Tutorial for mapping forest disturbances across the Southwestern Amazon using CODED, LandTrendr, and MTDD

SERVIR AMAZONIA

Reygadas Langarica, Yunuen

UNIVERSITY OF RICHMOND

Table of Contents

Tutorial for mapping forest disturbances across the Southwestern Amazon using CODED, LandTrendr, and MTDD

1. Background

Due to the urgent need to understand the effects of forest disturbances on ecosystem services, multiple remote sensing algorithms focused on detecting vegetation changes have been developed in the last decade. The challenge now lies in understanding which algorithm best suits the user´s study area and research objective. This tutorial shares our Google Earth Engine (GEE) work derived from the evaluation of the performance of three algorithms–Continuous Degradation Detection (CODED), Landsat-based detection of trends in disturbance and recovery (LandTrendr), and Multi-variate Time-series Disturbance Detection (MTDD)– to detect and characterize forest disturbances in the Southwestern Amazon (SWA). For the entire study, please refer to Reygadas et al. (2021) (currently under review).

In section 3.3, we share MTDD-based GEE codes to map disturbances in the SWA (i.e., Ucayali, Peru and Acre, Brazil). Complete GEE tutorials for CODED and LandTrendr are freely available online; in this tutorial, we share the specific configuration-parametrization for the SWA.

2. Definitions

Although it is hard to adopt a set of definitions that harmonize with the fundamental logic behind each algorithm, we refer to deforestation, forest degradation, and forest disturbances as follows:

- Deforestation. Long-term or permanent conversion of forested land to non-forested land (FAO, 2001).
- Forest degradation. While there is not a widely accepted definition, it is generally considered a long-term process that does not lead to a land cover change but negatively affects the forest´s structure and function (Sasaki and Putz, 2009; Schoene et al., 2007).
- Forest disturbances. Drivers of forest´s state and function that can vary from high-impact events, such as fires or deforestation, to subtle and gradual processes, such as those caused by prolonged droughts, insects, or diseases (Cohen et al., 2017; McDowell et al., 2015).

Accordingly, we consider both deforestation (i.e., high-impact event) and forest degradation (i.e., subtle and gradual process) types of forest disturbances.

3. Algorithms

All three algorithms use surface reflectance values of Landsat Thematic Mapper (TM), Landsat Enhanced Thematic Mapper+ (ETM+), and Landsat Operational Land Imager (OLI).

3.1. CODED

CODED v0 uses time series of endmember fractions and NDFI to detect and characterize disturbances which can later be classified as degradation or deforestation. See Bullock et al. (2020) for algorithm details and Bullock (2020) for GEE implementation.

3.1.1. Disturbances map

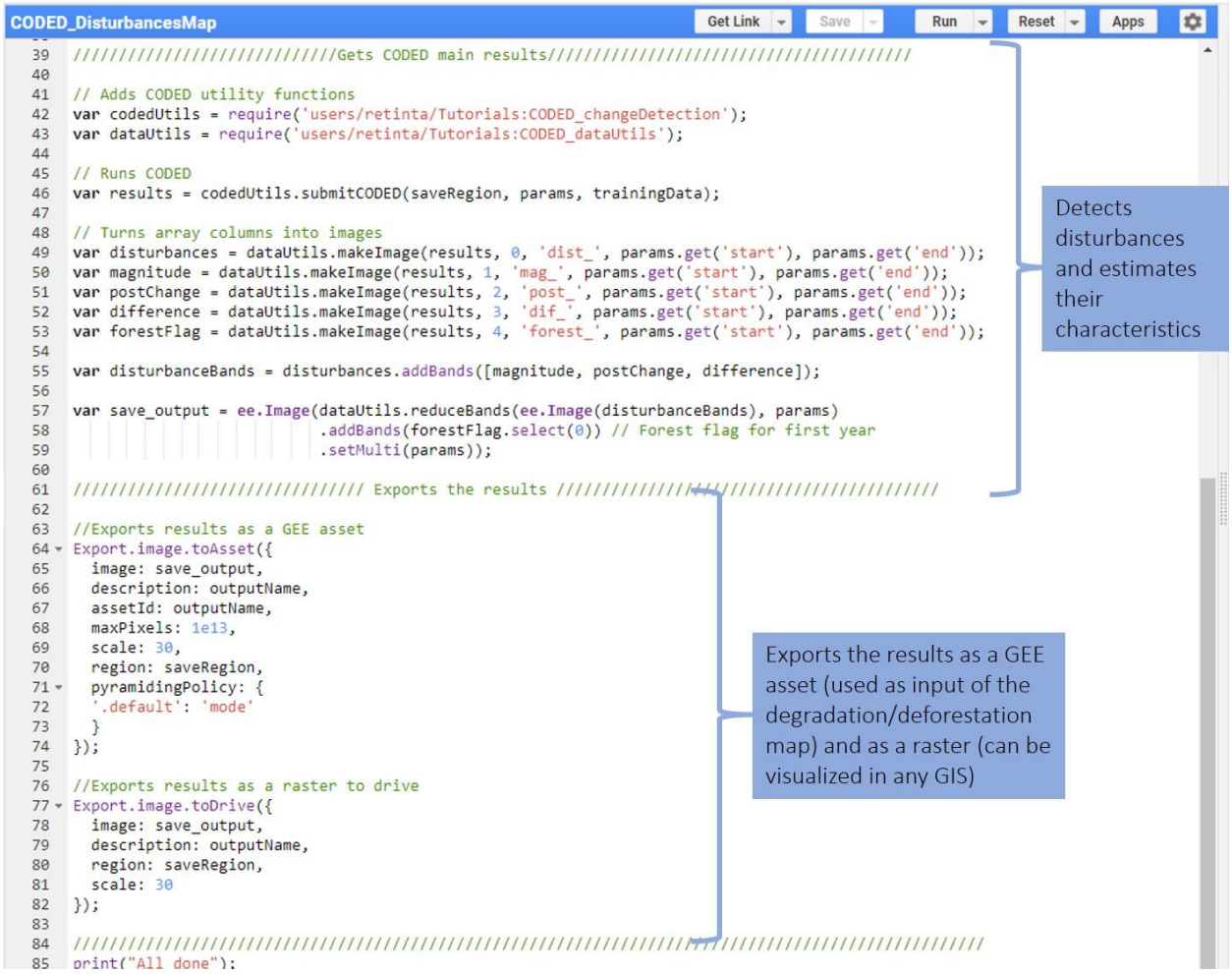
This section demonstrates how to run this code to produce a forest disturbance map.

Inputs

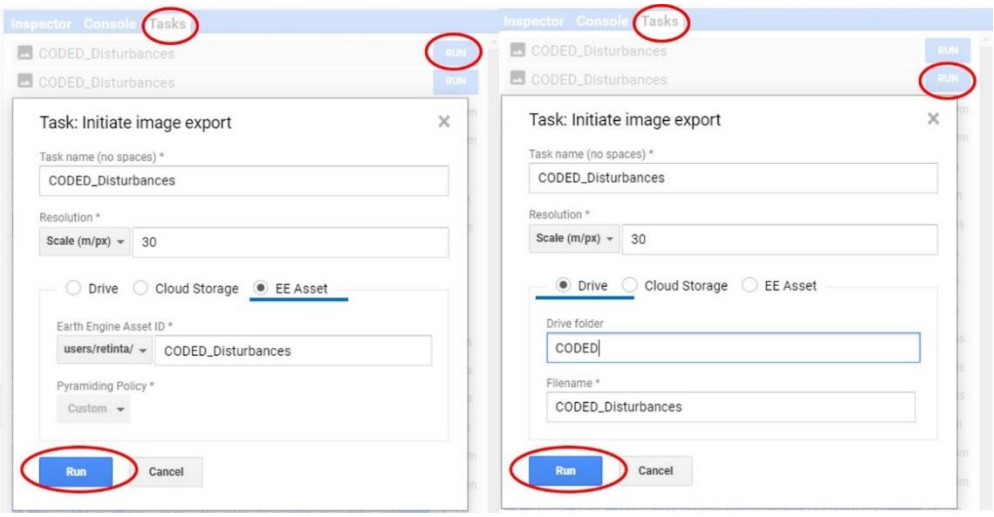
The user has to enter the following inputs:

Running the code

Execute the code by clicking run. Once it is all done, go to the task console and click run to export the results as a GEE asset and as a raster.




```
//////////////////////////////Gets CODED main results//////////////////////////////////////////

// Adds CODED utility functions
var codedUtils = require('users/retinta/Tutorials:CODED_changeDetection');
var dataUtils = require('users/retinta/Tutorials:CODED_dataUtils');

// Runs CODED
var results = codedUtils.submitCODED(saveRegion, params, trainingData);

// Turns array columns into images
var disturbances = dataUtils.makeImage(results, 0, 'dist_', params.get('start'), params.get('end'));
var magnitude = dataUtils.makeImage(results, 1, 'mag_', params.get('start'), params.get('end'));
var postChange = dataUtils.makeImage(results, 2, 'post_', params.get('start'), params.get('end'));
var difference = dataUtils.makeImage(results, 3, 'dif_', params.get('start'), params.get('end'));
var forestFlag = dataUtils.makeImage(results, 4, 'forest_', params.get('start'), params.get('end'));

var disturbanceBands = disturbances.addBands([magnitude, postChange, difference]);

var save_output = ee.Image(dataUtils.reduceBands(ee.Image(disturbanceBands), params)
                           .addBands(forestFlag.select(0)) // Forest flag for first year
                           .setMulti(params));

//////////////////////////////// Exports the results ////////////////////////////////////////

//Exports results as a GEE asset
Export.image.toAsset({
  image: save_output,
  description: outputName,
  assetId: outputName,
  maxPixels: 1e13,
  scale: 30,
  region: saveRegion,
  pyramidingPolicy: {
  '.default': 'mode'
  }
});

//Exports results as a raster to drive
Export.image.toDrive({
  image: save_output,
  description: outputName,
  region: saveRegion,
  scale: 30
});

////////////////////////////////////////////////////////////////////////////////////////////////
print("All done");
```



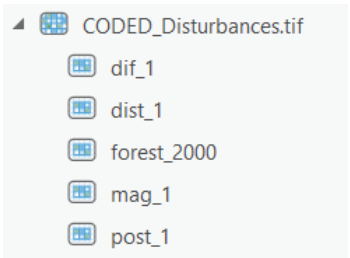
Output

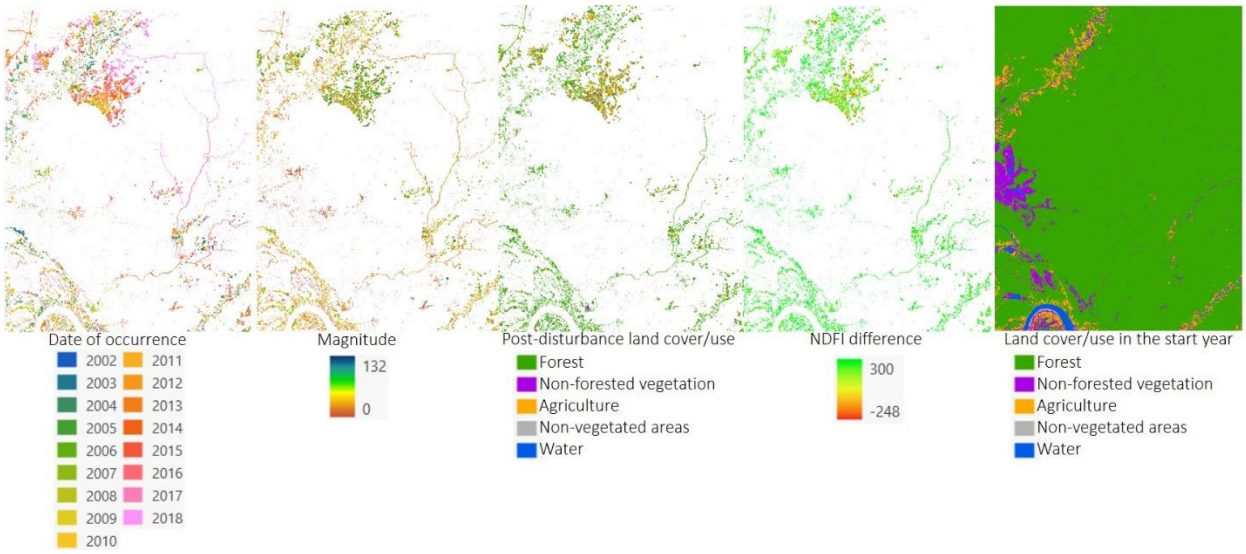
This code outputs a disturbance map with four layers per disturbance (i.e., date of occurrence, magnitude of change, post-disturbance land-cover, NDFI difference from before and after the disturbance) and one last layer indicating the land-cover type in the start year.

