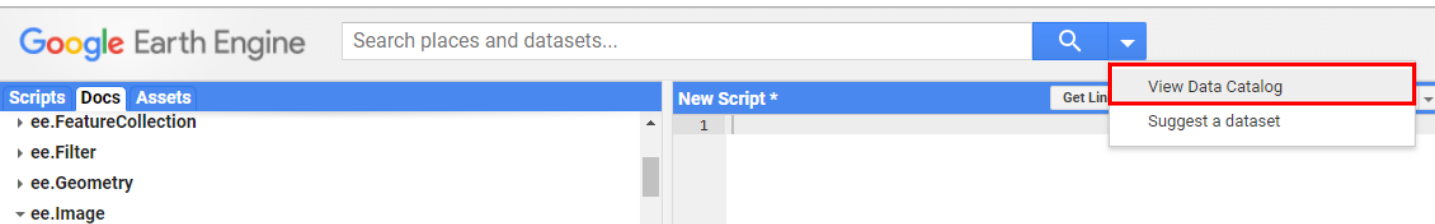
Tasks: Shows you when GEE is running your code.

Map: shows you the results of your code

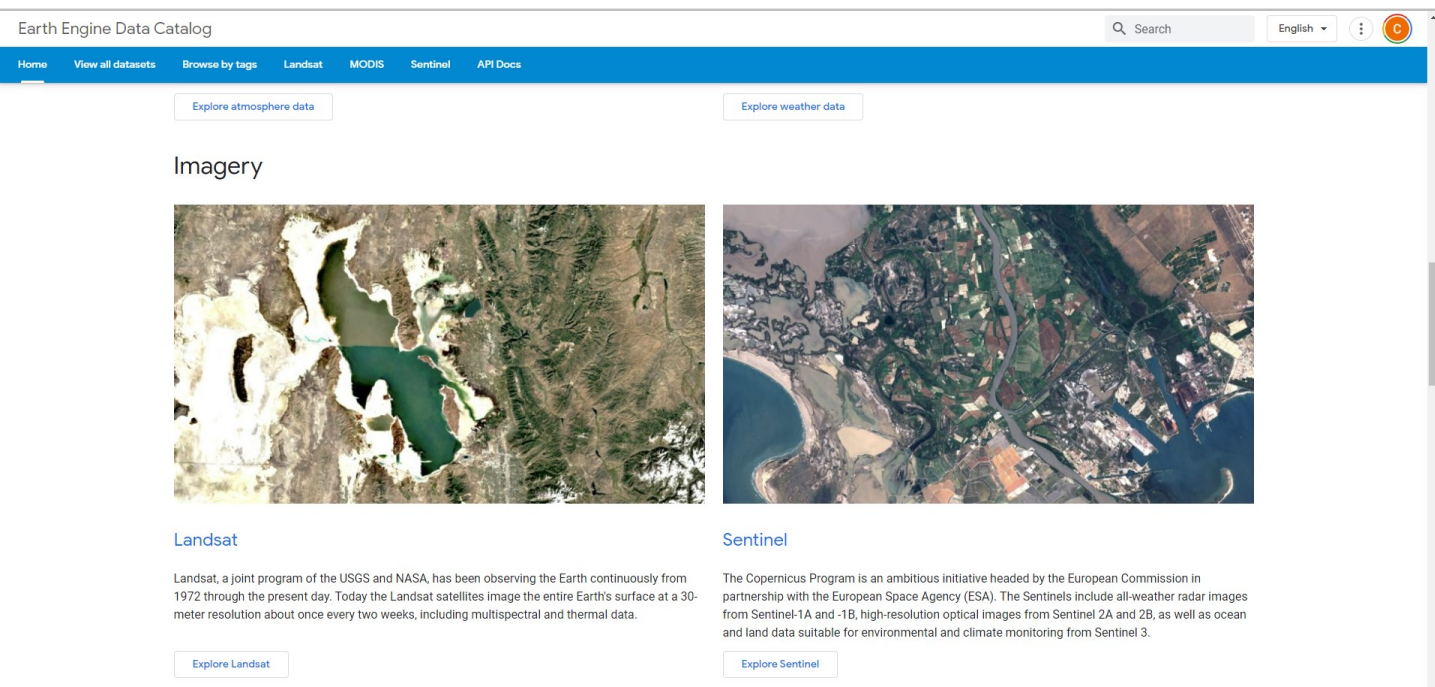
Practical: Image Data and Band Combinations in Google Earth Engine

Lets take a quick look at the Datasets

- Next to the search bar at the top, click on the drop-down menu and click on “View Data Catalog”



- In the new page, scroll down to “Imagery”



- Let’s “Explore Landsat”

- Here we can see a bunch of information about the Landsat Collections and Missions

Practical: Image Data and Band Combinations in Google Earth Engine

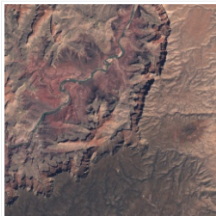
- If we now click on “Landsat 8 Surface Reflectance” in Collection 2, we can inspect some details about this dataset, and we can see relevant information for the code we will write.

