High Carbon Stock Approach and Analysis to Southeast Asia Regions

Hanyong Xu
Geospatial Software Design, LARP 743
Instructor: Dr. Dana Tomlin
December 16th, 2019

Contents

1. Introduction

1.1 Background

High Carbon Stock areas (HCS) refer to natural forest areas, especially for tropical forests, that stores excellent amount of carbon stock. They are crucial in terms of fostering an environment for biodiversity and balancing the eco-system. In recent years, with the ever-increasing industrialization and economic development, deforestation or forest fragmentation is becoming a more and more severe problem.

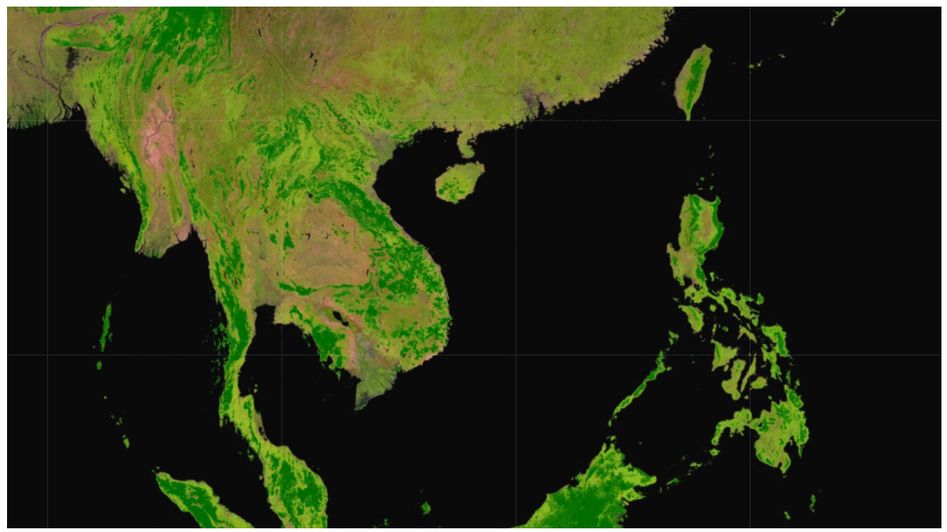
While there is broad consensus that these HCS areas need to be protected, other conflicts of interests made the process much difficult to execute. Conserving a patch of forest may require significant cost in terms of time and money. Furthermore, the cost of avoiding economic development, social construction, community enlivenment, or agricultural expansion in the region of a degrading forest might outweigh the benefit of maintaining the forest.

1.2 Purpose and Goal

To solve the dilemma, HCSA (High Carbon Stock Approach), published by the HCS Approach Steering Group will be used to find the viable forest patches that are of the most conservation importance through a series of patch analyses. In this way, the best results may be better achieved for all stakeholders.

1.3 Study Area

Forest fragmentation is most problematic in South or Central America and Southeast Asia. While the HCSA serves as a tool that can be applied to most areas, this report will be focusing on the Southeast Asian region, and several sites within the region will be identified for more specific analysis.



https://www.flickr.com/photos/10565417@N03/6246541918
http://earthenginepartners.appspot.com/science-2013-global-forest?hl=en&llbox=46.28%2C-8.8%2C159.32%2C51.3&t=ROADMAP&layers=layer0%2Clayer1%3A100%2C14%3A100%2Clayer9%3A54%2C6%2Clayer12%2Clayer8

2. Methods

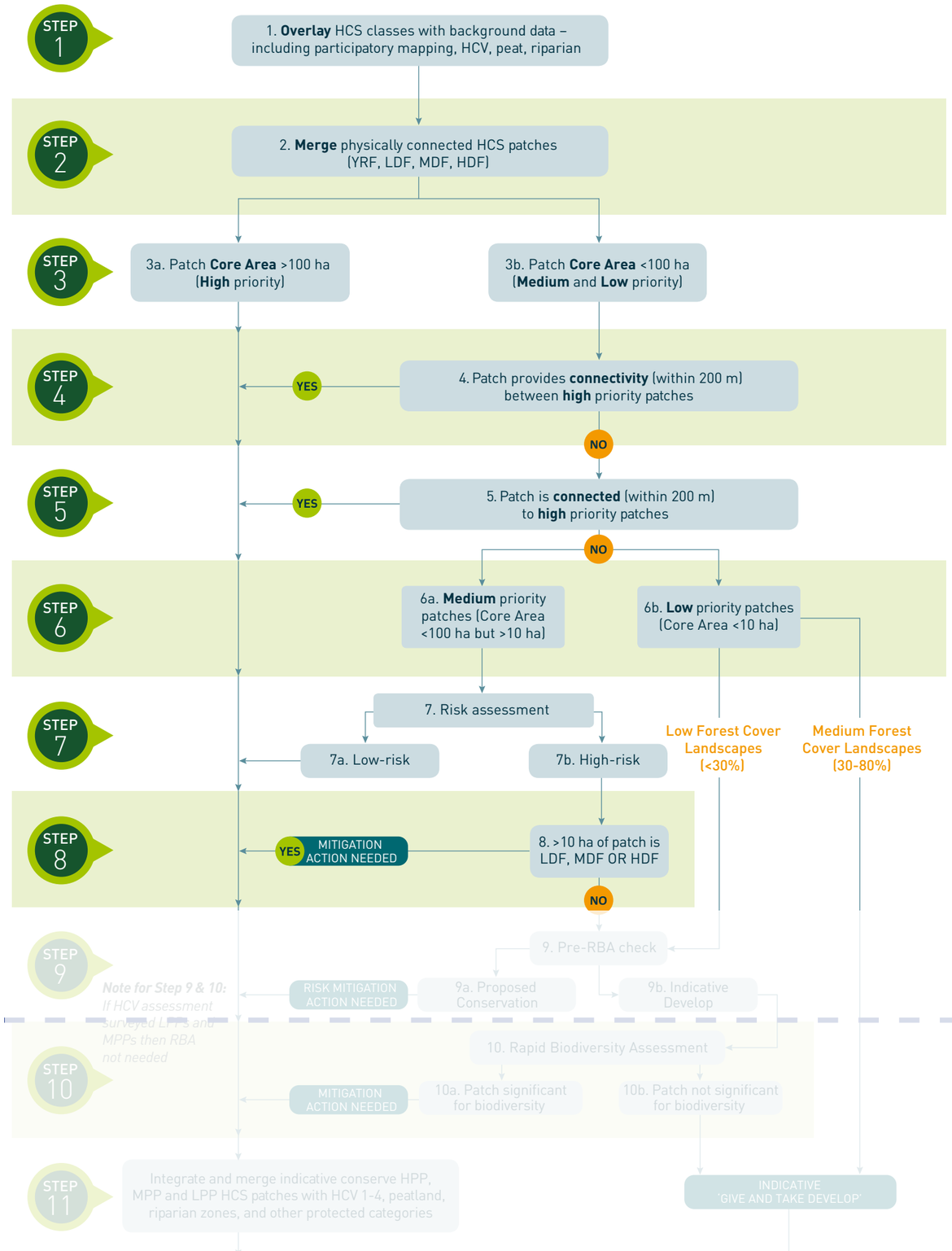
2.1 HCSA

The main framework of this project is closely following the workflow outlined by the HCSA tool kit Module 5. A decision tree of this workflow is demonstrated on the right. A summary of the major steps are presented below. Note that this project will only analyze until step 9. Analyses beyond require more in depth data and integration of the community support, thus, although they are briefly summarized here, they will not be covered in this project. The original document can be referred for more details.

1. Core area. The core area is identified by buffering inwards 100 m for each patch since edge forests are more easily affected by the exterior conditions. After we get core areas, the size of them are compared. Large patches tend to hold more species and allow more balanced ecosystems while small patches tend to be fragmented, isolated, and deteriorate more quickly. Thus, larger core area are given higher priority. Specifically, in HCSA, the priority is divided into three categories, < 10 ha, 10 – 100 ha, > 100 ha.

2. Connectivity. Connected corridors allow movement of diverse species. Therefore, both existing connected forests and the smaller patches that can serve as the stepping stone between the larger patches have conservation values. Specifically, patches with 200 m distance is considered as connected.

3. Risk levels. The median priority and lower priority patches are examined in terms of risk levels, which are identified by looking at the patch's proximity to human activities and constructions, including "public roads, settlements, waterways used for navigation/ transportation, and other anthropogenic activities such as mining, logging, or plantations" (30).



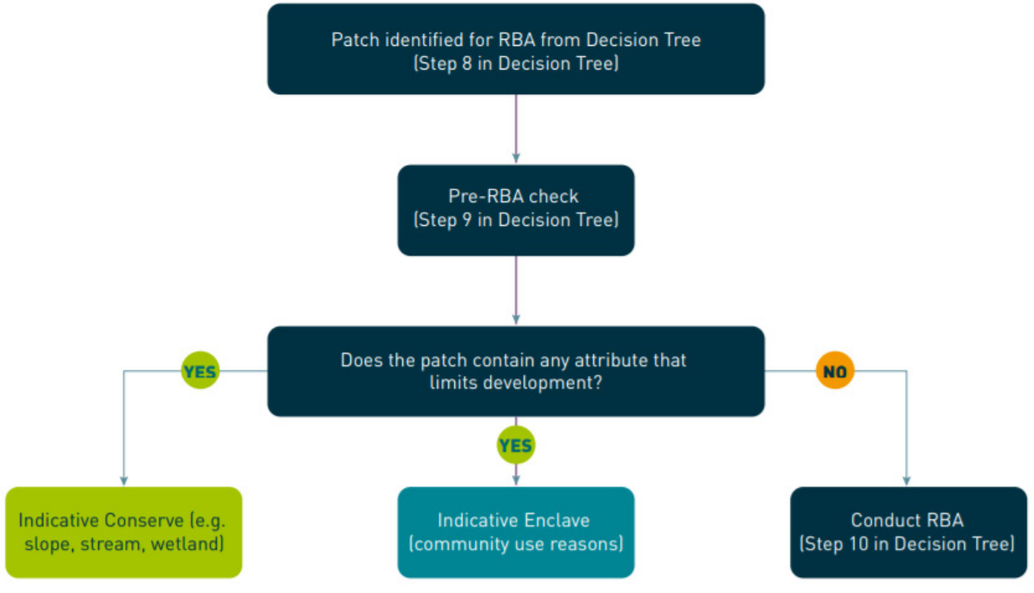
source: http://highcarbonstock.org/wp-content/uploads/2018/04/Def-HCSA-Module-5-16_04_2018_Web.pdf, page 25.

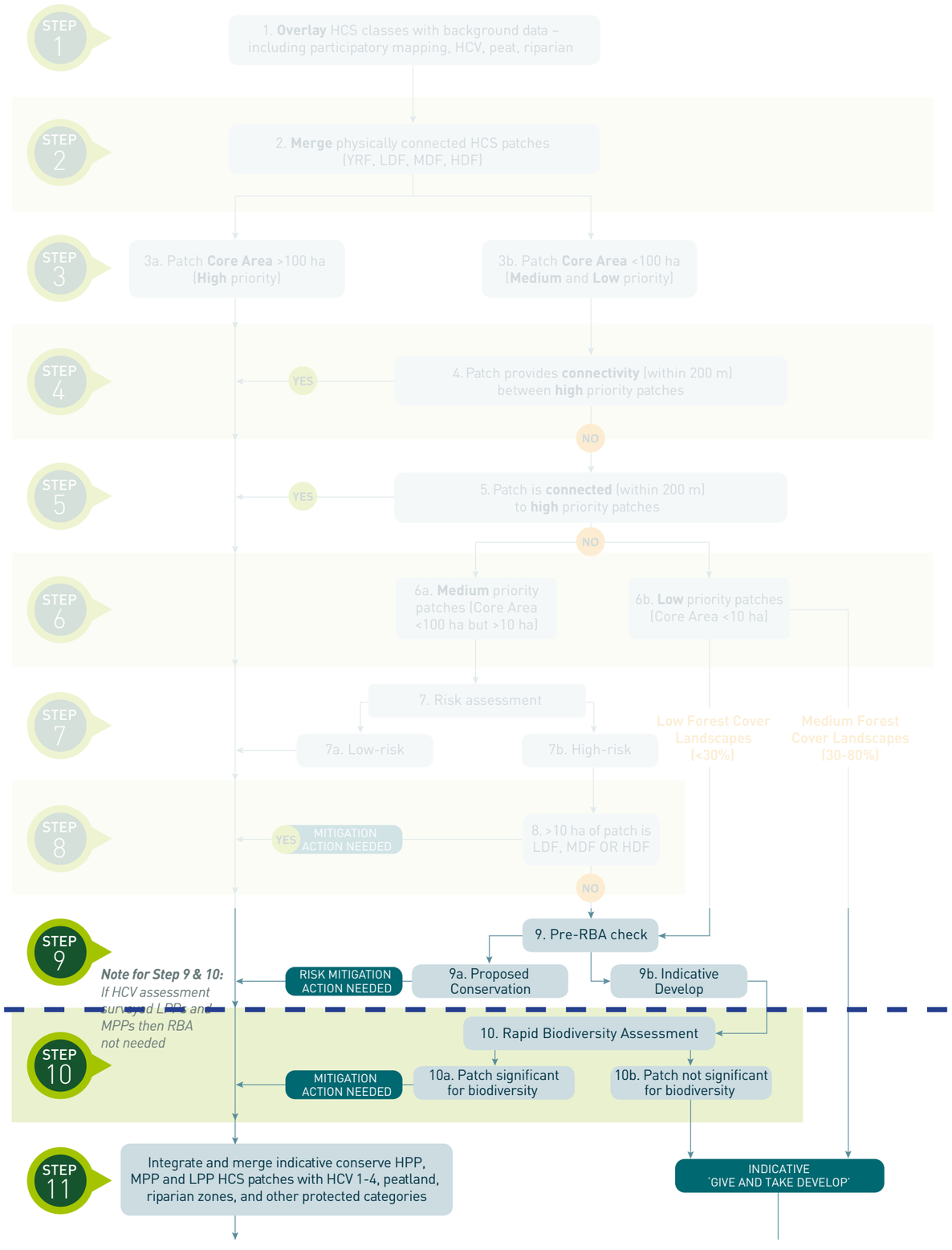
2. Methods

4. Pre-RBA check. In order to quickly filter out the layers that require RBA (Rapid Biodiversity Assessment) or not, a brief analysis is done before that. This includes checking whether the area is in close proximity with the streams and sloped areas. The decision tree provided by HCSA is shown below. I will only focus on the natural factors for pre-RBA check. The project will end here, although the HCSA continues to go on. In stead, I will categorize the patches into three categories: develop, conserve, and conserve with mitigation.

5. Habitat quality. The lower priority forest patches require an RBA to detect its importance before suggesting them for development.

6. Integrated conservation and land use planning. This step includes merging the forest patches with the HCV (high conservation value) data and suggesting corridors between patches.





source: http://highcarbonstock.org/wp-content/uploads/2018/04/Def-HCSA-Module-5-16_04_2018_Web.pdf, page 25 and 39.

2. Methods

2.2 Data

2.2.1 Data Collection

* A full citation of the datasets are appendixed to the end of this report.

