



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

1. Background	1
2. Definitions	1
3. Algorithms.....	2
3.1. CODED	2
3.1.1. <i>Disturbances map</i>	2
3.1.2. <i>Degradation and deforestation map</i>	5
3.2. LandTrendr	8
3.2.1. Disturbances map	8
3.3. MTDD.....	11
3.3.1. <i>Training samples</i>	11
3.3.2. <i>Degradation and deforestation map</i>	16
References.....	22

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:

```
CODED_DisturbancesMap
  ↴ Imports (2 entries)
    ↴ var studyArea: Table users/retinta/Tutorials/CaseStudy1
    ↴ var trainingPts: Table users/retinta/Tutorials/TrainingPointsCODED
1 //////////////////////////////////////////////////////////////////// CODED ALGORITHM ///////////////////////////////////////////////////////////////////
2 /// Retrieves a forest disturbances map ///////////////////////////////////////////////////////////////////
3 /// See Bullock et al., 2020 for algorithm details, and Bullock, 2020 for GEE implementation/////
4 /// This particular piece of code to obtain CODED outputs was written by Y. Reygadas and V. Galati /////
5
6 ////////////////////////////////////////////////////////////////// User-defined information //////////////////////////////////////////////////////////////////
7
8 // Defines the study area
9 var saveRegion = ee.FeatureCollection(studyArea);
10
11 // Defines training data
12 var trainingData = trainingPts;
13
14 // Defines parameters
15 var params = ee.Dictionary({
16   'cfThreshold': .01, // Minimum threshold to remove clouds based on cloud fraction
17   'consec': 4, // Number of consecutive observations below the change threshold needed to declare a disturbance
18   'thresh': 3, // Change threshold (observation residual normalized by the training model RMSE)
19   'start': 2000, // Start year of the study period
20   'end': 2018, // End year of the study period
21   'trainDataEnd': 2016, // End year of the training period
22   'trainDataStart': 2013, // Start year of the training period
23   'trainLength': 3, // Number of years in the training period
24   'soil': [2000, 3000, 3400, 5800, 6000, 5800], // Soil endmember reflectance value for each band
25   'gv': [500, 900, 400, 6100, 3000, 1000], // Green vegetation endmember reflectance value for each band
26   'npv': [1400, 1700, 2200, 3000, 5500, 3000], // Non-photosynthetic vegetation endmember reflectance value for each band
27   'shade': [0, 0, 0, 0, 0, 0], // Shade endmember reflectance value for each band
28   'cloud': [9000, 9600, 8000, 7800, 7200, 6500], // Cloud endmember reflectance value for each band
29   'forestLabel': 1, // Label assigned to forest in the training data
30   'window': 2, // Maximum number of years to use in the monitoring period at any given time
31   'minYears': 2, // Minimum years between disturbances
32   'numChanges': 1, // Maximum number of changes to output
33   'minObs': 5 // Minimum number of observations needed to fit a model for training
34 });
35
36 // Defines output name
37 var outputName = 'CODED_Disturbances';
```

1. Study Area: polygon of the study area.

2. Training data: CODED requires the user to collect land-cover and land-use samples from a selected training period within the entire study period. This is usually obtained through a stratified random sample based on a combination of land-use and land-cover maps and high spatial resolution imagery available in GEE. The field that contains the land-cover identifiers must be named "label".

3. Set of parameters

4. Output name: a desired name for the disturbances map.

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.

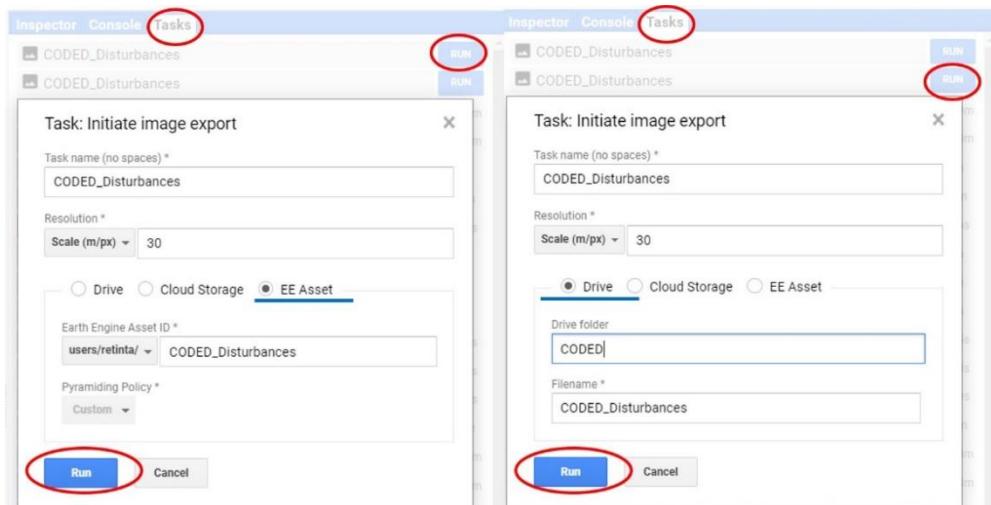
```

39 //////////////////////////////////////////////////////////////////Gets CODED main results////////////////////////////////////////////////////////////////
40
41 // Adds CODED utility functions
42 var codedUtils = require('users/retinta/Tutorials:CODED_changeDetection');
43 var dataUtils = require('users/retinta/Tutorials:CODED_dataUtils');
44
45 // Runs CODED
46 var results = codedUtils.submitCODED(saveRegion, params, trainingData);
47
48 // Turns array columns into images
49 var disturbances = dataUtils.makeImage(results, 0, 'dist_', params.get('start'), params.get('end'));
50 var magnitude = dataUtils.makeImage(results, 1, 'mag_', params.get('start'), params.get('end'));
51 var postChange = dataUtils.makeImage(results, 2, 'post_', params.get('start'), params.get('end'));
52 var difference = dataUtils.makeImage(results, 3, 'dif_', params.get('start'), params.get('end'));
53 var forestFlag = dataUtils.makeImage(results, 4, 'forest_', params.get('start'), params.get('end'));
54
55 var disturbanceBands = disturbances.addBands([magnitude, postChange, difference]);
56
57 var save_output = ee.Image(dataUtils.reduceBands(ee.Image(disturbanceBands), params)
58 |.addBands(forestFlag.select(0)) // Forest flag for first year
59 |.setMulti(params));
60
61 ////////////////////////////////////////////////////////////////// Exports the results //////////////////////////////////////////////////////////////////
62
63 //Exports results as a GEE asset
64 Export.image.toAsset({
65   image: save_output,
66   description: outputName,
67   assetId: outputName,
68   maxPixels: 1e13,
69   scale: 30,
70   region: saveRegion,
71   pyramidingPolicy: {
72     '.default': 'mode'
73   }
74 });
75
76 //Exports results as a raster to drive
77 Export.image.toDrive({
78   image: save_output,
79   description: outputName,
80   region: saveRegion,
81   scale: 30
82 });
83
84 //////////////////////////////////////////////////////////////////
85 print("All done");

```

Detects disturbances and estimates their characteristics

Exports the results as a GEE asset (used as input of the degradation/deforestation map) and as a raster (can be visualized in any GIS)



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.

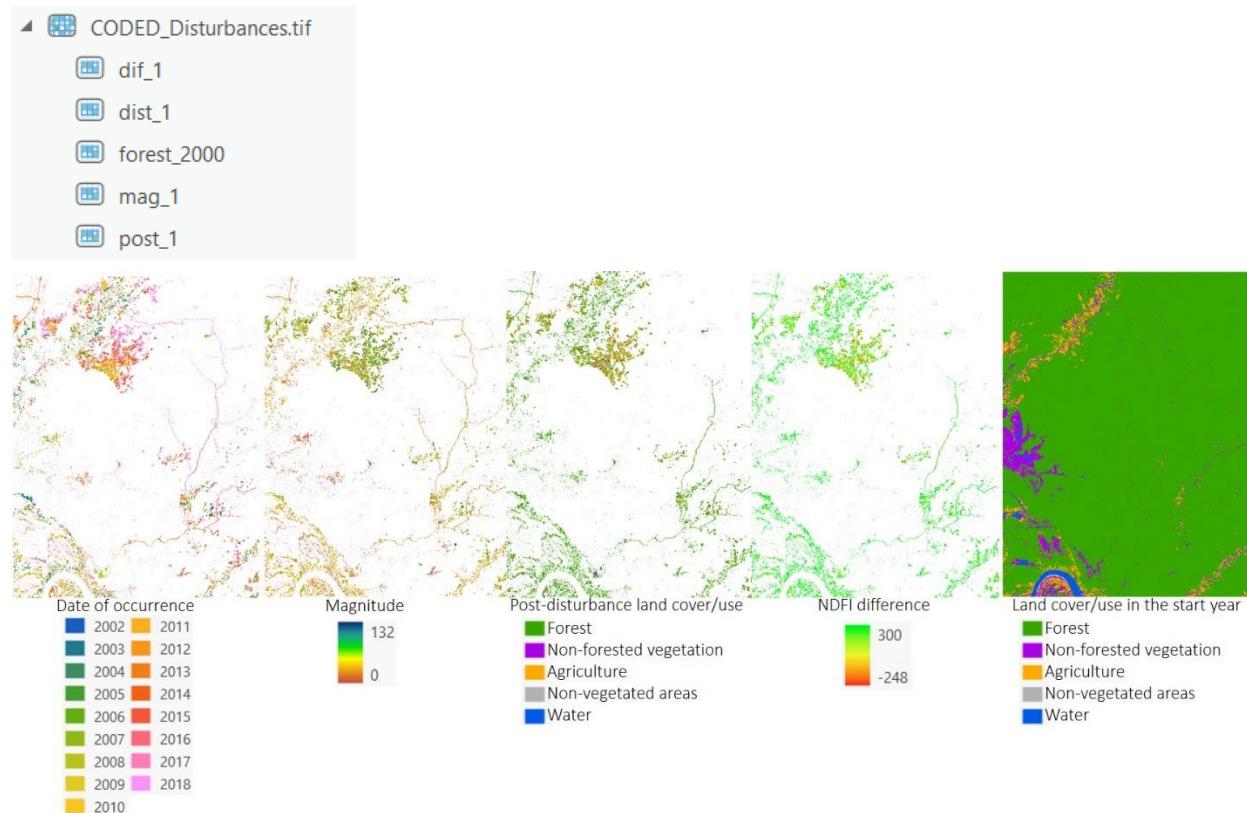
GEE asset

Image: CODED_Disturbances

Index	Name	Type	Dimensions	CRS	Nominal Scale	Min	Max
0	dist_1	signed int16	981x1287 px	EPSG:4326	30.00000000000004	2002	2018
1	mag_1	signed int16	981x1287 px	EPSG:4326	30.00000000000004	3	132
2	post_1	signed int16	981x1287 px	EPSG:4326	30.00000000000004	0	5
3	dif_1	signed int16	981x1287 px	EPSG:4326	30.00000000000004	-248	300
4	forest_2000	signed int16	981x1287 px	EPSG:4326	30.00000000000004	1	5