Earth Engine Data Catalog

Search

Home View all datasets Browse by tags Landsat MODIS Sentinel API Docs

USGS Landsat 8 Level 2, Collection 2, Tier 1



Dataset Availability

2013-04-11T00:00:00Z - 2021-11-04T00:00:00

Dataset Provider

USGS

Earth Engine Snippet

```
ee.ImageCollection("LANDSAT/LC08/C02/T1_L2")
```

Dataset ID

Tags

cfmask cloud fmask global l8sr landsat lasrc
lc08 lst reflectance sr usgs

Description Bands Image Properties Terms of Use

This dataset contains atmospherically corrected surface reflectance and land surface temperature derived from the data produced by the Landsat 8 OLI/TIRS sensors. These images contain 5 visible and near-infrared (VNIR) bands and 2 short-wave infrared (SWIR) bands processed to orthorectified surface reflectance, and one thermal infrared (TIR) band processed to orthorectified surface temperature. They also contain intermediate bands used in calculation of the ST products, as well as QA bands.

Landsat 8 SR products are created with the Land Surface Reflectance Code (LaSRC). All Collection 2 ST products are created with a single-channel algorithm jointly created by the Rochester Institute of Technology (RIT) and National Aeronautics and Space Administration (NASA) Jet Propulsion Laboratory (JPL).

Strips of collected data are packaged into overlapping "scenes" covering approximately 170km x 183km using a [standardized reference grid](#).

Some assets have only SR data, in which case ST bands are present but empty. For assets with both ST and SR bands, 'PROCESSING_LEVEL' is set to 'L2SP'. For assets with only SR bands, 'PROCESSING_LEVEL' is set to 'L2SR'.

[Additional documentation and usage examples](#).

Data provider notes:

- Data products must contain both optical and thermal data to be successfully processed to surface temperature, as ASTER NDVI is required to temporally adjust the ASTER GED product to the target Landsat scene. Therefore, night time acquisitions cannot be processed to surface temperature.

- When we specify which bands to show, the Band Names are important. The naming convention may change from dataset to dataset, so it's always good to check!

Description	Bands	Image Properties	Terms of Use				
Resolution							
30 meters							
Bands							
Name	Units	Min	Max	Scale	Offset	Wavelength	Description
SR_B1		1	65455	2.75e-05	-0.2	0.435-0.451 µm	Band 1 (ultra blue, coastal aerosol) surface reflectance
SR_B2	Band Names	1	65455	2.75e-05	-0.2	0.452-0.512 µm	Band 2 (blue) surface reflectance
SR_B3		1	65455	2.75e-05	-0.2	0.533-0.590 µm	Band 3 (green) surface reflectance
SR_B4		1	65455	2.75e-05	-0.2	0.636-0.673 µm	Band 4 (red) surface reflectance
SR_B5		1	65455	2.75e-05	-0.2	0.851-0.879 µm	Band 5 (near infrared) surface reflectance
SR_B6		1	65455	2.75e-05	-0.2	1.566-1.651 µm	Band 6 (shortwave infrared 1) surface reflectance
SR_B7		1	65455	2.75e-05	-0.2	2.107-2.294 µm	Band 7 (shortwave infrared 2) surface reflectance
ST_B10	Kelvin	0	65535	0.00341802	149	10.60-11.19 µm	Band 10 surface temperature. If 'PROCESSING_LEVEL' is set to 'L2SR', this band is fully masked out.

Practical: Image Data and Band Combinations in Google Earth Engine

- The same goes for the Image Properties. Often we will want to use the image properties to filter a dataset (for example for cloud cover), so we need to make sure we are using the correct image property name.

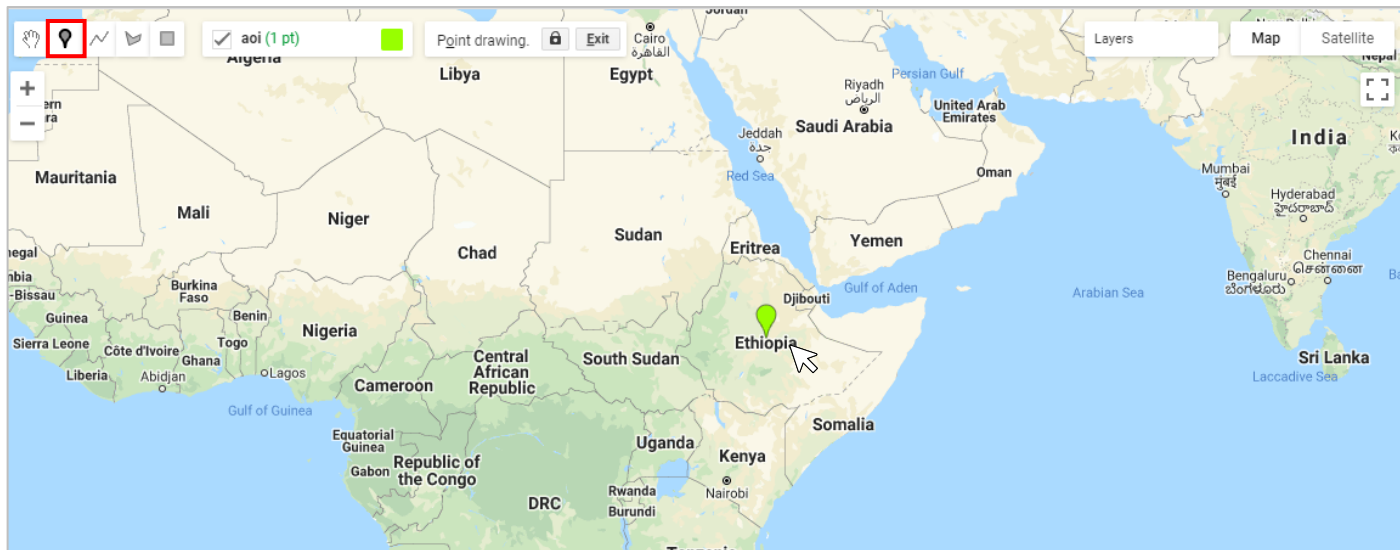
Description	Bands	<u>Image Properties</u>	Terms of Use
Image Properties			
Name	Property Names	Type	Description
ALGORITHM_SOURCE_SURFACE_REFLECTANCE		STRING	Name and version of the surface reflectance algorithm.
ALGORITHM_SOURCE_SURFACE_TEMPERATURE		STRING	Name and version of the surface temperature algorithm.
CLOUD_COVER		DOUBLE	Percentage cloud cover (0-100), -1 = not calculated.
CLOUD_COVER_LAND		DOUBLE	Percentage cloud cover over land (0-100), -1 = not calculated.
COLLECTION_CATEGORY		STRING	Scene collection category, "T1" or "T2".
DATA_SOURCE_AIR_TEMPERATURE		STRING	Air temperature data source.
DATA_SOURCE_ELEVATION		STRING	Elevation data source.
DATA_SOURCE_OZONE		STRING	Ozone data source.
DATA_SOURCE_PRESSURE		STRING	Pressure data source.
DATA_SOURCE_REANALYSIS		STRING	Reanalysis data source.
DATA_SOURCE_TIRS_STRAY_LIGHT_CORRECTION		STRING	TIRS stray light correction data source.
DATA_SOURCE_WATER_VAPOR		STRING	Water vapor data source.
DATE_PRODUCT_GENERATED		DOUBLE	Timestamp of the date when the product was generated.