GEE asset

Index	Name	Type	Dimensions	CRS	Nominal Scale	Min	Max
0	dist_1	signed int16	981x1287 px	EPSG:4326	30.000000000000004	2002	2018
1	mag_1	signed int16	981x1287 px	EPSG:4326	30.000000000000004	3	132
2	post_1	signed int16	981x1287 px	EPSG:4326	30.000000000000004	0	5
3	dif_1	signed int16	981x1287 px	EPSG:4326	30.000000000000004	-248	300
4	forest_2000	signed int16	981x1287 px	EPSG:4326	30.000000000000004	1	5

Raster (".tif" file)



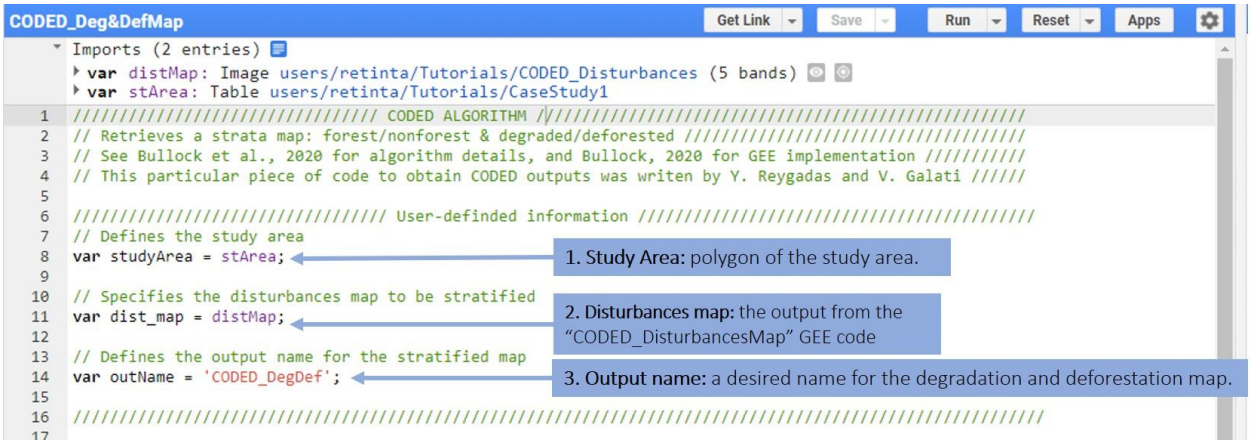


3.1.2. Degradation and deforestation map

This section demonstrates how to run this code to produce a map classified into forest, non-forest, degradation, and deforestation.

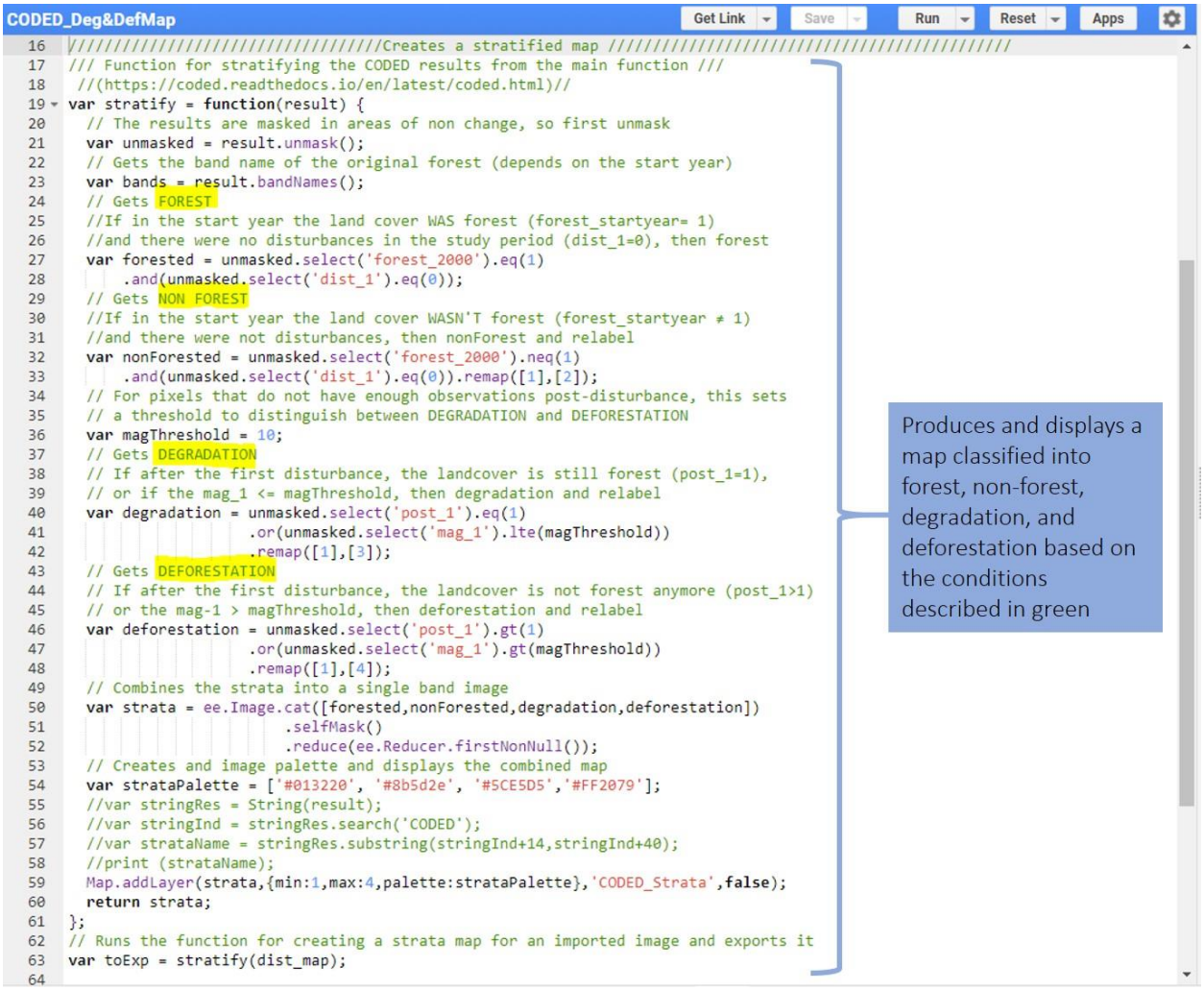
Inputs

There are three inputs the user has to enter:

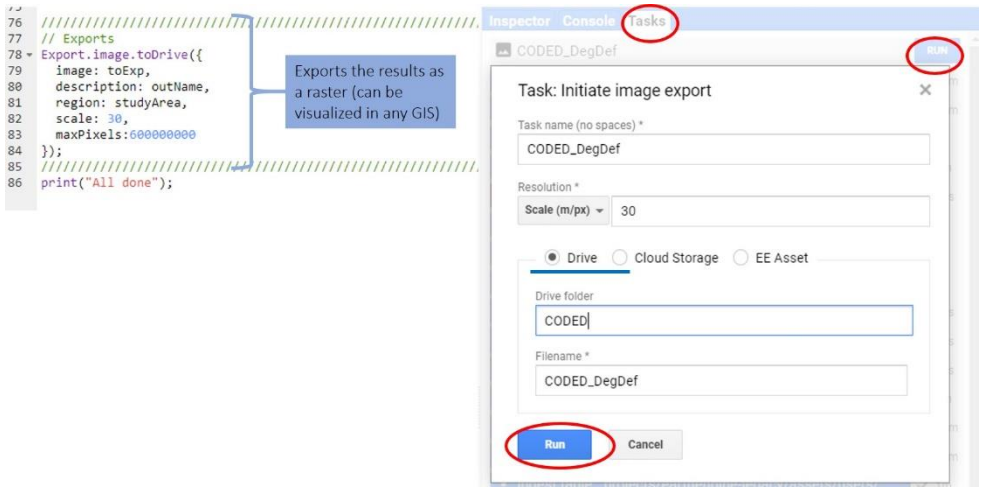


Running the code

Execute the code by clicking run. Once it is all done, go to the task console and click run to export the results as a raster.