Image ID: users/retinta/Tutorials/CODED_Disturbances

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:

```
CODED_Deg&DefMap
Get Link Save Run Reset Apps
Imports (2 entries)
var distMap: Image users/retinta/Tutorials/CODED_Disturbances (5 bands)
var stArea: Table users/retinta/Tutorials/CaseStudy1
1 //////////////// CODED ALGORITHM /////////////
2 // Retrieves a strata map: forest/nonforest & degraded/deforested ///////////
3 // See Bullock et al., 2020 for algorithm details, and Bullock, 2020 for GEE implementation ///////////
4 // This particular piece of code to obtain CODED outputs was written by Y. Reygadas and V. Galati ///////////
5
6 //////////////// User-defined information ///////////
7 // Defines the study area
8 var studyArea = stArea; 1. Study Area: polygon of the study area.
9
10 // Specifies the disturbances map to be stratified
11 var dist_map = distMap; 2. Disturbances map: the output from the "CODED_DisturbancesMap" GEE code
12
13 // Defines the output name for the stratified map
14 var outName = 'CODED_DegDef'; 3. Output name: a desired name for the degradation and deforestation map.
15
16
17
```

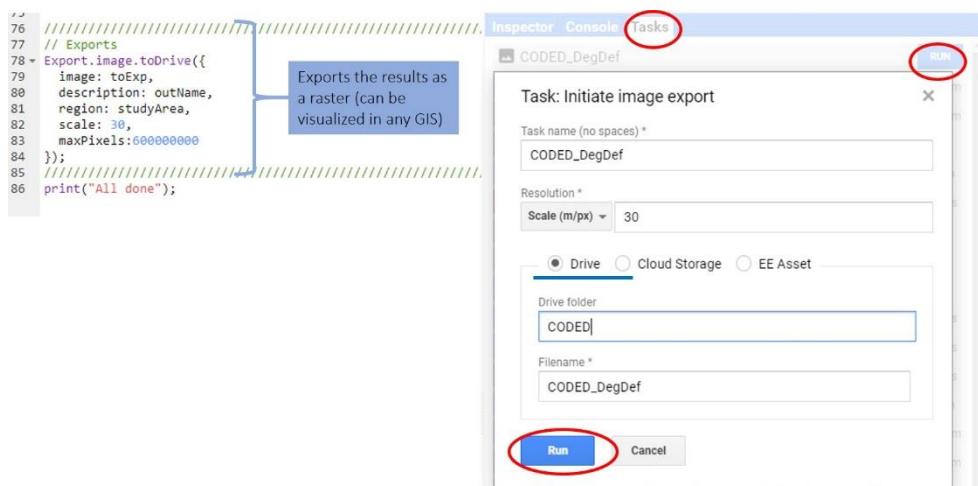
Running the code

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.

```

16 //////////////////////////////////////////////////////////////////Creates a stratified map ///////////////////////////////////////////////////////////////////
17 // Function for stratifying the CODED results from the main function //
18 //(https://coded.readthedocs.io/en/latest/coded.html)//
19 var stratify = function(result) {
20   // The results are masked in areas of non change, so first unmask
21   var unmasked = result.unmask();
22   // Gets the band name of the original forest (depends on the start year)
23   var bands = result.bandNames();
24   // Gets FOREST
25   //If in the start year the land cover WAS forest (forest_startyear= 1)
26   //and there were no disturbances in the study period (dist_1=0), then forest
27   var forested = unmasked.select('forest_2000').eq(1)
28     .and(unmasked.select('dist_1').eq(0));
29   // Gets NON FOREST
30   //If in the start year the land cover WASN'T forest (forest_startyear != 1)
31   //and there were not disturbances, then nonForest and relabel
32   var nonForested = unmasked.select('forest_2000').neq(1)
33     .and(unmasked.select('dist_1').eq(0)).remap([1],[2]);
34   // For pixels that do not have enough observations post-disturbance, this sets
35   // a threshold to distinguish between DEGRADATION and DEFORESTATION
36   var magThreshold = 10;
37   // Gets DEGRADATION
38   // If after the first disturbance, the landcover is still forest (post_1=1),
39   // or if the mag_1 <= magThreshold, then degradation and relabel
40   var degradation = unmasked.select('post_1').eq(1)
41     .or(unmasked.select('mag_1').lte(magThreshold))
42     .remap([1],[3]);
43   // Gets DEFORESTATION
44   // If after the first disturbance, the landcover is not forest anymore (post_1>1)
45   // or the mag_1 > magThreshold, then deforestation and relabel
46   var deforestation = unmasked.select('post_1').gt(1)
47     .or(unmasked.select('mag_1').gt(magThreshold))
48     .remap([1],[4]);
49   // Combines the strata into a single band image
50   var strata = ee.Image.cat([forested,nonForested,degradation,deforestation])
51     .selfMask()
52     .reduce(ee.Reducer.firstNonNull());
53   // Creates and image palette and displays the combined map
54   var strataPalette = ['#013220', '#8b5d2e', '#5CE5D5','#FF2079'];
55   //var stringRes = String(result);
56   //var strataName = stringRes.substring(stringInd+14,stringInd+40);
57   //print (strataName);
58   Map.addLayer(strata,{min:1,max:4,palette:strataPalette}, 'CODED_Strata', false);
59   return strata;
60 };
61 // Runs the function for creating a strata map for an imported image and exports it
62 var toExp = stratify(dist_map);
63
64 
```

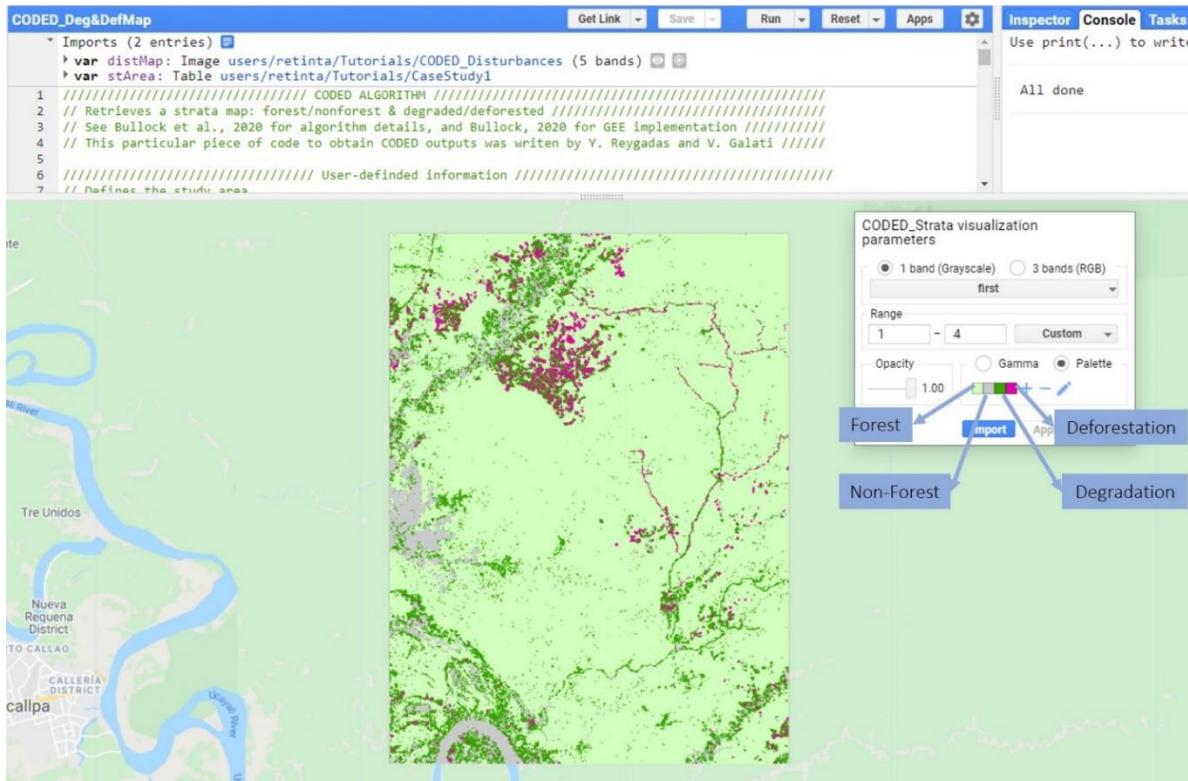
Produces and displays a map classified into forest, non-forest, degradation, and deforestation based on the conditions described in green