- With this information we can already start to create our own code.
 - Switch back to the Code Editor Tab.

Practical: Image Data and Band Combinations in Google Earth Engine

1. Create a Point of Interest anywhere in the world

- Use the Geometry Drawing Tools in the map to draw a point
- Don't go too close to the oceans



- As soon as you finish drawing it, notice the geometry appear in the Code Editor
- Rename it to 'poi' (simply click on the word 'geometry' to edit it)

```
Imports (1 entry)
var poi: Point (39.60, 8.92)
```

- Now we can create a very large AOI, by adding a buffer of 1 million meters (1000 km)
- Add this code at the top of your script

```
// 1. create area of interest around point, with a radius of 1 million meters
var aoi = poi.buffer(1000000);
```

- With a radius of 1 million meters, we now have an AOI of about 3.1 million km²

Practical: Image Data and Band Combinations in Google Earth Engine

2. Create Variables for Start and End Dates for the Search

The // indicate a comment.

Its best practice to use comments to keep track of what your code is doing

```
// 2. Set start and end date range  
var start = ee.Date('2019-01-01');  
var end = ee.Date('2021-01-01');
```

Create variables to contain start and end dates

ee.Date is a built-in function that can read and understand dates

The date: yyyy-mm-dd

Practical: Image Data and Band Combinations in Google Earth Engine

3. Create a Variable for our ImageCollection and filter it to find fitting images

```
// 3. Call and filter an image collection  
var myImageCollection = ee.ImageCollection('LANDSAT/LC08/C02/T1_L2')  
.filterBounds(aoi) ←  
.filterDate(start, end) ←  
.filter('CLOUD_COVER < 15');
```

ee.ImageCollection is a built-in function that calls image collections

Remember the Dataset ID we saw in the Landsat Image Collection? This is where we need it

The filtering function excludes any images that don't fit the criteria

less than 15 %

Remember the Image Properties we saw in the Landsat Image Collection? This is where we need it

filterBounds means we are filtering by our AOI (use the name that you gave your AOI here)

filterDate means we are filtering by our start and end dates as the time period to search

Practical: Image Data and Band Combinations in Google Earth Engine

4. Print out some Information

Create a variable that contains the size (= count) of images that are now in our “myImageCollection”

```
// 4. print out information about how many images were found to fit our criteria
var size = myImageCollection.size();
print('Number of images that fit the criteria: ', size);
```

The print function will print something to the console

Descriptive text to be printed to the console

Print out the value of the variable “size”

- Now we can click “Run” to see if everything is going well so far. If all is well, we will see a result in the Console.

The screenshot shows the Google Earth Engine web interface. The top navigation bar includes the Google Earth Engine logo, a search bar, and utility icons. Below the navigation bar is the 'New Script' editor with a menu bar containing 'Get Link', 'Save', 'Run' (highlighted with a red box), 'Reset', and 'Apps'. The script editor contains the following code:

```
5 var start = ee.Date('2019-01-01');
6 var end = ee.Date('2020-01-01');
7
8 // 3. Call and filter an image collection
9 var myImageCollection = ee.ImageCollection('LANDSAT/LC08/C02/T1_L2')
10 .filterBounds(aoi)
11 .filterDate(start, end)
12 .filter('CLOUD_COVER < 15');
13
14 // 4. print out information about how many images were found to fit our crit
15 var size = myImageCollection.size();
16 print('Number of images that fit the criteria: ', size);
17
18
19
20
```

On the right side of the interface is the 'Inspector Console' panel. It contains the text 'Use print(...) to write to this console.' Below this, the output of the script is displayed: 'Number of images that fit the criteria: 1543'. The output is enclosed in a red box. The bottom part of the screenshot shows a map of the region, with a green location pin placed over Ethiopia. The map includes labels for countries like Sudan, Eritrea, Yemen, Ethiopia, and Somalia, as well as cities like Khartoum, Sana'a, and Djibouti. The map also shows the Gulf of Aden and the Red Sea.