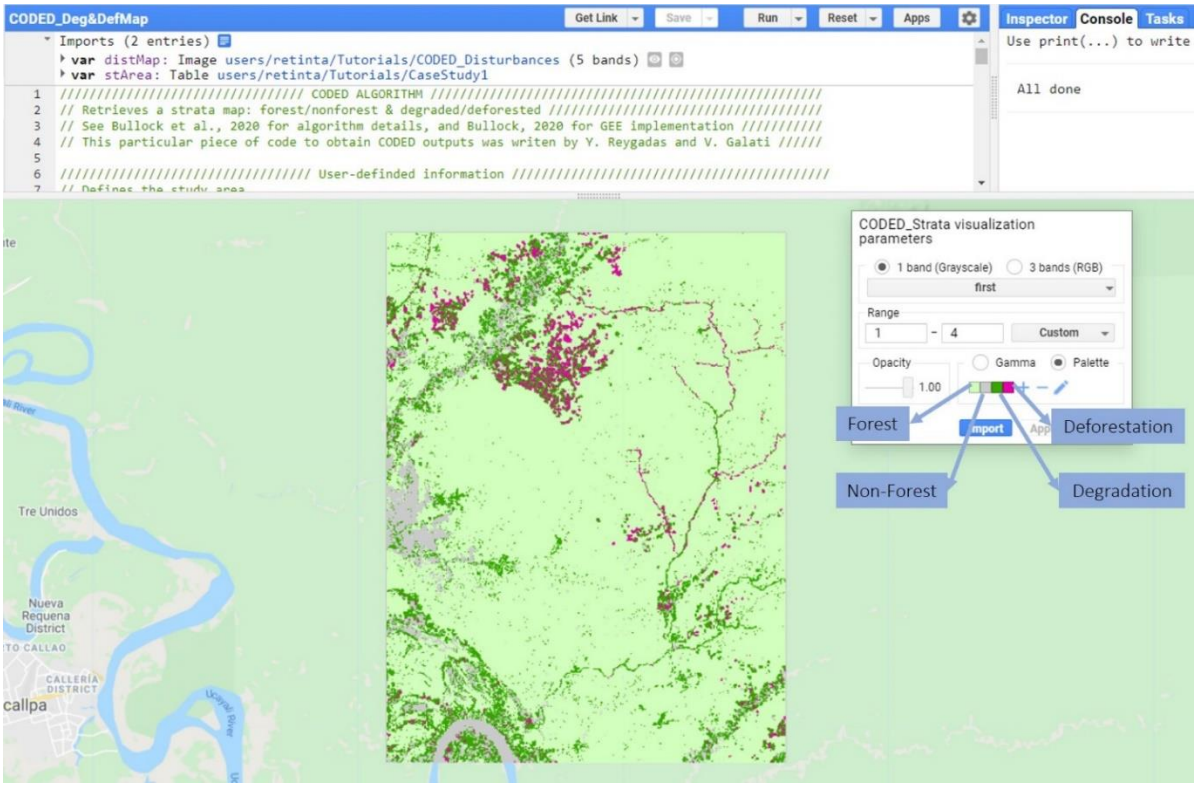

```
//////////////////////////////////Creates a stratified map ////////////////////////////////////////////////
/// Function for stratifying the CODED results from the main function ///
 //(https://coded.readthedocs.io/en/latest/coded.html)//
var stratify = function(result) {
  // The results are masked in areas of non change, so first unmask
  var unmasked = result.unmask();
  // Gets the band name of the original forest (depends on the start year)
  var bands = result.bandNames();
  // Gets FOREST
  //If in the start year the land cover WAS forest (forest_startyear= 1)
  //and there were no disturbances in the study period (dist_1=0), then forest
  var forested = unmasked.select('forest_2000').eq(1)
      .and(unmasked.select('dist_1').eq(0));
  // Gets NON FOREST
  //If in the start year the land cover WASN'T forest (forest_startyear ≠ 1)
  //and there were not disturbances, then nonForest and relabel
  var nonForested = unmasked.select('forest_2000').neq(1)
      .and(unmasked.select('dist_1').eq(0)).remap([1],[2]);
  // For pixels that do not have enough observations post-disturbance, this sets
  // a threshold to distinguish between DEGRADATION and DEFORESTATION
  var magThreshold = 10;
  // Gets DEGRADATION
  // If after the first disturbance, the landcover is still forest (post_1=1),
  // or if the mag_1 <= magThreshold, then degradation and relabel
  var degradation = unmasked.select('post_1').eq(1)
                    .or(unmasked.select('mag_1').lte(magThreshold))
                    .remap([1],[3]);
  // Gets DEFORESTATION
  // If after the first disturbance, the landcover is not forest anymore (post_1>1)
  // or the mag-1 > magThreshold, then deforestation and relabel
  var deforestation = unmasked.select('post_1').gt(1)
                    .or(unmasked.select('mag_1').gt(magThreshold))
                    .remap([1],[4]);
  // Combines the strata into a single band image
  var strata = ee.Image.cat([forested,nonForested,degradation,deforestation])
                        .selfMask()
                        .reduce(ee.Reducer.firstNonNull());
  // Creates and image palette and displays the combined map
  var strataPalette = ['#013220', '#8b5d2e', '#5CE5D5','#FF2079'];
  //var stringRes = String(result);
  //var stringInd = stringRes.search('CODED');
  //var strataName = stringRes.substring(stringInd+14,stringInd+40);
  //print (strataName);
  Map.addLayer(strata,{min:1,max:4,palette:strataPalette},'CODED_Strata',false);
  return strata;
};
// Runs the function for creating a strata map for an imported image and exports it
var toExp = stratify(dist_map);
```



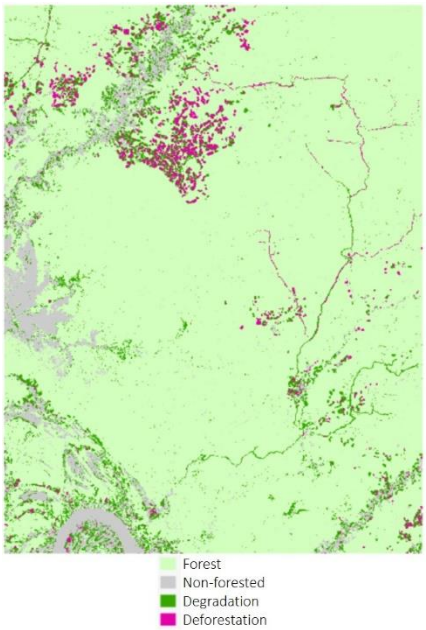
Output

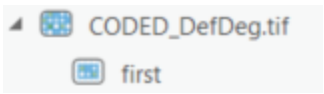
This code outputs a map classified into forest, non-forest, degradation, and deforestation.

Map displayed in GEE



Raster (".tif" file)





3.2. LandTrendr

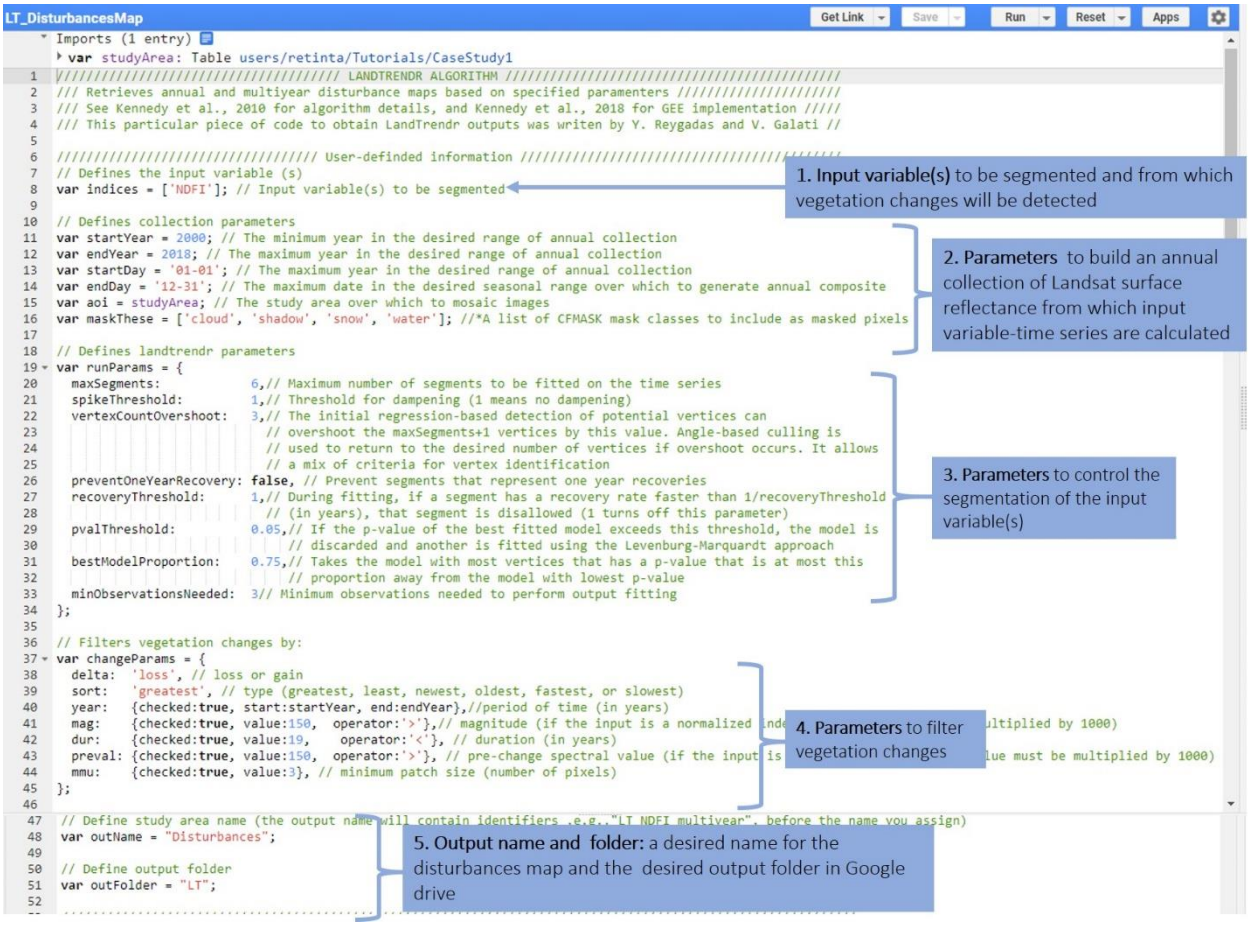
LandTrendr detects and characterizes vegetation loss or gain by segmenting and fitting temporal trajectories of a user-defined variable (possible input variables: NDFI*added, NBR, NDVI, NDSI, NDMI, TCB, TCG, TCW, TCA, and Landsat TM-equivalent bands 1-5 and 7). See Kennedy et al. (2010) for algorithm details and Kennedy et al. (2018) for GEE implementation.

3.2.1. Disturbances map

This section demonstrates how to run this code to produce a forest disturbance map.