From Earth Engine Data Catalog:

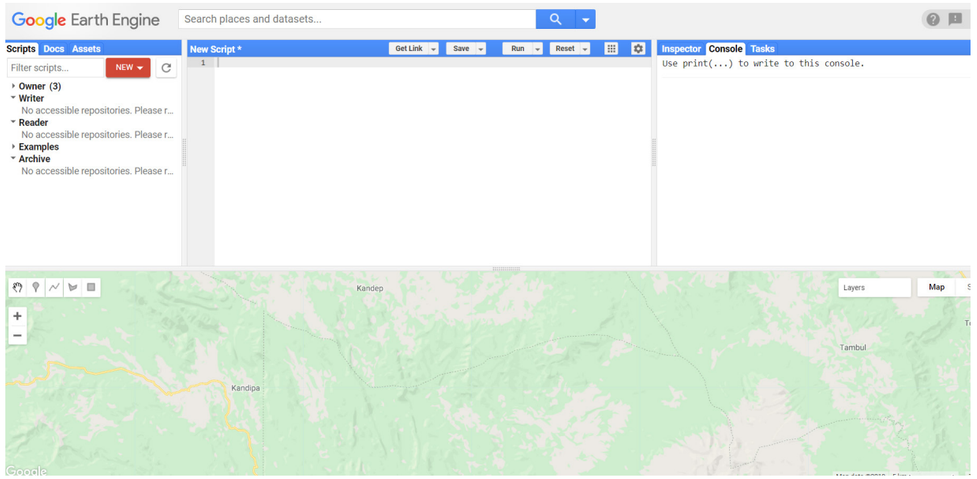
- WHRC: Pantropical National Level Carbon Stock Dataset
- GLCF: Landsat Tree Cover Continuous Fields
- DMSP OLS: Nighttime Lights Time Series Version 4, Defense Meteorological Program Operational Linescan System
- GlobCover: Global Land Cover Map
- JRC: Yearly Water Classification History, v1.1
- GMTED2010: Global Multi-resolution Terrain Elevation Data 2010

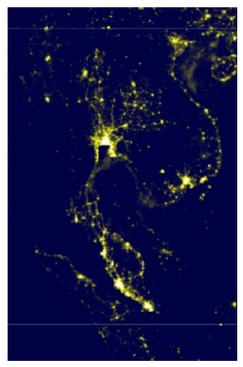
Found elsewhere and uploaded to GEE:

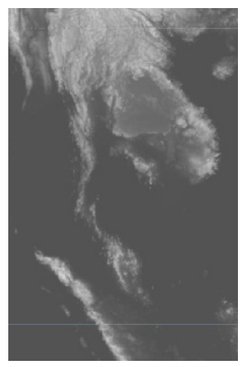
- WRI: The Universal Mill List (UML)
- GISTA The Thai Space Agency: Thailand Roads (subsetted in ArcGIS before upload)

2.3 Tools

GEE (Google Earth Engine) was be used to conduct all the analysis. Due to the fact that GEE can take inputs from landsat images from its own repository, this analysis can be easily replicated in other regions of the world.

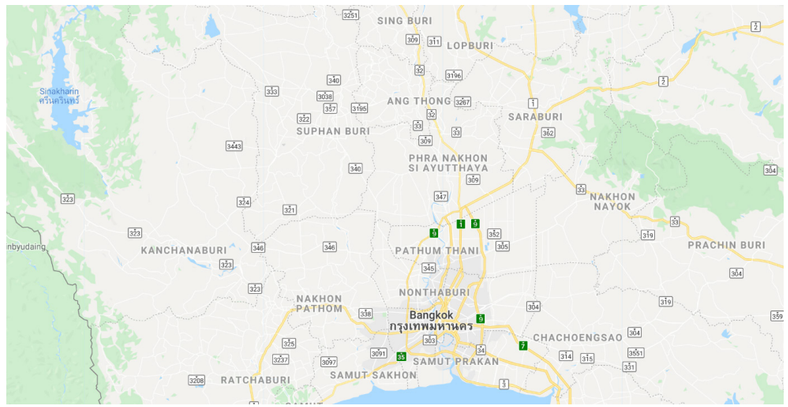






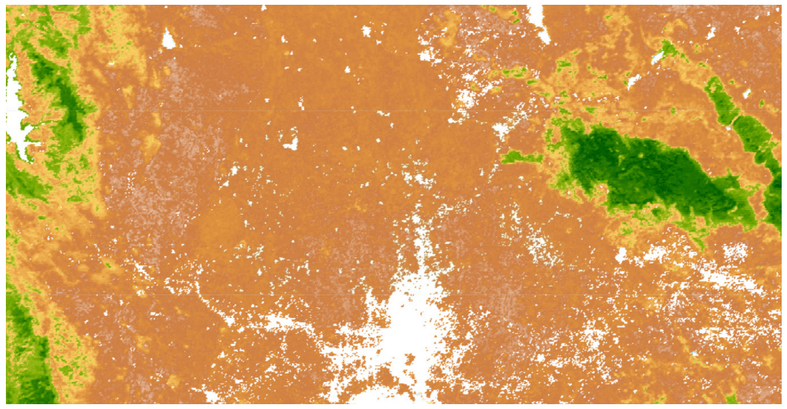
3. Analysis

3.1 Classify High/Medium/Low Priority Patches



All analysis below are focusing on the region near Bangkok, Thailand.

```
var studyarea = ee.Geometry.Rectangle([99,13.5,102,15]); // near Bangkok
```

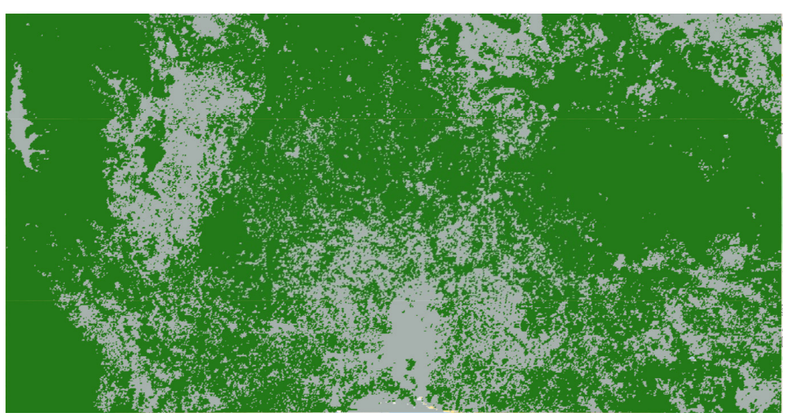


Import carbon stock data

WHRC: Pantropical National Level Carbon Stock Dataset

This dataset describes the above-ground carbon stock for tropical countries. Value ranges from 0 to 503 Mg/Ha. Resolution is at 500 m.

```
// Importa Woody Biomass Data
var dataset = ee.Image('WHRC/biomass/tropical');
var CarbonStock = dataset.select('Mg');
var visParams = {
  min: 0.0,
  max: 503.0,
  palette: [
    'FFFFFF', 'CE7E45', 'DF923D', 'F1B555', 'FCD163', '99B718', '74A901',
    '66A000', '529400', '3E8601', '207401', '056201', '004C00', '023B01',
    '012E01', '011D01', '011301'
  ],
};
Map.centerObject(studyarea, 11);
var CarbonStock_c = CarbonStock.clip(studyarea);
Map.addLayer(CarbonStock_c, visParams, 'Aboveground Live Woody Biomass', false);
```



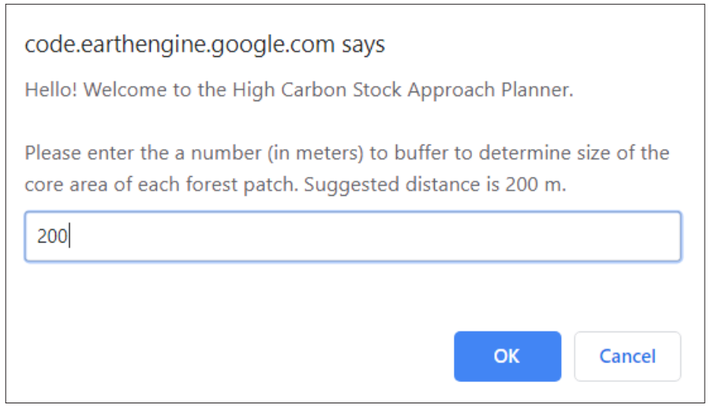
Identify High/Low Carbon Stock

As outlined in *Identifying High Carbon Stock (HCS) Forest for Protection* by Grennpeace, high carbon stock forests have carbon stock larger than 35 Mg/Ha.

```
var lcs = CarbonStock_c.lte(35);
var hcs = CarbonStock_c.gt(35);
Map.addLayer(lcs, {palette:['aedbf5','bab98c']}, 'Low Carbon Stock', false);
```

3. Analysis

3.1 Classify High/Medium/Low Priority Patches



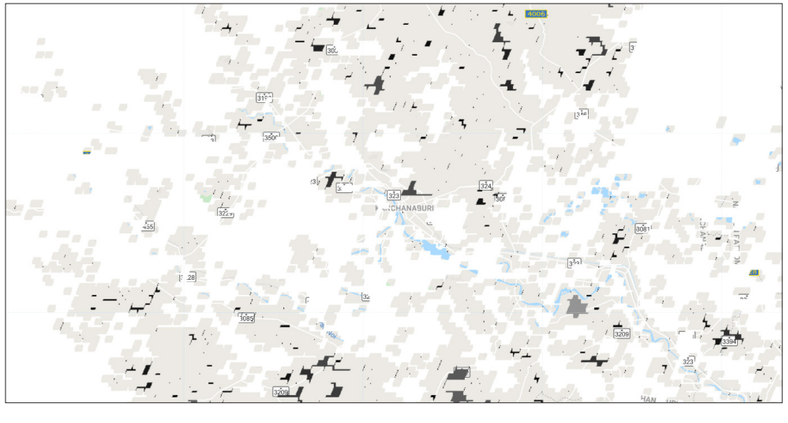
The user is able to define the buffer distance. Suggested distance is 200m.

```
// User Input
var response = prompt( "Hello! Welcome to the High Carbon Stock Approach Planner. \n"
+ "\nPlease enter the a number (in meters) to buffer to determine size of the core area "
+ "of each forest patch. Suggested distance is 200 m.");
var response = ee.Number.parse(response);
```

Identify Core Patches

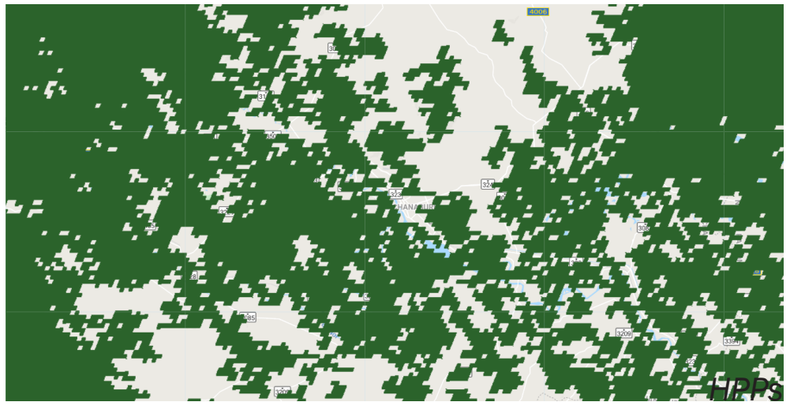
Screen shots are zoomed in order to show the small-scale changes. Buffer inwards user defined distance to find the core of each forest patch. The area of each unconnected patch is calculated using connectedPixelCount() and pixelArea(). The resulting map has each patch colored by the area size.

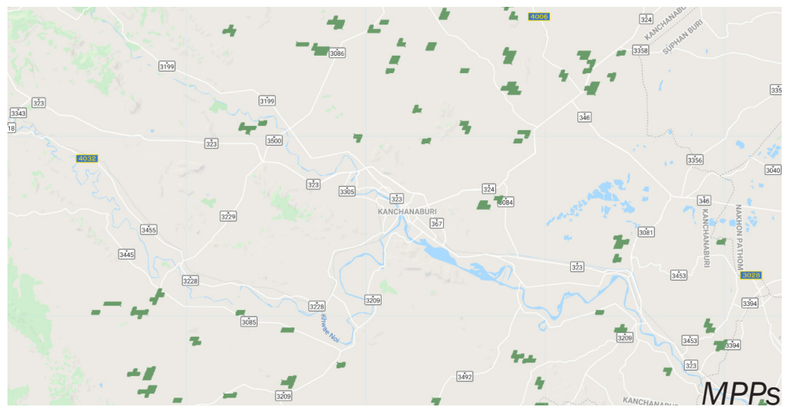
```
var bufferlcs = lcs
    .cumulativeCost({
      source: lcs,
      maxDistance: response,
    }).lt(35);

// Get core hcs
var raster1 = ee.Image(1);
var bufferlcs = bufferlcs.unmask(-1);
var core_hcs = raster1.subtract(bufferlcs).clip(studyarea);
var core_hcs = core_hcs.remap([0,1,2],[0,0,1]);
var core_hcs = core_hcs.mask(core_hcs);
Map.addLayer(core_hcs, {palette:['000000','4a8762']}, 'Core HCS', false);

// Compute the number of pixels in each patch.
var patchsize = core_hcs.connectedPixelCount(1024, false);
var patcharea = patchsize.multiply(ee.Image.pixelArea());
Map.addLayer(patchsize, {}, 'patch size', false);
Map.addLayer(patcharea, {}, 'patch area', false);
```

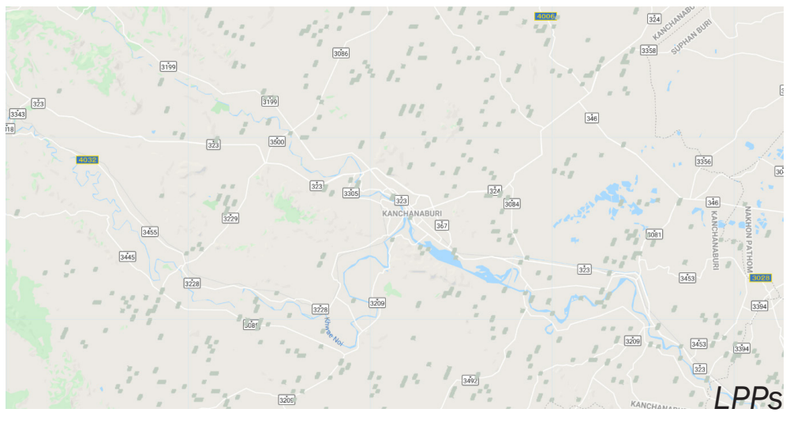


3. Analysis

3.1 Classify High/Medium/Low Priority Patches







Filter High/Med/Low Priorities

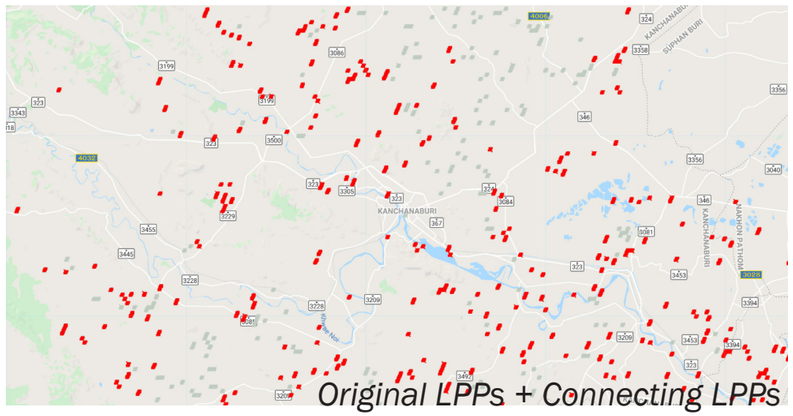
Patches are filtered according to HCSA. Core areas of greater than 100 ha, between 10 - 100 ha, and less than 10 ha are labeled as high, medium, low priority forest patches (HPPs, MPPs, LPPs). In the screenshots on the left, the HPPs, MPPs, and LPPs are respectively the dark green, normal green, and light green colors.

```
// Filter high, medium, low priority patches
var HPP = patcharea.gt(1000000); // greater than 100 ha
var MPP = patcharea.lte(1000000).and(patcharea.gte(100000)); // 10-100 ha
var LPP = patcharea.lt(100000); // less than 10 ha

// Buffer around the core HPPs, MPPs, LPPs to get original carbon stock area
var HPP_ = HPP.mask(hcs).distance(ee.Kernel.euclidean(response, 'meters')).gt(0).remap([0,1],[1,1]);
var MPP_ = MPP.mask(hcs).distance(ee.Kernel.euclidean(response, 'meters')).gt(0).remap([0,1],[1,1]);
var LPP_ = LPP.mask(hcs).distance(ee.Kernel.euclidean(response, 'meters')).gt(0).remap([0,1],[1,1]);
Map.addLayer(HPP_, {palette:['000000', '2b632b']}, 'High Priority Patch', false);
Map.addLayer(MPP_, {palette:['000000', '6a9c6a']}, 'Medium Priority Patch', false);
Map.addLayer(LPP_, {palette:['000000', 'c0ccc0']}, 'Low Priority Patch', false);

var HPP_ = HPP_.unmask(0);
var MPP_ = MPP_.unmask(0);
var LPP_ = LPP_.unmask(0);
```