5. Create Median Collection

- The `.median()` function calculates the median of all values at each pixel across the stack of all matching bands.

```
// 5. Create collection of median pixel values of all valid images  
var median = myImageCollection.median();
```

Practical: Image Data and Band Combinations in Google Earth Engine

6. Define Band Combinations

- We can prepare the selection of bands by defining them in a variable

```
// 6. Define Band Combinations
var bandsRGB = ['SR_B4', 'SR_B3', 'SR_B2'];
var bandsNIR = ['SR_B5', 'SR_B4', 'SR_B3'];
var bandsSWIR = ['SR_B7', 'SR_B6', 'SR_B4'];
```

Here we use the band names that we saw in the data catalog for Landsat 8

- Now we can test to see how the visualisation looks without any stretching

```
// 7. Test non-stretched visualisation
Map.addLayer(median.clip(aoi), {bands: bandsRGB}, 'Not Stretched RGB');
```

Generic function to add a layer to the map

Layer to add

Layer Visualisation Parameters

Layer Name

■ “Run”

- Looks a bit dark, doesn't it?



Practical: Image Data and Band Combinations in Google Earth Engine

8. Create Histograms to View Min and Max Values

- With the help of the histogram we can decide where to set the Min and Max values for our visualisations

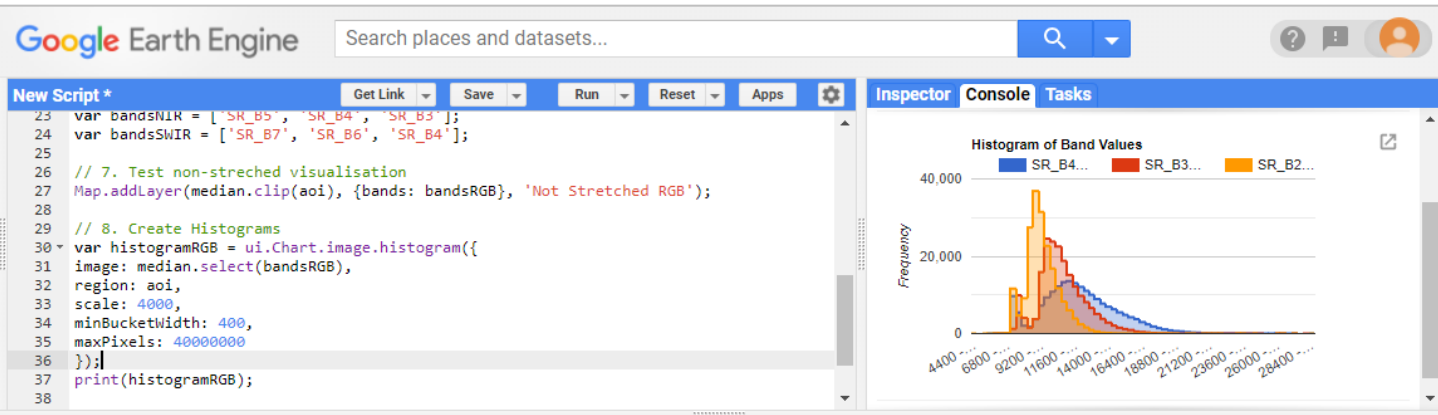
```
// 8. Create Histogram
```

```
var histogramRGB = ui.Chart.image.histogram({  
  image: median.select(bandsRGB),  
  region: aoi,  
  scale: 4000,  
  minBucketWidth: 400,  
  maxPixels: 40000000  
});  
print(histogramRGB);
```

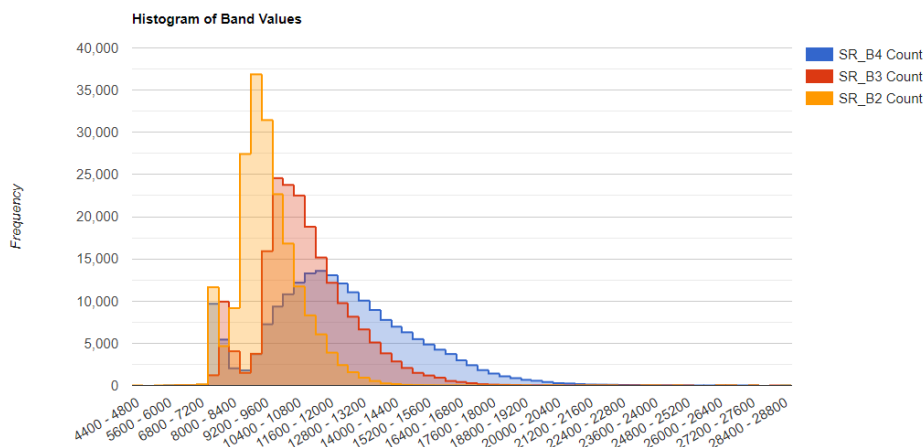
← Generic function to create a histogram

← Print the histogram

- “Run”



- You can click on the  “Open in New Window” button to see the histogram in full size.



Practical: Image Data and Band Combinations in Google Earth Engine

9. Set Visualisation for Min and Max Values and add the Stretched Results to the Map

- With the help of the histogram we can now see which min and max values to define