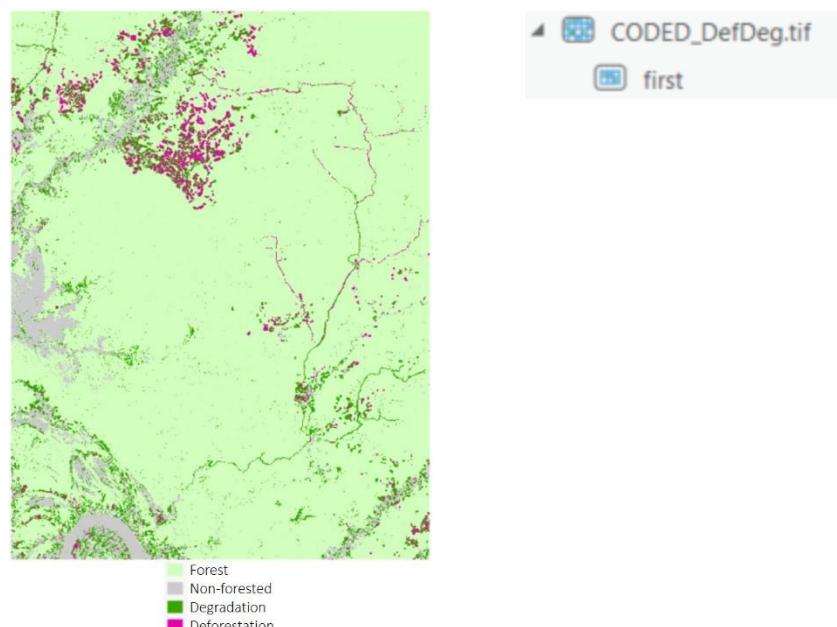
Output

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:

```
LT_DisturbancesMap
  * Imports (1 entry)
    > var studyArea: Table users/retinta/Tutorials/CaseStudy1
    // LANDTRENDR ALGORITHM
    // Retrieves annual and multiyear disturbance maps based on specified paramenters
    // See Kennedy et al., 2010 for algorithm details, and Kennedy et al., 2018 for GEE implementation
    // This particular piece of code to obtain LandTrendr outputs was written by Y. Reygadas and V. Galati //
    // User-defined information
    // Defines the input variable (s)
    var indices = ['NDFI']; // Input variable(s) to be segmented
    // Defines collection parameters
    var startYear = 2000; // The minimum year in the desired range of annual collection
    var endYear = 2018; // The maximum year in the desired range of annual collection
    var startDay = '01-01'; // The maximum date in the desired range of annual collection
    var endDay = '12-31'; // The maximum date in the desired seasonal range over which to generate annual composite
    var aoi = studyArea; // The study area over which to mosaic images
    var maskThese = ['cloud', 'shadow', 'snow', 'water']; //A list of CFMASK mask classes to include as masked pixels
    // Defines landtrendr parameters
    var runParams = {
      maxSegments: 6, // Maximum number of segments to be fitted on the time series
      spikeThreshold: 1, // Threshold for dampening (1 means no dampening)
      vertexCountOvershoot: 3, // The initial regression-based detection of potential vertices can overshoot the maxSegments+1 vertices by this value. Angle-based culling is used to return to the desired number of vertices if overshoot occurs. It allows a mix of criteria for vertex identification
      preventOneYearRecovery: false, // Prevent segments that represent one year recoveries
      recoveryThreshold: 1, // During fitting, if a segment has a recovery rate faster than 1/recoveryThreshold (in years), that segment is disallowed (1 turns off this parameter)
      pvalThreshold: 0.05, // If the p-value of the best fitted model exceeds this threshold, the model is discarded and another is fitted using the Levenburg-Marquardt approach
      bestModelProportion: 0.75, // Takes the model with most vertices that has a p-value that is at most this proportion away from the model with lowest p-value
      minObservationsNeeded: 3 // Minimum observations needed to perform output fitting
    };
    // Filters vegetation changes by:
    var changeParams = {
      delta: 'loss', // loss or gain
      sort: 'greatest', // type (greatest, least, newest, oldest, fastest, or slowest)
      year: {checked:true, start:startYear, end:endYear}, //period of time (in years)
      mag: {checked:true, value:150, operator:'>'}, // magnitude (if the input is a normalized index)
      dur: {checked:true, value:19, operator:'<'}, // duration (in years)
      preval: {checked:true, value:150, operator:'>'}, // pre-change spectral value (if the input is a normalized index)
      mmu: {checked:true, value:3}, // minimum patch size (number of pixels)
    };
    // Define study area name (the output name will contain identifiers .e.g., "LT_NDFI multivear", before the name you assign)
    var outName = "Disturbances";
    // Define output folder
    var outFolder = "LT";
```

1. Input variable(s) to be segmented and from which vegetation changes will be detected

2. Parameters to build an annual collection of Landsat surface reflectance from which input variable-time series are calculated

3. Parameters to control the segmentation of the input variable(s)

4. Parameters to filter vegetation changes

5. Output name and folder: a desired name for the disturbances map and the desired output folder in Google drive

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.

```
LT_DisturbancesMap
22 //////////////////////////////////////////////////////////////////
23 // Require the LandTrendr Library that has NDFI added
24 var ltgee = require('users/retinta/Tutorials:LT_Utilitys_NDFI');
25
26 //////////////////////////////////////////////////////////////////
27 // Function for running LandTrendR per index
28 var doLT = function(indices) {
29   for(var i in indices) {
30     // Add index to changeParams object
31     changeParams.index = indices[i];
32     // Run landtrendr
33     var LTResult = ltgee.runLT(startYear, endYear, startDay, endDay, aoi, indices[i], [], runParams, maskThese);
34     // Call function for getting the disturbances map
35     combinedMap(LTResult,indices[i]);
36   }
37 };
38
39 // Function for getting a disturbances map per index
40 var combinedMap = function(result,index) {
41   changeParams.year = {checked:true, start:startYear, end:endYear};
42   // Get the change map layers (yod, mag, dur, preval, rate, dsnr)
43   var changeImg = ltgee.getChangeMap(result, changeParams);
44   // Display the change attribute map (specifically year of disturbance-yor)
45   Map.addLayer(changeImg.select(['yod']).clip(aoi), yodVizParms, String(index));
46   //Export results
47   exportResults(changeImg,index,outName,outFolder);
48 };
49
50 // Function for exporting data
51 var exportResults = function (img, index, name,folder){
52   Export.Image.toDrive({
53     image: img.clip(aoi).unmask(@).short(),
54     description: 'LT_'+String(index)+'_'+name,
55     folder: folder,
56     region: aoi,
57     scale: 30,
58     maxPixels: 10000000000000
59   });
60 };
61
62 // Visualization parameters (year of disturbance)
63 var palette = ['#940033', '#4B0082', '#0000FF', '#00FF00', '#FFFF00', '#FF7F00', '#FF0000'];
64 var yodVizParms = {
65   min: startYear,
66   max: endYear,
67   palette: palette
68 };
69
70 // Adjusts map
71 Map.centerObject(aoi,10);
72
73 // Runs everything ///
74 doLT(indices);
75
76 //////////////////////////////////////////////////////////////////
77 print("All done");
78
```

Requires the Landtrendr library that also supports NDFI

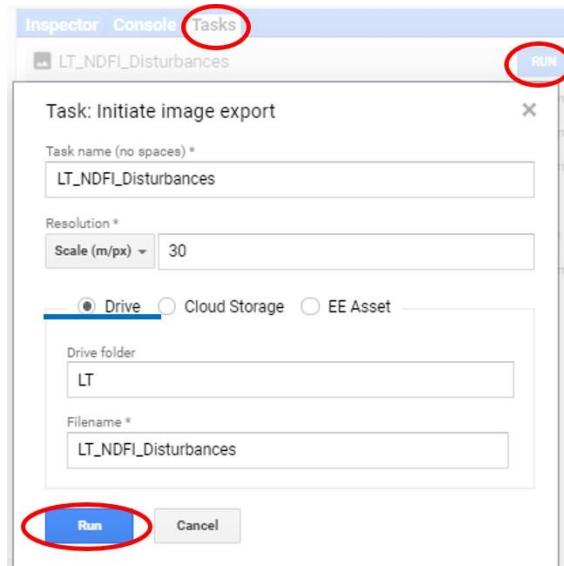
Segments the input variable(s)

Produces a disturbances map with the user-defined filtering criteria

Exports the results as raster(s)

