Restoring Floodplains in the Conterminous US

Yujing Wu

LARP 743

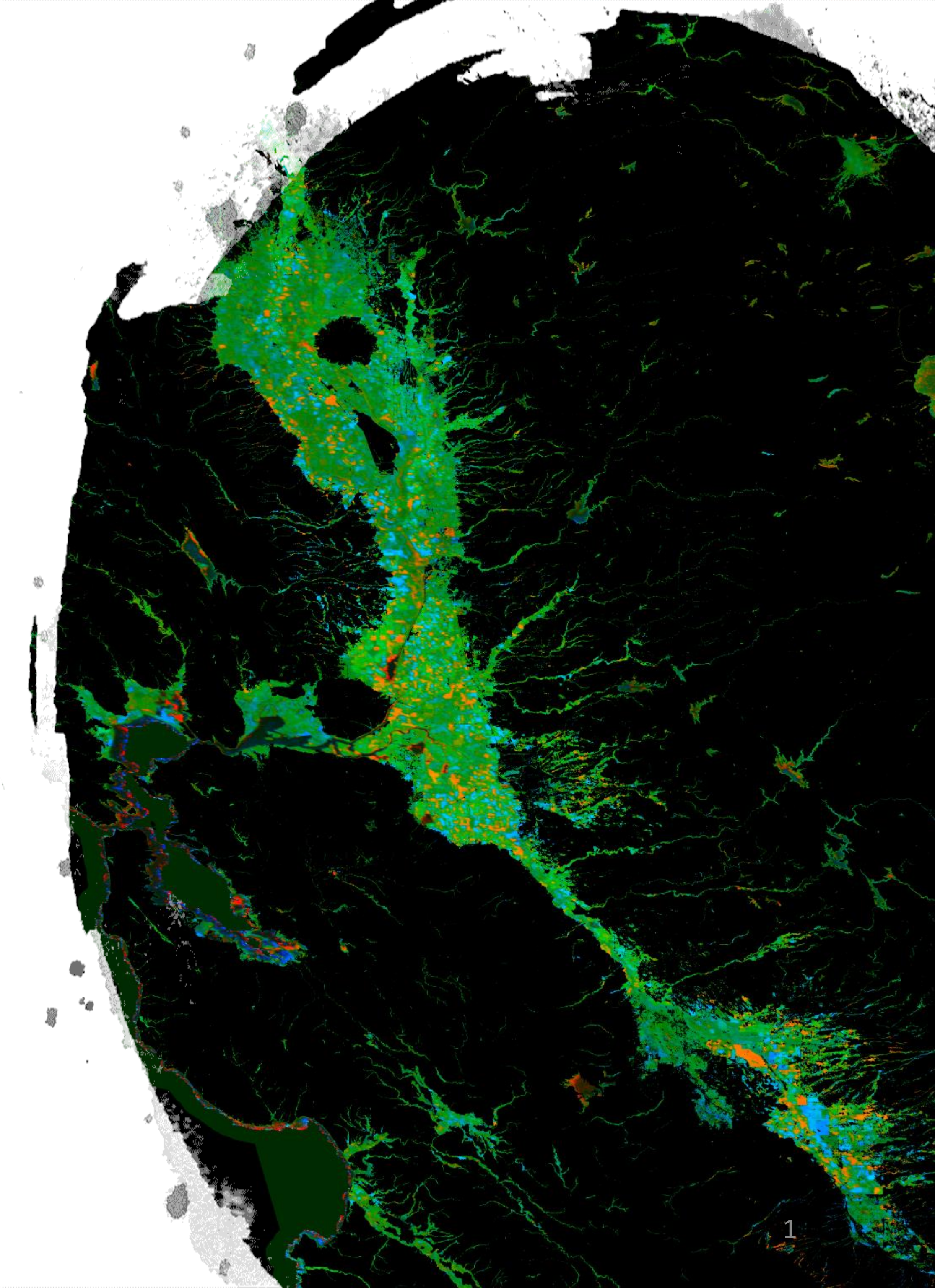
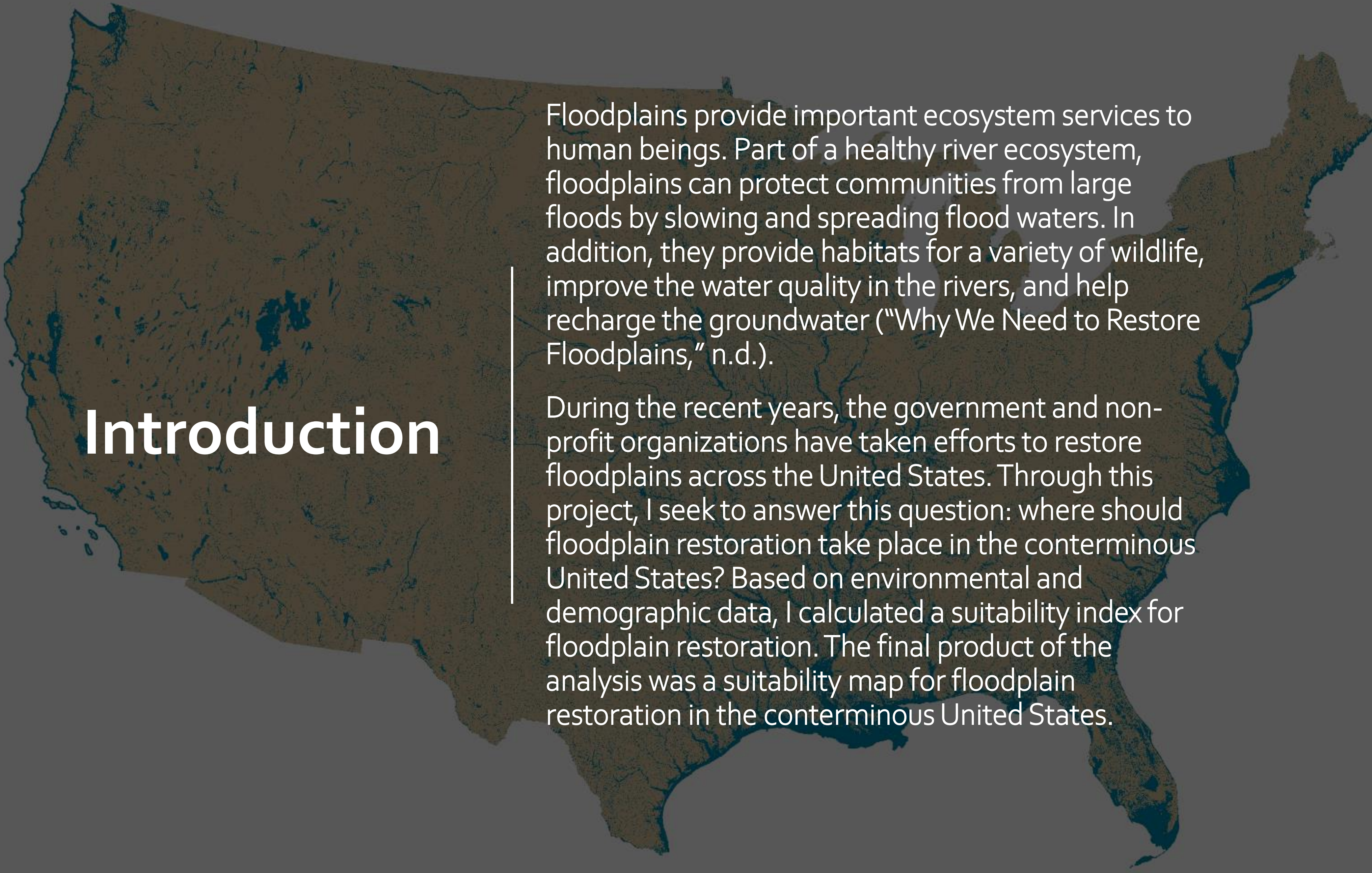