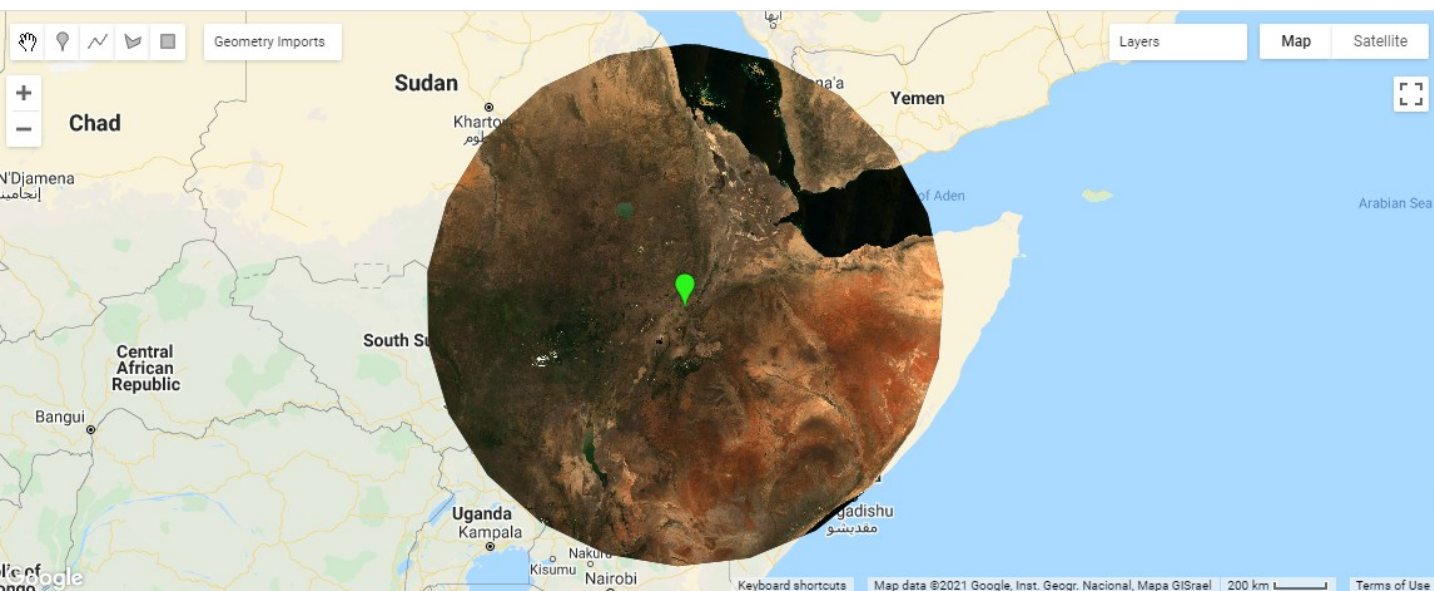
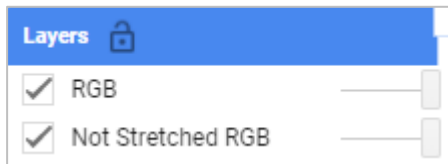
```
// 9. Add images to the map  
Map.addLayer(median.clip(aoi), {bands: bandsRGB, min: 7200, max: 20800}, 'RGB');
```

Image

Layer Visualisation

Layer Name

- Click “Run”
- Use the Layers Menu in the Map to turn your layers on and off to see the others



Practical: Image Data and Band Combinations in Google Earth Engine

10. Do the same for other band combinations

- Go back to 8. and add two more histograms (you can copy and paste the first one) for these new band combinations:

```
// 8. Create Histograms

var histogramRGB = ui.Chart.image.histogram({
  image: median.select(bandsRGB),
  region: aoi,
  scale: 4000,
  minBucketWidth: 400,
  maxPixels: 40000000
});
print(histogramRGB);

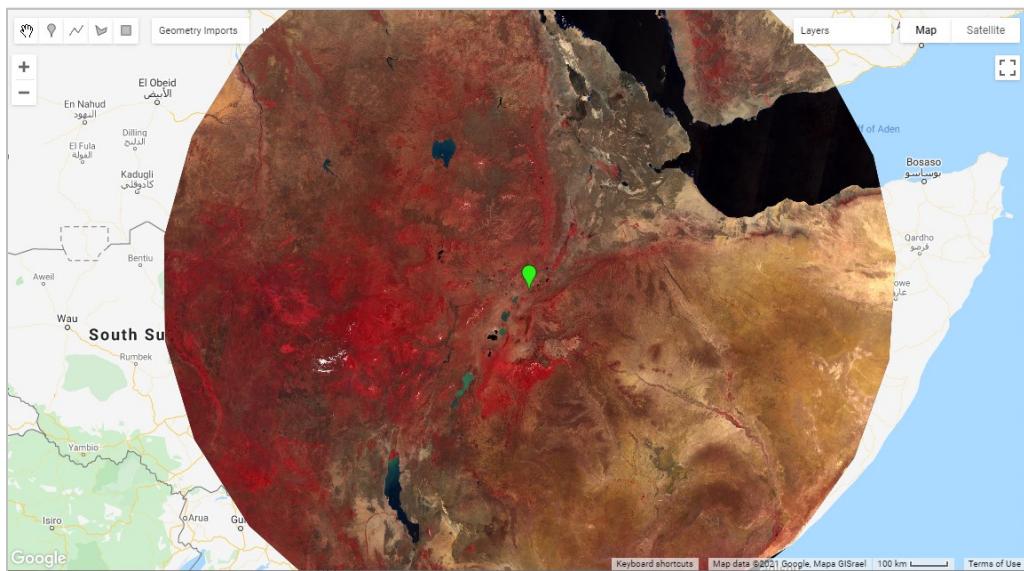
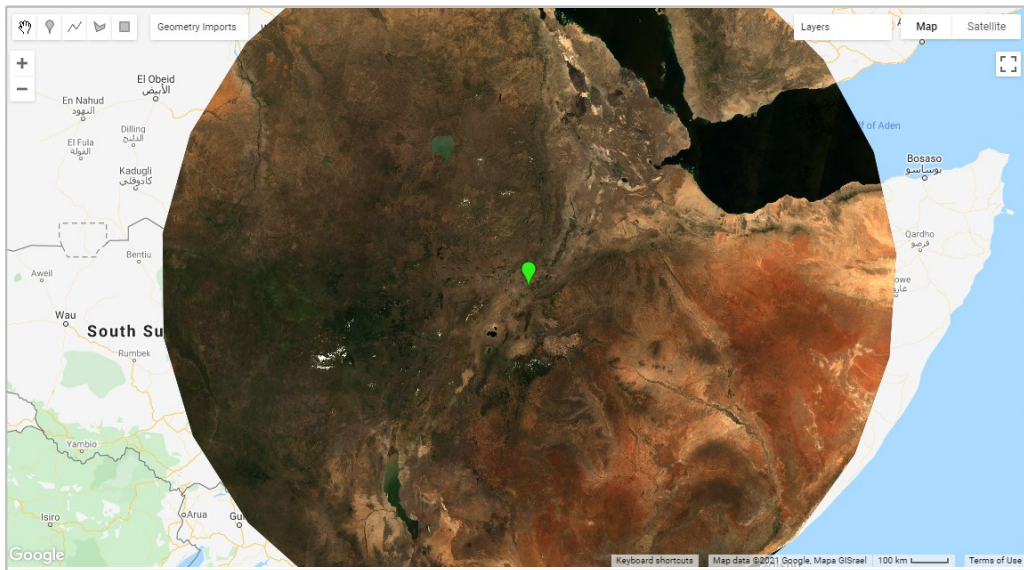
var histogramNIR = ui.Chart.image.histogram({
  image: median.select(bandsNIR),
  region: aoi,
  scale: 4000,
  minBucketWidth: 400,
  maxPixels: 40000000
});
print(histogramNIR);

var histogramSWIR = ui.Chart.image.histogram({
  image: median.select(bandsSWIR),
  region: aoi,
  scale: 4000,
  minBucketWidth: 400,
  maxPixels: 40000000
});
print(histogramSWIR);
```