Sets the map visualization parameters

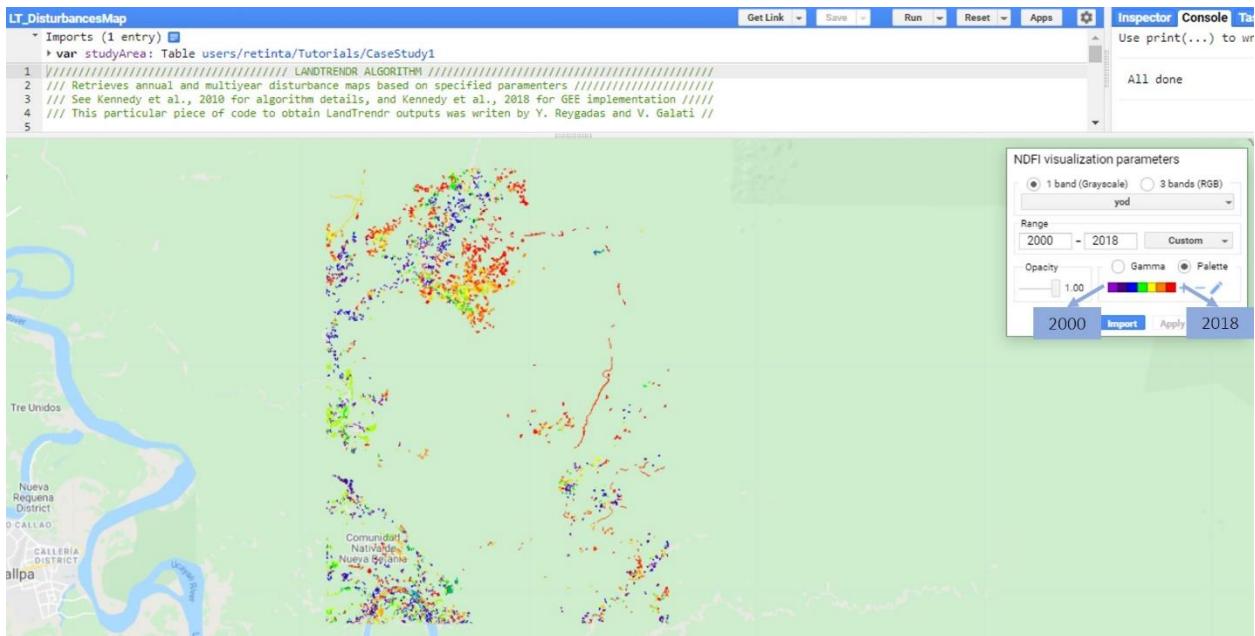
Runs all functions



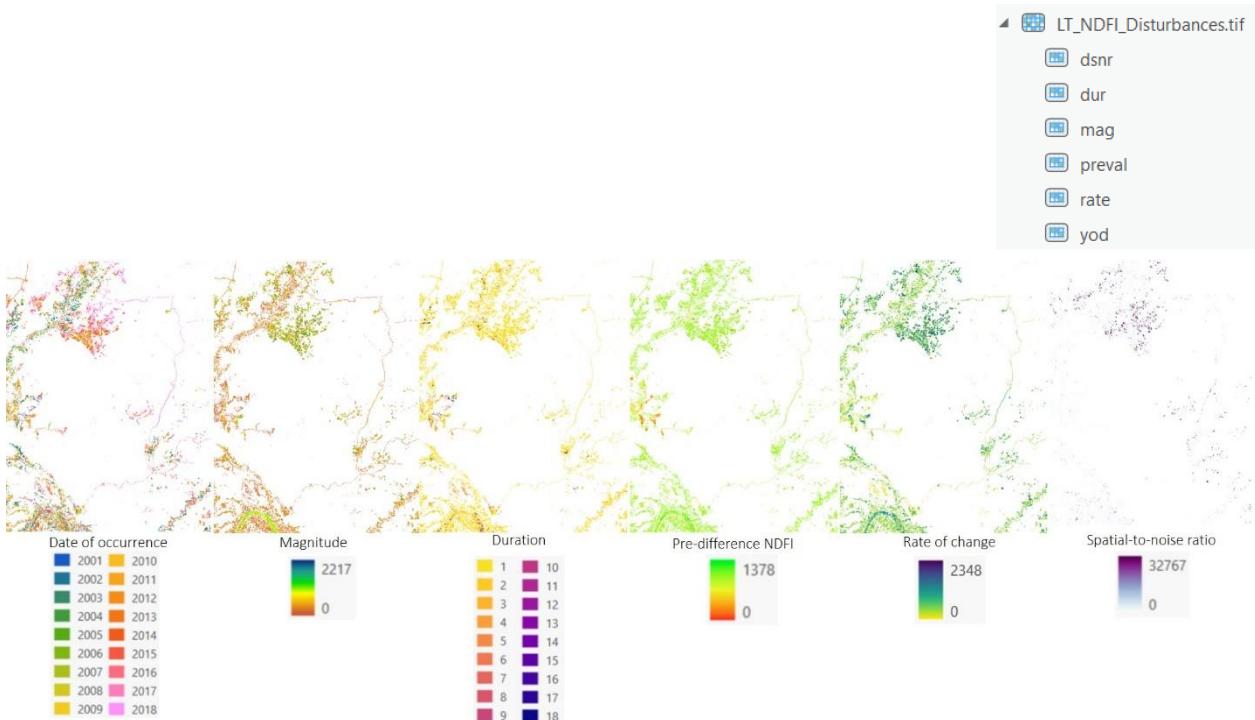
Output

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	MapBiomass (MapBiomass, 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	MapBiomass (MapBiomass, 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	MapBiomass (MapBiomass, 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:

```
MTDD_Sampling
* Imports (5 entries) 
  > var GFW: (Deprecated) Image "Hansen Global Forest Change v1.7 (2000-2019)" (13 bands) A 
  > var USGS_TC: ImageCollection "Global Forest Cover Change (GFCC) Tree Cover Multi-Year Global 30m"
  > var MAPBIOMAS: ImageCollection users/retinta/MachineLearning/MapBiomassLcCol
  > var RAISG: Image users/retinta/MachineLearning/RAISG_2000_2018_fix1 (1 band) C 
  > var studyArea: Table users/retinta/Tutorials/CaseStudy1

1 //////////////////////////////////////////////////// MTDD ALGORITHM ///////////////////////////////////
2 ////////////////////////////////// Sampling /////////////////////
3 /// Outputs a user-defined number (90% of which are for training and 10% for validation) ///
4 /// of Intact Forest, Degraded Forest, and Deforested sampling points. If the output has ///
5 /// less points than expected, it is because the code ensures that the training samples ///
6 /// are uncorrelated with the validation samples by removing samples that are within some ///
7 /// distance to other sample(s) /////////////////////
8 /// Based on Wang et al. (2019) and adapted by Reygadas et al. 2021 to the Southwestern ///
9 /// Amazon to detect intact, disturbed, and deforested areas///////////////////

10 ////////////////// User-defined information ///////////////////
11
12 // Defines the study period
13 var startYear = 2000;
14 var endYear = 2018; 1. Study period: start and end year

15
16 // Defines the number of desired sampling points per class:
17 // intact forest, degraded forest, and deforested
18 // (90% will be used for training and 10% for validation)
19 var samPoints = 500; 2. Sampling points: number of desired sampling points per class

20
21 // Defines the study area
22 var aoi = studyArea; 3. Study Area: polygon of the study area

23
24 // Defines the datasets to build conditional to detect intact, degraded, and deforested areas
25 var gfw = GFW; // Global Forest Watch (GFC)
26 var usgs = USGS_TC; // Global Forest Cover Change (GFCC) Tree Cover Multi-Year Global 30m
27 var mapBiomass = MAPBIOMAS; // Annual Land Cover, MapBiomass
28 var raisg = RAISG; // Deforestation, RAISG 4. Reference datasets: sources to build the conditions

29
30
31 // Defines a and b (see table in section 3.3.1 of the tutorial) tree cover thresholds
32 var tcMaxThres = 75; 5. Three cover thresholds: desired tree cover threshold for some of the conditions (see table above)
33 var tcMinThres = 30;
34
35 // Defines a desired name that will be added to identifiers (e.g. validationPnts "outputName")
36 var outputName = "MTDT"; 6. Output name: desired name for the training and validation points
```

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.

```

MTDD_Sampling
Get Link | Save | Run | Reset | Apps
All conditions
37 ////////////////////////////////////////////////////////////////// All conditions //////////////////////////////////////////////////////////////////
38
39 ////////////////////////////////////////////////////////////////// Deforested at some point / not deforested (RAISG) //////////////////////////////////////////////////////////////////
40 var rStartYear; // RAISG starts in 2001, this conditional selects 2001 as startYear if the actual startYear is lower (2000)
41 if (startYear < 2001) {rStartYear = 2001;} else {rStartYear = startYear}
42 var def = raisg.lte(endYear).and(raisg.gte(rStartYear));//Deforestation appears as 1
43 var noDef = def.not();// No deforestation appears as 1
44
45 ////////////////////////////////////////////////////////////////// Forest loss at any time / no loss (GFW) //////////////////////////////////////////////////////////////////
46 var endYearN = endYear-2000;// converts the end year to a number of two digits
47 var startYearN = startYear-2000; // converts the end start to a number of two digits
48 var loss = gfw.select("lossyear").lte(endYearN).and(gfw.select("lossyear").gte(startYearN)).unmask();
49 var noLoss = loss.not();
50
51 ////////////////////////////////////////////////////////////////// Tree cover greater than 75% in 2000 (GFW) //////////////////////////////////////////////////////////////////
52 var tc75_2000 = gfw.select("treecover2000").gt(tcMaxThres);
53
54 ////////////////////////////////////////////////////////////////// Tree Cover criteria (USGS): //////////////////////////////////////////////////////////////////
55 // The USGS dataset has TC data for 2005, 2010, and 2015, so it sets a conditional
56 // to select the TC dataset closest to the endYear
57 var tcYear;
58 if (endYear < 2007) {tcYear = 2005;} // this conditional helps to select the TC dataset closest to the endYear
59 else if (endYear >= 2008 && endYear <= 2012) {tcYear = 2010;}
60 else if (endYear > 2013) {tcYear = 2015;}
61 // Function for applying a filter to reduce noise in the TC USGS outputs (years 2005 and 2010 are specially noisy)
62 var filter = function (image) {
63   // Applies a 90m X 90m filter (default options)
64   var imageFiltered = image.focal_mode();
65   // This is only for visualization purposes. The reproject is sufficient to force the computation to occur at native scale.
66   return imageFiltered.reproject('EPSG:4326', null, 30);
67 };
68
69 // Tree cover greater than 75% in 2015 or around the end of the study period (USGS)
70 var tc75_endYear = usgs.filterBounds(aoi).select("tree_canopy_cover")
71   .filterMetadata("year","equals",tcYear).mean().gt(tcMaxThres); // mean is only to convert it into a single image rather than image collection with only one
72 var tc75_endYear_fil = filter (tc75_endYear);
73
74 // Tree cover greater or equal than 30% and lower or equal than 75% in 2015 or around the end of the study period (USGS)
75 var tc30_endYear = usgs.filterBounds(aoi).select("tree_canopy_cover")
76   .filterMetadata("year","equals",tcYear).mean().gt(tcMinThres); // mean is only to convert it into a single image rather than image collection with only one
77 var tcLess75_endYear = usgs.filterBounds(aoi).select("tree_canopy_cover")
78   .filterMetadata("year","equals",tcYear).mean().lte(tcMaxThres);
79 var tc30_75_endYear = tc30_endYear.and(tcLess75_endYear);
80 var tc30_75_endYear_fil = filter (tc30_75_endYear);
81
82 ////////////////////////////////////////////////////////////////// Forest cover during all the study period (MapBiomass) //////////////////////////////////////////////////////////////////
83 var rename = function (image){// Funcion for assigning the same name to all bands
84   return image.select(0).rename("b1");
85 }
86 var mapBiomass_r = mapBiomass.map(rename);
87 var mapBiomass_fil = mapBiomass_r.filter(ee.Filter.and//Creates an collection containing the years within the study period (untill 2018)
88   ee.Filter.gte('year', startYear),
89   ee.Filter.lte('year', endYear));
90 var mapBiomass_minMax = mapBiomass_fil.reduce(ee.Reducer.minMax()).rename("min","max");
91 var mapBiomass_range = mapBiomass_minMax.expression('b\'\min\'-b\'\max\'').rename('range');
92 var forestAll = mapBiomass_minMax.select("max").lte(6)//https://storage.googleapis.com/mapbiomas-public/RAISG/COLECAO2/LEGENDA/codigo_de_la_leyenda_coleccion-2.pdf
93   .and(mapBiomass_range.eq(0)).rename("allForest");//forest is 3 or 6 in the entire Acre-Ucayali region during all years
94
95 ////////////////////////////////////////////////////////////////// Forest / non-forest at the end of the study period (MapBiomass) //////////////////////////////////////////////////////////////////
96 var forestEndYear = mapBiomass_r.filter(ee.Filter.eq("year",endYear)).mean().lte(6); //mean is only to convert it into a single image rather than image collection with only one image
97 var noForestEndYear = forestEndYear.not();
98
99 ////////////////////////////////////////////////////////////////// Selection of intact, disturbed, and deforested based on the above criteria //////////////////////////////////////////////////////////////////
100
101 ////////////////////////////////////////////////////////////////// Intact forest (1)
102 // (criteria: never classified as deforested, TC>75% in 2000,
103 // no forest loss, TC>75% in 2015 (or around the end of the study period), classified as forest during all the study period)
104 var intactForest = noDef.and(tc75_2000).and(noLoss)
105   .and(tc75_endYear).and(forestAll).clip(aoi);
106
107 ////////////////////////////////////////////////////////////////// Degraded forest (2)
108 // (criteria: Deforested at some point, TC>30<=75% in 2015 (or around the end of the study period),
109 // and classified as forest at the end of the study period)
110 var degradedForest = def.and(tc30_75_endYear_fil)
111   .and(forestEndYear).remap([0,1],[0,2]).clip(aoi);
112
113 ////////////////////////////////////////////////////////////////// Deforested (3)
114 // (criteria: classified as deforested at some point, forest loss at some point,
115 // and classified as non-forest at the end of the study period)
116 var deforested = def.and(loss)
117   .and(forestEndYear).remap([0,1],[0,3]).clip(aoi);
118
119 // Merges classes and adds the result to the map (Using Image.cat to combine provokes an error in the sampling)
120 var IntactDefDef = intactForest.add(degradedForest).add(deforested).remap([1,2,3],[1,2,3]).rename("class");
121 Map.addLayer(IntactDefDef, {"opacity":1,"bands":["class"],"palette":["#0ff93","ff6131","b05aff"]}, "Intact, Disturbed, and Deforested");
122 Map.centerObject(aoi,10);
123