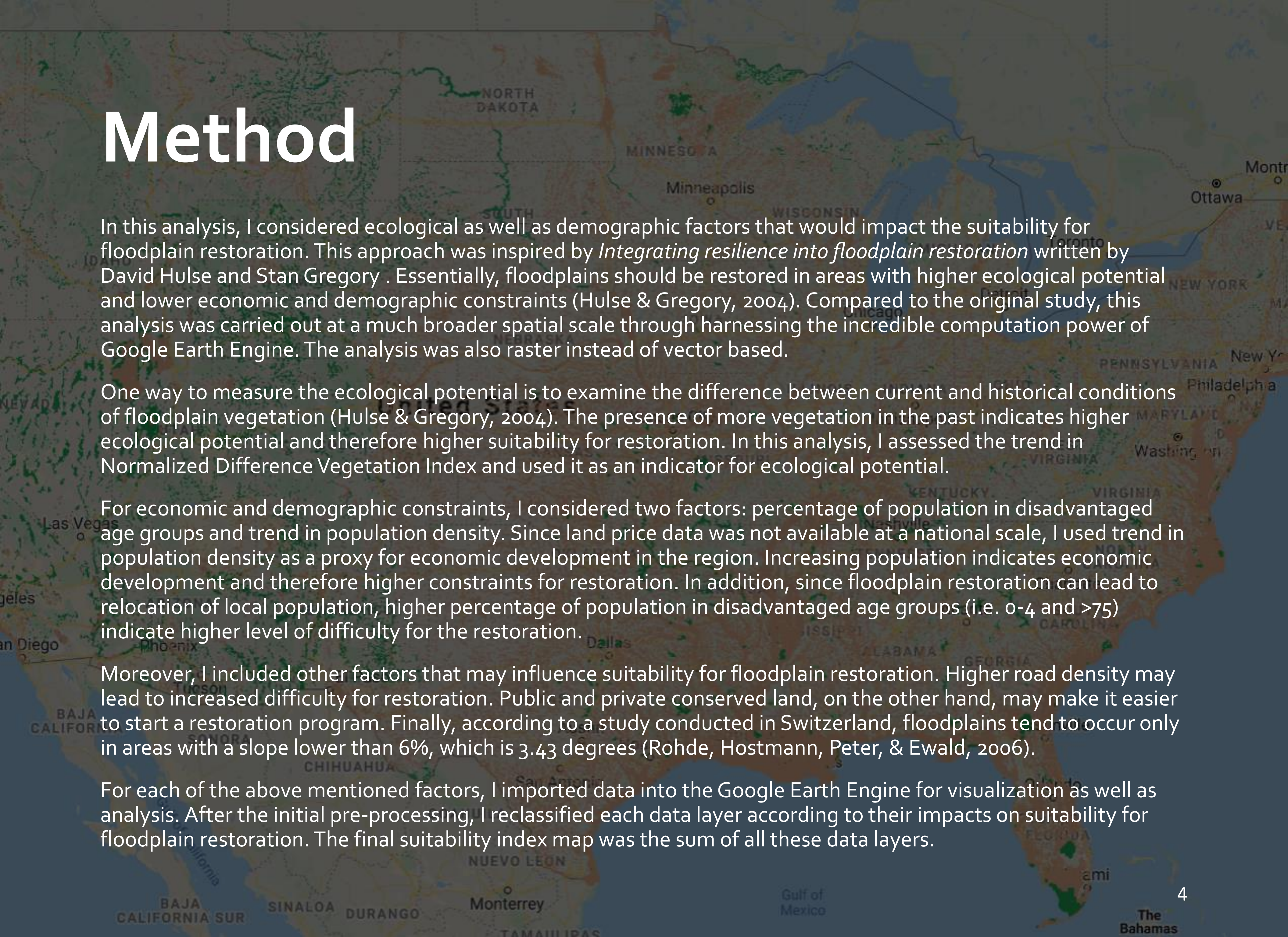
Table of Content



Introduction

Floodplains provide important ecosystem services to human beings. Part of a healthy river ecosystem, floodplains can protect communities from large floods by slowing and spreading flood waters. In addition, they provide habitats for a variety of wildlife, improve the water quality in the rivers, and help recharge the groundwater ("Why We Need to Restore Floodplains," n.d.).

During the recent years, the government and non-profit organizations have taken efforts to restore floodplains across the United States. Through this project, I seek to answer this question: where should floodplain restoration take place in the conterminous United States? Based on environmental and demographic data, I calculated a suitability index for floodplain restoration. The final product of the analysis was a suitability map for floodplain restoration in the conterminous United States.



Method

In this analysis, I considered ecological as well as demographic factors that would impact the suitability for floodplain restoration. This approach was inspired by *Integrating resilience into floodplain restoration* written by David Hulse and Stan Gregory . Essentially, floodplains should be restored in areas with higher ecological potential and lower economic and demographic constraints (Hulse & Gregory, 2004). Compared to the original study, this analysis was carried out at a much broader spatial scale through harnessing the incredible computation power of Google Earth Engine. The analysis was also raster instead of vector based.

One way to measure the ecological potential is to examine the difference between current and historical conditions of floodplain vegetation (Hulse & Gregory, 2004). The presence of more vegetation in the past indicates higher ecological potential and therefore higher suitability for restoration. In this analysis, I assessed the trend in Normalized Difference Vegetation Index and used it as an indicator for ecological potential.