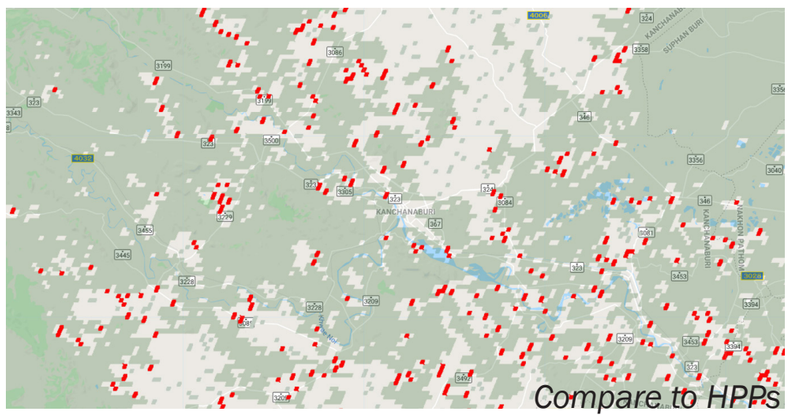
3. Analysis

3.2 Check Connectivity between Patches

Original LPPs



Original LPPs + Connecting LPPs



Compare to HPPs

Connectivity between LPPs and HPPs

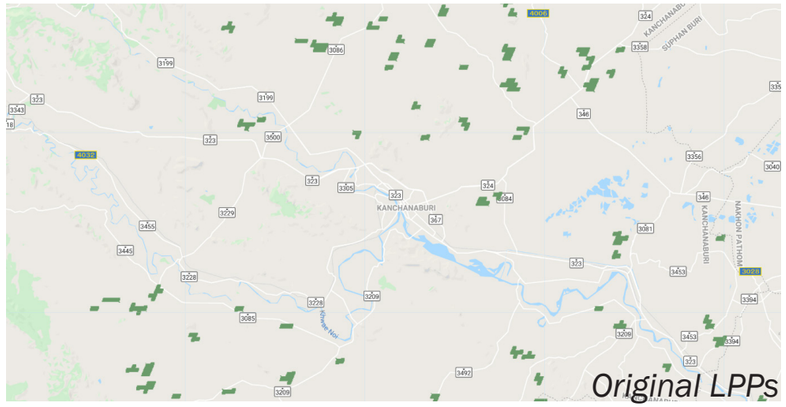
LPPs that are within 200 meters distance with HPPs are subject to conservation in order to establish connectivity between forests and foster biodiversity. LPPs that are too far from HPPs will be checked later on for either “Pre-Rapid Biodiversity Check” or “Indicative to develop”.

Checking connectivity is a major challenge in this project. I approached this problem by adding the buffered LPPs’ layer and the HPPs’ layer, and compare the results with the original LPPs’ layer using connectedComponents() and Reducer functions. On the screenshot on the left, the red represents the LPPs that are within 200 meters distance to HPPs.

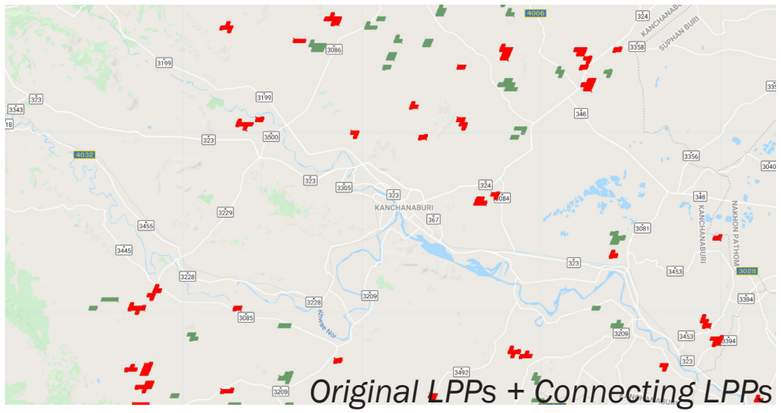
```
// Isolate Low Priority Patches that connect to the High Priority Patches within 200 meters
var distL = LPP_.distance(ee.Kernel.euclidean(200, 'meters')).gt(0).remap([0,1],[1,1]).unmask(0);
//Map.addLayer(distL, {}, 'Distance to Low Priority Patch', false);
var L_H = distL.add(HPP_).mask(LPP_).unmask(0);
Map.addLayer(L_H, {}, 'L+H', false);
var low_fsum0 = LPP_.addBands(LPP_);
var low_fsum = L_H.addBands(LPP_);
var low_fsum0 = low_fsum0.reduceConnectedComponents(TheREDUCER, 'remapped_1', 256);
var low_fsum = low_fsum.reduceConnectedComponents(TheREDUCER, 'remapped_1', 256);
Map.addLayer(low_fsum0, {}, 'L+H focal sum previous', false);
Map.addLayer(low_fsum, {}, 'L+H focal sum after', false);
var low_connect = low_fsum.subtract(low_fsum0).gt(0);
var low_connect = low_connect.mask(low_connect);
print("low connecting", low_connect);
Map.addLayer(low_connect, {palette: "fc0303"}, 'Low Connecting', false);
```

3. Analysis

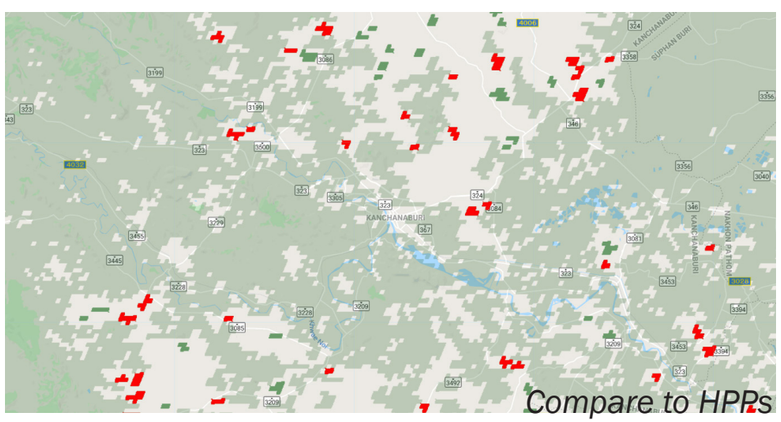
3.2 Check Connectivity between Patches



Original LPPs



Original LPPs + Connecting LPPs



Compare to HPPs

Connectivity between MPPs and HPPs
Similaryly, MPPs that are within 200 meters distance with HPPs are subject to conservation in order to establish connectivity between forests and foster biodiversity. Unconnected MPPs will move on for Risk Assessments.

The method used to check connectivity is the same as the previous page. On the screenshot on the left, the red represents the MPPs that are within 200 meters distance to HPPs.

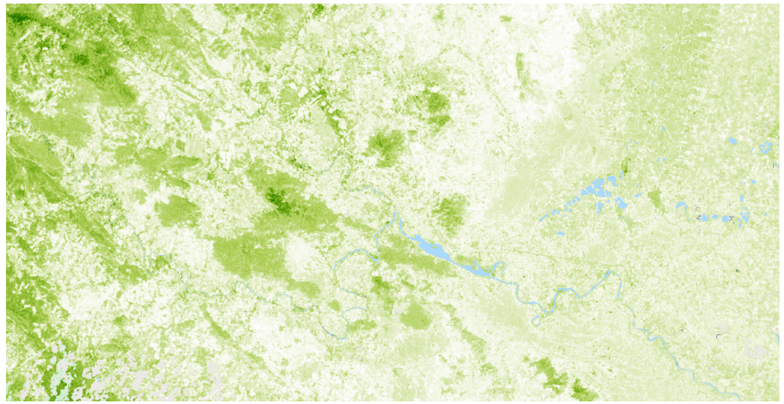
```
var distM = MPP_.distance(ee.Kernel.euclidean(200, 'meters')).gt(0).remap([0,1],[1,1]).unmask(0);
//Map.addLayer(distM, {}, 'Distance to Medium Priority Patch', false);
var M_H = distM.add(HPP_).mask(MPP_).unmask(0);
Map.addLayer(M_H, {}, 'M+H', false);
var TheREDUCER = ee.Reducer.sum();
var med_fsum0 = MPP_.addBands(MPP_);
var med_fsum = M_H.addBands(MPP_);
var med_fsum0 = med_fsum0.reduceConnectedComponents(TheREDUCER, 'remapped_1', 256);
var med_fsum = med_fsum.reduceConnectedComponents(TheREDUCER, 'remapped_1', 256);
Map.addLayer(med_fsum0, {}, 'M+H focal sum previous', false);
Map.addLayer(med_fsum, {}, 'M+H focal sum after', false);
var med_connect = med_fsum.subtract(med_fsum0).gt(0);
var med_connect = med_connect.mask(med_connect);
Map.addLayer(med_connect, {palette: "fc0303"}, 'Median Connecting', false);
```

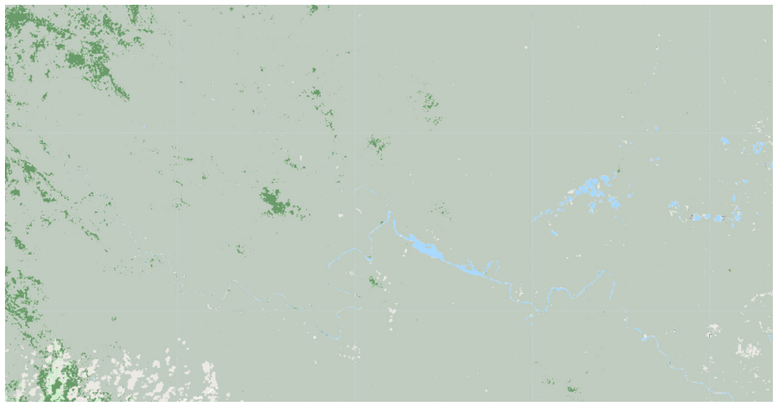
Finally, the patches that are in proximity with HPPs are merger with HPPs in the same layer.

```
// Merge Low and Med patches that can connect with High patches together
var HPP_1 = HPP_.add(low_connect.unmask(0)).add(med_connect.unmask(0)).gt(0);
Map.addLayer(HPP_1, {}, 'Old HPPs and New conserve from MPPs and LPPs', false);
```

3. Analysis

3.3.1 Check Unconnected LPPs





Load Forest Cover Dataset

GlobCover: Global Land Cover Map

The forest cover shows how much of each pixel is covered by forest. Value ranges from 0 to 100 percent. Resolution is 30 m.

```
// Import tree cover data and classify the 3 types of tree covers
var treeCanopyCover = ee.ImageCollection('GLCF/GLS_TCC').select('tree_canopy_cover')
    .filter(ee.Filter.date('2010-01-01', '2010-12-31'));
var treeCanopyCover = treeCanopyCover.mean();
var treeCanopyCoverVis = {
  min: 0.0,
  max: 100.0,
  palette: ['ffffff', 'afce56', '5f9c00', '0e6a00', '003800'],
};
Map.addLayer(treeCanopyCover, treeCanopyCoverVis, 'Tree Canopy Cover',false);
var area = ee.Image(treeCanopyCover).clip(studyarea);
```

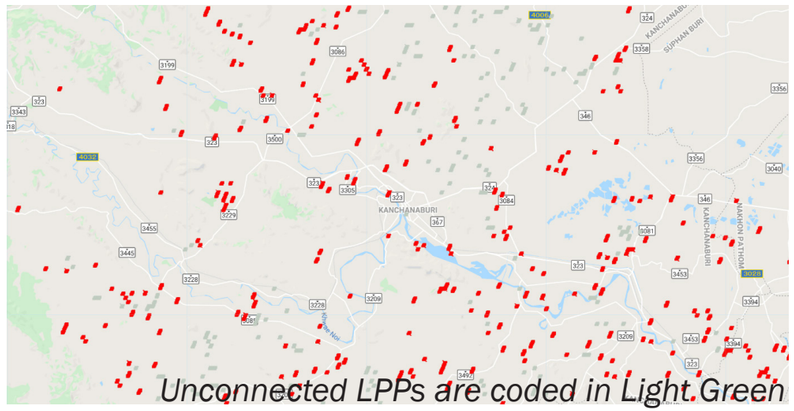
Reclassify the Forest Cover

According to HCSA, forests with cover rates of below 30%, 30% - 80%, and above 80% foster dramatically different environments. I reclassified the forest cover data into this three categories. Note that in this study region, only a small portion of the forests are medium and high cover.

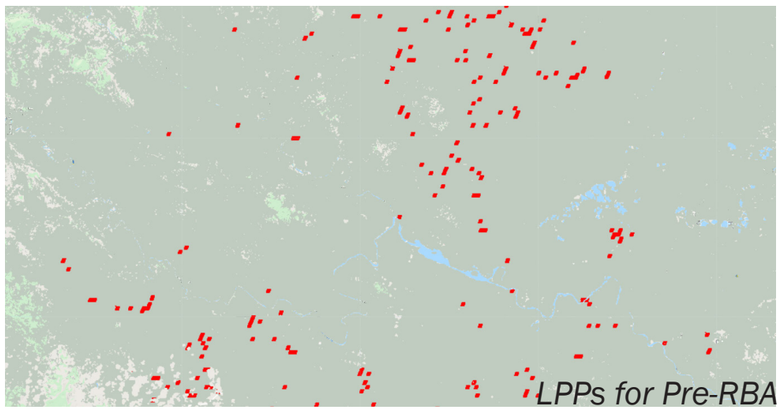
```
// classification into high/med/low forest cover
var area = ee.Image(treeCanopyCover);
var highcover = area.gt(80);
var highcover = highcover.mask(highcover);
var medcover = area.lte(80).and(area.gt(30));
var medcover = medcover.mask(medcover);
var lowcover = area.lte(30);
var lowcover = lowcover.mask(lowcover);
Map.addLayer(highcover, {palette: "2b632b"}, 'high forest cover', false);
Map.addLayer(medcover, {palette: "6a9c6a"}, 'med forest cover', false);
Map.addLayer(lowcover, {palette: "c0ccc0"}, 'low forest cover', false);
```

3. Analysis

3.3.1 Check Unconnected LPPs



Unconnected LPPs are coded in Light Green



LPPs for Pre-RBA

Label Unconnected LPPs

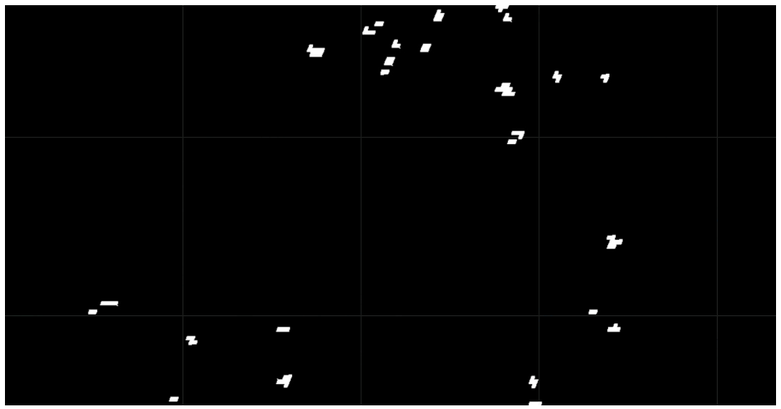
According to HCSA, unconnected LPPs that have a medium forest cover will be labeled as “Indicative develop” or “Give and Take” for community development. On ther other hand, unconnected LPPs that have a low forest cover will be subject to Pre Rapid Biodiversity Assessment (Pre-RBA) since they may be suitable to conserve for the biodiversity sake.

```
// Find LPPs that are subject to "give and take approach"
var give_take = LPP_.subtract(low_connect.unmask(0)).remap([-1,0,1],[0,0,1]).add(medcover.unmask(0)).eq(2);
var give_take = give_take.mask(give_take);
print("give and take", give_take);
Map.addLayer(give_take, {palette: "fc0303"}, 'Give and Take', false);

// Find LPPs that are subject to Pre-RBA check
var LPP_pre_RBA = LPP_.subtract(low_connect.unmask(0)).remap([-1,0,1],[0,0,1]).add(lowcover.unmask(0)).eq(2);
var LPP_pre_RBA = LPP_pre_RBA.mask(LPP_pre_RBA);
print("LPPs for pre RBA assessment", LPP_pre_RBA);
Map.addLayer(LPP_pre_RBA, {palette: "fc0303"}, 'LPPs for Pre-RBA', false);
```

Here, there is no suitable LPPs that are medium forest cover so screenshots only shows the LPPs for Pre-RBA. However, the code is written out above, and one can change the study region to perform the same analysis.