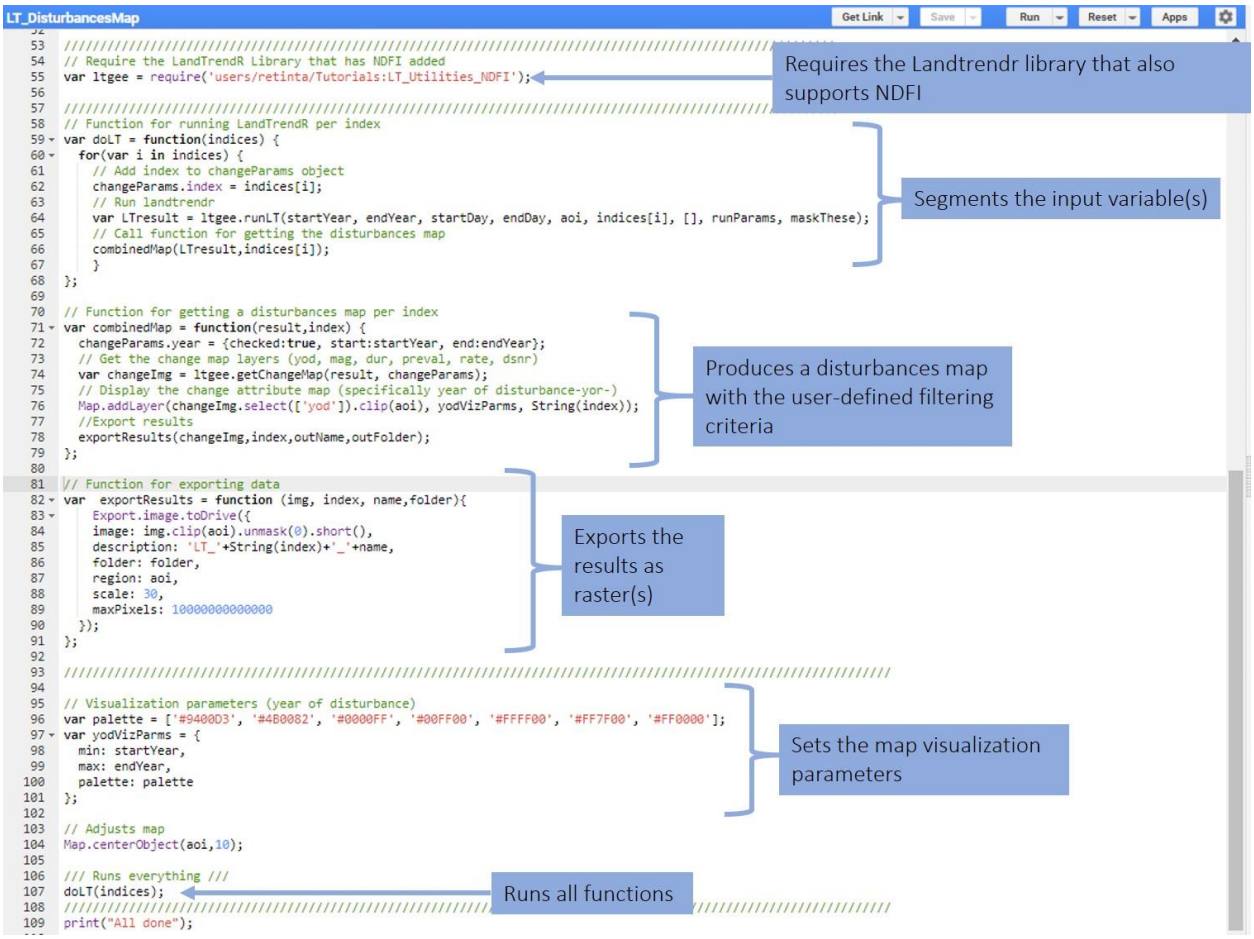
Inputs

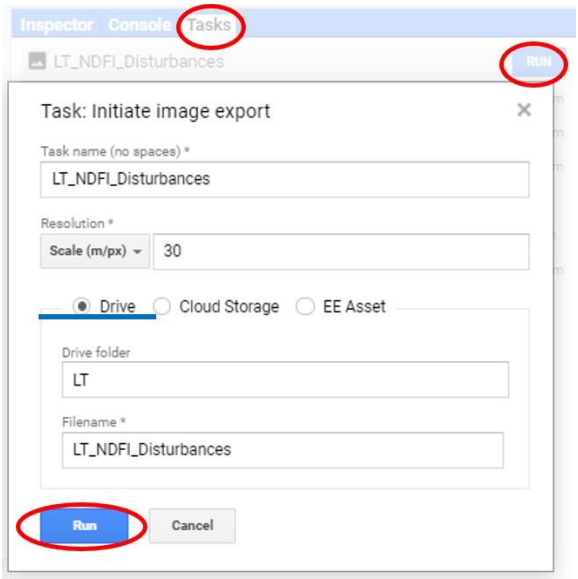
The user has to enter the following inputs:



Running the code

Execute the code by clicking run. Once it is all done, go to the task console and click run to export the results as a raster.

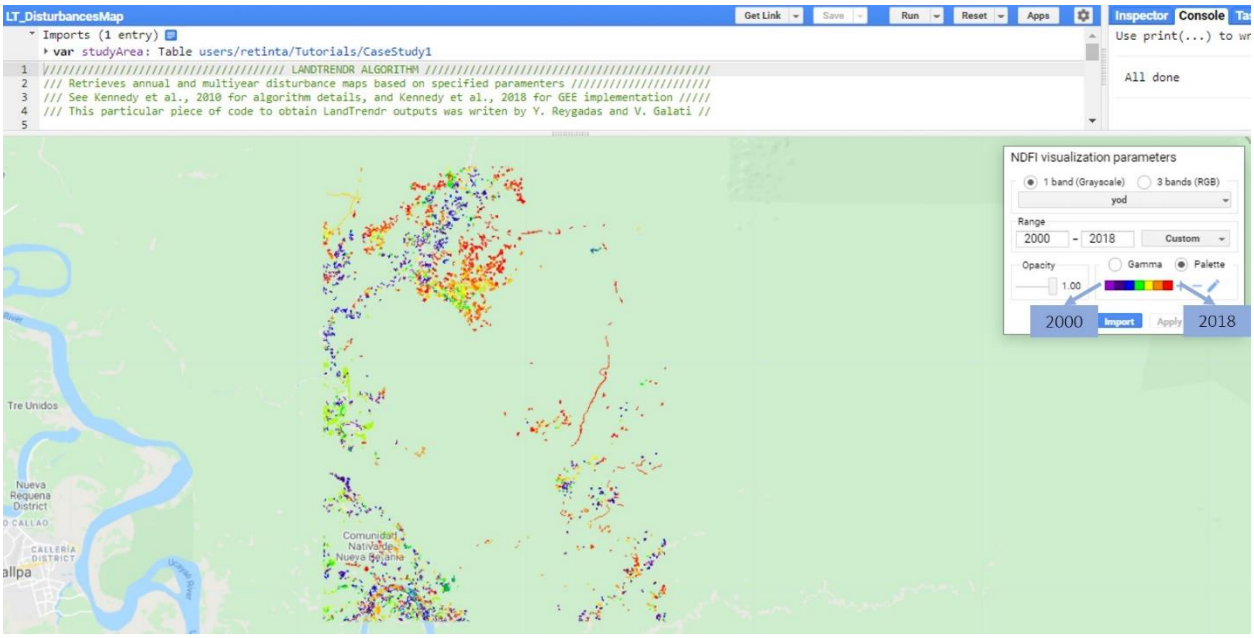




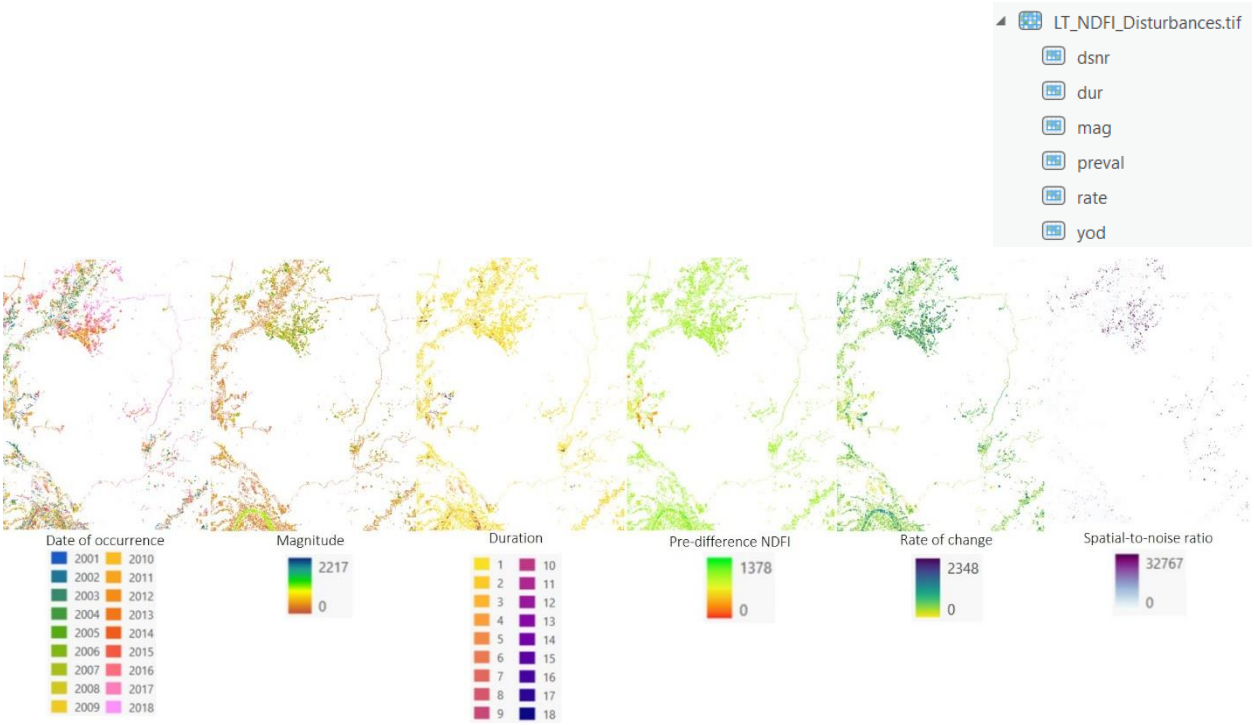
Output

This code outputs a disturbance map with six layers: date of occurrence, magnitude of change, duration, pre-change spectral value, rate of change, and signal-to-noise ratio.

Map displayed in GEE (only "year of disturbance" layer)



Raster (".tif" file)



3.3. MTDD

This MTDD-based algorithm classifies initially forested areas into intact, degraded, and deforested by training a random forest model with sixty-six metrics derived from six annual time series (i.e., NDVI, two SWIR spectral regions, two NDWI indices, and SAVI) from which eleven descriptive statistics (i.e., minimum, maximum, range, mean, standard deviation, coefficient of variation, kurtosis, skewness, slope, maximum 5-year slope, and most recent value) are calculated. We built this MTDD GEE code based on Wang et al. (2019) and adapted it to the SWA.

3.3.1. Training samples

This section demonstrates how to run this code to produce samples which are later used to train a random forest classifier.

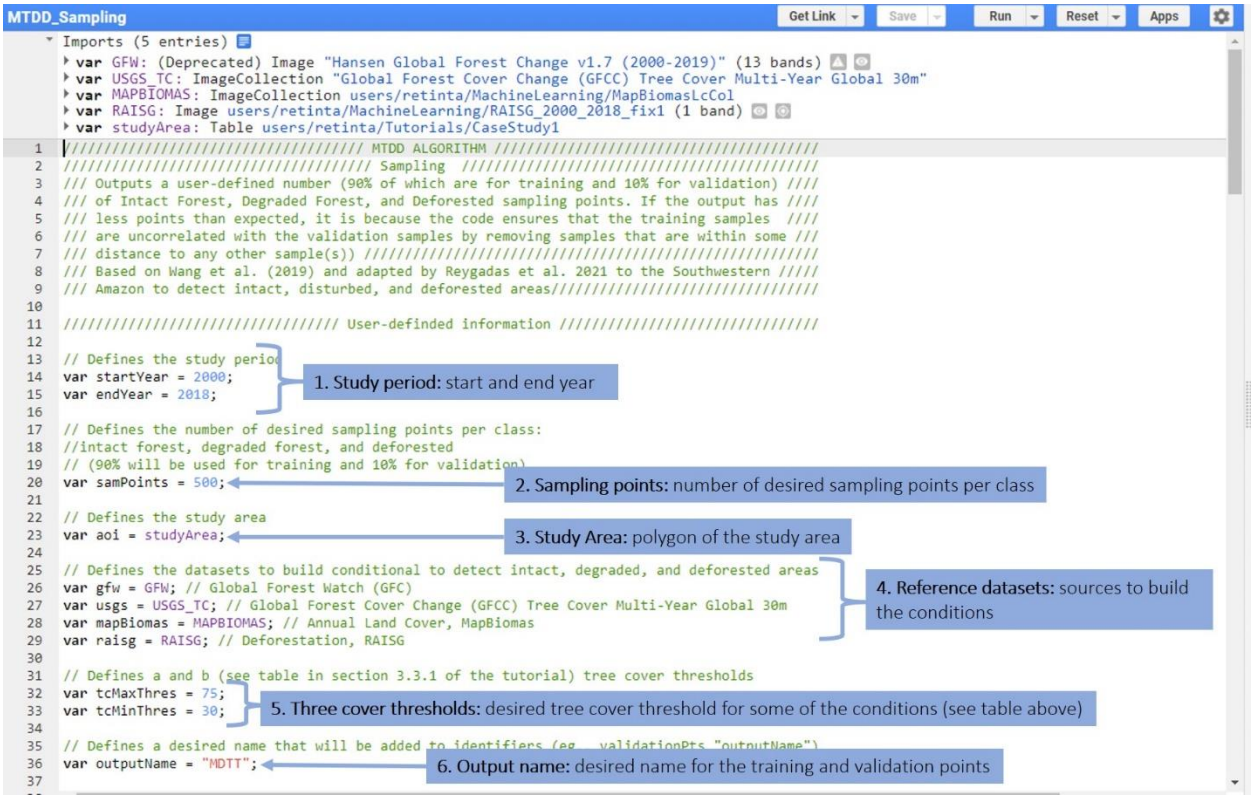
The samples are randomly selected from the following set of conditions:

Class	Condition	Reference dataset
Intact	Forest cover in the entire study period	MapBiomas (MapBiomas, 2020)
	No deforested in the entire study period	RAISG (RAISG, 2020)
	No forest loss in the entire study period	GFC (Hansen et al., 2013)
	Tree cover greater than 75%[a] in 2000	GFC (Hansen et al., 2013)
	Tree cover greater than 75%[a] in 2015	GFCC (Sexton et al., 2013)
Degraded	Forest cover in the end-year	MapBiomas (MapBiomas, 2020)
	Deforested at some point in the study period	RAISG (RAISG, 2020)
	Tree cover greater than 30%[b] and lower than 75%[a] in 2015	GFCC (Sexton et al., 2013)
Deforested	No forest cover in the end-year	MapBiomas (MapBiomas, 2020)
	Deforested at some point in the study period	RAISG (RAISG, 2020)
	Forest loss at some point in the study period	GFC (Hansen et al., 2013)

[a] and [b] tree cover thresholds can be modified in the GEE code

Inputs

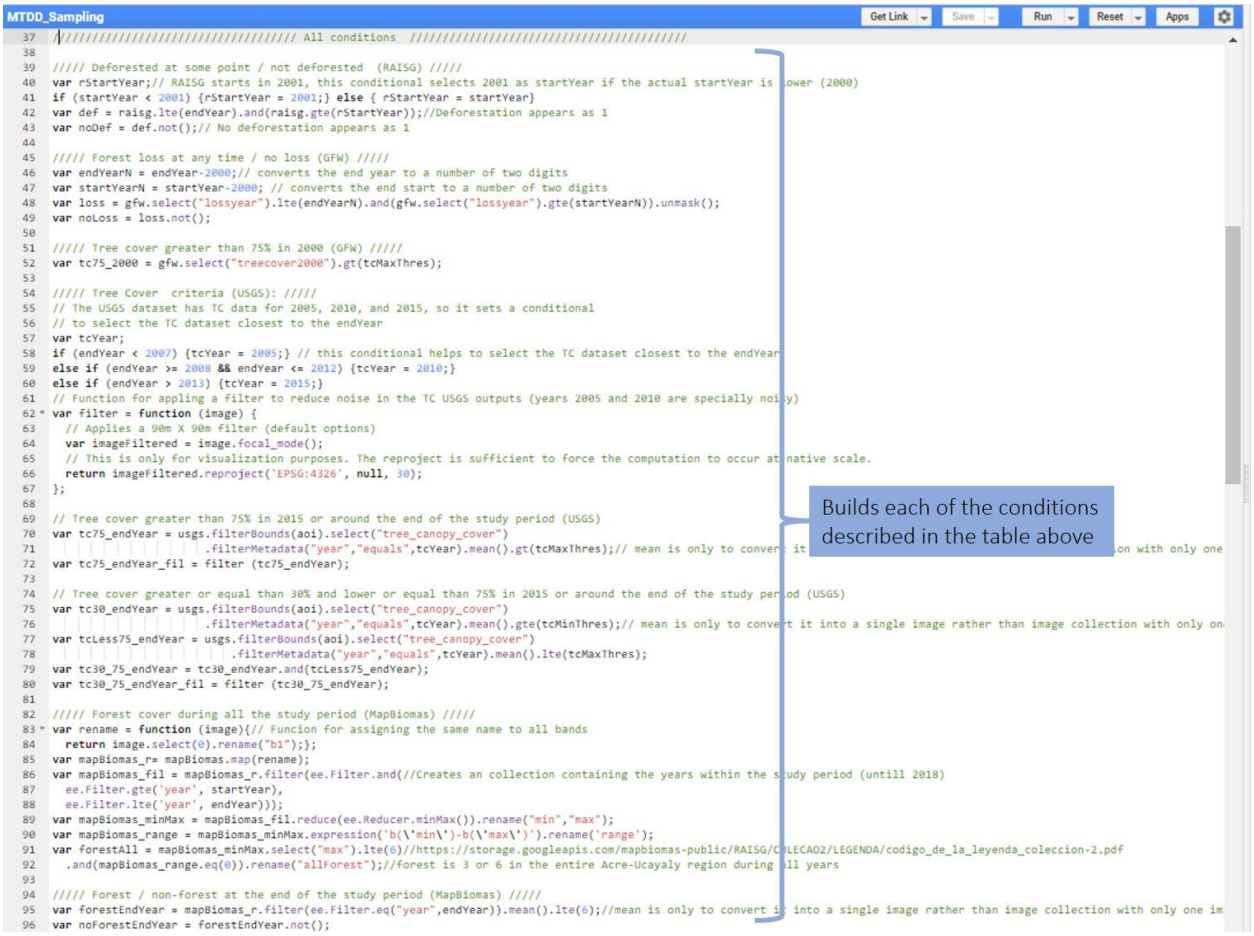
The user has to enter the following inputs:

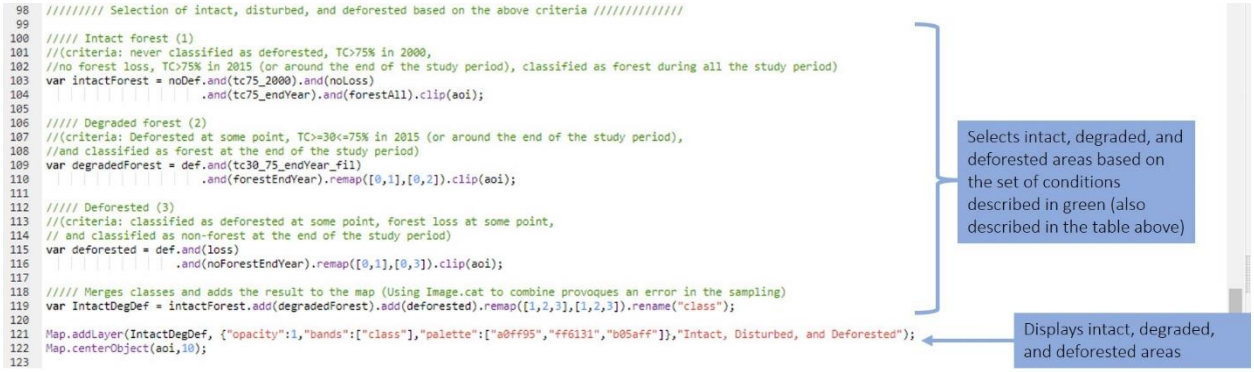


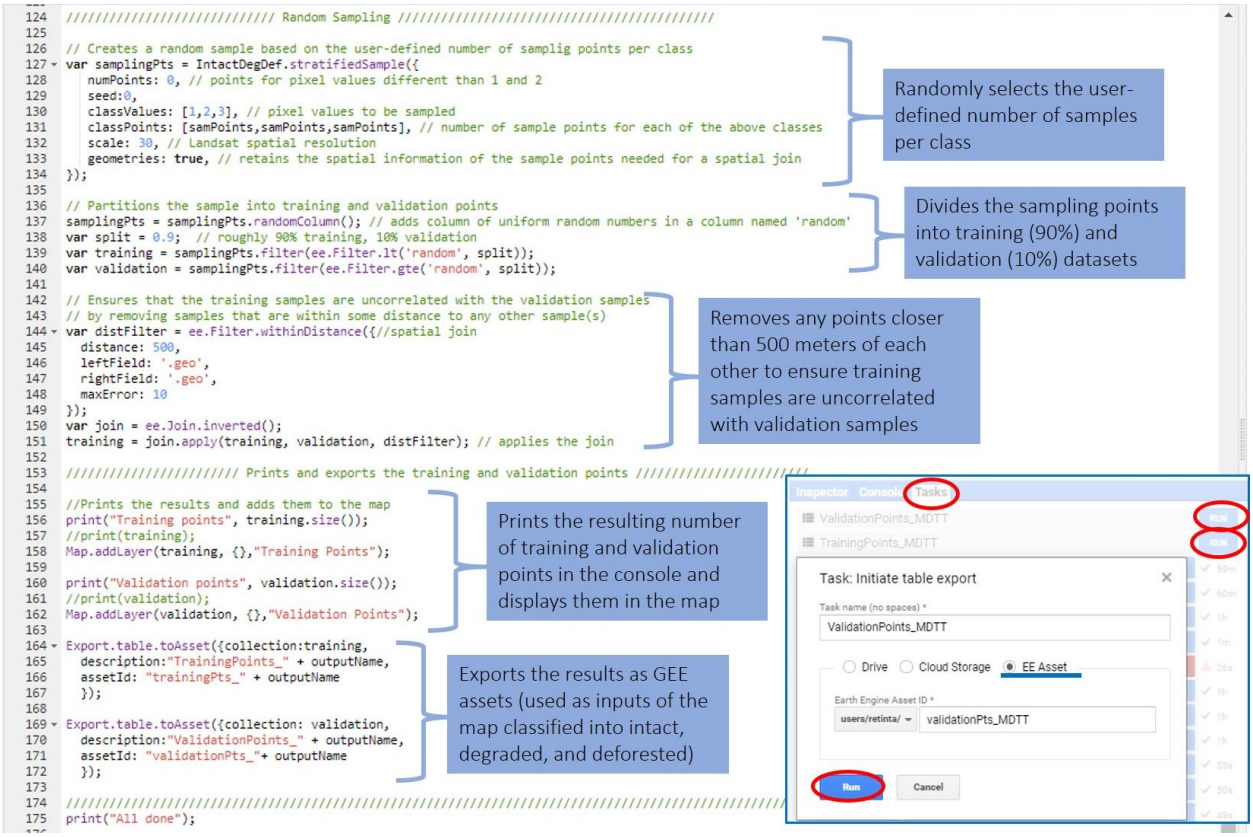
Note: Some of the datasets used to build the conditions date back to 2018. Therefore, the study period has to fall between 2000 and 2018.

Running the code

Execute the code by clicking run. Once it is all done, go to the task console and click run to export the results as a GEE asset.