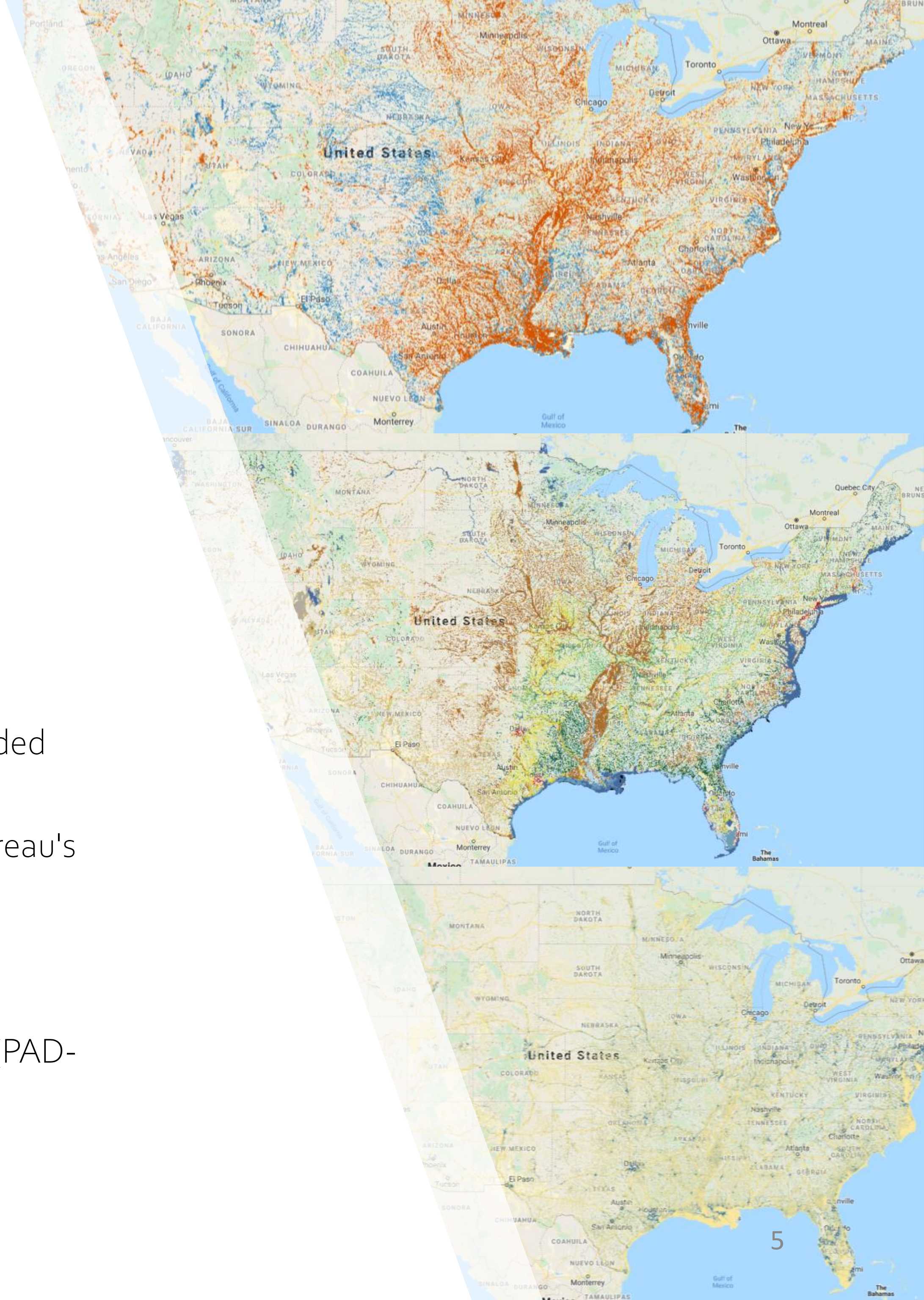
For economic and demographic constraints, I considered two factors: percentage of population in disadvantaged age groups and trend in population density. Since land price data was not available at a national scale, I used trend in population density as a proxy for economic development in the region. Increasing population indicates economic development and therefore higher constraints for restoration. In addition, since floodplain restoration can lead to relocation of local population, higher percentage of population in disadvantaged age groups (i.e. 0-4 and >75) indicate higher level of difficulty for the restoration.

Moreover, I included other factors that may influence suitability for floodplain restoration. Higher road density may lead to increased difficulty for restoration. Public and private conserved land, on the other hand, may make it easier to start a restoration program. Finally, according to a study conducted in Switzerland, floodplains tend to occur only in areas with a slope lower than 6%, which is 3.43 degrees (Rohde, Hostmann, Peter, & Ewald, 2006).