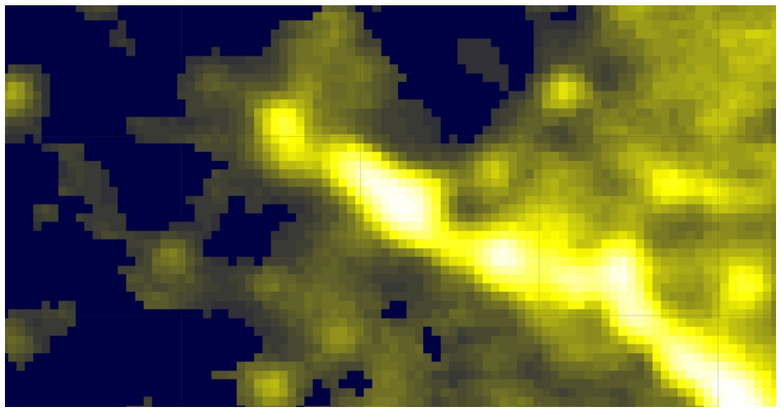
3. Analysis

3.3.2 Unconnected MPPs - Risk Assessments

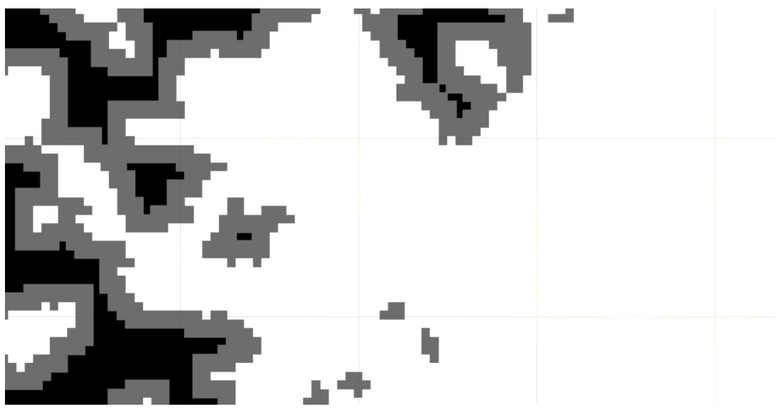


Isolate Unconnected MPPs
(white represents 1 and black represents 0 throughout this document)

```
// Find MPPs that are not conserved
var MPP_ra = MPP_.subtract(med_connect.unmask(0)).remap([-1,0,1],[0,0,1]);
print(MPP_ra);
Map.addLayer(MPP_ra, {}, 'MPPs for Risk Assessment', false);
```



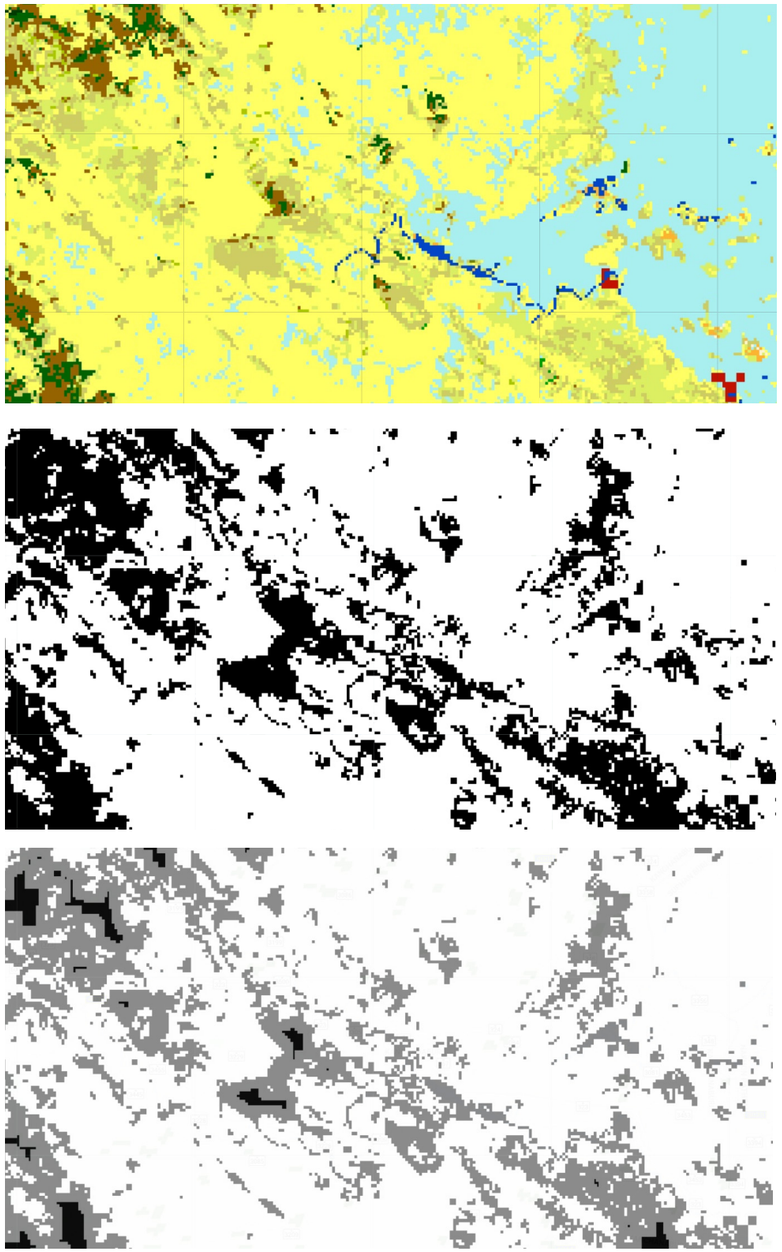
Risk Factors
1. Load Night Time Images
MSP OLS: Nighttime Lights Time Series Version 4, Defense Meteorological Program Operational Linescan System
This dataset describes the night light received from satelites. Resolution is at 30 arc seconds. Stable lights band is used to represent human settlements from cities, suburbs, and villages. The settlement layer is then buffered outwards for 2 km for later analysis.



```
// use light change to represent urban cities / human activities / settlements
// NightLighting in 2012
var lights = ee.Image( 'NOAA/DMSP-OLS/NIGHTTIME_LIGHTS/F182012').clip(studyarea).select("stable_lights");
var BandOneIMAGE = lights.expression( 'b("stable_lights")' ); // Band 1: Persistent Lighting
Map.addLayer( BandOneIMAGE, {min: 0, max: 63, palette: ['000044','ffff00','ffffff']}, 'persistnet lighting', false);
// Buffer Around Settlements (Urban Built Up) 2km
var urban = BandOneIMAGE.gt(0);
Map.addLayer(urban, {}, 'Settlements / Urban Built Up', false);
var urban_buff = urban.distance(ee.Kernel.euclidean(2000, 'meters')).gt(0).remap([0,1],[1,1]).unmask(0);
Map.addLayer(urban_buff, {}, 'Settlements Buffer', false);
```

3. Analysis

3.3.2 Unconnected MPPs - Risk Assessments



2. Load Land Cover Data
GlobCover: Global Land Cover Map
This dataset describes the land use based on ENVISAT's Medium Resolution Imaging Spectrometer (MERIS) Level 1B data. Resolution is at 300 m. Plantation and croplands categories are used to further represent human settlements. This layer is then buffered outwards for 1 km for later analysis.

```
// Import Landuse Data
var dataset = ee.Image('ESA/GLOBCOVER_L4_200901_200912_V2_3');
var landcover = dataset.select('landcover').clip(studyarea);
Map.addLayer(landcover, {}, 'Landcover', false);
print(landcover);
// plantations / croplands
var settlement_list = ee.List([11, 14, 20]);
settlement_list = ee.Image.constant(settlement_list);
var crops = landcover.eq(settlement_list);
// Buffer around crop lands for 1km
Map.addLayer(crops, {}, 'crops', false);
var crop_buff = crops.distance(ee.Kernel.euclidean(1000, 'meters')).gt(0).remap([0,1],[1,1]).unmask(0);
Map.addLayer(crop_buff, {}, 'crops Buffer', false);
```

3. Load Road Data
GISTA The Thai Space Agency: Thailand Roads (subsetted in ArcGIS before upload)

4. Load Palm Oil Mill Data
WRI: The Universal Mill List (UML)

3. Analysis

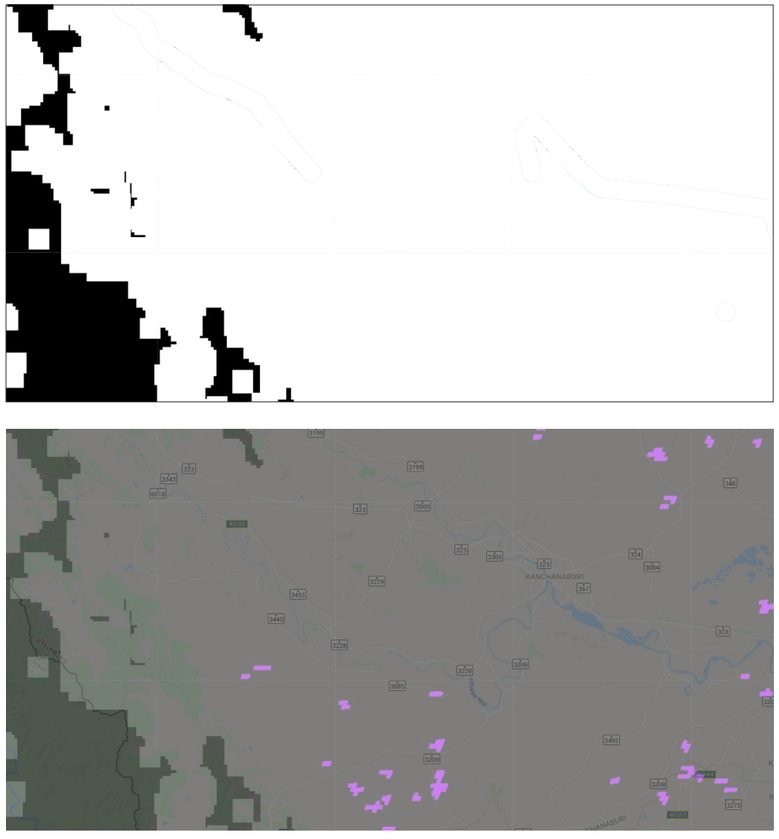
3.3.2 Unconnected MPPs - Risk Assessments

3. Load Road Data
GISTA The Thai Space Agency: Thailand Roads (subsetted in ArcGIS before upload)
This data has the Thailand road network in shapefile. I subsetted the rail network out and uploaded it to Google Earth Engine.
4. Load Palm Oil Mill Data
WRI: The Universal Mill List (UML)
This dataset has the location of palm oil mills' locations in a shape file. I uploaded the file to Google Earth Engine.
Both shape files are buffered outwards 1 km fore further analysis. Buffering results are transformed into rater format.

```
// rail road network and buffer 1km
var road = ee.FeatureCollection("users/hanyongxu/rail");
Map.addLayer(road, {}, 'Thai Road Network', false);
var BufferFeature = function(f) {
  f = ee.Feature(f);
  var buffer_size = f.get('buffer_size');
  return f.buffer(buffer_size);
};
var BufferFeaturesByDistance = function (fc, buffer_size) {
  var SetBufferSize = function(f) {
    return f.set({'buffer_size': buffer_size});
  };
  return fc.map(SetBufferSize).map(BufferFeature);
};
var road_buff = BufferFeaturesByDistance(road, 1000);
Map.addLayer(road_buff, {}, 'Thai Road Network buffer', false);

// palm oil mills and buffer 1km
var mill = ee.FeatureCollection("users/hanyongxu/Mills");
Map.addLayer(mill, {}, 'mills', false);
var mill_buff = BufferFeaturesByDistance(mill,1000);
Map.addLayer(mill_buff, {}, 'palm oil mills buffer', false);
// clip the regions within 1 km of the rail and set the result to raster
var rail_clip = raster1.clip(road_buff);
var mill_clip = raster1.clip(mill_buff);
var vec_clip = rail_clip.unmask(0).add(mill_clip.unmask(0));
Map.addLayer(vec_clip, {}, 'Thai Road + Palm Mill buffer raster', false);
```

3. Analysis

3.3.2 Unconnected MPPs - Risk Assessments



Combine All Risk Factors

```
// add all risk factors together
var risk = vec_clip.add(crop_buff).add(urban_buff).gt(0);
Map.addLayer(risk, {}, 'All Risk Factors', false);
```

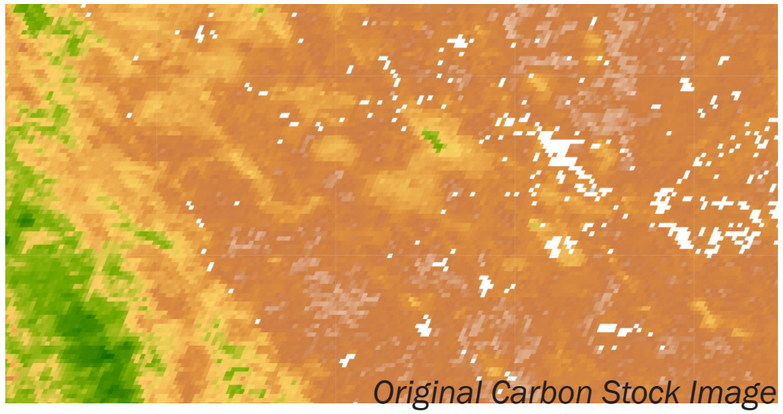
Filter High/Low Risk MPPs

Unconnected MPPs within risky regions (derived above) are labeled as “High Risk” and will be further examined for forest density. These patches are colored purple on the left. On the other hand, unconnected MPPs outside the risky regions will be labeled as “Indicative to Conserve”. Unfortunately, there is not such patches in the study region so I did not provide a screenshot for it. However, one can use the algorithm to find such patches in other regions.

