Introduction to Google Earth Engine



1. Introduction

Hello,

Thanks for joining in the webinar today. My name is Warren Scott and I am a trainer with the Geospatial Technology and Applications Center (GTAC). In this quick presentation and demo, you will learn what Google Earth Engine is, how to conduct geospatial analyses as well as how to access data.

OK, let’s get started.

* 1. What is Earth Engine?

It is a cloud-based geospatial processing platform for executing large-scale environmental data analysis. Applications of environmental analyses in Earth Engine include detecting deforestation, classifying land cover, and urban mapping.

Three aspects of Earth Engine make it quite powerful as a geospatial processing tool. The first is its extensive public data catalog, the second is the processing power of the computation engine, and the third is its interactive development platforms.

* 1. Data Catalog

In the data catalog, there is over 40 years of current and historical imagery and data in the Earth Engine data library. Available data sets include

1. **Imagery** – these imagery sets include both raw and pre-processed, georeferenced data products that are ready for use. Examples include NAIP, comprehensive Landsat archive, and MODIS data and ancillary products such as global composites and max NDVI.
2. **Geophysical data** – such as digital elevation models, surface temperature, and land cover
3. **Climate and weather** – NASA’s TOMS atmospheric data and – forecast weather and climate model data
4. **Demographic data** – such as population density estimates

Users are also able to load in their own data – both imagery and vector files to use for analysis.

* 1. Computation Engine

*The computation engine is a just in time distributed computation model****,*** *a cloud-based processing infrastructure that automatically parallelizes analyses on many CPUs across many computers in Google’s data centers.*

This means that the analysis requests are sent out to many different computers; each computer runs the calculations on a small area (small number of pixels) and returns the results. Requests are also efficient – it only computes for the input data of your requested values or to the extent of your screen. Results are then cached so that multiple requests for the same image or values do not result in re-computation. This results in:

– unprecedented speed: reduce processing times by orders of magnitude by using the cloud-based computing power

– Ease of use and lower costs. Online platform with easy access to data, scientific algorithms, computational power.

Read more here: <https://developers.google.com/earth-engine/>

* 1. Interactive Development Platforms:

Earth Engine’s geospatial processing algorithms and API interface enables interactive algorithm development. These include basic spatial analysis operations (overlay, map algebra, vector based extraction of image statistics, etc.) and more advanced algorithms such as image classification, terrain modeling, and time series analysis.

Results can be reported as charts, maps, tables, or image exports. Google Earth Engine algorithms and user-submitted sample scripts – are constantly being added, enhanced, and updated – enabled by an open source environment.

Highly interactive algorithm development – a web based IDE for rapid proto-typing and visualization of complex spatial analyses across the world. Read more here: <https://earthengine.google.com/platform/>

* 1. Two Platforms
1. Explorer - a light-weight web app with point and click processing.
2. Code Editor - an IDE where users can program analysis scripts in either JavaScript or python. This allows the user to take advantage of a larger suite of Earth Engine processing algorithms and methods and create more complicated workflows.

Both platforms include a mapping window for instant visualization of results and data export capabilities. You have to register to access computational power in either platform; although you can visualize data in the Explorer without having an account.

* 1. Registration Process
* Sign up for an Evaluator account – may take up to one week to be granted access. You can register here: <https://earthengine.google.com/signup/>.

Part 2: Explorer Demo

We’ll first explore the capacity of Explorer: access and visualize data, simple computations, and data export.

Here’s the link to my workspace with the loaded data sets and computations: <https://explorer.earthengine.google.com/#workspace/eqqRwhqPJ8k>

* + 1. You can access Explorer here: <https://explorer.earthengine.google.com/#workspace>
		2. Make sure you log in if you would like to run computations, save your workspace, or export your data.
	1. Data Catalog
		1. There are two buttons in the upper right hand panel that allow you toggle between the data catalog and workspace.
	2. Landsat 8
		1. At the top of the webpage there is a search bar where users can search for data available in Earth Engine. All the data that was mentioned in the Datasets page on the Earth Engine website can be accessed and loaded into the workspace from here.
		2. Search for and load the data: Landsat 8 8 Day composite.
		3. Now notice that the data is displayed in the workspace.
		4. Explore metadata/asset details by clicking on the “eye” symbol next to the specific asset in the workspace view.
		5. Change date.
		6. Export data.
		7. Show the manage workspace.
	3. Classification and Analysis
		1. Explore NDVI calculations

img1[5]-img1[4]/( img1[5]+img1[4])