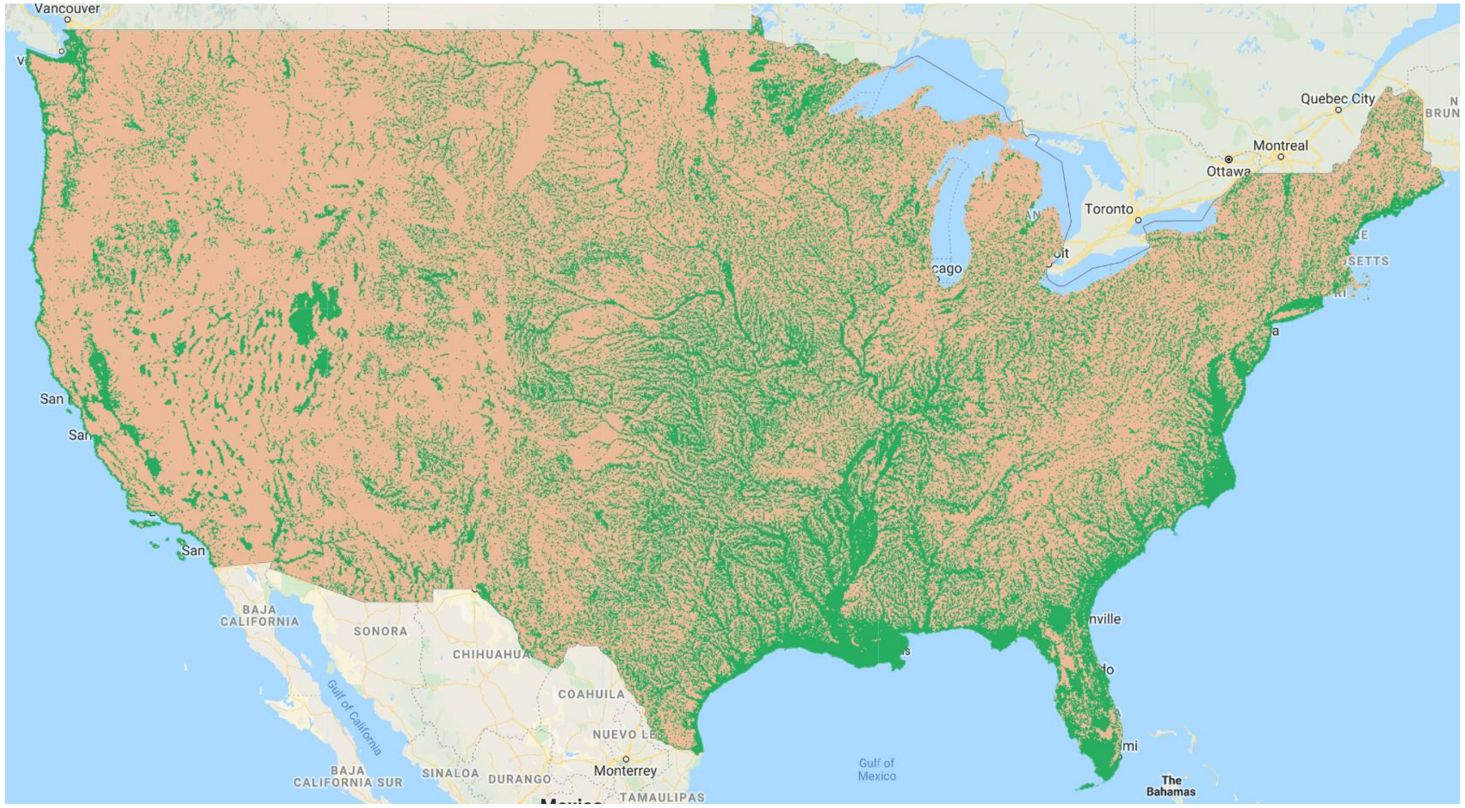
For each of the above mentioned factors, I imported data into the Google Earth Engine for visualization as well as analysis. After the initial pre-processing, I reclassified each data layer according to their impacts on suitability for floodplain restoration. The final suitability index map was the sum of all these data layers.

Data

- Estimated floodplain map for the conterminous United States
- MOD13Q1.006 Terra Vegetation Indices 16-Day Global 250m
- NLCD: USGS National Land Cover Database
- ALOS DSM: Global 30m
- GPWv411: UN-Adjusted Population Count (Gridded Population of the World Version 4.11)
- USA Road Density derived from U.S. Census Bureau's 2014 TIGER database
- GPWv411: Basic Demographic Characteristics (Gridded Population of the World Version 4.11)
- Protected Areas Database of the United States (PAD-US) version 2.0 dataset