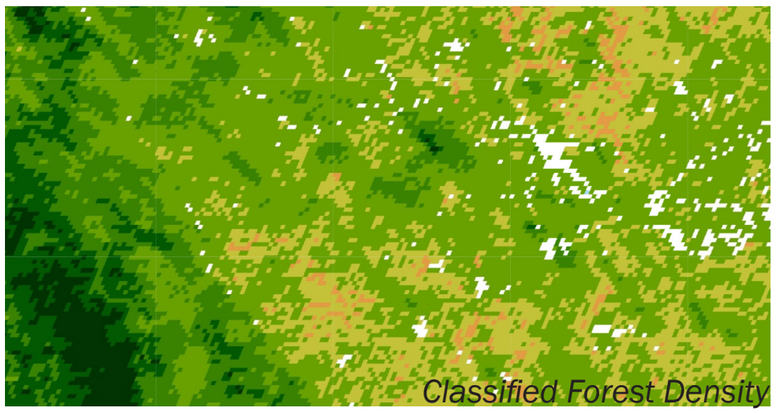
```
// Filter MPPs that are within/out of the risky regions
var risk_M = risk.add(MPP_ra).mask(MPP_ra).unmask(0);
var rm_fmean = risk_M.addBands(MPP_ra);
print(rm_fmean);
var rm_fmean = rm_fmean.reduceConnectedComponents(ee.Reducer.mean(), 'remapped', 256);
Map.addLayer(rm_fmean, {}, 'Risks + MPP focal mean after', false);
var MPP_con = risk_M.eq(rm_fmean).unmask(0);
var MPP_con = MPP_con.subtract(risk_M.eq(2)).remap([-1,0,1],[0,0,1]);
print(MPP_con);
Map.addLayer(MPP_con, {}, 'MPPs Indicative Conserve', false);
var MPP_ncon = MPP_ra.subtract(MPP_con).remap([-1,0,1],[0,0,1]);
print("MPP_ncon",MPP_ncon);
Map.addLayer(MPP_ncon, {palette:['000000','a503fc']}, 'MPPs High Risk', false);
```

3. Analysis

3.3.2 Unconnected MPPs - Risk Assessments



Original Carbon Stock Image



Classified Forest Density

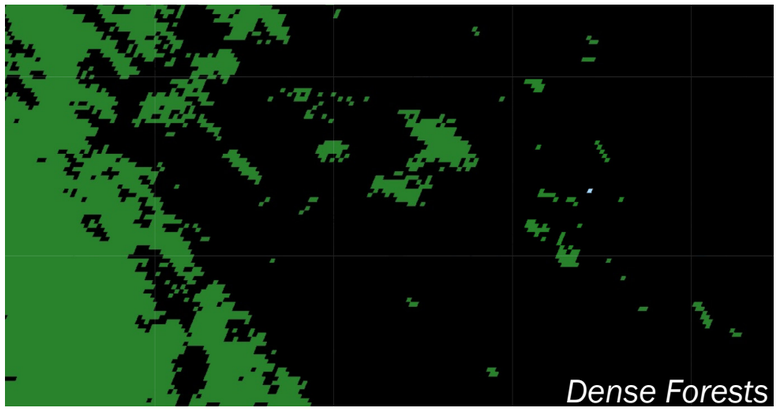
Reclassify Forest Density

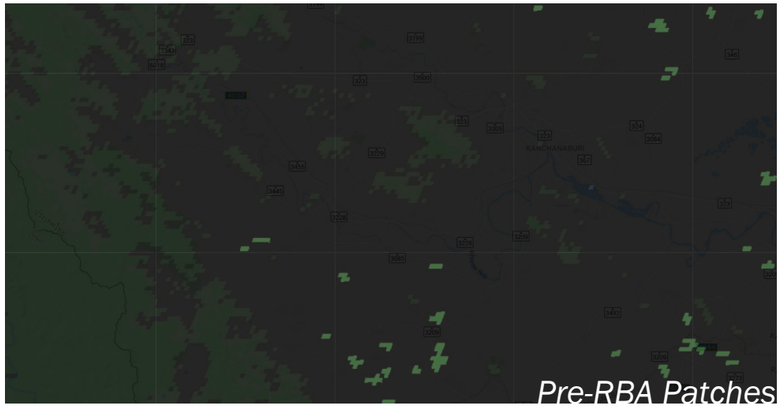
Forest density map is derived from the carbon stock dataset. The *Identifying High Carbon Stock (HCS) Forest for Protection* by Grennpeace outlines the possible ranges for six types of vegetation types: Cleared/Open Land, Young Scrub, Old Scrub, Low Density Forest (LDF), Medium Density Forest (MDF), and High Density Forest(HDF). LDF, MDF, and HDFs are considered as higher density.

```
// Reclassify Carbon Stock to get Forest Cover
var limits = [0, 22, 35, 83.5, 136.5, 179];
var forest_cov = ee.ImageCollection.fromImages(ee.List(limits).map(function(limit){
  return CarbonStock_c.gt(ee.Number(limit));
})).sum();
/*
Cleared/Open Land(LT) ~ 1, Young Scrub (BM) ~ 2, Old Scrub (BT) ~ 3,
Low Density Forest (HK1/LDF) ~ 4, Medium Density Forest (HK2/MDF) ~ 5,
High Density Forest (HK3/HDF) ~ 6
*/
print(forest_cov);
var forestvis = {
  min: 0,
  max: 7,
  palette: [
    'FFFFFF', 'CE7E45', 'DF923D', 'F1B555', 'FCD163', '99B718', '74A901',
    '66A000', '529400', '3E8601', '207401', '056201', '004C00', '023B01',
    '012E01', '011D01', '011301'
  ],
};
Map.addLayer(forest_cov, forestvis, 'HCS classification', false);
```

3. Analysis

3.3.2 Unconnected MPPs - Risk Assessments





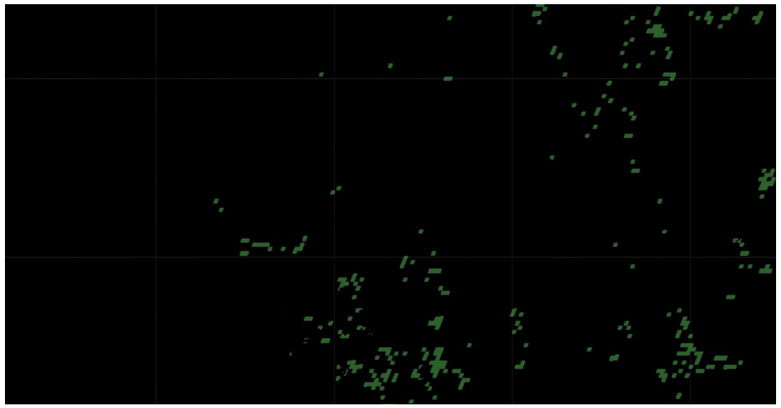
Check High Risk Patches Density

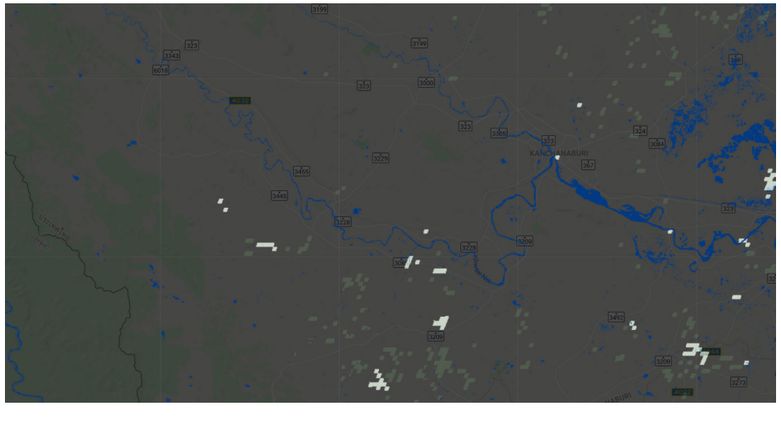
Patches that were labeled as “High Risk” are checked for their area size. Patches that are larger than 10 ha will be labeled as “Indicative to Conserve with Mitigation”. Other patches will be moved forward for Pre-RBA. There is no patches in the study region that belongs to the conservation category and all the previous high-risk patches move forward to Pre-RBA.

```
// Further filter MPPs according to forest density
// first find all MPPs that are LDF, MDF, HDF:
var forden_higher = forest_cov.gt(3.5);
Map.addLayer(forden_higher, {palette:['000000','fc0303']}, 'Forest cover more than low', false);
var MPP_miticheck = MPP_ncon.add(forden_higher).eq(2).unmask(0);
var MPP_miticheck = MPP_miticheck.mask(MPP_miticheck);
print(MPP_miticheck);
Map.addLayer(MPP_miticheck, {palette:['000000','03f8fc']}, 'MPPs check for mitigation', false);
// check the above MPPs for size (>10 ha)
var MPPm_patchsize = MPP_miticheck.connectedPixelCount(1024, false);
var MPPm_patcharea = MPPm_patchsize.multiply(ee.Image.pixelArea());
print("MPPm_patchsize", MPPm_patchsize);
print("MPPm_patcharea", MPPm_patcharea);
Map.addLayer(MPPm_patcharea, {}, 'MPPm_patcharea',false);
var MPPm = MPPm_patcharea.gt(100000); // greater than 10 ha
var MPPm = MPPm.mask(MPPm);
print("MPPm",MPPm);
var MPP_preRBA = MPP_ncon.subtract(MPPm.unmask(0)).remap([-1,0,1],[0,0,1]);
print("MPP_preRBA",MPP_preRBA);
Map.addLayer(MPP_preRBA, {palette:['000000', '2b632b']}, 'MPP for pre-RBA',false);
```

3. Analysis

3.4 Pre-Rapid Biodiversity Assessment





Merge All Patches that Require Pre-RBA

```
// merge all layers subject to Pre-RBA check
var preRBA = MPP_preRBA.unmask(0).add(LPP_pre_RBA.unmask(0)).gt(0);
print("preRBA",preRBA);
Map.addLayer(preRBA, {palette:['000000', '2b632b']}, 'pre-RBA patches',false);
```

Load Water Layer

JRC: Yearly Water Classification History, v1.1

This data shows water classifications generated from Landsat 5, 7, and 8. Data from 2018 to 2019 are used. Resolution is at 30 m. Permanent Water band is used. As outlined in the HCSA, close to or contain water could make the patch has more conservation value.

```
// conduct Pre RBA check
// Import water data
var water = ee.ImageCollection("JRC/GSW1_1/YearlyHistory").select('waterClass')
    .filter(ee.Filter.date('2018-01-01', '2019-12-31'));
var water = water.mean().eq(3);
Map.addLayer(water, {palette: '036ffc'}, 'water',false);
```

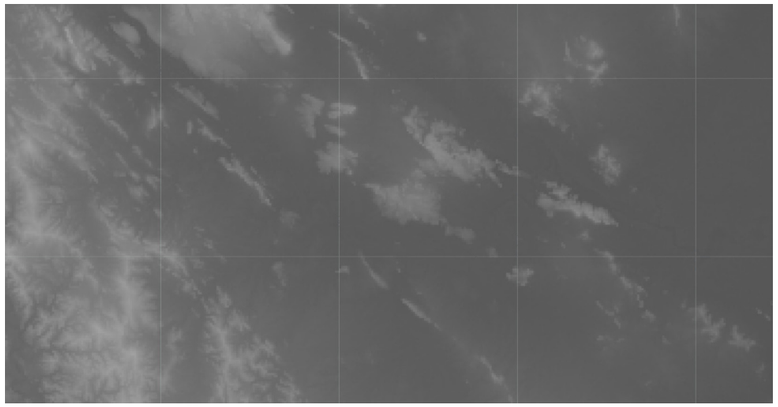
Proximity to Water

I define proximity as within 1 km distance to the water feature. Patches within this distance will be labeled as “near water” and are shown on left as white patches. Method to check proximity is the same as the one checking connectivity in section 3.2.

```
// check proximity to water
var water_buff = water.distance(ee.Kernel.euclidean(1000, 'meters')).gt(0).remap([0,1],[1,1]).unmask(0);
var pRBA_water = preRBA.add(water_buff).mask(preRBA).unmask(0);
Map.addLayer(pRBA_water,{},"pRBA + water",false);
var pRBA_water_fmean = pRBA_water.addBands(preRBA);
print("pRBA_water_fmean",pRBA_water_fmean);
var pRBA_water_fmean = pRBA_water_fmean.reduceConnectedComponents(ee.Reducer.mean(), 'remapped_1', 256);
var pRBA_near_water = pRBA_water.neq(pRBA_water_fmean).unmask(0);
print("pRBA_near_water",pRBA_near_water);
Map.addLayer(pRBA_near_water, {}, 'preRBA patches near water', false);
```

3. Analysis

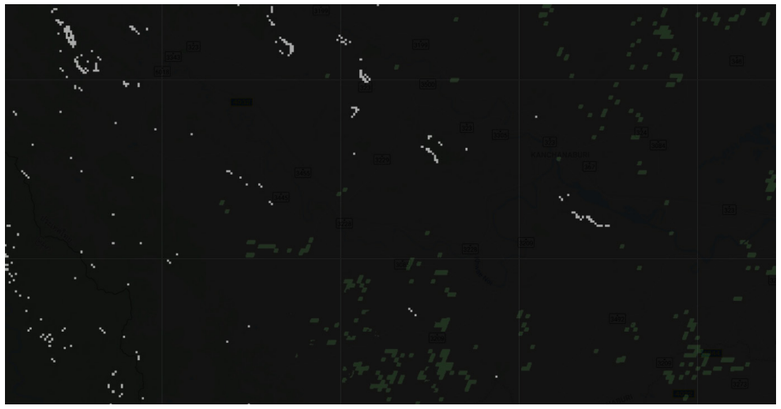
3.4 Pre-Rapid Biodiversity Assessment



Load Elevation Layer

GMTED2010: Global Multi-resolution Terrain Elevation Data 2010
This data shows the elevation world wide. Value ranges from -457m to 8746 m. Resolution is at 7.5 arc seconds.

```
// Import dem data
var elevation = ee.Image('USGS/GMTED2010').select('be75');
var elevationVis = {
  min: -100.0,
  max: 5000.0,
  gamma: 3.5,
};
Map.addLayer(elevation, elevationVis, 'Elevation',false);
```



Find Slope and Filter Steep Areas

Slope is identified by using the Terrain.slope() function.

```
// check slope
var slope = ee.Terrain.slope(elevation);
// find where slope is greater than 30 degrees
var steep = slope.gt(30);
Map.addLayer(steep, {}, 'steep slope',false);
```

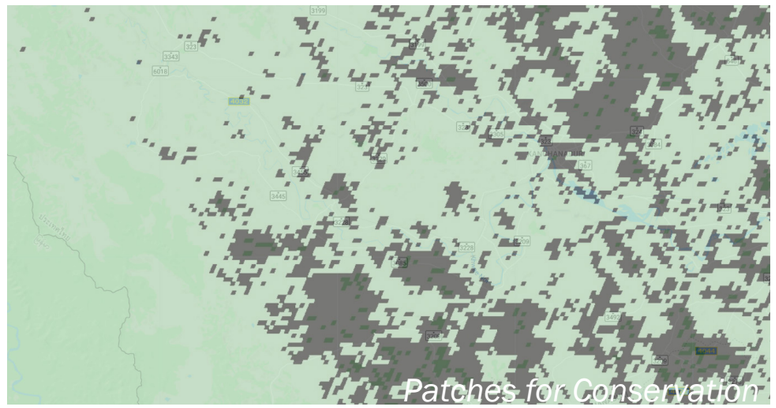
Proximity to Steep Areas

Since patches with steep slope might not be appropriate for development, thus steeper regions are categorized as conservation with mitigation. The same method as water proximity was used. Again, there is no patches near steeped regions in the study area.