```

Builds each of the conditions described in the table above

Selects intact, degraded, and deforested areas based on the set of conditions described in green (also described in the table above)

Displays intact, degraded, and deforested areas

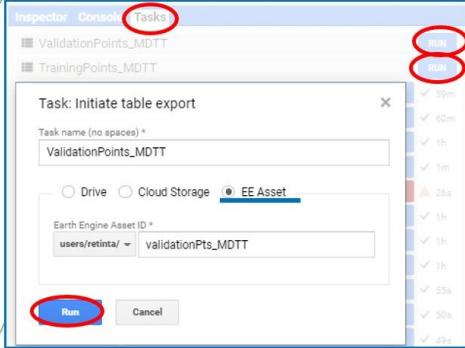
```

124 ////////////////////////////////////////////////////////////////// Random Sampling //////////////////////////////////////////////////////////////////
125
126 // Creates a random sample based on the user-defined number of sampling points per class
127 var samplingPts = IntactDegDef.stratifiedSample({
128   numPoints: 0, // points for pixel values different than 1 and 2
129   seed: 0,
130   classValues: [1,2,3], // pixel values to be sampled
131   classPoints: [samPoints,samPoints,samPoints], // number of sample points for each of the above classes
132   scale: 30, // Landsat spatial resolution
133   geometries: true, // retains the spatial information of the sample points needed for a spatial join
134 });
135
136 // Partitions the sample into training and validation points
137 samplingPts = samplingPts.randomColumn(); // adds column of uniform random numbers in a column named 'random'
138 var split = 0.9; // roughly 90% training, 10% validation
139 var training = samplingPts.filter(ee.Filter.lt('random', split));
140 var validation = samplingPts.filter(ee.Filter.gte('random', split));
141
142 // Ensures that the training samples are uncorrelated with the validation samples
143 // by removing samples that are within some distance to any other sample(s)
144 var distFilter = ee.Filter.withinDistance({//spatial join
145   distance: 500,
146   leftField: '.geo',
147   rightField: '.geo',
148   maxError: 10
149 });
150 var join = ee.Join.inverted();
151 training = join.apply(training, validation, distFilter); // applies the join
152
153 ////////////////////////////////////////////////////////////////// Prints and exports the training and validation points //////////////////////////////////////////////////////////////////
154
155 //Prints the results and adds them to the map
156 print("Training points", training.size());
157 //print(training);
158 Map.addLayer(training, {}, "Training Points");
159
160 print("Validation points", validation.size());
161 //print(validation);
162 Map.addLayer(validation, {}, "Validation Points");
163
164 Export.table.toAsset({collection:training,
165   description:"TrainingPoints_" + outputName,
166   assetId: "trainingPts_" + outputName
167 });
168
169 Export.table.toAsset({collection: validation,
170   description:"ValidationPoints_" + outputName,
171   assetId: "validationPts_" + outputName
172 });
173
174 //////////////////////////////////////////////////////////////////
175 print("All done");
176

```

Annotations for the code:

- Line 127: Randomly selects the user-defined number of samples per class
- Line 138: Divides the sampling points into training (90%) and validation (10%) datasets
- Line 144: Removes any points closer than 500 meters of each other to ensure training samples are uncorrelated with validation samples
- Line 156: Prints the resulting number of training and validation points in the console and displays them in the map
- Line 164: Exports the results as GEE assets (used as inputs of the map classified into intact, degraded, and deforested)



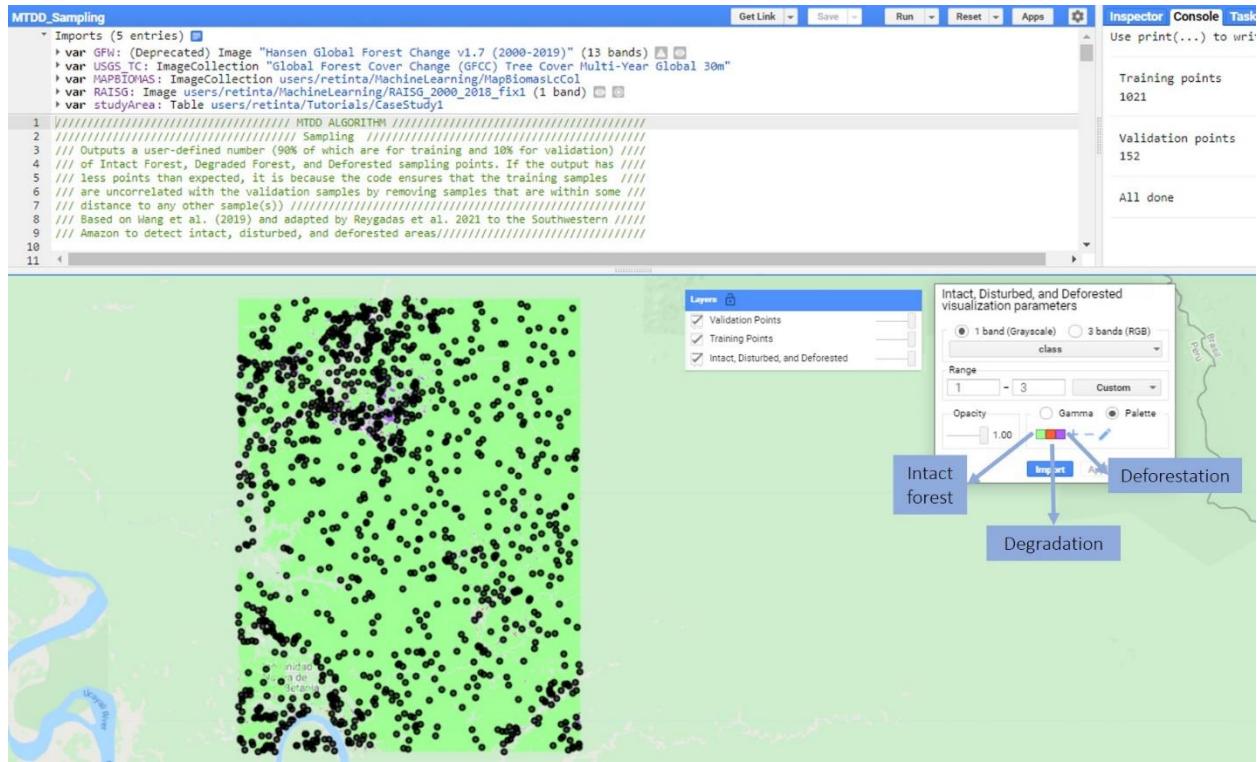
Output

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

GEE assets

Table: trainingPts_MDTT				Table: validationPts_MDTT			
DESCRIPTION FEATURES PROPERTIES				DESCRIPTION FEATURES PROPERTIES			
							
Table ID	<input type="checkbox"/>	users/retinta/Tutorials/trainingPts_MDTT	Table ID	<input type="checkbox"/>	users/retinta/Tutorials/validationPts_MDTT	Table ID	<input type="checkbox"/>
Feature Index	class (Long)	random (Float)	system:index (String)	Feature Index	class (Long)	random (Float)	system:index (String)
0	1	0.5397374669840993		0	1	0.9287363229116009	
1	1	0.17310481412728573		1	1	0.9962982589520231	
2	1	0.3011276520055456		2	1	0.9668643293210513	
3	1	0.5262166033693878		3	1	0.9393284494480373	
4	1	0.08721550293471692		4	1	0.952914024892686	

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:

```
MTDD_DisturbancesMap
1 //////////////////////////////////////////////////////////////////// MTDD ALGORITHM //////////////////////////////
2 //////////////////////////////////////////////////////////////////// Clasification //////////////////////////////
3 // Outputs an image classified as Non Forest (0), Intact Forest (1), Disturbed Forest (2), //////////////////
4 // and Deforestation (3), also VALIDATES the model and exports the results in csv format //////////////////
5 //////////////////////////////////////////////////////////////////// Based on Wang et al. (2019) and adapted by Reygadas et al. 2021 to the Southwestern //////////////////
6 //////////////////////////////////////////////////////////////////// Amazon to detect intact, disturbed, and deforested areas //////////////////
7 //////////////////////////////////////////////////////////////////// User-defined information //////////////////////////////
8
9 // Go to this link before starting: https://code.earthengine.google.com/?accept_repo=users/emaprlab/public;
10 // If you want to display the map and print the accuracies in the console,
11 // you have to uncomment lines 270-273 & 387-388, but if the study area is big, it usually provokes a time computation or memory limit error
12
13 //////////////////////////////////////////////////////////////////// Defines the years of the training data (possible years: 2000-2018)
14 var startYearT = 2000;
15 var endYearT = 2018;
16
17 //////////////////////////////////////////////////////////////////// Defines years for classification (possible years: 2000-present)
18 var startYearC = 2000;
19 var endYearC = 2020;
20
21 //////////////////////////////////////////////////////////////////// Defines parameters to build a Landsat collection
22 var startDayB = '01-01';
23 var endDayB = '12-31';
24 var aoiB = studyArea;
25 var maskTheseB = ['cloud', 'shadow', 'snow', 'water'];
26
27 //////////////////////////////////////////////////////////////////// Defines training and validation samples
28 var trainingPts = tp;
29 var validationPts = vp;
30
31 //////////////////////////////////////////////////////////////////// Defines datasets to define forest and non forest areas
32 var mapBiomass = MAPBIOMAS; // Annual Land Cover
33
34 //////////////////////////////////////////////////////////////////// Defines an output name that will be added to these identifiers: MTDD_Classified
35 var outName = "2018";
```