Study Region

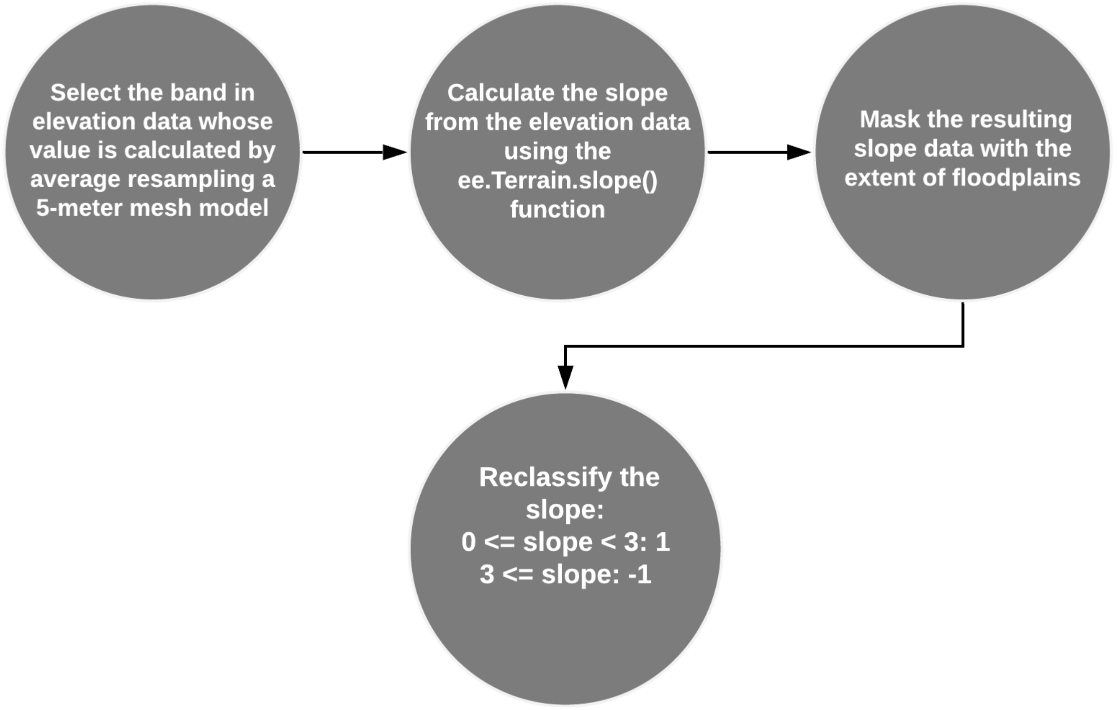


🟧 non-floodplain 🟩 floodplain

The study region of the suitability analysis is determined by the estimated floodplain map for the conterminous United States created by U.S. EPA. The U.S. Federal Emergency Management Agency has created Flood Insurance Rate Maps for only 60% of the conterminous United States. Thus, I have chosen to use this dataset since it provides an estimate of floodplain extent for the whole conterminous U.S (Woznicki, Baynes, Panlasigui, Mehaffey, & Neale, 2019).

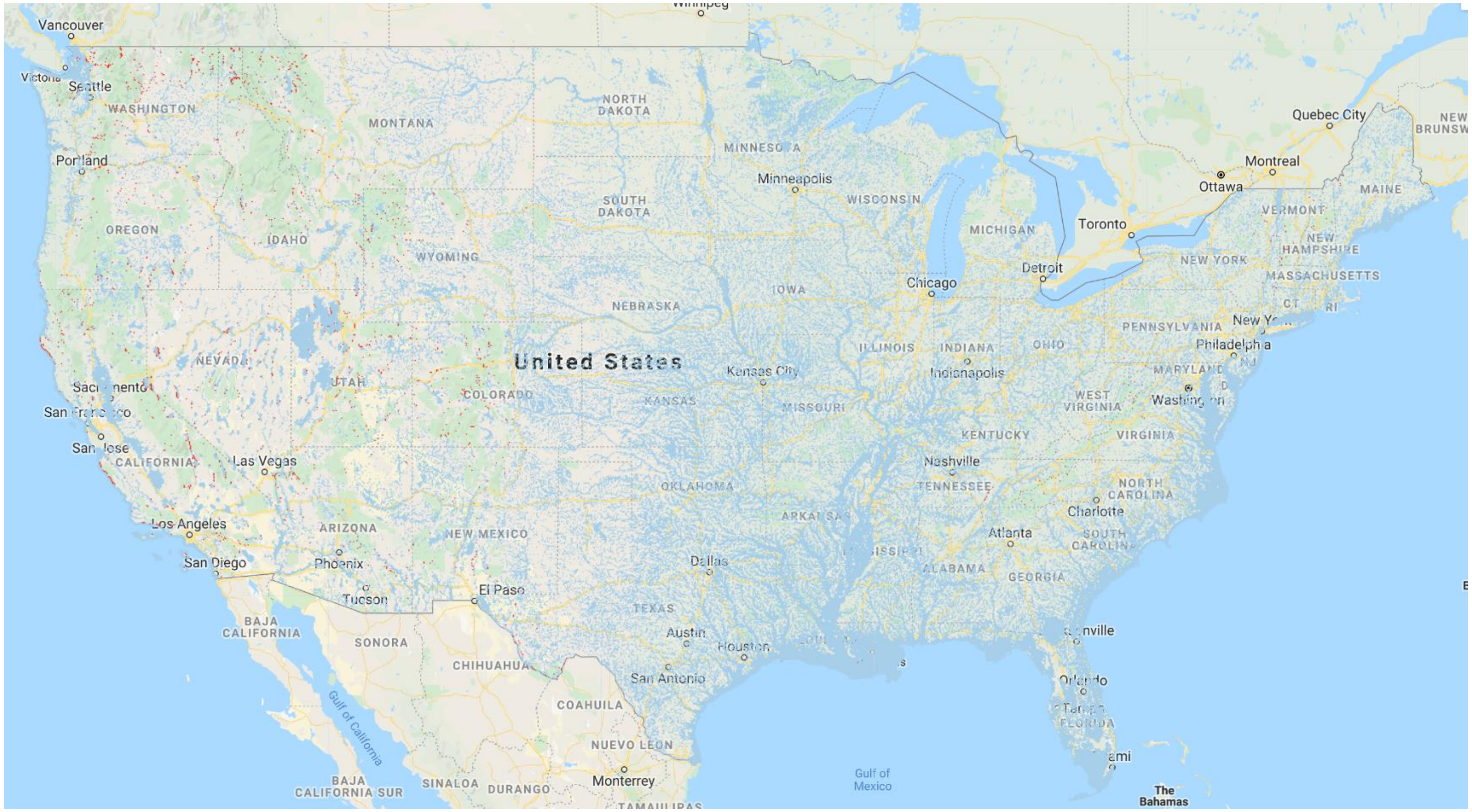
```
// Import the downloaded dataset.
var Extent =
ee.Image("users/sarahwyj97/Estimated_floodplain_CONUS");
```