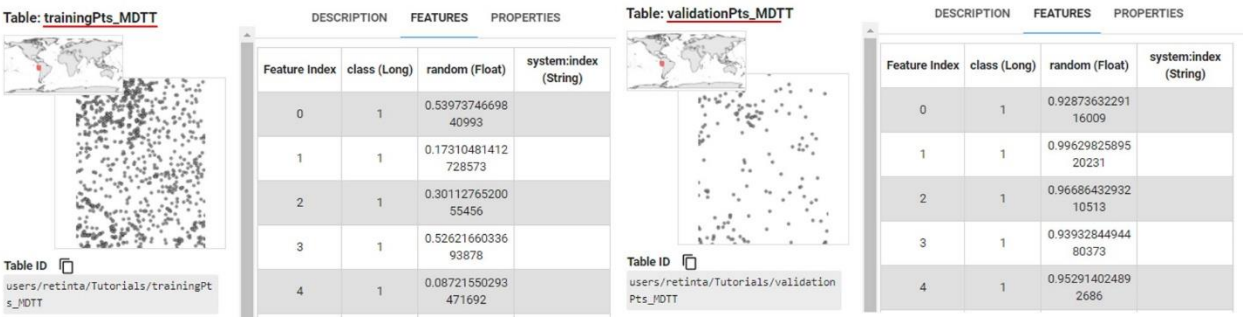




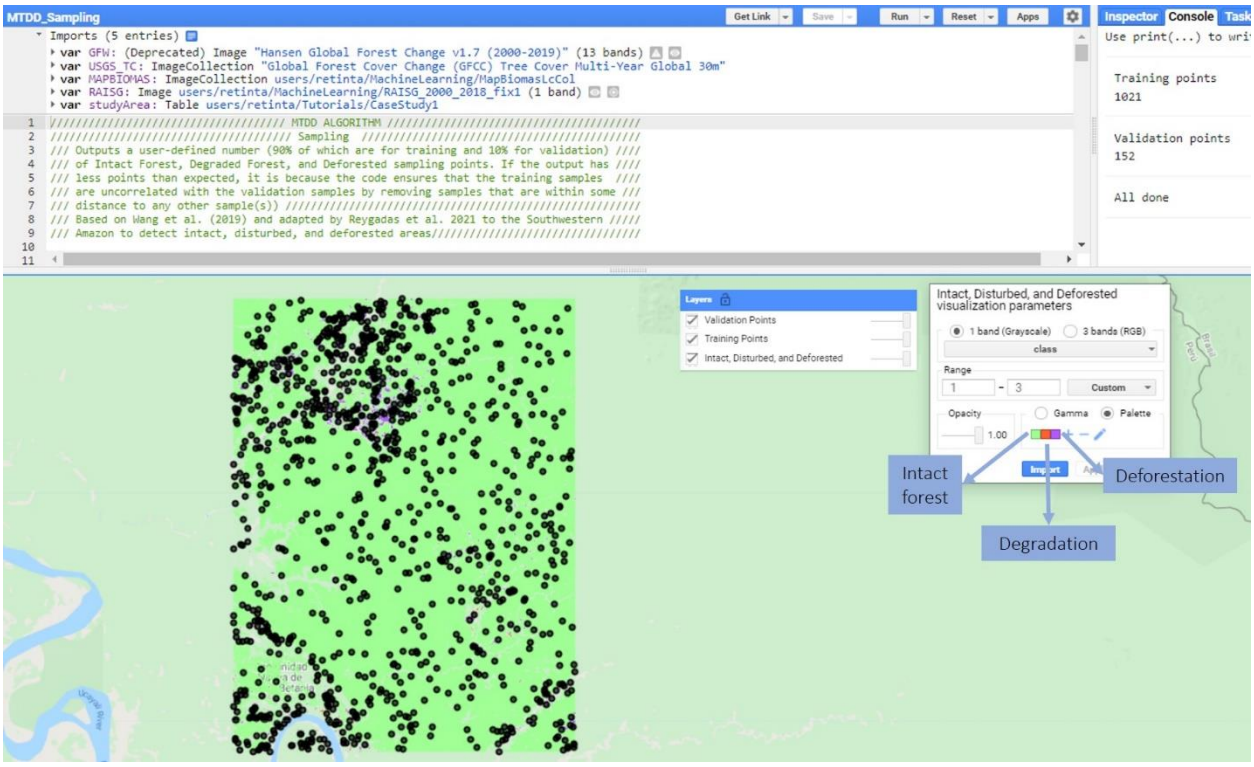
Output

This code outputs two datasets: training points and validation points.

GEE assets



Map displayed in GEE

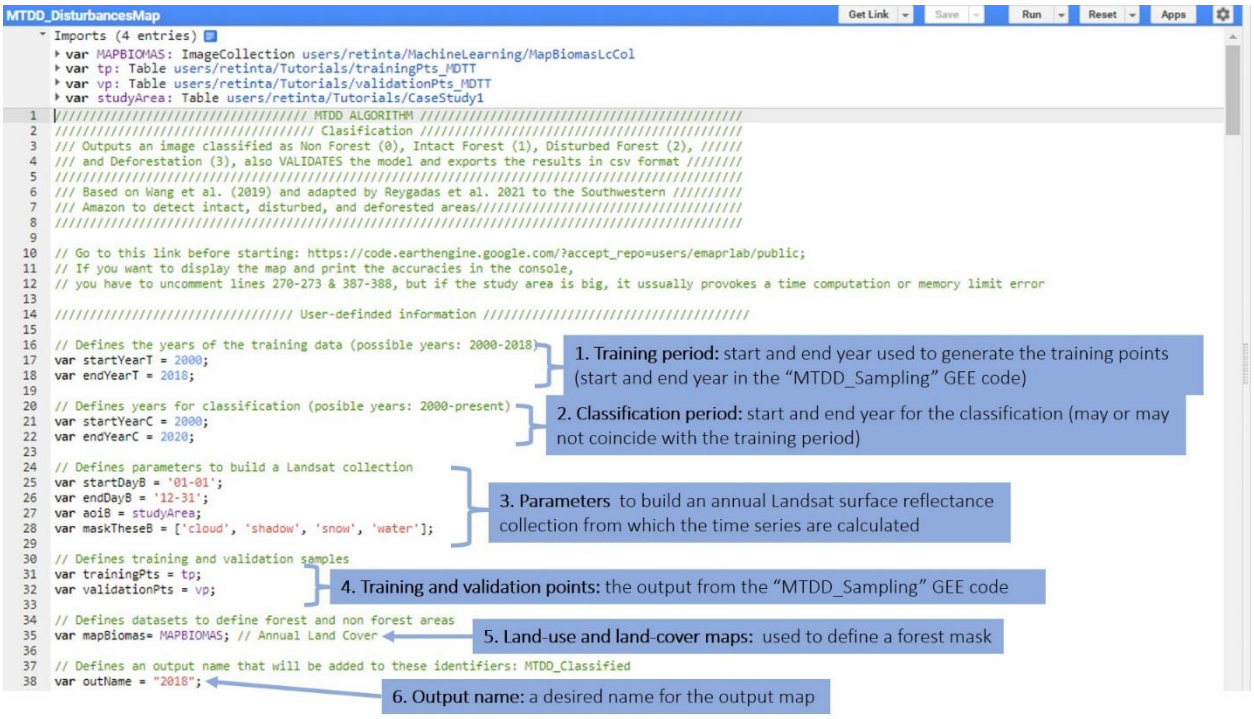


3.3.2. Degradation and deforestation map

This section demonstrates how to run this code to produce a map classified into intact forest, degradation, and deforestation.

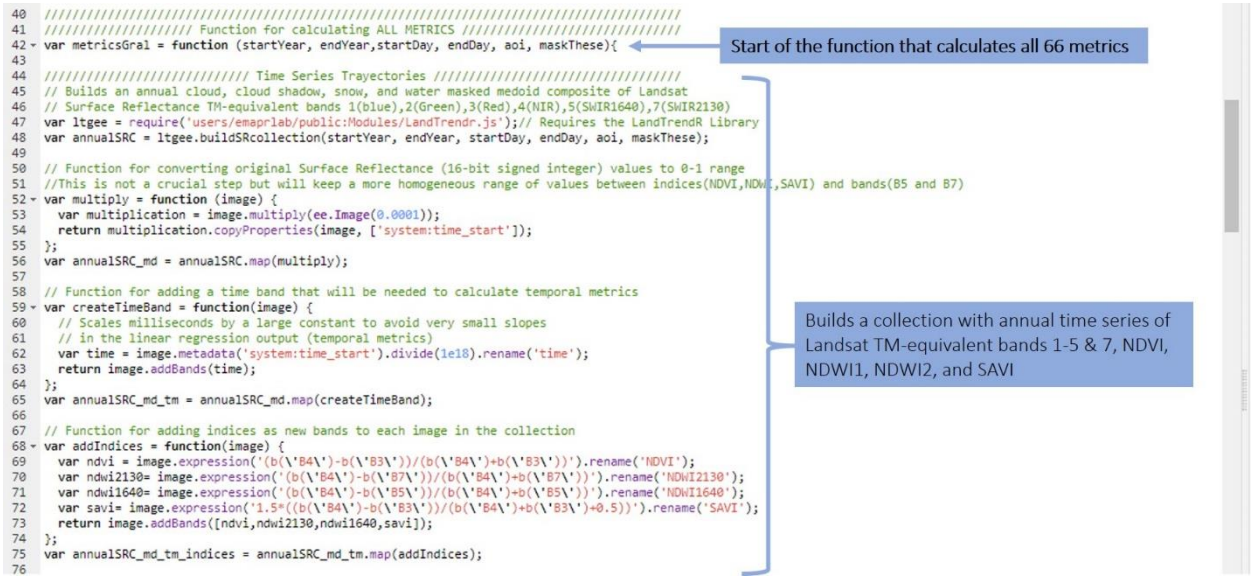
Inputs

The user has to enter the following inputs:



Running the code

Execute the code by clicking run. Once it is all done, go to the task console and click run to export the results as a raster (map classified) and csv files (overall accuracy and error matrix).




```
40  ////////////////////////////////////////////////////////////////////////////////
41  ////////////////////// Function for calculating ALL METRICS ////////////////////////////////
42  var metricsGral = function (startYear, endYear,startDay, endDay, aoi, maskThese){
43
44  ////////////////////////////// Time Series Trayectories ////////////////////////////////
45  // Builds an annual cloud, cloud shadow, snow, and water masked medoid composite of Landsat
46  // Surface Reflectance TM-equivalent bands 1(blue),2(Green),3(Red),4(NIR),5(SWIR1640),7(SWIR2130)
47  var ltgee = require('users/emaprlab/public:Modules/LandTrendr.js');// Requires the LandTrendR Library
48  var annualSRC = ltgee.buildSRcollection(startYear, endYear, startDay, endDay, aoi, maskThese);
49
50  // Function for converting original Surface Reflectance (16-bit signed integer) values to 0-1 range
51  //This is not a crucial step but will keep a more homogeneous range of values between indices(NDVI,NDWI,SAVI) and bands(B5 and B7)
52  var multiply = function (image) {
53    var multiplication = image.multiply(ee.Image(0.0001));
54    return multiplication.copyProperties(image, ['system:time_start']);
55  };
56  var annualSRC_md = annualSRC.map(multiply);
57
58  // Function for adding a time band that will be needed to calculate temporal metrics
59  var createTimeBand = function(image) {
60    // Scales milliseconds by a large constant to avoid very small slopes
61    // in the linear regression output (temporal metrics)
62    var time = image.metadata('system:time_start').divide(1e18).rename('time');
63    return image.addBands(time);
64  };
65  var annualSRC_md_tm = annualSRC_md.map(createTimeBand);
66
67  // Function for adding indices as new bands to each image in the collection
68  var addIndices = function(image) {
69    var ndvi = image.expression('(b(\'B4\')-b(\'B3\'))/(b(\'B4\')+b(\'B3\'))').rename('NDVI');
70    var ndwi2130= image.expression('(b(\'B4\')-b(\'B7\'))/(b(\'B4\')+b(\'B7\'))').rename('NDWI2130');
71    var ndwi1640= image.expression('(b(\'B4\')-b(\'B5\'))/(b(\'B4\')+b(\'B5\'))').rename('NDWI1640');
72    var savi= image.expression('1.5*((b(\'B4\')-b(\'B3\'))/(b(\'B4\')+b(\'B3\')+0.5))').rename('SAVI');
73    return image.addBands([ndvi,ndwi2130,ndwi1640,savi]);
74  };
75  var annualSRC_md_tm_indices = annualSRC_md_tm.map(addIndices);
76
```

```
////////////////////////////// Descriptive statistics /////////////////////////////////////////

///// Location metrics: min, max, range, mean /////
var min = annualSRC_md_tm_indices.select(['B5','B7','NDVI','NDWI2130','NDWI1640','SAVI'])
    .reduce(ee.Reducer.min());
var max = annualSRC_md_tm_indices.select(['B5','B7','NDVI','NDWI2130','NDWI1640','SAVI'])
    .reduce(ee.Reducer.max());
var range = max.subtract(min).rename(['B5_range','B7_range','NDVI_range','NDWI2130_range','NDWI1640_range','SAVI_range']);
var mean = annualSRC_md_tm_indices.select(['B5','B7','NDVI','NDWI2130','NDWI1640','SAVI'])
    .reduce(ee.Reducer.mean());

///// Scale metrics: stdDev, C.V., kurtosis, skewness /////
var stdDev = annualSRC_md_tm_indices.select(['B5','B7','NDVI','NDWI2130','NDWI1640','SAVI'])
    .reduce(ee.Reducer.stdDev());
var cv = mean.divide(stdDev).rename(['B5_cv','B7_cv','NDVI_cv','NDWI2130_cv','NDWI1640_cv','SAVI_cv']);
var kurt = annualSRC_md_tm_indices.select(['B5','B7','NDVI','NDWI2130','NDWI1640','SAVI'])
    .reduce(ee.Reducer.kurtosis());
var skew = annualSRC_md_tm_indices.select(['B5','B7','NDVI','NDWI2130','NDWI1640','SAVI'])
    .reduce(ee.Reducer.skew());

///// Temporal metrics /////
/// Slope (outpus two bands: 'offset' (y-intercept) and 'scale' (slope))
var slopeB5 = annualSRC_md_tm_indices.select(['time','B5'])
    .reduce(ee.Reducer.linearFit()).select('scale').rename('B5_slp');
var slopeB7 = annualSRC_md_tm_indices.select(['time','B7'])
    .reduce(ee.Reducer.linearFit()).select('scale').rename('B7_slp');
var slopeNDVI = annualSRC_md_tm_indices.select(['time','NDVI'])
    .reduce(ee.Reducer.linearFit()).select('scale').rename('NDVI_slp');
var slopeNDWI2130 = annualSRC_md_tm_indices.select(['time','NDWI2130'])
    .reduce(ee.Reducer.linearFit()).select('scale').rename('NDWI2130_slp');
var slopeNDWI1640 = annualSRC_md_tm_indices.select(['time','NDWI1640'])
    .reduce(ee.Reducer.linearFit()).select('scale').rename('NDWI1640_slp');
var slopeSAVI = annualSRC_md_tm_indices.select(['time','SAVI'])
    .reduce(ee.Reducer.linearFit()).select('scale').rename('SAVI_slp');

/// Max-slope (maximum absolute linear regression slope of 5-year windows)
// Creates a moving 5-year window
var join = ee.Join.saveAll({//Returns a join that pairs each element from the annual SR collection
    matchesKey: 'images'    //with a group of matching elements from the 5-year window collection.
});                         //The list of matches is added to each result as an additional property
var diffFilter = ee.Filter.maxDifference({ // Establishes the 5-year filter
  difference: 78892380000, // 2.5 years in milliseconds, the filter selects 2 years before and after the target year
  leftField: 'system:time_start', // therefore, the extreme windows are made of only 3-4 years
  rightField: 'system:time_start'
});
var fiveWindowJoin = join.apply({// Collection in which each image is associated with the 5-year window images (see properties:
  primary: annualSRC_md_tm_indices,
  secondary: annualSRC_md_tm_indices,
  condition: diffFilter
});
// Function for calculating the absolute maximum slope over 5-year moving windows
var maxAbsMovSlope = function (invar, outvar){
  // Calculates slopes over 5-year windows
  var mvSlp = ee.ImageCollection(fiveWindowJoin.map(function(image) {
    var annualSRC_md_tm_indices = ee.ImageCollection.fromImages(image.get('images'));
    return ee.Image(image).addBands(annualSRC_md_tm_indices.select(["time",invar])
    .reduce(ee.Reducer.linearFit()));
  }));
  // Gets rid of the extreme windows, which are made of only 3 or 4 years
  var range1 = mvSlp.reduceColumns(ee.Reducer.minMax(), ["system:time_start"]); //Gets the date range of images in the collecti
  var filter1 = mvSlp.filter(ee.Filter.date(range1.get('min')).not());
  var filter2 = filter1.filter(ee.Filter.date(range1.get('max')).not());
  var range2 = filter2.reduceColumns(ee.Reducer.minMax(), ["system:time_start"]);
  var filter3 = filter2.filter(ee.Filter.date(range2.get('min')).not());
  var filter4 = filter3.filter(ee.Filter.date(range2.get('max')).not());// Print this to see all 5-yer window slopes
  // Converts slope values to absolute numbers
  var abs = function (image) {return image.select('scale').abs()};
  // Extracts the maximum absolute slope
  return filter4.map(abs).reduce(ee.Reducer.max()).rename(outvar);
};
// Applies the absolute moving slope function to all variables
var mxSlpB5 = maxAbsMovSlope('B5', 'B5_mxSlp');
var mxSlpB7 = maxAbsMovSlope('B7','B7_mxSlp');
var mxSlpNDVI = maxAbsMovSlope('NDVI','NDVI_mxSlp');
var mxSlpNDWI2130 = maxAbsMovSlope('NDWI2130','NDWI2130_mxSlp');
var mxSlpNDWI1640 = maxAbsMovSlope('NDWI1640','NDWI1640_mxSlp');
var mxSlpSAVI = maxAbsMovSlope('SAVI','SAVI_mxSlp');

//Value at last year
var rangeDates = annualSRC_md_tm_indices.reduceColumns(ee.Reducer.minMax(), ["system:time_start"]);
var lastYearFil = annualSRC_md_tm_indices.filter(ee.Filter.date(rangeDates.get('max')));
var lastYear = lastYearFil.select(['B5','B7','NDVI','NDWI2130','NDWI1640','SAVI']).toBands();
var lastYearVal= lastYear.rename(['B5_last','B7_last','NDVI_last','NDWI2130_last','NDWI1640_last','SAVI_last']);

///// All metrics /////
// Creates a collection from all metrics
var metricsCollection = ee.ImageCollection([min,max,range,mean,stdDev,cv,kurt,skew,
  slopeB5,slopeB7,slopeNDVI,slopeNDWI2130,slopeNDWI1640,slopeSAVI,
  mxSlpB5,mxSlpB7,mxSlpNDVI,mxSlpNDWI2130,mxSlpNDWI1640,mxSlpSAVI,lastYearVal]);

// Converts the metrics collection to a single multi-band image
var metrics= metricsCollection.toBands();
return metrics;
};
////////////////////////////////////////////////////////////////////////////////////////////
```

Calculates 11 descriptive statistics (i.e., minimum, maximum, range, mean, standard deviation, coefficient of variation, kurtosis, skewness slope, maximum 5-year slope, and most recent value) for each of the 6 time series (i.e., NDVI, two SWIR spectral regions, two NDWI indices, and SAVI)

End of the function that calculates all 66 metrics

```
////////////////////// Masking: delimitation of forested areas ////////////////////////////////

//// Selects areas that were forest at leat 3 consecutive years during the study period///
var rename = function (image){// Funcion for assigning the same name to all bands
  return image.select(0).rename("b1");};
var mapBiomas_r= mapBiomas.map(rename);
var mapBiomas_fil = mapBiomas_r.filter(ee.Filter.and(//Creates an collection containing the years within the study period (untill 2018)
  ee.Filter.gte('year', startYearT),
  ee.Filter.lte('year', endYearT)));
// Function for detecting pixels covered by forest
//https://storage.googleapis.com/mapbiomas-public/RAISG/COLECAO2/LEGENDA/codigo_de_la_leyenda_coleccion-2.pdf
//forest is 3 or 6 in the entire Acre-Ucayaly region during all years
var forest = function (image) {
  return image.eq(3).or(image.eq(6)).rename("forest").copyProperties(image, ['year'])};
var forestCol = mapBiomas.map(forest);

// Function for adding a time band that will be needed to calculate land covers per year
var createTimeBandMask = function(image) {
  var time = image.metadata('year').rename('time');
  return image.addBands(time);
};
var forestCol_tm = forestCol.map(createTimeBandMask);

/// Forest in three consecutive years
// Creates a moving 3-year window
var join1 = ee.Join.saveAll({matchesKey: 'images'});
var diffFilter1 = ee.Filter.maxDifference({difference: 1.5, leftField: 'year', rightField: 'year'});
var threeWindowJoin = join1.apply({primary: forestCol_tm, secondary: forestCol_tm, condition: diffFilter1});

// Function for detecting forest cover over 3-year moving windows
var forest3years = function (invar, outvar){
  // Calculates sum over 3-year windows
  var f3y = ee.ImageCollection(threeWindowJoin.map(function(image) {
    var forestCol_tm = ee.ImageCollection.fromImages(image.get('images'));
    return ee.Image(image).addBands(forestCol_tm .select(["time",invar])
    .reduce(ee.Reducer.sum()));
  }));
  // Extracts the maximum sum
  return f3y.reduce(ee.Reducer.max());
};

// Applies the "detecting forest cover over 3-year" function to forest presence per year
var MaxForest3years = forest3years('forest', 'forest_3years').select("forest_sum_max");
// Gets the forest mask (areas that were forest at least three consecutive years)
var forestMask= MaxForest3years.gte(3);

```

Creates a forest mask composed of all areas that were forest al least three consecutive years during the study period

```
//////////////////////////////// Classificador random forest ////////////////////////////////////////////

// *Crea una colección con todas las métricas con base en los años del muestreo
var metricsTraining = metricsGral(startYearT, endYearT,startDayB, endDayB, aoiB, maskTheseB);
print("Métricas muestreo", metricsTraining);

// **Crea una colleción con todas las métricas con base en los años para la clasificación
var metricsClassification = metricsGral(startYearC, endYearC,startDayB, endDayB, aoiB, maskTheseB);
print("Métricas clasificación", metricsClassification);

// Sobrepone los puntos de entrenamiento con las métricas*
var collectMetrics = function (feature){// usar una funcion permite a GEE trabajar con más puntos antes de exceder su capacidad
    return metricsTraining.sampleRegions({collection:feature,
    properties: ["class"],
    scale: 30,
    tileScale: 2});
};
var trainingCol = trainingPts.map(collectMetrics);
var trainingPoints = trainingCol.flatten();

// Crea un classificador random forest con 500 árboles y lo entrena
var classifier = ee.Classifier.smileRandomForest({numberOfTrees: 50})
  .train({features: trainingPoints, classProperty:"class",
  inputProperties:['0_B5_min','0_B7_min','0_NDVI_min','0_NDWI2130_min','0_NDWI1640_min','0_SAVI_min',
    '1_B5_max','1_B7_max','1_NDVI_max','1_NDWI2130_max','1_NDWI1640_max','1_SAVI_max',
    '2_B5_range','2_B7_range','2_NDVI_range','2_NDWI2130_range','2_NDWI1640_range','2_SAVI_range',
    '3_B5_mean','3_B7_mean','3_NDVI_mean','3_NDWI2130_mean','3_NDWI1640_mean','3_SAVI_mean',
    '4_B5_stdDev','4_B7_stdDev','4_NDVI_stdDev','4_NDWI2130_stdDev','4_NDWI1640_stdDev','4_SAVI_stdDev',
    '5_B5_cv','5_B7_cv','5_NDVI_cv','5_NDWI2130_cv','5_NDWI1640_cv','5_SAVI_cv',
    '6_B5_kurtosis','6_B7_kurtosis','6_NDVI_kurtosis','6_NDWI2130_kurtosis','6_NDWI1640_kurtosis','6_SAVI_kurtosis',
    '7_B5_skew','7_B7_skew','7_NDVI_skew','7_NDWI2130_skew','7_NDWI1640_skew','7_SAVI_skew',
    '8_B5_slp','9_B7_slp','10_NDVI_slp','11_NDWI2130_slp','12_NDWI1640_slp','13_SAVI_slp',
    '14_B5_mxSlp','15_B7_mxSlp','16_NDVI_mxSlp','17_NDWI2130_mxSlp','18_NDWI1640_mxSlp','19_SAVI_mxSlp',
    '20_B5_last','20_B7_last','20_NDVI_last','20_NDWI2130_last','20_NDWI1640_last','20_SAVI_last']

  });

// Classifica las métricas deseadas**
var metricsMask = metricsClassification.mask(forestMask);// Enmascara las areas no forestales
var classified = metricsMask.classify(classifier);//No bosque (0), bosque intacto (1), bosque degradado (2), áreas deforestadas (3)
print("Mapa clasificado");