* + 1. Show how to change display – palette.
	+ hex colors here: <http://www.rapidtables.com/web/color/brown-color.htm>
		1. Export data – goes into Downloads folder on your c drive.
		2. Manage workspace: <https://explorer.earthengine.google.com/#workspace/PJUVhFXBkln>
1. Code Editor Demo

In the Code Editor, the API, users can script more complex analysis than what is available in the Explorer – by creatively recombining existing algorithms.

Open the Code editor at: <https://code.earthengine.google.com/>

* 1. Layout
		1. Point out the
			1. Data search bar – this is similar to the one we saw in Explorer
			2. Scripts and docs –
				1. Scripts Manager - a wide variety of preloaded example scripts to provide simple examples of API usage;
				2. Shared and your scripts can also be saved here – in the private scripts area;
				3. Documentation List – a searchable list of documentation for the predefined GEE object types and methods (functions).
			3. Text editor – write and edit code here. Click run to execute script.
			4. Map output area – visualize the results of your query or analysis
			5. Information panel - inspector, console, and tasks.
				1. Inspector - locate information about the layers in your map,
				2. Console - return messages as the scripts run and record any errors,
				3. Manage the exporting of data and results to your google drive.
		2. The **Help** dropdown menu in the upper right hand corner of the screen has many nice documents. The **Feature Tour** is a nice introduction that points out the different panels and features of the Code Editor.
	2. Text Editor
		1. Let’s get started by first running an example script. In the **Scripts** tab in the upper left hand corner, select the **normalized difference** from the list of **Examples** (under **Image**). You can also search for this in the **Filter Scripts…** search bar.
		2. Let’s quickly look at the code that populates the Text Editor.
			1. You set variables with **var**.
			2. **ee.Image** sets up an image as an Earth Engine image object.
			3. Palette is set up as a variable holding an array of hexadecimal colors that range from yellow to dark green – read more about hexadecimal color codes here <http://www.rapidtables.com/web/color/brown-color.htm>.
			4. **Map.setCenter** aligns the map display to an area of interest at a specified zoom level.
			5. **Map.addLayer** adds the
			6. Running the normalized difference function – show the function in the docs
		3. Click **Run**.
		4. Adjust layers.
		5. Show them the Inspector to ID pixel values.
	3. Modify Script

Let’s change the script to run on a Landsat image.

* + 1. Setting variables – change the data to point to the Landsat image.

// Define an image

var LC8\_image = ee.image('LC8\_L1T/LC81280502014042LGN00');

**Note**: if you copy and paste the text from here into the Code Editor, sometimes the font has issues with the quotes – you may have to delete the quotes and retype them in the Code Editor.

* + 1. Change the bands to 5 and 4 creating an NDVI image

//Create NDVI image using normalized difference operator

var ndvi = Lc8\_image.normalizedDifference([‘B5, 'B4']);

* + 1. Adjust the centering options for map display.

//Map.setCenter(-94.84497, 39.01918, 8);

//center map on the LC8 image
Map.centerObject(LC8\_image);

* + 1. Update the export process.
			1. Change the image variable name to the referring to the Landsat image.
			2. Change the band names to match the Landsat bands.
			3. Change the stretch parameters
			4. Rename the legend entry – Landsat8scene

//Add LC8\_image to map

Map.addLayer(LC8\_image.select('B6', 'B5', 'B4'), {'min': 10000, 'max': 30000},'Landsat8Scene');

//Add NDVI layer to Map

Map.addLayer(ndvi);

//Export both NDVI and Landsat 8 image to your google Drive

Export.image.toDrive(LC8\_image, "Landsat8Image");

Export.image.toDrive (ndvi, "NDVI\_Image");

* + 1. Here’s the link to my code:

<https://code.earthengine.google.com/4222dc7af661ae3f7caebd7b9e48dad6>

**Notes about JavaScript syntax:**

// notes

Variables declared with var

ee.Image – function that loads an image as an object

can select and clip the image…

can use ee.Feature for vectors

semicolon to end statements