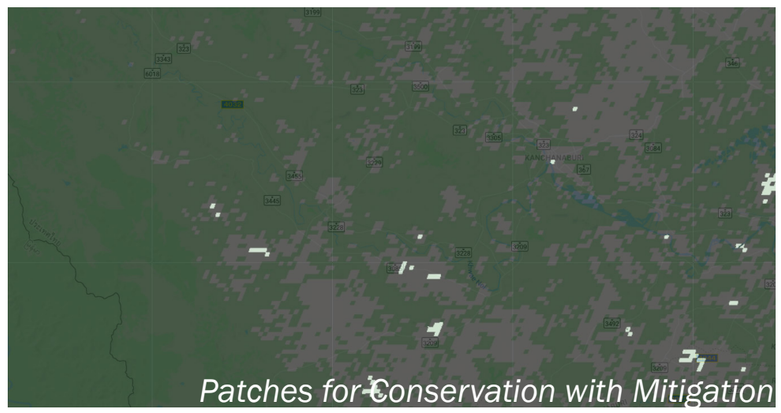
```
var steep_buff = steep.distance(ee.Kernel.euclidean(1000, 'meters')).gt(0).remap([0,1],[1,1]).unmask(0);
var pRBA_steep = preRBA.add(steep_buff).mask(preRBA).unmask(0);
Map.addLayer(pRBA_steep,{},"pRBA + steep",false);
var pRBA_steep_fmean = pRBA_steep.addBands(preRBA);
print("pRBA_steep_fmean",pRBA_steep_fmean);
var pRBA_steep_fmean = pRBA_steep_fmean.reduceConnectedComponents(ee.Reducer.mean(), 'remapped_1', 256);
var pRBA_near_steep = pRBA_steep.neq(pRBA_steep_fmean).unmask(0);
print("pRBA_near_steep",pRBA_near_steep);
Map.addLayer(pRBA_near_steep, {}, 'preRBA patches near steep', false);
```

3. Analysis

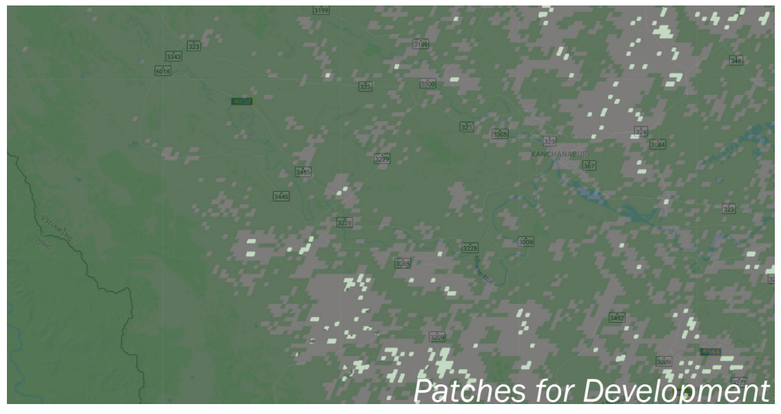
3.5 Results



Patches for Conservation



Patches for Conservation with Mitigation



Patches for Development

Merge Layers Together

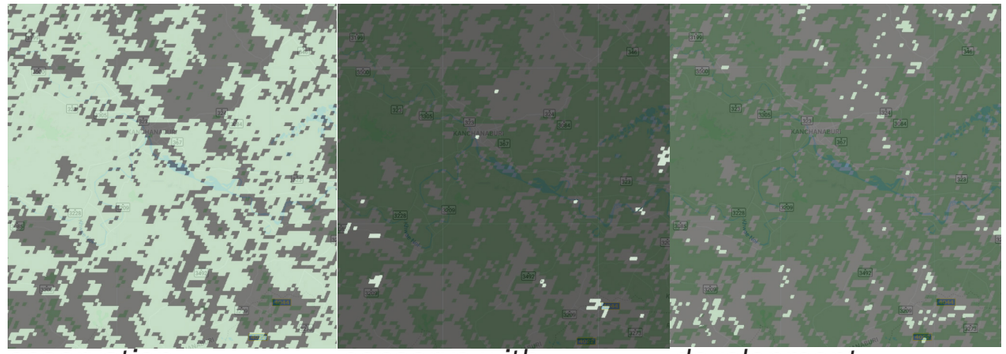
- *Conservation* layer includes original HPPs, connecting MPPs and LPPs, and low-risk unconnected MPPs.
- *Conservation with mitigation* layer includes high-risk unconnected MPPs that are either greater than 10 ha or pass pre-RBA check.
- Development layer includes unconnected LPPs with medium forest cover and unconnected LPPs with low forest cover that do not pass pre-RBA check.

```
// FINALIZE all the layers
// conservations that require mitigation
var con_miti = pRBA_near_steep.add(pRBA_near_water).add(MPPm.unmask(0)).gt(0);
print("con_miti",con_miti);
Map.addLayer(con_miti, {}, 'conservations that require mitigation');
// conservation for sure
var con = MPP_con.add(HPP_1).gt(0);
print("conservation", con);
Map.addLayer(con, {}, 'conservation for sure');
// development/ give and take
var develop =  preRBA.neq(pRBA_near_steep.add(pRBA_near_water).gt(0)).add(give_take.unmask(0)).gt(0);
print("develop",develop);
Map.addLayer(develop, {}, 'development');
```

**Original high carbon stock layer is layered at the background with dark green to show comparison.*

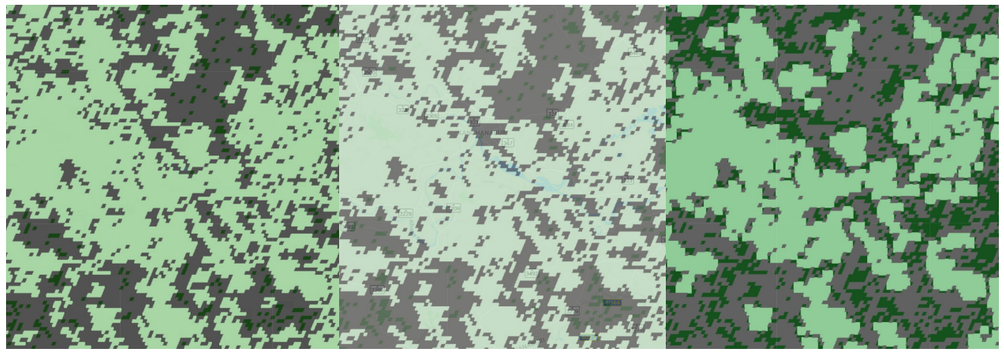
4. Conclusion

Through four major steps, including classifying High/Medium/Low Priority Patches, check connectivity between patches, check unconnected patches and conduct risk assessments, and lastly pre-Rapid Biodiversity Assessment, I was able to create a tool that identifies the forest patches that are suitable for either conservation or development. The tool allows the planner to quickly understand which patches are more valuable and what are less valuable so that they can make the right decisions for the natural environment.



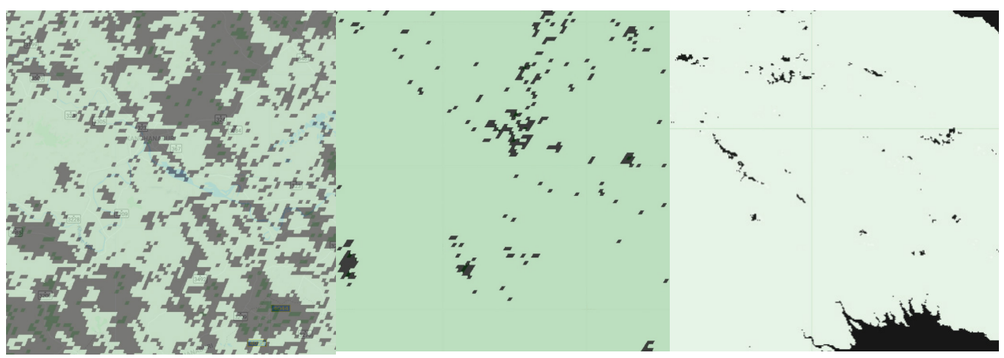
coservation *conserve with mitigation* *development*

Furthermore, I have included a prompt asking the user to input the buffer distance. This allow the user to customize the algorithm easily and receive feedback with different inputs. The images on the right shows outputs from different buffer distances.



100m *200m* *500m*

Lastly, even though for the sake of this report, the study region focuses a small region in Thailand near Bangkok, since the tool is designed in Google Earth Engine, it is easily replicable to different regions. The images on the right shows several sets of ouputs from different parts of the earth.



Bangkok, Thailand *Hainan, China* *New Guinea, Indonesia*

5. Appendix

5.1 Citations

5.1.1 Datasets

A. Baccini, S J. Goetz, W.S. Walker, N. T. Laporte, M. Sun, D. Sulla-Menashe, J. Hackler, P.S.A. Beck, R. Dubayah, M.A. Friedl, S. Samanta and R. A. Houghton. Estimated carbon dioxide emissions from tropical deforestation improved by carbon-density maps. 2012 Nature Climate Change, https://doi.org/10.1038/NCLIMATE1354

ESA 2010 and UCLouvain. http://due.esrin.esa.int/page_globcover.php

Global Multi-resolution Terrain Elevation Data 2010 courtesy of the U.S. Geological Survey

Jean-Francois Pekel, Andrew Cottam, Noel Gorelick, Alan S. Belward, High-resolution mapping of global surface water and its long-term changes. Nature 540, 418-422 (2016). (doi:10.1038/nature20584)

Thailand - Roads, GISTA The Thai Space Agency, https://data.humdata.org/dataset/thailand-roads

Tree Canopy Cover, 2010, Global Land Cover Facility, www.landcover.org.

Version 4 DMSP-OLS Nighttime Lights Time Series, 2012, National Oceanic and Atmospheric Administration, https://ngdc.noaa.gov/eog/dmsp/downloadV4composites.html.

World Resources Institute, Rainforest Alliance, Proforest, Daemeter, Trase, Earthworm, Auriga, CIFOR, Transitions, Jason Benedict, Robert Heilmayr, Kim Carlson“Universal Mill List.” October 2019. Accessed through Global Forest Watch on December 2019. www.globalforestwatch.org.

5.1.2 Reports

“Identifying High Carbon Stock (HCS) Forest for Protection.” Identifying High Carbon Stock (HCS) Forest for Protection. Greenpeace, March 2013. https://goldenagri.com.sg/pdfs/misc/HCS Briefer FINAL with graphic Eng REVISED.pdf.

“THE HCS APPROACH TOOLKIT Photo: Ardiles Rante © High Carbon Stock Forest Patch Analysis and Protection MODULE 5.” THE HCS APPROACH TOOLKIT Photo: Ardiles Rante © High Carbon Stock Forest Patch Analysis and Protection MODULE 5. HCS Approach Steering Group, May 2017. http://highcarbonstock.org/wp-content/uploads/2018/04/Def-HCSA-Module-5-16_04_2018_Web.pdf.

5. Appendix

5.2 Code

```
//var studyarea = ee.Geometry.Rectangle([99,13,106,19]);
//var studyarea = ee.Geometry.Rectangle([100,15,102,16]);
var studyarea = ee.Geometry.
Rectangle([108.6,17.8,111.6,20.24]);

// Importa Woody Biomass Data
var dataset = ee.Image('WHRC/biomass/tropical');
var CarbonStock = dataset.select('Mg');
var visParams = {
  min: 0.0,
  max: 503.0,
  palette: [
    'FFFFFF', 'CE7E45', 'DF923D', 'F1B555', 'FCD163',
'99B718', '74A901',
    '66A000', '529400', '3E8601', '207401', '056201',
'004C00', '023B01',
    '012E01', '011D01', '011301'
  ],
};
Map.centerObject(studyarea, 11);
var CarbonStock_c = CarbonStock.clip(studyarea);
Map.addLayer(CarbonStock_c, visParams, 'Aboveground Live
Woody Biomass', false);

// Get low carbon stock areas
var lcs = CarbonStock_c.lte(35);
var hcs = CarbonStock_c.gt(35);
Map.addLayer(lcs, {palette:['aedbf5','bab98c']}, 'Low Carbon
Stock', false);

// User Input
var response = prompt( "Hello! Welcome to the High Carbon
Stock Approach Planner. \n"
+ "\nPlease enter the a number (in meters) to buffer
to determine size of the core area of each forest patch.
Suggested distance is 200 m.");
var response = ee.Number.parse(response);

// Buffer around the low carbon stock and subtract the result
from the whole
var bufferlcs = lcs
    .cumulativeCost({
      source: lcs,
      maxDistance: response,
    }).lt(35);

// Get core hcs
var raster1 = ee.Image(1);
var bufferlcs = bufferlcs.unmask(-1);
var core_hcs = raster1.subtract(bufferlcs).clip(studyarea);
var core_hcs = core_hcs.remap([0,1,2],[0,0,1]);
var core_hcs = core_hcs.mask(core_hcs);
Map.addLayer(core_hcs, {palette:['000000','4a8762']}, 'Core
HCS', false);

// Compute the number of pixels in each patch.
var patchsize = core_hcs.connectedPixelCount(1024, false);
var patcharea = patchsize.multiply(ee.Image.pixelArea());
Map.addLayer(patchsize, {}, 'patch size', false);
Map.addLayer(patcharea, {}, 'patch area', false);

// Filter high, medium, low priority patches
var HPP = patcharea.gt(1000000); // greater than 100 ha
var MPP = patcharea.lte(1000000).and(patcharea.
gte(100000)); // 10-100 ha
var LPP = patcharea.lt(100000); // less than 10 ha

// Buffer around the core HPPs, MPPs, LPPs to get original
carbon stock area
var HPP_ = HPP.mask(hcs).distance(ee.Kernel.
euclidean(response, 'meters')).gt(0).remap([0,1],[1,1]);
var MPP_ = MPP.mask(hcs).distance(ee.Kernel.
euclidean(response, 'meters')).gt(0).remap([0,1],[1,1]);
var LPP_ = LPP.mask(hcs).distance(ee.Kernel.
euclidean(response, 'meters')).gt(0).remap([0,1],[1,1]);
Map.addLayer(HPP_, {palette:['000000', '2b632b']}, 'High
Priority Patch', false);
Map.addLayer(MPP_, {palette:['000000', '6a9c6a']}, 'Medium
Priority Patch', false);
Map.addLayer(LPP_, {palette:['000000', 'c0ccc0']}, 'Low
Priority Patch', false);

var HPP_ = HPP_.unmask(0);
var MPP_ = MPP_.unmask(0);
var LPP_ = LPP_.unmask(0);

// Check Connectivity
// Isolate Median Priority Patches that connect to the High
Priority Patches within 200 meters
var distM = MPP_.distance(ee.Kernel.euclidean(200,
'meters')).gt(0).remap([0,1],[1,1]).unmask(0);
//Map.addLayer(distM, {}, 'Distance to Medium Priority
Patch', false);
var M_H = distM.add(HPP_).mask(MPP_).unmask(0);
Map.addLayer(M_H, {}, 'M+H', false);
var TheREDUCER = ee.Reducer.sum();
var med_fsum0 = MPP_.addBands(MPP_);
var med_fsum = M_H.addBands(MPP_);
var med_fsum0 = med_fsum0.
reduceConnectedComponents(TheREDUCER, 'remapped_1',
256);
var med_fsum = med_fsum.
reduceConnectedComponents(TheREDUCER, 'remapped_1',
256);
Map.addLayer(med_fsum0, {}, 'M+H focal sum previous',
false);
Map.addLayer(med_fsum, {}, 'M+H focal sum after', false);
var med_connect = med_fsum.subtract(med_fsum0).gt(0);
var med_connect = med_connect.mask(med_connect);
Map.addLayer(med_connect, {palette: "fc0303"}, 'Median
Connecting', false);

// Isolate Low Priority Patches that connect to the High
Priority Patches within 200 meters
var distL = LPP_.distance(ee.Kernel.euclidean(200,
'meters')).gt(0).remap([0,1],[1,1]).unmask(0);
//Map.addLayer(distL, {}, 'Distance to Low Priority Patch',
false);
var L_H = distL.add(HPP_).mask(LPP_).unmask(0);
Map.addLayer(L_H, {}, 'L+H', false);
var low_fsum0 = LPP_.addBands(LPP_);
var low_fsum = L_H.addBands(LPP_);
var low_fsum0 = low_fsum0.
reduceConnectedComponents(TheREDUCER, 'remapped_1',
256);
```