1. Training period: start and end year used to generate the training points (start and end year in the “MTDD_Sampling” GEE code)

2. Classification period: start and end year for the classification (may or may not coincide with the training period)

3. Parameters to build an annual Landsat surface reflectance collection from which the time series are calculated

4. Training and validation points: the output from the “MTDD_Sampling” GEE code

5. Land-use and land-cover maps: used to define a forest mask

6. Output name: a desired name for the output map

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 //////////////////////////////////////////////////////////////////// Function for calculating ALL METRICS ///////////////////////////////////////////////////////////////////
41 var metricsGral = function (startYear, endYear, startDay, endDay, aoi, maskThese){ ← Start of the function that calculates all 66 metrics
42
43
44 //////////////////////////////////////////////////////////////////// Time Series Trajectories ///////////////////////////////////////////////////////////////////
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/emaplab/public:Modules/LandTrendr.js');// Requires the LandTrendR Library
48 var annualSRC = ltgee.buildSRCcollection(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); ← Builds a collection with annual time series of
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
279
280
281
282
283
284
285
286
287
288
289
289
290
291
292
293
294
295
296
297
298
299
299
300
301
302
303
304
305
306
307
308
309
309
310
311
312
313
314
315
316
317
318
319
319
320
321
322
323
324
325
326
327
328
329
329
330
331
332
333
334
335
336
337
338
339
339
340
341
342
343
344
345
346
347
348
349
349
350
351
352
353
354
355
356
357
358
359
359
360
361
362
363
364
365
366
367
368
369
369
370
371
372
373
374
375
376
377
378
379
379
380
381
382
383
384
385
386
387
388
389
389
390
391
392
393
394
395
396
397
398
399
399
400
401
402
403
404
405
406
407
408
409
409
410
411
412
413
414
415
416
417
418
419
419
420
421
422
423
424
425
426
427
428
429
429
430
431
432
433
434
435
436
437
438
439
439
440
441
442
443
444
445
446
447
448
449
449
450
451
452
453
454
455
456
457
458
459
459
460
461
462
463
464
465
466
467
468
469
469
470
471
472
473
474
475
476
477
478
479
479
480
481
482
483
484
485
486
487
488
489
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
509
510
511
512
513
514
515
516
517
518
519
519
520
521
522
523
524
525
526
527
528
529
529
530
531
532
533
534
535
536
537
538
539
539
540
541
542
543
544
545
546
547
548
549
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
568
569
569
570
571
572
573
574
575
576
577
578
579
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
687
688
689
689
690
691
692
693
694
695
696
697
697
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
717
718
719
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
787
788
789
789
790
791
792
793
794
795
796
797
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
817
818
819
819
820
821
822
823
824
825
826
827
828
828
829
830
831
832
833
834
835
836
837
838
838
839
840
841
842
843
844
845
846
847
847
848
849
849
850
851
852
853
854
855
856
857
858
858
859
860
861
862
863
864
865
866
867
867
868
869
869
870
871
872
873
874
875
876
877
877
878
879
879
880
881
882
883
884
885
886
887
887
888
889
889
890
891
892
893
894
895
896
897
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
917
918
919
919
920
921
922
923
924
925
926
927
927
928
929
929
930
931
932
933
934
935
936
937
937
938
939
939
940
941
942
943
944
945
946
947
947
948
949
949
950
951
952
953
954
955
956
957
957
958
959
959
960
961
962
963
964
965
966
966
967
968
968
969
969
970
971
972
973
974
975
976
976
977
978
978
979
979
980
981
982
983
984
985
986
986
987
987
988
988
989
989
990
991
992
993
994
995
996
997
998
999
999
1000
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1017
1018
1019
1019
1020
1021
1022
1023
1024
1025
1026
1027
1027
1028
1029
1029
1030
1031
1032
1033
1034
1035
1036
1037
1037
1038
1039
1039
1040
1041
1042
1043
1044
1045
1046
1046
1047
1048
1048
1049
1049
1050
1051
1052
1053
1054
1055
1056
1057
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1066
1067
1068
1068
1069
1069
1070
1071
1072
1073
1074
1075
1076
1076
1077
1078
1078
1079
1079
1080
1081
1082
1083
1084
1085
1086
1086
1087
1088
1088
1089
1089
1090
1091
1092
1093
1094
1095
1095
1096
1097
1097
1098
1098
1099
1099
1100
1101
1102
1103
1104
1105
1105
1106
1107
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1115
1116
1117
1117
1118
1118
1119
1119
1120
1121
1122
1123
1124
1125
1125
1126
1127
1127
1128
1128
1129
1129
1130
1131
1132
1133
1134
1135
1135
1136
1137
1137
1138
1138
1139
1139
1140
1141
1142
1143
1144
1145
1145
1146
1147
1147
1148
1148
1149
1149
1150
1151
1152
1153
1154
1155
1155
1156
1157
1157
1158
1158
1159
1159
1160
1161
1162
1163
1164
1165
1165
1166
1167
1167
1168
1168
1169
1169
1170
1171
1172
1173
1174
1175
1175
1176
1177
1177
1178
1178
1179
1179
1180
1181
1182
1183
1184
1185
1185
1186
1187
1187
1188
1188
1189
1189
1190
1191
1192
1193
1194
1194
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1201
1202
1203
1204
1204
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1211
1212
1213
1214
1214
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1221
1222
1223
1224
1225
1225
1226
1227
1227
1228
1228
1229
1229
1230
1231
1232
1233
1234
1235
1235
1236
1237
1237
1238
1238
1239
1239
1240
1241
1242
1243
1244
1245
1245
1246
1247
1247
1248
1248
1249
1249
1250
1251
1252
1253
1254
1255
1255
1256
1257
1257
1258
1258
1259
1259
1260
1261
1262
1263
1264
1265
1265
1266
1267
1267
1268
1268
1269
1269
1270
1271
1272
1273
1274
1275
1275
1276
1277
1277
1278
1278
1279
1279
1280
1281
1282
1283
1284
1285
1285
1286
1287
1287
1288
1288
1289
1289
1290
1291
1292
1293
1294
1294
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1301
1302
1303
1304
1304
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1311
1312
1313
1314
1314
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1321
1322
1323
1324
1325
1325
1326
1327
1327
1328
1328
1329
1329
1330
1331
1332
1333
1334
1335
1335
1336
1337
1337
1338
1338
1339
1339
1340
1341
1342
1343
1344
1345
1345
1346
1347
1347
1348
1348
1349
1349
1350
1351
1352
1353
1354
1355
1355
1356
1357
1357
1358
1358
1359
1359
1360
1361
1362
1363
1364
1365
1365
1366
1367
1367
1368
1368
1369
1369
1370
1371
1372
1373
1374
1375
1375
1376
1377
1377
1378
1378
1379
1379
1380
1381
1382
1383
1384
1385
1385
1386
1387
1387
1388
1388
1389
1389
1390
1391
1392
1393
1394
1394
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1401
1402
1403
1404
1404
1405
1406
1406
1407
1407
1408
1408
1409
1409
1410
1411
1412
1413
1414
1414
1415
1416
1416
1417
1417
1418
1418
1419
1419
1420
1421
1422
1423
1424
1425
1425
1426
1427
1427
1428
1428
1429
1429
1430
1431
1432
1433
1434
1435
1435
1436
1437
1437
1438
1438
1439
1439
1440
1441
1442
1443
1444
1445
1445
1446
1447
1447
1448
1448
1449
1449
1450
1451
1452
1453
1454
1455
1455
1456
1457
1457
1458
1458
1459
1459
1460
1461
1462
1463
1464
1465
1465
1466
1467
1467
1468
1468
1469
1469
1470
1471
1472
1473
1474
1475
1475
1476
1477
1477
1478
1478
1479
1479
1480
1481
1482
1483
1484
1485
1485
1486
1487
1487
1488
1488
1489
1489
1490
1491
1492
1493
1494
1494
1495
1496
1496
1497
1497
1498
1498
1499
1499
1500
1501
1502
1503
1504
1504
1505
1506
1506
1507
1507
1508
1508
1509
1509
1510
1511
1512
1513
1514
1514
1515
1516
1516
1517
1517
1518
1518
1519
1519
1520
1521
1522
1523
1524
1525
1525
1526
1527
1527
1528
1528
1529
1529
1530
1531
1532
1533
1534
1535
1535
1536
1537
1537
1538
1538
1539
1539
1540
1541
1542
1543
1544
1545
1545
1546
1547
1547
1548
1548
1549
1549
1550
1551
1552
1553
1554
1555
1555
1556
1557
1557
1558
1558
1559
1559
1560
1561
1562
1563
1564
1565
1565
1566
1567
1567
1568
1568
1569
1569
1570
1571
1572
1573
1574
1575
1575
1576
1577
1577
1578
1578
1579
1579
1580
1581
1582
1583
1584
1585
1585
1586
1587
1587
1588
1588
1589
1589
1590
1591
1592
1593
1594
1595
1595
1596
1597
1597
1598
1598
1599
1599
1600
1601
1602
1603
1604
1604
1605
1606
1606
1607
1607
1608
1608
1609
1609
1610
1611
1612
1613
1614
1614
1615
1616
1616
1617
1617
1618
1618
1619
1619
1620
1621
1622
1623
1624
1625
1625
1626
1627
1627
1628
1628
1629
1629
1630
1631
1632
1633
1634
1635
1635
1636
1637
1637
1638
1638
1639
1639
1640
1641
1642
1643
1644
1645
1645
1646
1647
1647
1648
1648
1649
1649
1650
1651
1652
1653
1654
1655
1655
1656
1657
1657
1658
1658
1659
1659
1660
1661
1662
1663
1664
1665
1665
1666
1667
1667
1668
1668
1669
1669
1670
1671
1672
1673
1674
1675
1675
1676
1677
1677
1678
1678
1679
1679
1680
1681
1682
1683
1684
1685
1685
1686
1687
1687
1688
1688
1689
1689
1690
1691
1692
1693
1694
1695
1695
1696
1697
1697
1698
1698
1699
1699
1700
1701
1702
1703
1704
1704
1705
1706
1706
1707
1707
1708
1708
1709
1709
1710
1711
1712
1713
1714
1714
1715
1716
1716
1717
1717
1718
1718
1719
1719
1720
1721
1722
1723
1724
1725
1725
1726
1727
1727
1728
1728
1729
1729
1730
1731
1732
1733
1734
1735
1735
1736
1737
1737
1738
1738
1739
1739
1740
1741
1742
1743
1744
1745
1745
1746
1747
1747
1748
1748
1749
1749
1750
1751
1752
1753
1754
1755
1755
1756
1757
1757
1758
1758
1759
1759
1760
1761
1762
1763
1764
1765
1765
1766
1767
1767
1768
1768
1769
1769
1770
1771
1772
1773
1774
1775
1775
1776
1777
1777
1778
1778
1779
1779
1780
1781
1782
1783
1784
1785
1785
1786
1787
1787
1788
1788
1789
1789
1790
1791
1792
1793
1794
1795
1795
1796
1797
1797
1798
1798
1799
1799
1800
1801
1802
1803
1804
1804
1805
1806
1806
1807
1807
1808
1808
1809
1809
1810
1811
1812
1813
1814
1814
1815
1816
1816
1817
1817
1818
1818
1819
1819
1820
1821
1822
1823
1824
1825
1825
1826
1827
1827
1828
1828
1829
1829
1830
1831
1832
1833
1834
1835
1835
1836
1837
1837
1838
1838
1839
1839
1840
1841
1842
1843
1844
1845
1845
1846
1847
1847
1848
1848
1849
1849
1850
1851
1852
1853
1854
1855
1855
1856
1857
1857
1858
1858
1859
1859
1860
1861
1862
1863
1864
1865
1865
1866
1867
1867
1868
1868
1869
1869
1870
1871
1872
1873
1874
1875
1875
1876
1877
1877
1878
1878
1879
1879
1880
1881
1882
1883
1884
1885
1885
1886
1887
1887
1888
1888
1889
1889
1890
1891
1892
1893
1894
1895
1895
1896
1897
1897
1898
1898
1899
1899
1900
1901
1902
1903
1904
1904
1905
1906
1906
1907
1907
1908
1908
1909
1909
1910
1911
1912
1913
1914
1914
1915
1916
1916
1917
1917
1918
191
```