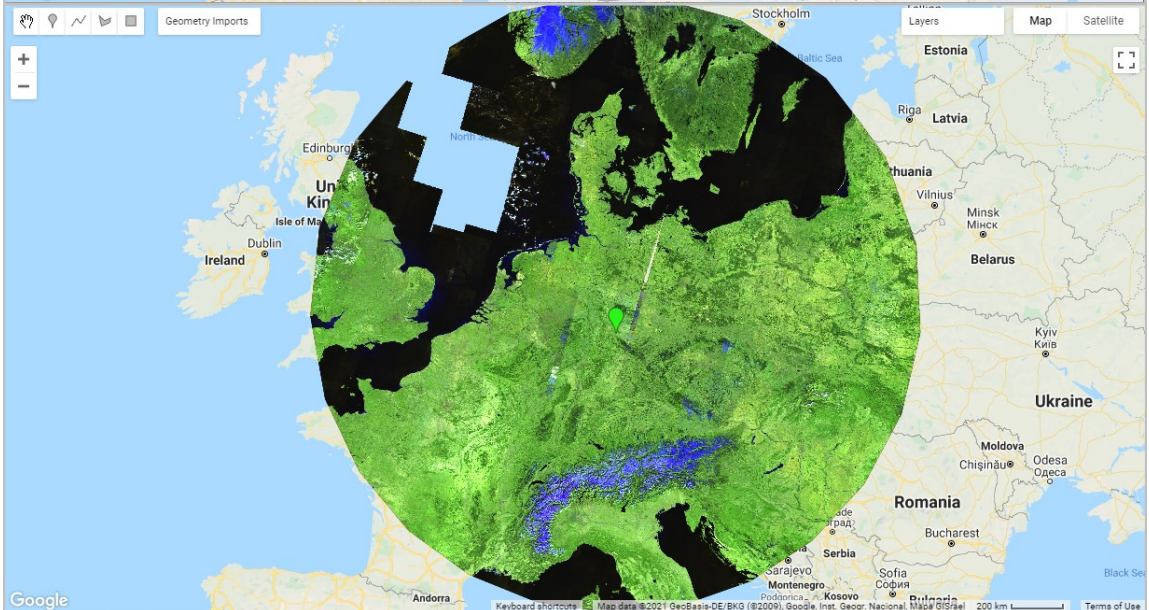
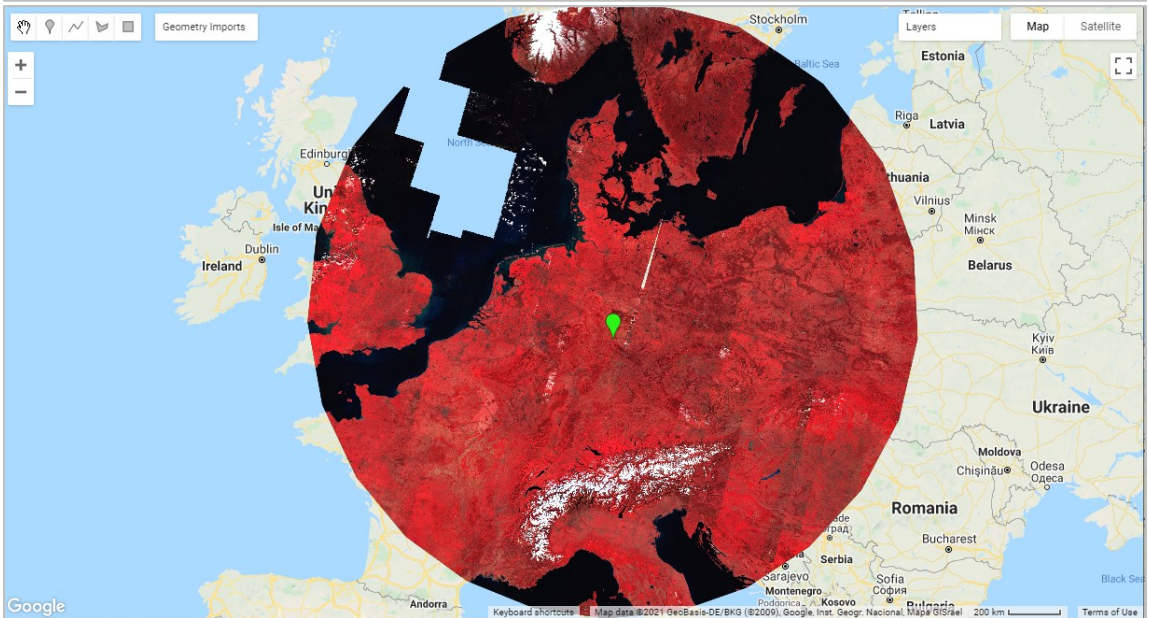
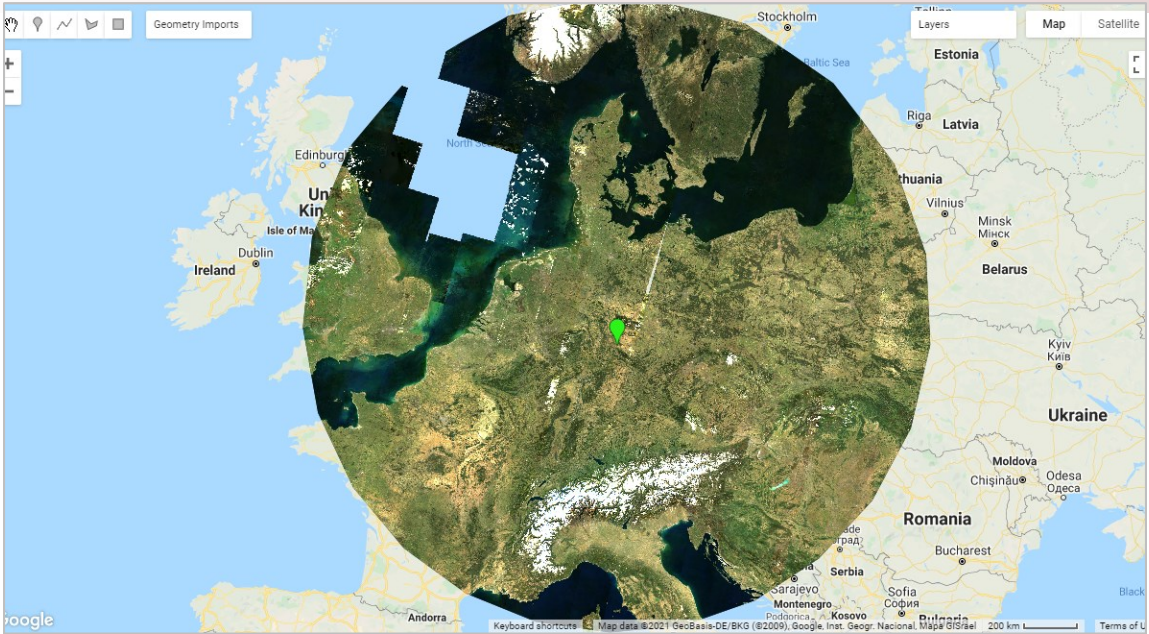
- “Run”
- Use the histograms to set new visualisations for the NIR and SWIR visualisations

```
// 9. Add images to the map
Map.addLayer(bestImage, {bands: bandsRGB, min: 7200, max: 19200}, 'RGB');
Map.addLayer(bestImage, {bands: bandsNIR, min: 7200, max: 24000}, 'NIR');
Map.addLayer(bestImage, {bands: bandsSWIR, min: 7200, max: 27600}, 'SWIR');
```