5. Appendix

```
var low_fsum = low_fsum.
reduceConnectedComponents(TheREDUCER, 'remapped_1',
256);
Map.addLayer(low_fsum0, {}, 'L+H focal sum previous', false);
Map.addLayer(low_fsum, {}, 'L+H focal sum after', false);
var low_connect = low_fsum.subtract(low_fsum0).gt(0);
var low_connect = low_connect.mask(low_connect);
print("low connecting", low_connect);
Map.addLayer(low_connect, {palette: "fc0303"}, 'Low
Connecting', false);

// Merge Low and Med patches that can connect with High
patches together
var HPP_1 = HPP_.add(low_connect.unmask(0)).add(med_
connect.unmask(0)).gt(0);
Map.addLayer(HPP_1, {}, 'Old HPPs and New conserve from
MPPs and LPPs', false);

// Reclassify Carbon Stock to get Forest Cover
var limits = [0, 22, 35, 83.5, 136.5, 179];
var forest_cov = ee.ImageCollection.fromImages(ee.List(limits).
map(function(limit){
  return CarbonStock_c.gt(ee.Number(limit));
})).sum();
/*
Cleared/Open Land(LT) ~ 1, Young Scrub (BM) ~ 2, Old Scrub
(BT) ~ 3,
Low Density Forest (HK1/LDF) ~ 4, Medium Density Forest
(HK2/MDF) ~ 5,
High Density Forest (HK3/HDF) ~ 6
*/
print(forest_cov);
var forestvis = {
  min: 0,
  max: 7,
  palette: [
    'FFFFFF', 'CE7E45', 'DF923D', 'F1B555', 'FCD163',
'99B718', '74A901',
    '66A000', '529400', '3E8601', '207401', '056201',
'004C00', '023B01',
    '012E01', '011D01', '011301'
  ],
};
Map.addLayer(forest_cov, forestvis, 'HCS classification', false);


// Import tree cover data and classify the 3 types of tree covers
var treeCanopyCover = ee.ImageCollection('GLCF/GLS_TCC').
select('tree_canopy_cover').filter(ee.Filter.date('2010-01-01',
'2010-12-31'));
var treeCanopyCover = treeCanopyCover.mean();
var treeCanopyCoverVis = {
  min: 0.0,
  max: 100.0,
  palette: ['ffffff', 'afce56', '5f9c00', '0e6a00', '003800'],
};
Map.addLayer(treeCanopyCover, treeCanopyCoverVis, 'Tree
Canopy Cover',false);
var area = ee.Image(treeCanopyCover).clip(studyarea);
// classification into high/med/low forest cover
var area = ee.Image(treeCanopyCover);
var highcover = area.gt(80);
var highcover = highcover.mask(highcover);
var medcover = area.lte(80).and(area.gt(30));
var medcover = medcover.mask(medcover);
var lowcover = area.lte(30);
var lowcover = lowcover.mask(lowcover);
Map.addLayer(highcover, {palette: "2b632b"}, 'high forest cover',
false);
Map.addLayer(medcover, {palette: "6a9c6a"}, 'med forest cover',
false);
Map.addLayer(lowcover, {palette: "c0ccc0"}, 'low forest cover',
false);

// Find LPPs that are subject to "give and take approach"
var give_take = LPP_.subtract(low_connect.unmask(0)).
remap([-1,0,1],[0,0,1]).add(medcover.unmask(0)).eq(2);
var give_take = give_take.mask(give_take);
print("give and take", give_take);
Map.addLayer(give_take, {palette: "fc0303"}, 'Give and Take',
false);

// Find LPPs that are subject to Pre-RBA check
var LPP_pre_RBA = LPP_.subtract(low_connect.unmask(0)).
remap([-1,0,1],[0,0,1]).add(lowcover.unmask(0)).eq(2);
var LPP_pre_RBA = LPP_pre_RBA.mask(LPP_pre_RBA);
print("LPPs for pre RBA assessment", LPP_pre_RBA);
Map.addLayer(LPP_pre_RBA, {palette: "fc0303"}, 'LPPs for
Pre-RBA', false);

// Find MPPs that are not conserved
var MPP_ra = MPP_.subtract(med_connect.unmask(0)).
remap([-1,0,1],[0,0,1]);
print(MPP_ra);
Map.addLayer(MPP_ra, {}, 'MPPs for Risk Assessment', false);

// Find all risk factors for risk assessment

// use light change to represent urban cities / human
activities / settlements
var lights = ee.Image( 'NOAA/DMSP-OLS/NIGHTTIME_
LIGHTS/F182012').clip(studyarea).select("stable_lights");//
Night Lighting in 2013
var BandOneIMAGE = lights.expression( 'b("stable_lights")' );
// Band 1: Persistent Lighting
Map.addLayer( BandOneIMAGE, {min: 0, max: 63, palette:
['000044','ffff00','ffffff']}, 'persistnet lighting', false);
// Buffer Around Settlements (Urban Built Up) 2km
var urban = BandOneIMAGE.gt(0);
Map.addLayer(urban, {}, 'Settlements / Urban Built Up', false);
var urban_buff = urban.distance(ee.Kernel.euclidean(2000,
'meters')).gt(0).remap([0,1],[1,1]).unmask(0);
Map.addLayer(urban_buff, {}, 'Settlements Buffer', false);

// Import Landuse Data
var dataset = ee.Image('ESA/GLOBCOVER_
L4_200901_200912_V2_3');
var landcover = dataset.select('landcover').clip(studyarea);
Map.addLayer(landcover, {}, 'Landcover', false);
print(landcover);
// plantations / croplands
var settlement_list = ee.List([11, 14, 20]);
settlement_list = ee.Image.constant(settlement_list);
var crops = landcover.eq(settlement_list);
// Buffer around crop lands for 1km
Map.addLayer(crops, {}, 'crops', false);
```

5. Appendix

```
var crop_buff = crops.distance(ee.Kernel.euclidean(1000,
'meters')).gt(0).remap([0,1],[1,1]).unmask(0);
Map.addLayer(crop_buff, {}, 'crops Buffer', false);

// rail road network and buffer 1km
var road = ee.FeatureCollection("users/hanyongxu/rail");
Map.addLayer(road, {}, 'Thai Road Network', false);
var BufferFeature = function(f) {
  f = ee.Feature(f);
  var buffer_size = f.get('buffer_size');
  return f.buffer(buffer_size);
};
var BufferFeaturesByDistance = function (fc, buffer_size) {
  var SetBufferSize = function(f) {
    return f.set({'buffer_size': buffer_size});
  };
  return fc.map(SetBufferSize).map(BufferFeature);
};
var road_buff = BufferFeaturesByDistance(road, 1000);
Map.addLayer(road_buff, {}, 'Thai Road Network buffer', false);

// palm oil mills and buffer 1km
var mill = ee.FeatureCollection("users/hanyongxu/Mills");
Map.addLayer(mill, {}, 'mills', false);
var mill_buff = BufferFeaturesByDistance(mill,1000);
Map.addLayer(mill_buff, {}, 'palm oil mills buffer', false);

// clip the regions within 1 km of the rail and set the result to
raster
var rail_clip = raster1.clip(road_buff);
var mill_clip = raster1.clip(mill_buff);
var vec_clip = rail_clip.unmask(0).add(mill_clip.unmask(0));
Map.addLayer(vec_clip, {}, 'Thai Road + Palm Mill buffer
raster', false);

// add all risk factors together
var risk = vec_clip.add(crop_buff).add(urban_buff).gt(0);
Map.addLayer(risk, {}, 'All Risk Factors', false);

// Filter MPPs that are within/out of the risky regions
var risk_M = risk.add(MPP_ra).mask(MPP_ra).unmask(0);
var rm_fmean = risk_M.addBands(MPP_ra);
print(rm_fmean);
var rm_fmean = rm_fmean.reduceConnectedComponents(ee.
Reducer.mean(), 'remapped', 256);
Map.addLayer(rm_fmean, {}, 'Risks + MPP focal mean after',
false);
var MPP_con = risk_M.eq(rm_fmean).unmask(0);
var MPP_con = MPP_con.subtract(risk_M.eq(2)).
remap([-1,0,1],[0,0,1]);
print(MPP_con);
Map.addLayer(MPP_con, {}, 'MPPs Indicative Conserve', false);
var MPP_ncon = MPP_ra.subtract(MPP_con).
remap([-1,0,1],[0,0,1]);
print("MPP_ncon",MPP_ncon);
Map.addLayer(MPP_ncon, {palette:['000000','a503fc']}, 'MPPs
High Risk', false);

// Further filter MPPs according to forest density
// first find all MPPs that are LDF, MDF, HDF:
var forden_higher = forest_cov.gt(3.5);
Map.addLayer(forden_higher, {palette:['000000','fc0303']},
'Forest cover more than low', false);
var MPP_miticheck = MPP_ncon.add(forden_higher).eq(2).
unmask(0);
var MPP_miticheck = MPP_miticheck.mask(MPP_miticheck);
print(MPP_miticheck);
Map.addLayer(MPP_miticheck, {palette:['000000','03f8fc']},
'MPPs check for mitigation', false);
// check the above MPPs for size (>10 ha)
var MPPm_patchsize = MPP_miticheck.
connectedPixelCount(1024, false);
var MPPm_patcharea = MPPm_patchsize.multiply(ee.Image.
pixelArea());
print("MPPm_patchsize", MPPm_patchsize);
print("MPPm_patcharea", MPPm_patcharea);
Map.addLayer(MPPm_patcharea, {}, 'MPPm_patcharea',false);
var MPPm = MPPm_patcharea.gt(100000); // greater than 10 ha
var MPPm = MPPm.mask(MPPm);
print("MPPm",MPPm);
var MPP_preRBA = MPP_ncon.subtract(MPPm.unmask(0)).
remap([-1,0,1],[0,0,1]);
print("MPP_preRBA",MPP_preRBA);
Map.addLayer(MPP_preRBA, {palette:['000000', '2b632b']},
'MPP for pre-RBA',false);

// merge all layers subject to Pre-RBA check
var preRBA = MPP_preRBA.unmask(0).add(LPP_pre_RBA.
unmask(0)).gt(0);
print("preRBA",preRBA);
Map.addLayer(preRBA, {palette:['000000', '2b632b']}, 'pre-
RBA patches',false);

// conduct Pre RBA check
// Import water data
var water = ee.ImageCollection("JRC/GSW1_1/
YearlyHistory").select('waterClass').filter(ee.Filter.date('2018-
01-01', '2019-12-31'));
var water = water.mean().eq(3);
Map.addLayer(water, {palette: '036ffc'}, 'water',false);

// check proximity to water
var water_buff = water.distance(ee.Kernel.euclidean(1000,
'meters')).gt(0).remap([0,1],[1,1]).unmask(0);
var pRBA_water = preRBA.add(water_buff).mask(preRBA).
unmask(0);
Map.addLayer(pRBA_water,{},"pRBA + water",false);
var pRBA_water_fmean = pRBA_water.addBands(preRBA);
print("pRBA_water_fmean",pRBA_water_fmean);
var pRBA_water_fmean = pRBA_water_fmean.
reduceConnectedComponents(ee.Reducer.mean(),
'remapped_1', 256);
var pRBA_near_water = pRBA_water.neq(pRBA_water_
fmean).unmask(0);
print("pRBA_near_water",pRBA_near_water);
Map.addLayer(pRBA_near_water, {}, 'preRBA patches near
water', false);

// Import dem data
var elevation = ee.Image('USGS/GMTED2010').select('be75');
var elevationVis = {
  min: -100.0,
  max: 5000.0,
  gamma: 3.5,
};
Map.addLayer(elevation, elevationVis, 'Elevation',false);
// check slope
var slope = ee.Terrain.slope(elevation);
// find where slope is greater than 45 degrees
```

5. Appendix

```
var steep = slope.gt(30);
Map.addLayer(steep, {}, ‘steep slope’,false);
var steep_buff = steep.distance(ee.Kernel.euclidean(1000,
‘meters’)).gt(0).remap([0,1],[1,1]).unmask(0);
var pRBA_steep = preRBA.add(steep_buff).mask(preRBA).
unmask(0);
Map.addLayer(pRBA_steep,{},”pRBA + steep”,false);
var pRBA_steep_fmean = pRBA_steep.addBands(preRBA);
print(“pRBA_steep_fmean”,pRBA_steep_fmean);
var pRBA_steep_fmean = pRBA_steep_fmean.
reduceConnectedComponents(ee.Reducer.mean(),
‘remapped_1’, 256);
var pRBA_near_steep = pRBA_steep.neq(pRBA_steep_fmean).
unmask(0);
print(“pRBA_near_steep”,pRBA_near_steep);
Map.addLayer(pRBA_near_steep, {}, ‘preRBA patches near
steep’, false);


// FINALIZE all the layers
// conservations that require mitigation
var con_miti = pRBA_near_steep.add(pRBA_near_water).
add(MPPm.unmask(0)).gt(0);
print(“con_miti”,con_miti);
Map.addLayer(con_miti, {}, ‘conservations that require
mitigation’);
// conservation for sure
var con = MPP_con.add(HPP_1).gt(0);
print(“conservation”, con);
Map.addLayer(con, {}, ‘conservation for sure’);
// development/ give and take
var develop =  preRBA.neq(pRBA_near_steep.add(pRBA_near_
water).gt(0)).add(give_take.unmask(0)).gt(0);
print(“develop”,develop);
Map.addLayer(develop, {}, ‘development’);

var forden_higher = forest_cov.gt(3.5);
Map.addLayer(forden_higher, {palette:[‘000000’,’fc0303’]},
‘Forest cover more than low’, false);
var MPP_miticheck = MPP_ncon.add(forden_higher).eq(2).
unmask(0);
var MPP_miticheck = MPP_miticheck.mask(MPP_miticheck);
print(MPP_miticheck);
Map.addLayer(MPP_miticheck, {palette:[‘000000’,’03f8fc’]},
‘MPPs check for mitigation’, false);
// check the above MPPs for size (>10 ha)
var MPPm_patchsize = MPP_miticheck.
connectedPixelCount(1024, false);
var MPPm_patcharea = MPPm_patchsize.multiply(ee.Image.
pixelArea());
print(“MPPm_patchsize”, MPPm_patchsize);
print(“MPPm_patcharea”, MPPm_patcharea);
Map.addLayer(MPPm_patcharea, {}, ‘MPPm_patcharea’,false);
var MPPm = MPPm_patcharea.gt(100000); // greater than 10 ha
var MPPm = MPPm.mask(MPPm);
print(“MPPm”,MPPm);
var MPP_preRBA = MPP_ncon.subtract(MPPm.unmask(0)).
remap([-1,0,1],[0,0,1]);
print(“MPP_preRBA”,MPP_preRBA);
Map.addLayer(MPP_preRBA, {palette:[‘000000’, ‘2b632b’]},
‘MPP for pre-RBA’,false);

// merge all layers subject to Pre-RBA check
var preRBA = MPP_preRBA.unmask(0).add(LPP_pre_RBA.
unmask(0)).gt(0);
print(“preRBA”,preRBA);
Map.addLayer(preRBA, {palette:[‘000000’, ‘2b632b’]}, ‘pre-RBA
patches’,false);

// conduct Pre RBA check
// Import water data
var water = ee.ImageCollection(“JRC/GSW1_1/YearlyHistory”).
select(‘waterClass’).filter(ee.Filter.date(‘2018-01-01’, ‘2019-12-
31’));
var water = water.mean().eq(3);
Map.addLayer(water, {palette: ‘036ffc’}, ‘water’,false);

// check proximity to water
var water_buff = water.distance(ee.Kernel.euclidean(1000,
‘meters’)).gt(0).remap([0,1],[1,1]).unmask(0);
var pRBA_water = preRBA.add(water_buff).mask(preRBA).
unmask(0);
Map.addLayer(pRBA_water,{},”pRBA + water”,false);
var pRBA_water_fmean = pRBA_water.addBands(preRBA);
print(“pRBA_water_fmean”,pRBA_water_fmean);
var pRBA_water_fmean = pRBA_water_fmean.
reduceConnectedComponents(ee.Reducer.mean(),
‘remapped_1’, 256);
var pRBA_near_water = pRBA_water.neq(pRBA_water_
fmean).unmask(0);
print(“pRBA_near_water”,pRBA_near_water);
Map.addLayer(pRBA_near_water, {}, ‘preRBA patches near
water’, false);

// Import dem data
var elevation = ee.Image(‘USGS/GMTED2010’).select(‘be75’);
var elevationVis = {
  min: -100.0,
  max: 5000.0,
  gamma: 3.5,
};
Map.addLayer(elevation, elevationVis, ‘Elevation’,false);
// check slope
var slope = ee.Terrain.slope(elevation);
// find where slope is greater than 45 degrees
var steep = slope.gt(30);
Map.addLayer(steep, {}, ‘steep slope’,false);
var steep_buff = steep.distance(ee.Kernel.euclidean(1000,
‘meters’)).gt(0).remap([0,1],[1,1]).unmask(0);
var pRBA_steep = preRBA.add(steep_buff).mask(preRBA).
unmask(0);
Map.addLayer(pRBA_steep,{},”pRBA + steep”,false);
var pRBA_steep_fmean = pRBA_steep.addBands(preRBA);
print(“pRBA_steep_fmean”,pRBA_steep_fmean);
var pRBA_steep_fmean = pRBA_steep_fmean.
reduceConnectedComponents(ee.Reducer.mean(),
‘remapped_1’, 256);
var pRBA_near_steep = pRBA_steep.neq(pRBA_steep_
fmean).unmask(0);
print(“pRBA_near_steep”,pRBA_near_steep);
Map.addLayer(pRBA_near_steep, {}, ‘preRBA patches near
steep’, false);


// FINALIZE all the layers
// conservations that require mitigation
```