```

74 ////////////////////////////////////////////////////////////////// Descriptive statistics //////////////////////////////////////////////////////////////////
75
76 ////////////////////////////////////////////////////////////////// Location metrics: min, max, range, mean /////
77 var min = annualSRC_md_tm_indices.select(['B5','B7','NDVI','NDWI2130','NDWI1640','SAVI'])
78 .reduce(ee.Reducer.min());
79 var max = annualSRC_md_tm_indices.select(['B5','B7','NDVI','NDWI2130','NDWI1640','SAVI'])
80 .reduce(ee.Reducer.max());
81 var range = max.subtract(min).rename(['B5_range','B7_range','NDVI_range','NDWI2130_range','NDWI1640_range','SAVI_range']);
82 var mean = annualSRC_md_tm_indices.select(['B5','B7','NDVI','NDWI2130','NDWI1640','SAVI'])
83 .reduce(ee.Reducer.mean());
84
85 ////////////////////////////////////////////////////////////////// Scale metrics: stdDev, C.V., kurtosis, skewness /////
86 var stdDev = annualSRC_md_tm_indices.select(['B5','B7','NDVI','NDWI2130','NDWI1640','SAVI'])
87 .reduce(ee.Reducer.stdDev());
88 var cv = mean.divide(stdDev).rename(['B5_cv','B7_cv','NDVI_cv','NDWI2130_cv','NDWI1640_cv','SAVI_cv']);
89 var kurt = annualSRC_md_tm_indices.select(['B5','B7','NDVI','NDWI2130','NDWI1640','SAVI'])
90 .reduce(ee.Reducer.kurtosis());
91 var skew = annualSRC_md_tm_indices.select(['B5','B7','NDVI','NDWI2130','NDWI1640','SAVI'])
92 .reduce(ee.Reducer.skew());
93
94 ////////////////////////////////////////////////////////////////// Temporal metrics /////
95 // Slope (outputs two bands: 'offset' (y-intercept) and 'scale' (slope))
96 var slopeB5 = annualSRC_md_tm_indices.select(['time','B5'])
97 .reduce(ee.Reducer.linearFit()).select('scale').rename('B5_slp');
98 var slopeB7 = annualSRC_md_tm_indices.select(['time','B7'])
99 .reduce(ee.Reducer.linearFit()).select('scale').rename('B7_slp');
100 var slopeNDVI = annualSRC_md_tm_indices.select(['time','NDVI'])
101 .reduce(ee.Reducer.linearFit()).select('scale').rename('NDVI_slp');
102 var slopeNDWI2130 = annualSRC_md_tm_indices.select(['time','NDWI2130'])
103 .reduce(ee.Reducer.linearFit()).select('scale').rename('NDWI2130_slp');
104 var slopeNDWI1640 = annualSRC_md_tm_indices.select(['time','NDWI1640'])
105 .reduce(ee.Reducer.linearFit()).select('scale').rename('NDWI1640_slp');
106 var slopeSAVI = annualSRC_md_tm_indices.select(['time','SAVI'])
107 .reduce(ee.Reducer.linearFit()).select('scale').rename('SAVI_slp');
108
109 // Max-slope (maximum absolute linear regression slope of 5-year windows)
110 // Creates a moving 5-year window
111 var join = ee.Join.saveAll({//Returns a join that pairs each element from the annual SR collection
112   matchesKey: 'images' // with a group of matching elements from the 5-year window collection.
113 });
114 var diffFilter = ee.Filter.maxDifference({ // Establishes the 5-year filter
115   difference: 78892380000, // 2.5 years in milliseconds, the filter selects 2 years before and after the target year
116   leftField: 'system:time_start', // therefore, the extreme windows are made of only 3-4 years
117   rightField: 'system:time_start'
118 });
119 var fiveWindowJoin = join.apply({// Collection in which each image is associated with the 5-year window images (see properties:
120   primary: annualSRC_md_tm_indices,
121   secondary: annualSRC_md_tm_indices,
122   condition: diffFilter
123 });
124 // Function for calculating the absolute maximum slope over 5-year moving windows
125 var maxAbsMovSlope = function (invar, outvar){
126   // Calculates slopes over 5-year windows
127   var mvSlp = ee.ImageCollection(fiveWindowJoin.map(function(image) {
128     var annualSRC_md_tm_indices = ee.ImageCollection.fromImages(image.get('images'));
129     return ee.Image(image).addBands(annualSRC_md_tm_indices.select(['time',invar])
130 .reduce(ee.Reducer.linearFit())));
131   }));
132   // Gets rid of the extreme windows, which are made of only 3 or 4 years
133   var range1 = mvSlp.reduceColumns(ee.Reducer.minMax(), ["system:time_start"]); //Gets the date range of images in the collection
134   var filter1 = mvSlp.filter(ee.Filter.date(range1.get('min')).not());
135   var filter2 = filter1.filter(ee.Filter.date(range1.get('max')).not());
136   var range2 = filter2.reduceColumns(ee.Reducer.minMax(), ["system:time_start"]);
137   var filter3 = filter2.filter(ee.Filter.date(range2.get('min')).not());
138   var filter4 = filter3.filter(ee.Filter.date(range2.get('max')).not()); // Print this to see all 5-yr window slopes
139   // Converts slope values to absolute numbers
140   var abs = function (image) {return image.select('scale').abs()};
141   // Extracts the maximum absolute slope
142   return filter4.map(abs).reduce(ee.Reducer.max()).rename(outvar);
143 };
144 // Applies the absolute moving slope function to all variables
145 var mxSlpB5 = maxAbsMovSlope('B5', 'B5_mxSlp');
146 var mxSlpB7 = maxAbsMovSlope('B7', 'B7_mxSlp');
147 var mxSlpNDVI = maxAbsMovSlope('NDVI', 'NDVI_mxSlp');
148 var mxSlpNDWI2130 = maxAbsMovSlope('NDWI2130', 'NDWI2130_mxSlp');
149 var mxSlpNDWI1640 = maxAbsMovSlope('NDWI1640', 'NDWI1640_mxSlp');
150 var mxSlpSAVI = maxAbsMovSlope('SAVI', 'SAVI_mxSlp');
151
152 //Value at last year
153 var rangeDates = annualSRC_md_tm_indices.reduceColumns(ee.Reducer.minMax(), ["system:time_start"]);
154 var lastYearFil = annualSRC_md_tm_indices.filter(ee.Filter.date(rangeDates.get('max')));
155 var lastYear = lastYearFil.select(['B5','B7','NDVI','NDWI2130','NDWI1640','SAVI']).toBands();
156 var lastYearVal = lastYear.rename(['B5_last','B7_last','NDVI_last','NDWI2130_last','NDWI1640_last','SAVI_last']);
157
158 ////////////////////////////////////////////////////////////////// All metrics /////
159 // Creates a collection from all metrics
160 var metricsCollection = ee.ImageCollection([min,max,range,mean,stdDev,cv,kurt,skew,
161 slopeB5,slopeB7,slopeNDVI,slopeNDWI2130,slopeNDWI1640,slopeSAVI,
162 mxSlpB5,mxSlpB7,mxSlpNDVI,mxSlpNDWI2130,mxSlpNDWI1640,mxSlpSAVI,lastYearVal]);
163
164 // Converts the metrics collection to a single multi-band image
165 var metrics= metricsCollection.toBands();
166 return metrics;
167 };
168 //////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

```

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