Practical: Image Data and Band Combinations in Google Earth Engine



Practical: Image Data and Band Combinations in Google Earth Engine



Practical: Image Data and Band Combinations in Google Earth Engine

- You can easily change this code for anywhere else in the world by **simply changing the point coordinates** at the very top and clicking “run” again!!
 - You may need to adjust the min and max values in the visualisation based on the histogram

Here: my “poi” is in the middle of Afghanistan

The screenshot displays the Google Earth Engine web interface. At the top, there is a search bar and navigation tabs for Scripts, Docs, and Assets. The main area is divided into a code editor and a console. The code editor shows a script named 'Practical3_LargeAOI' with the following code:

```
1 // point of interest
2 var poi = ee.Geometry.Point(66.25, 34.17);
3
4 // create area of interest around point, with a radius of 1 million km
5 var aoi = poi.buffer(1000000);
6
7 // Set start and end date range
8 var start = ee.Date('2019-01-01');
9 var end = ee.Date('2021-01-01');
10
11 // Call and filter an image collection
12 var myImageCollection = ee.ImageCollection('LANDSAT/LC08/C02/T1_L2')
13   .filterBounds(aoi)
14   .filterDate(start, end);
```

The console on the right shows the results of the script execution: "Number of images that fit the criteria: 4566". Below the code editor, a map of the region around Afghanistan is displayed. A large circular visualization is overlaid on the map, showing a composite of satellite images in a false-color scheme (red, brown, and white), representing the filtered data for the specified area and time period. The map includes labels for various countries and cities, such as Georgia, Azerbaijan, Iran, Saudi Arabia, and India.