5. Appendix

```
var con_miti = pRBA_near_steep.add(pRBA_near_water).
add(MPPm.unmask(0)).gt(0);
print("con_miti",con_miti);
Map.addLayer(con_miti, {}, 'conservations that require
mitigation');
// conservation for sure
var con = MPP_con.add(HPP_1).gt(0);
print("conservation", con);
Map.addLayer(con, {}, 'conservation for sure');
// development/ give and take
var develop =  preRBA.neq(pRBA_near_steep.add(pRBA_near_
water).gt(0)).add(give_take.unmask(0)).gt(0);
print("develop",develop);
Map.addLayer(develop, {}, 'development');

mean(), 'remapped_1', 256);
var pRBA_near_steep = pRBA_steep.neq(pRBA_steep_fmean).
unmask(0);
print("pRBA_near_steep",pRBA_near_steep);
Map.addLayer(pRBA_near_steep, {}, 'preRBA patches near
steep', false);


// FINALIZE all the layers
// conservations that require mitigation
var con_miti = pRBA_near_steep.add(pRBA_near_water).
add(MPPm.unmask(0)).gt(0);
print("con_miti",con_miti);
Map.addLayer(con_miti, {}, 'conservations that require
mitigation');
// conservation for sure
var con = MPP_con.add(HPP_1).gt(0);
print("conservation", con);
Map.addLayer(con, {}, 'conservation for sure');
// development/ give and take
var develop =  preRBA.neq(pRBA_near_steep.add(pRBA_near_
water).gt(0)).add(give_take.unmask(0)).gt(0);
print("develop",develop);
Map.addLayer(develop, {}, 'development');

var forden_higher = forest_cov.gt(3.5);
Map.addLayer(forden_higher, {palette:['000000','fc0303']},
'Forest cover more than low', false);
var MPP_miticheck = MPP_ncon.add(forden_higher).eq(2).
unmask(0);
var MPP_miticheck = MPP_miticheck.mask(MPP_miticheck);
print(MPP_miticheck);
Map.addLayer(MPP_miticheck, {palette:['000000','03f8fc']},
'MPPs check for mitigation', false);
// check the above MPPs for size (>10 ha)
var MPPm_patchsize = MPP_miticheck.
connectedPixelCount(1024, false);
var MPPm_patcharea = MPPm_patchsize.multiply(ee.Image.
pixelArea());
print("MPPm_patchsize", MPPm_patchsize);
print("MPPm_patcharea", MPPm_patcharea);
Map.addLayer(MPPm_patcharea, {}, 'MPPm_patcharea',false);
var MPPm = MPPm_patcharea.gt(100000); // greater than 10 ha
var MPPm = MPPm.mask(MPPm);
print("MPPm",MPPm);
var MPP_preRBA = MPP_ncon.subtract(MPPm.unmask(0)).
remap([-1,0,1],[0,0,1]);
print("MPP_preRBA",MPP_preRBA);
Map.addLayer(MPP_preRBA, {palette:['000000', '2b632b']},
'MPP for pre-RBA',false);

// merge all layers subject to Pre-RBA check
var preRBA = MPP_preRBA.unmask(0).add(LPP_pre_RBA.
unmask(0)).gt(0);
print("preRBA",preRBA);
Map.addLayer(preRBA, {palette:['000000', '2b632b']}, 'pre-RBA
patches',false);

// conduct Pre RBA check
// Import water data
var water = ee.ImageCollection("JRC/GSW1_1/YearlyHistory").
select('waterClass').filter(ee.Filter.date('2018-01-01', '2019-12-
31'));
var water = water.mean().eq(3);
Map.addLayer(water, {palette: '036ffc'}, 'water',false);

// check proximity to water
var water_buff = water.distance(ee.Kernel.euclidean(1000,
'meters')).gt(0).remap([0,1],[1,1]).unmask(0);
var pRBA_water = preRBA.add(water_buff).mask(preRBA).
unmask(0);
Map.addLayer(pRBA_water,{},"pRBA + water",false);
var pRBA_water_fmean = pRBA_water.addBands(preRBA);
print("pRBA_water_fmean",pRBA_water_fmean);
var pRBA_water_fmean = pRBA_water_fmean.
reduceConnectedComponents(ee.Reducer.mean(),
'remapped_1', 256);
var pRBA_near_water = pRBA_water.neq(pRBA_water_
fmean).unmask(0);
print("pRBA_near_water",pRBA_near_water);
Map.addLayer(pRBA_near_water, {}, 'preRBA patches near
water', false);

// Import dem data
var elevation = ee.Image('USGS/GMTED2010').select('be75');
var elevationVis = {
  min: -100.0,
  max: 5000.0,
  gamma: 3.5,
};
Map.addLayer(elevation, elevationVis, 'Elevation',false);
// check slope
var slope = ee.Terrain.slope(elevation);
// find where slope is greater than 45 degrees
var steep = slope.gt(30);
Map.addLayer(steep, {}, 'steep slope',false);
var steep_buff = steep.distance(ee.Kernel.euclidean(1000,
'meters')).gt(0).remap([0,1],[1,1]).unmask(0);
var pRBA_steep = preRBA.add(steep_buff).mask(preRBA).
unmask(0);
Map.addLayer(pRBA_steep,{},"pRBA + steep",false);
var pRBA_steep_fmean = pRBA_steep.addBands(preRBA);
print("pRBA_steep_fmean",pRBA_steep_fmean);
var pRBA_steep_fmean = pRBA_steep_fmean.
reduceConnectedComponents(ee.Reducer.mean(),
'remapped_1', 256);
var pRBA_near_steep = pRBA_steep.neq(pRBA_steep_
fmean).unmask(0);
print("pRBA_near_steep",pRBA_near_steep);
Map.addLayer(pRBA_near_steep, {}, 'preRBA patches near
steep', false);
```

5. Appendix

```
// FINALIZE all the layers
// conservations that require mitigation
var con_miti = pRBA_near_steep.add(pRBA_near_water).
add(MPPm.unmask(0)).gt(0);
print(“con_miti”,con_miti);
Map.addLayer(con_miti, {}, ‘conservations that require
mitigation’);
// conservation for sure
var con = MPP_con.add(HPP_1).gt(0);
print(“conservation”, con);
Map.addLayer(con, {}, ‘conservation for sure’);
// development/ give and take
var develop =  preRBA.neq(pRBA_near_steep.add(pRBA_near_
water).gt(0)).add(give_take.unmask(0)).gt(0);
print(“develop”,develop);
Map.addLayer(develop, {}, ‘development’);

var elevation = ee.Image(‘USGS/GMTED2010’).select(‘be75’);
var elevationVis = {
  min: -100.0,
  max: 5000.0,
  gamma: 3.5,
};
Map.addLayer(elevation, elevationVis, ‘Elevation’,false);
// check slope
var slope = ee.Terrain.slope(elevation);
// find where slope is greater than 45 degrees
var steep = slope.gt(30);
Map.addLayer(steep, {}, ‘steep slope’,false);
var steep_buff = steep.distance(ee.Kernel.euclidean(1000,
‘meters’)).gt(0).remap([0,1],[1,1]).unmask(0);
var pRBA_steep = preRBA.add(steep_buff).mask(preRBA).
unmask(0);
Map.addLayer(pRBA_steep,{},”pRBA + steep”,false);
var pRBA_steep_fmean = pRBA_steep.addBands(preRBA);
print(“pRBA_steep_fmean”,pRBA_steep_fmean);
var pRBA_steep_fmean = pRBA_steep_fmean.
reduceConnectedComponents(ee.Reducer.mean(),
‘remapped_1’, 256);
var pRBA_near_steep = pRBA_steep.neq(pRBA_steep_fmean).
unmask(0);
print(“pRBA_near_steep”,pRBA_near_steep);
Map.addLayer(pRBA_near_steep, {}, ‘preRBA patches near steep’,
false);


// FINALIZE all the layers
// conservations that require mitigation
var con_miti = pRBA_near_steep.add(pRBA_near_water).
add(MPPm.unmask(0)).gt(0);
print(“con_miti”,con_miti);
Map.addLayer(con_miti, {}, ‘conservations that require
mitigation’);
// conservation for sure
var con = MPP_con.add(HPP_1).gt(0);
print(“conservation”, con);
Map.addLayer(con, {}, ‘conservation for sure’);
// development/ give and take
var develop =  preRBA.neq(pRBA_near_steep.add(pRBA_near_
water).gt(0)).add(give_take.unmask(0)).gt(0);
print(“develop”,develop);
Map.addLayer(develop, {}, ‘development’);

mean(), ‘remapped_1’, 256);
var pRBA_near_steep = pRBA_steep.neq(pRBA_steep_fmean).
unmask(0);
print(“pRBA_near_steep”,pRBA_near_steep);
Map.addLayer(pRBA_near_steep, {}, ‘preRBA patches near steep’,
false);


// FINALIZE all the layers
// conservations that require mitigation
var con_miti = pRBA_near_steep.add(pRBA_near_water).
add(MPPm.unmask(0)).gt(0);
print(“con_miti”,con_miti);
Map.addLayer(con_miti, {}, ‘conservations that require
mitigation’);
// conservation for sure
var con = MPP_con.add(HPP_1).gt(0);
print(“conservation”, con);
Map.addLayer(con, {}, ‘conservation for sure’);
// development/ give and take
var develop =  preRBA.neq(pRBA_near_steep.add(pRBA_
near_water).gt(0)).add(give_take.unmask(0)).gt(0);
print(“develop”,develop);
Map.addLayer(develop, {}, ‘development’);

var forden_higher = forest_cov.gt(3.5);
Map.addLayer(forden_higher, {palette:[‘000000’,’fc0303’]},
‘Forest cover more than low’, false);
var MPP_miticheck = MPP_ncon.add(forden_higher).eq(2).
unmask(0);
var MPP_miticheck = MPP_miticheck.mask(MPP_miticheck);
print(MPP_miticheck);
Map.addLayer(MPP_miticheck, {palette:[‘000000’,’03f8fc’]},
‘MPPs check for mitigation’, false);
// check the above MPPs for size (>10 ha)
var MPPm_patchsize = MPP_miticheck.
connectedPixelCount(1024, false);
var MPPm_patcharea = MPPm_patchsize.multiply(ee.Image.
pixelArea());
print(“MPPm_patchsize”, MPPm_patchsize);
print(“MPPm_patcharea”, MPPm_patcharea);
Map.addLayer(MPPm_patcharea, {}, ‘MPPm_
patcharea’,false);
var MPPm = MPPm_patcharea.gt(100000); // greater than
10 ha
var MPPm = MPPm.mask(MPPm);
print(“MPPm”,MPPm);
var MPP_preRBA = MPP_ncon.subtract(MPPm.unmask(0)).
remap([-1,0,1],[0,0,1]);
print(“MPP_preRBA”,MPP_preRBA);
Map.addLayer(MPP_preRBA, {palette:[‘000000’, ‘2b632b’]},
‘MPP for pre-RBA’,false);

// merge all layers subject to Pre-RBA check
var preRBA = MPP_preRBA.unmask(0).add(LPP_pre_RBA.
unmask(0)).gt(0);
print(“preRBA”,preRBA);
Map.addLayer(preRBA, {palette:[‘000000’, ‘2b632b’]}, ‘pre-
RBA patches’,false);

// conduct Pre RBA check
```

5. Appendix

```
// Import water data
var water = ee.ImageCollection("JRC/GSW1_1/YearlyHistory").
select('waterClass').filter(ee.Filter.date('2018-01-01', '2019-
12-31'));
var water = water.mean().eq(3);
Map.addLayer(water, {palette: '036ffc'}, 'water',false);

// check proximity to water
var water_buff = water.distance(ee.Kernel.euclidean(1000,
'meters')).gt(0).remap([0,1],[1,1]).unmask(0);
var pRBA_water = preRBA.add(water_buff).mask(preRBA).
unmask(0);
Map.addLayer(pRBA_water,{},"pRBA + water",false);
var pRBA_water_fmean = pRBA_water.addBands(preRBA);
print("pRBA_water_fmean",pRBA_water_fmean);
var pRBA_water_fmean = pRBA_water_fmean.
reduceConnectedComponents(ee.Reducer.mean(),
'remapped_1', 256);
var pRBA_near_water = pRBA_water.neq(pRBA_water_fmean).
unmask(0);
print("pRBA_near_water",pRBA_near_water);
Map.addLayer(pRBA_near_water, {}, 'preRBA patches near
water', false);

// Import dem data
var elevation = ee.Image('USGS/GMTED2010').select('be75');
var elevationVis = {
  min: -100.0,
  max: 5000.0,
  gamma: 3.5,
};
Map.addLayer(elevation, elevationVis, 'Elevation',false);
// check slope
var slope = ee.Terrain.slope(elevation);
// find where slope is greater than 45 degrees
var steep = slope.gt(30);
Map.addLayer(steep, {}, 'steep slope',false);
var steep_buff = steep.distance(ee.Kernel.euclidean(1000,
'meters')).gt(0).remap([0,1],[1,1]).unmask(0);
var pRBA_steep = preRBA.add(steep_buff).mask(preRBA).
unmask(0);
Map.addLayer(pRBA_steep,{},"pRBA + steep",false);
var pRBA_steep_fmean = pRBA_steep.addBands(preRBA);
print("pRBA_steep_fmean",pRBA_steep_fmean);
var pRBA_steep_fmean = pRBA_steep_fmean.
reduceConnectedComponents(ee.Reducer.mean(), 'remapped_1',
256);
var pRBA_near_steep = pRBA_steep.neq(pRBA_steep_fmean).
unmask(0);
print("pRBA_near_steep",pRBA_near_steep);
Map.addLayer(pRBA_near_steep, {}, 'preRBA patches near steep',
false);


// FINALIZE all the layers
// conservations that require mitigation
var con_miti = pRBA_near_steep.add(pRBA_near_water).
add(MPPm.unmask(0)).gt(0);
print("con_miti",con_miti);
Map.addLayer(con_miti, {}, 'conservations that require
mitigation');
// conservation for sure
var con = MPP_con.add(HPP_1).gt(0);
print("conservation", con);
Map.addLayer(con, {}, 'conservation for sure');
// development/ give and take
var develop =  preRBA.neq(pRBA_near_steep.add(pRBA_near_
water).gt(0)).add(give_take.unmask(0)).gt(0);
print("develop",develop);
Map.addLayer(develop, {}, 'development');
```