```

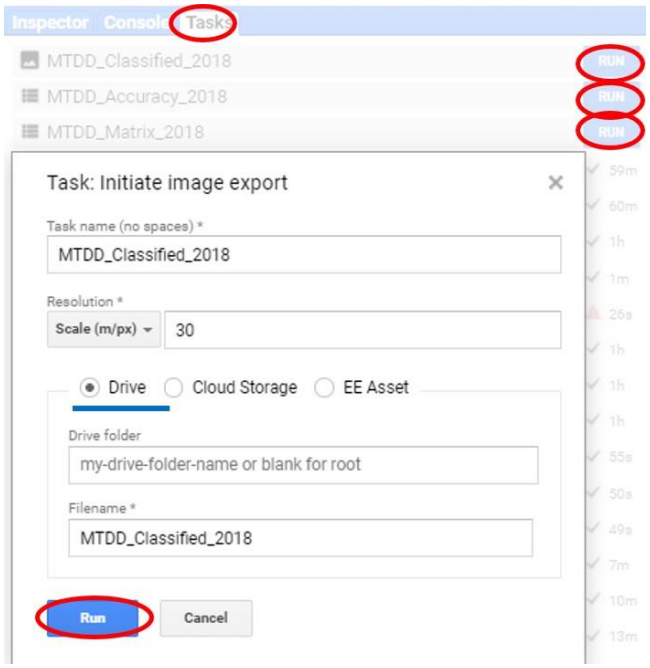
Entrena un clasificador *random forest* con las 66 métricas de los datos de entrenamiento y la clasifica las imágenes deseadas

```
//////////////////////////Post-classification filtering to reduce noise//////////////////////////////////

// Applies a 90m X 90m filter (default options)
var classified_fil = classified.focal_mode().mask(forestMask);
// This is only for visualization purposes. The reproject is sufficient
// to force the computation to occur at native scale.
var classified_fil_vis = classified_fil.reproject('EPSG:4326', null, 30);
// Displays the results
//Map.centerObject(aoi,10);
//Map.addLayer(classified_fil_vis.clip(aoiC),
            //  {"opacity":1,"bands":["classification"],"min":0,"max":3,"palette":["ffffff","7dff5c","0a9b15","ff5514"]},
            //  "Classification");

//////////////////////////////////////// Validation ////////////////////////////////////////////////////

// Overlays the validation points over the metrics
var validationCol = validationPts.map(collectMetrics);
var validationPoints = validationCol.flatten();
//print ("Metrics collected in validation points",validationPoints);

// Classify the validation data
var validated = validationPoints.classify(classifier);

// Gets a confusion matrix representing expected accuracy
var valAccuracy = validated.errorMatrix("class","classification");
//print('Validation error matrix: ', valAccuracy);
//print('Validation overall accuracy: ', valAccuracy.accuracy());

//////////////////////////////////// Results exportation ////////////////////////////////////////////////

// Exports the image classified into Intanct, Degraded forest, and deforested
Export.image.toDrive({image:classified_fil,
  description: "MTDD_Classified_" + outName ,
  fileNamePrefix: "MTDD_Classified_" + outName,
  region:aoiT,
  scale:30,
  maxPixels:600000000
});

// Exports the confusion matrix
var toExportMatrix = ee.Feature(null, {matrix: valAccuracy.array()});
Export.table.toDrive({
  collection: ee.FeatureCollection(toExportMatrix),
  description: "MTDD_Matrix_" + outName,
  fileFormat: 'CSV'
});

// Exports the validation overall accuracy
var toExportAccuracy = ee.Feature(null, {matrix: valAccuracy.accuracy()});
Export.table.toDrive({
  collection: ee.FeatureCollection(toExportAccuracy),
  description: "MTDD_Accuracy_" + outName,
  fileFormat: 'CSV'
});

////////////////////////////////////////////////////////////////////////////////////////////////////////

print("All done");
```

Applies a filter to reduce noise

Validates the results by using the validation dataset and generating a confusion matrix

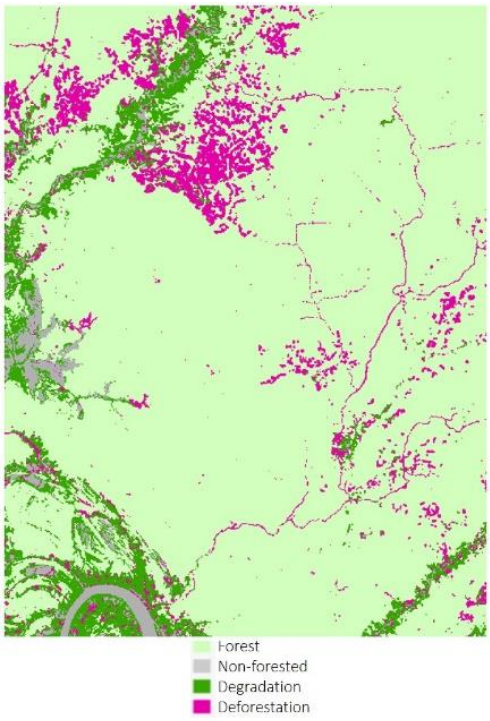
Exports the classified image as a raster, and the confusion matrix and overall accuracy as a csv file



Output

This code outputs a map classified into non-forest, intact forest, degradation, and deforestation.

Raster (".tif" file)



Overall accuracy and confusion matrix (".csv" files)

	A	B	C
1	system:in	matrix	.geo
2	0	0.927632	
3			

	A	B	C	D	E	F
1	system:in	matrix	.geo			
2	0	[[0, 0, 0, 0], [0, 60, 0, 0], [0, 0, 47, 4], [0, 1, 6, 34]]				
3						
4						

References

Bullock, E., 2020. Continuous Degradation Detection (CODED) — coded 0.2 documentation [WWW Document]. URL https://coded.readthedocs.io/en/latest/old.html (accessed 2.9.21).

Bullock, E.L., Woodcock, C.E., Olofsson, P., 2020. Monitoring tropical forest degradation using spectral unmixing and Landsat time series analysis. Remote Sens. Environ. 238, 110968. https://doi.org/10.1016/j.rse.2018.11.011

Cohen, W.B., Healey, S.P., Yang, Z., Stehman, S. V., Brewer, C.K., Brooks, E.B., Gorelick, N., Huang, C., Hughes, M.J., Kennedy, R.E., Loveland, T.R., Moisen, G.G., Schroeder, T.A., Vogelmann, J.E., Woodcock, C.E., Yang, L., Zhu, Z., 2017. How Similar Are Forest Disturbance Maps Derived from Different Landsat Time Series Algorithms? Forests 8, 98. https://doi.org/10.3390/f8040098

FAO, 2001. Global Forest Resources Assessment 2000. Rome, Italy.

Hansen, M.C., Potapov, P. V., Moore, R., Hancher, M., Turubanova, S.A., Tyukavina, A., Thau, D., Stehman, S. V., Goetz, S.J., Loveland, T.R., Kommareddy, A., Egorov, A., Chini, L., Justice, C.O., Townshend, J.R.G., 2013. High-resolution global maps of 21st-century forest cover change. Science (80-.). 342, 850–853. https://doi.org/10.1126/science.1244693

Kennedy, R.E., Yang, Z., Cohen, W.B., 2010. Detecting trends in forest disturbance and recovery using yearly Landsat time series: 1. LandTrendr-Temporal segmentation algorithms. https://doi.org/10.1016/j.rse.2010.07.008

Kennedy, R.E., Yang, Z., Gorelick, N., Braaten, J., Cavalcante, L., Cohen, W.B., Healey, S., 2018. Implementation of the LandTrendr algorithm on Google Earth Engine. Remote Sens. 10, 691. https://doi.org/10.3390/rs10050691

MapBiomas, 2020. MapBiomas Amazonia Project-Collection 2.0 of annual land-cover and land-use maps [WWW Document]. URL https://code.earthengine.google.com/toolkit-download

McDowell, N.G., Coops, N.C., Beck, P.S.A., Chambers, J.Q., Gangodagamage, C., Hicke, J.A., Huang, C. ying, Kennedy, R., Krofcheck, D.J., Litvak, M., Meddens, A.J.H., Muss, J., Negrón-Juarez, R., Peng, C., Schwantes, A.M., Swenson, J.J., Vernon, L.J., Williams, A.P., Xu, C., Zhao, M., Running, S.W., Allen, C.D., 2015. Global satellite monitoring of climate-induced vegetation disturbances. Trends Plant Sci. https://doi.org/10.1016/j.tplants.2014.10.008

RAISG, 2020. Amazon Geo-Referenced Socio-Environmental Information Network [WWW Document]. URL https://www.amazoniasocioambiental.org/ (accessed 3.30.21).

Reygadas, Y., Spera, S., Galati, V., Salisbury, D.S., Silva, S., Novoa, S., 2021. Mapping Forest Disturbances Across the Southwestern Amazon: Evaluation and Comparison of Optical Remote Sensing Algorithms. Remote Sens. Environ. Under review.

Sasaki, N., Putz, F.E., 2009. Critical need for new definitions of “forest” and “forest degradation” in global climate change agreements. Conserv. Lett. 2, 226–232. https://doi.org/10.1111/j.1755-263X.2009.00067.x

Schoene, D., Killmann, W., von Lüpke, H., LoycheWilkie, M., 2007. Definitional Issues Related to Reducing Emissions from Deforestation in Developing Countries, Vol. 5. ed. Food and Agriculture Organization of the United Nations, Rome.

Sexton, J.O., Song, X.-P., Feng, M., Noojipady, P., Anand, A., Huang, C., Kim, D.-H., Collins, K.M., Channan, S., Dimiceli, C., Townshend, J.R., 2013. International Journal of Digital Earth Global, 30-m resolution continuous fields of tree cover: Landsat-based rescaling of MODIS vegetation continuous fields with lidar-based estimates of error. Int. J. Digit. Earth 6, 427–448. https://doi.org/10.1080/17538947.2013.786146

Wang, Y., Ziv, G., Adami, M., Mitchard, E., Batterman, S.A., Buermann, W., Schwantes Marimon, B., Marimon Junior, B.H., Matias Reis, S., Rodrigues, D., Galbraith, D., 2019. Mapping tropical disturbed forests using multi-decadal 30 m optical satellite imagery. Remote Sens. Environ. 221, 474–488. https://doi.org/10.1016/j.rse.2018.11.028