Practical: Image Data and Band Combinations in Google Earth Engine

Here: my “poi” is in the middle of Japan, North and South Korea!

The screenshot displays the Google Earth Engine web interface. At the top, there is a search bar and navigation icons. Below the search bar, the 'Scripts' tab is active, showing a script named 'Practical3_LargeAOI'. The script code is as follows:

```
1 // point of interest
2 var poi = ee.Geometry.Point(134.78, 39.96);
3
4 // create area of interest around point, with a radius of 1 million km
5 var aoi = poi.buffer(1000000);
6
7 // Set start and end date range
8 var start = ee.Date('2019-01-01');
9 var end = ee.Date('2021-01-01');
10
11 // Call and filter an image collection
```

The 'Inspector' and 'Console' tabs are also visible. The console shows the output: 'Number of images that fit the criteria: 1572'. The main map area shows a satellite view of East Asia, with a large circular area of interest (AOI) overlaid. The AOI is a dark red circle centered over the Sea of Japan, encompassing parts of North Korea, South Korea, and Japan. The map includes labels for various cities and regions, such as Ulaanbaatar, Beijing, Tianjin, Dalian, Qingdao, and Shanghai.

Practical: Image Data and Band Combinations in Google Earth Engine

Here: my “poi” is in the middle of Australia!

The screenshot shows the Google Earth Engine interface. The top bar includes the Google Earth Engine logo, a search bar, and navigation icons. Below the search bar are tabs for Scripts, Docs, and Assets. The main area is divided into three panels: Scripts, Code Editor, and Inspector/Console. The Code Editor shows a script named 'Practical3_LargeAOI' with the following code:

```
1 // point of interest
2 var poi = ee.Geometry.Point(140.5, -25.34);
3
4 // create area of interest around point, with a radius of 1 million km
5 var aoi = poi.buffer(1000000);
6
7 // Set start and end date range
8 var start = ee.Date('2019-01-01');
9 var end = ee.Date('2021-01-01');
10
11 // Call and filter an image collection
```

The Inspector/Console panel shows the output of the script: 'Number of images that fit the criteria: 4846'. The map below shows Australia with a large circular area of interest (AOI) centered in the middle of the continent. The AOI is filled with a brownish-red color, representing the filtered image data. The map also shows major cities like Perth, Adelaide, Sydney, and Brisbane, and the Coral Sea to the east.

This screenshot shows a zoomed-in view of the AOI from the previous screenshot. The map is centered on the AOI, which is now a large, irregularly shaped area with a jagged black border. The AOI is filled with a brownish-red color, representing the filtered image data. A small green pin is visible in the center of the AOI, indicating the point of interest. The map also shows the Great Australian Bight to the south and the Tasman Sea to the east. The interface elements like the search bar, navigation icons, and map controls are visible at the top and right.

Practical: Image Data and Band Combinations in Google Earth Engine

Zoom in South of Baghdad



Practical: Image Data and Band Combinations in Google Earth Engine

Complete Script

- Set an geometry Point called “poi” with the geometry tools!
- Adjust the min and max values for the layers (use the histograms to decide!)

```
Imports (1 entry)   
▶ var poi: Point (39.41, 8.63)    
  
// 1. create area of interest around point, with a radius of 1 million meters  
var aoi = poi.buffer(1000000);  
  
// 2. set start and end dates  
var start = ee.Date('2019-01-01');  
var end = ee.Date('2020-01-01');  
  
// 3. Call and filter an image collection  
var myImageCollection = ee.ImageCollection('LANDSAT/LC08/C02/T1_L2')  
.filterBounds(aoi)  
.filterDate(start, end)  
.filter('CLOUD_COVER < 15');  
  
// 4. print out information about how many images were found to fit our criteria  
var size = myImageCollection.size();  
print('Number of images that fit the criteria: ', size);  
  
// 5. Create collection of median pixel values of all valid images  
var median = myImageCollection.median();  
  
// 6. Define Band Combinations  
var bandsRGB = ['SR_B4', 'SR_B3', 'SR_B2'];  
var bandsNIR = ['SR_B5', 'SR_B4', 'SR_B3'];  
var bandsSWIR = ['SR_B7', 'SR_B6', 'SR_B4'];  
  
// 7. Test non-streched visualisation  
Map.addLayer(median.clip(aoi), {bands: bandsRGB}, 'Not Stretched RGB');  
  
// 8. Create Histograms  
var histogramRGB = ui.Chart.image.histogram({  
  image: median.select(bandsRGB),  
  region: aoi,  
  scale: 4000,  
  minBucketWidth: 400,  
  maxPixels: 40000000  
});  
print(histogramRGB);  
  
var histogramNIR = ui.Chart.image.histogram({  
  image: median.select(bandsNIR),  
  region: aoi,  
  scale: 4000,  
  minBucketWidth: 400,  
  maxPixels: 40000000  
});  
print(histogramNIR);  
  
var histogramSWIR = ui.Chart.image.histogram({  
  image: median.select(bandsSWIR),  
  region: aoi,  
  scale: 4000,  
  minBucketWidth: 400,  
  maxPixels: 40000000  
});  
print(histogramSWIR);  
  
// 9. Add images to the map  
Map.addLayer(median.clip(aoi), {bands: bandsRGB, min: 7200, max: 20800}, 'RGB');  
Map.addLayer(median.clip(aoi), {bands: bandsNIR, min: 7200, max: 24000}, 'NIR');  
Map.addLayer(median.clip(aoi), {bands: bandsSWIR, min: 7200, max: 27600}, 'SWIR');
```