```

173 ////////////////////////////////////////////////////////////////// Masking: delimitation of forested areas //////////////////////////////////////////////////////////////////
174
175 // Selects areas that were forest at least 3 consecutive years during the study period///
176 var rename = function (image){// Funcion for assigning the same name to all bands
177   return image.select(0).rename("b1");
178 }
179 var mapBiomass_r= mapBiomass.map(rename);
180 ee.Filter.gte('year', startYearT,
181 ee.Filter.lte('year', endYearT));
182 // Function for detecting pixels covered by forest
183 //storage.googleapis.com/mapbiomas-public/RAISG/COLECAO2/LEGENDA/codigo_de_la_leyenda_coleccion-2.pdf
184 //forest is 3 or 6 in the entire Acre-Ucayali region during all years
185 var forest = function (image) {
186   return image.eq(3).or(image.eq(6)).rename("forest").copyProperties(image, ['year']);
187 }
188 var forestCol = mapBiomass.map(forest);
189 // Function for adding a time band that will be needed to calculate land covers per year
190 var createTimeBandMask = function(image) {
191   var time = image.metadata('year').rename('time');
192   return image.addBands(time);
193 };
194 var forestCol_tm = forestCol.map(createTimeBandMask);
195
196 // Forest in three consecutive years
197 // Creates a moving 3-year window
198 var join1 = ee.Join.saveAll({matchesKey: 'images'});
199 var difffilter1 = ee.Filter.maxDifference({difference: 1.5, leftField: 'year', rightField: 'year'});
200 var threeWindowJoin = join1.apply({primary: forestCol_tm, secondary: forestCol_tm, condition: difffilter1});
201
202 // Function for detecting forest cover over 3-year moving windows
203 var forest3years = function (invar, outvar){
204   // Calculates sum over 3-year windows
205   var f3y = ee.ImageCollection(threeWindowJoin.map(function(image) {
206     var forestCol_tm = ee.ImageCollection.fromImages(image.get('images'));
207     return ee.Image(image).addBands(forestCol_tm .select(["time",invar])
208       .reduce(ee.Reducer.sum()));
209   }));
210   // Extracts the maximum sum
211   return f3y.reduce(ee.Reducer.max());
212 };
213
214 // Applies the "detecting forest cover over 3-year" function to forest presence per year
215 var MaxForest3years = forest3years('forest', 'forest_3years').select("forest_sum_max");
216 // Gets the forest mask (areas that were forest at least three consecutive years)
217 var forestMask= MaxForest3years.gte(3);
218

```

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

```

218 ////////////////////////////////////////////////////////////////// Classificador random forest //////////////////////////////////////////////////////////////////
219
220 // *Crea una colección con todas las métricas con base en los años del muestreo
221 var metricsTraining = metricsGral(startYearT, endYearT,startDayB, endDayB, aoiB, maskTheseB);
222 print("Métricas muestreo", metricsTraining);
223
224 // **Crea una colección con todas las métricas con base en los años para la clasificación
225 var metricsClassification = metricsGral(startYearC, endYearC,startDayB, endDayB, aoiB, maskTheseB);
226 print("Métricas clasificación", metricsClassification);
227
228 // Sobrepone los puntos de entrenamiento con las métricas*
229 var collectMetrics = function (feature){// usar una funcion permite a GEE trabajar con más puntos antes de exceder su capacidad
230   return metricsTraining.sampleRegions({collection:feature,
231   properties: ["class"],
232   scale: 30,
233   tileSize: 2});
234 };
235 var trainingCol = trainingPts.map(collectMetrics);
236 var trainingPoints = trainingCol.flatten();
237
238 // Crea un clasificador random forest con 500 árboles y lo entrena
239 var classifier = ee.Classifier.smileRandomForest({numberOfTrees: 50})
240 .train({features: trainingPoints, classProperty:'class',
241 inputProperties:['0_B5_min','0_B7_min','0_NDWI_min','0_NDWI2130_min','0_NDWI1640_min','0_SAVI_min',
242 '1_B5_max','1_B7_max','1_NDWI_max','1_NDWI2130_max','1_NDWI1640_max','1_SAVI_max',
243 '2_B5_range','2_B7_range','2_NDWI_range','2_NDWI2130_range','2_NDWI1640_range','2_SAVI_range',
244 '3_B5_mean','3_B7_mean','3_NDWI_mean','3_NDWI2130_mean','3_NDWI1640_mean','3_SAVI_mean',
245 '4_B5_stdDev','4_B7_stdDev','4_NDWI_stdDev','4_NDWI2130_stdDev','4_NDWI1640_stdDev','4_SAVI_stdDev',
246 '5_B5_cv','5_B7_cv','5_NDWI2130_cv','5_NDWI1640_cv','5_SAVI_cv',
247 '6_B5_kurtosis','6_B7_kurtosis','6_NDWI_kurtosis','6_NDWI2130_kurtosis','6_NDWI1640_kurtosis','6_SAVI_kurtosis',
248 '7_B5_skew','7_B7_skew','7_NDWI_skew','7_NDWI2130_skew','7_NDWI1640_skew','7_SAVI_skew',
249 '8_B5_slp','9_B7_slp','10_NDWI_slp','11_NDWI2130_slp','12_NDWI1640_slp','13_SAVI_slp',
250 '14_B5_mxSlp','15_B7_mxSlp','16_NDWI_mxSlp','17_NDWI2130_mxSlp','18_NDWI1640_mxSlp','19_SAVI_mxSlp',
251 '20_B5_last','20_B7_last','20_NDWI1640_last','20_NDWI2130_last','20_SAVI_last']
252 });
253
254
255 // Clasifica las métricas deseadas**
256 var metricsMask = metricsClassification.mask(forestMask); // Enmascara las areas no forestales
257 var classified = metricsMask.classify(classifier); //No bosque (0), bosque intacto (1), bosque degradado (2), áreas deforestadas (3)
258 print("Mapa clasificado");
259

```

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

```

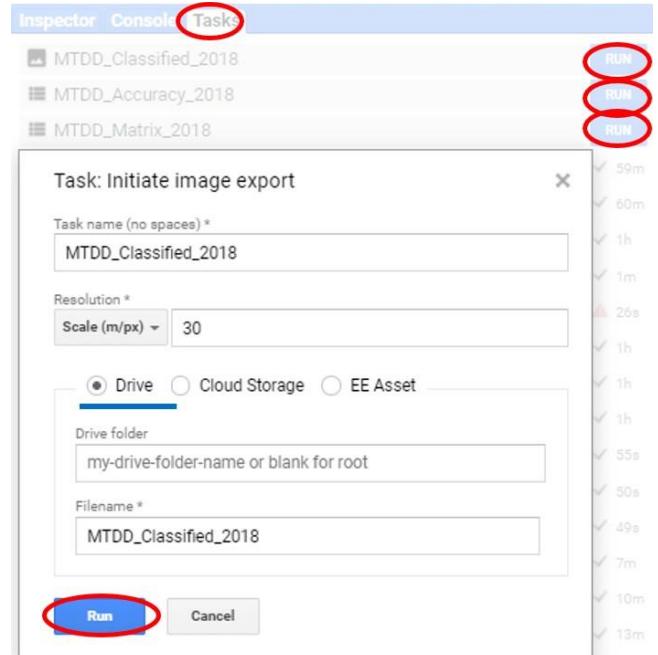
269 ///////////////////////////////////////////////////////////////////Post-classification filtering to reduce noise/////////////////////////////////////////////////////////////////
270
271 // Applies a 90m X 90m filter (default options)
272 var classified_fil = classified.focal_mode().mask(forestMask);
273 // This is only for visualization purposes. The reproject is sufficient
274 // to force the computation to occur at native scale.
275 var classified_fil_vis = classified_fil.reproject('EPSG:4326', null, 30);
276 // Displays the results
277 //Map.centerObject(aoi,10);
278 //Map.addLayer(classified_fil_vis.clip(aoiC),
279 //    {"opacity":1,"bands":["classification"],"min":0,"max":3,"palette":[ "#ffffff", "#7dff5c", "#0a9b15", "#ff5544"]},
280 //    "Classification");
281
282 ////////////////////////////////////////////////////////////////// Validation //////////////////////////////////////////////////////////////////
283
284 // Overlays the validation points over the metrics
285 var validationCol = validationPts.map(collectMetrics);
286 var validationPoints = validationCol.flatten();
287 //print("Metrics collected in validation points",validationPoints);
288
289 // Classify the validation data
290 var validated = validationPoints.classify(classifier);
291
292 // Gets a confusion matrix representing expected accuracy
293 var valAccuracy = validated.errorMatrix("class","classification");
294 //print('Validation error matrix: ', valAccuracy);
295 //print('Validation overall accuracy: ', valAccuracy.accuracy());
296
297 ////////////////////////////////////////////////////////////////// Results exportation //////////////////////////////////////////////////////////////////
298
299 // Exports the image classified into Intact, Degraded forest, and deforested
300 Export.image.toDrive({image:classified_fil,
301   description: "MTDD_Classified_" + outName ,
302   fileNamePrefix: "MTDD_Classified_" + outName,
303   region:aoiI,
304   scale:30,
305   maxPixels:60000000
306 });
307
308 // Exports the confusion matrix
309 var toExportMatrix = ee.Feature(null, {matrix: valAccuracy.array()});
310 Export.table.toDrive({
311   collection: ee.FeatureCollection(toExportMatrix),
312   description: "MTDD_Matrix_" + outName,
313   fileFormat: 'CSV'
314 });
315
316 // Exports the validation overall accuracy
317 var toExportAccuracy = ee.Feature(null, {matrix: valAccuracy.accuracy()});
318 Export.table.toDrive({
319   collection: ee.FeatureCollection(toExportAccuracy),
320   description: "MTDD_Accuracy_" + outName,
321   fileFormat: 'CSV'
322 });
323
324
325
326 print("All done");

```

Applies a filter to reduce noise

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

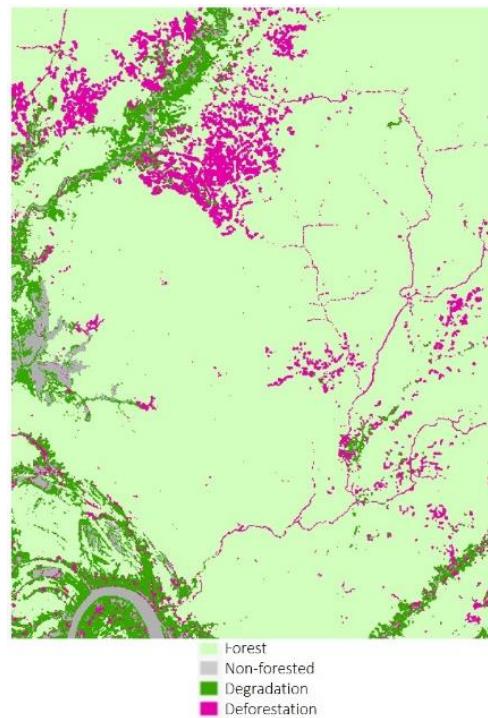
Exports the classified image as a raster, and the confusion matrix and overall accuracy as a csv file



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>
- MapBiomass, 2020. MapBiomass 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>