Slope


Select the band in elevation data whose value is calculated by average resampling a 5-meter mesh model
Calculate the slope from the elevation data using the ee.Terrain.slope() function
Mask the resulting slope data with the extent of floodplains
Reclassify the slope:
0 <= slope < 3: 1
3 <= slope: -1

```
// Import data.
var DEM = ee.Image("JAXA/ALOS/AW3D30_V1_1").select('AVE');

// Calculate slope.
var slope = ee.Terrain.slope(DEM);
var slope_floodplains = slope.mask(Extent);

// Good for restoration: below 3.43 degree (6%)
var suitable_mask =
slope_floodplains.gte(0).and(slope_floodplains.lt(3));
var slope_suitable = slope_floodplains.where(suitable_mask,
1).mask(suitable_mask).toInt();

// Bad for restoration: higher than or equal to 3.43
degree (6%)
var unsuitable_mask = slope_floodplains.gte(3);
var slope_unsuitable =
slope_floodplains.where(unsuitable_mask, -
1).mask(unsuitable_mask).toInt();

var slope_reclassed = ee.ImageCollection([slope_suitable,
slope_unsuitable]).reduce(ee.Reducer.max());
```

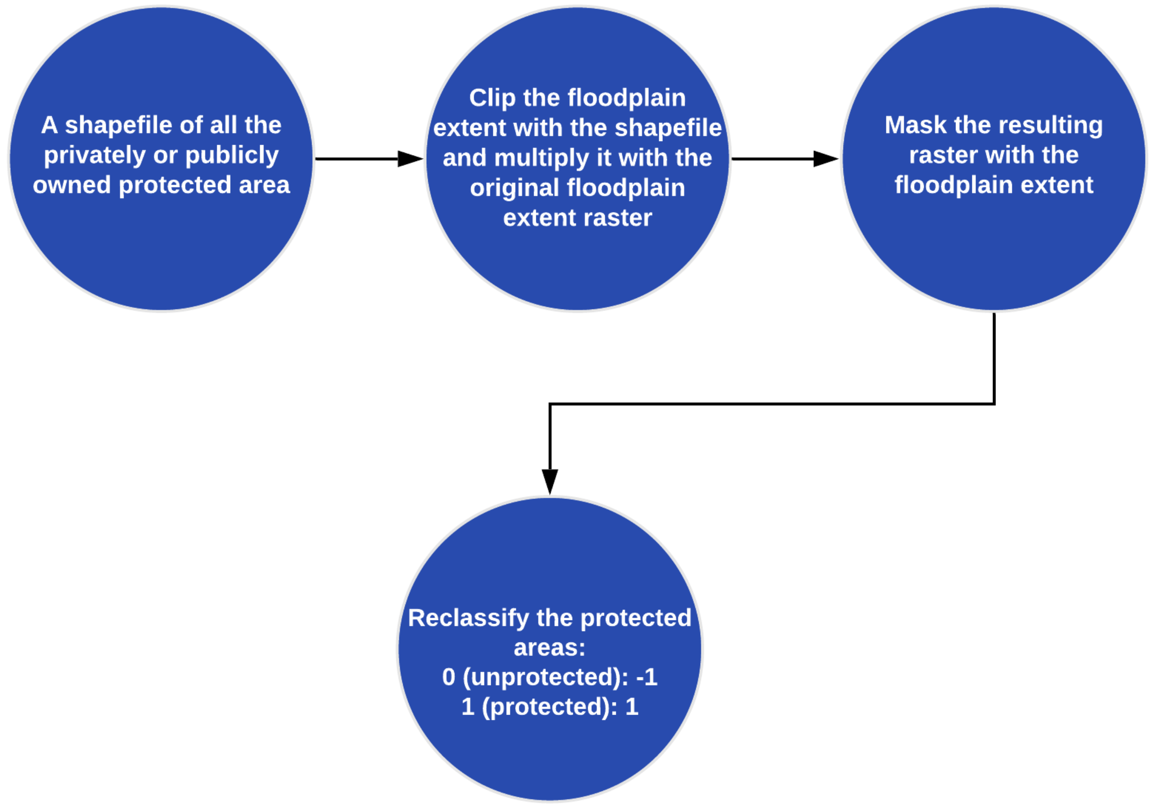
Slope



Reclassified Slope

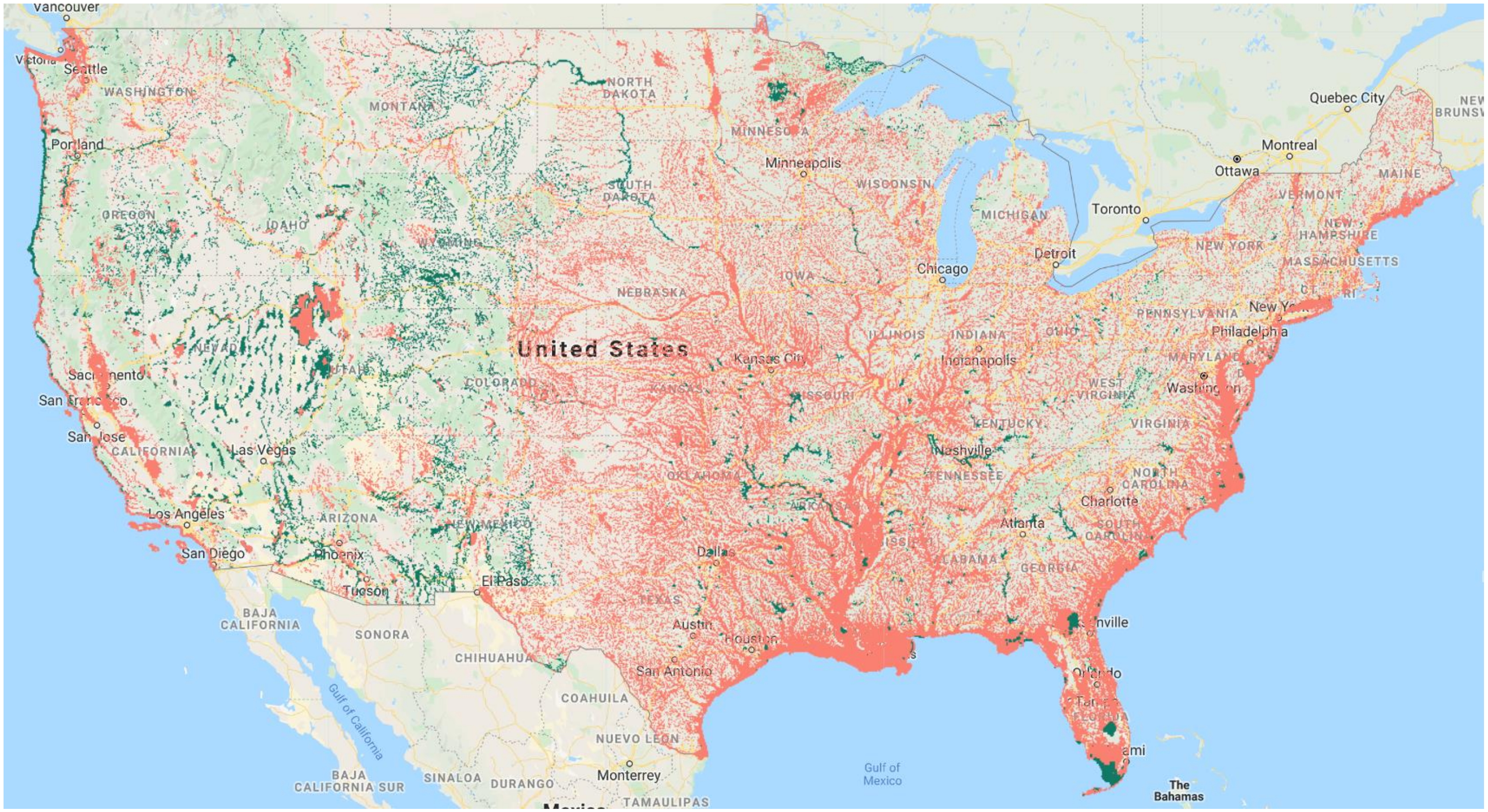
- 🟦 slope < 3 degree
- 🟥 slope >= 3 degree

Public/Private Protected Area


A shapefile of all the privately or publicly owned protected area
Clip the floodplain extent with the shapefile and multiply it with the original floodplain extent raster
Mask the resulting raster with the floodplain extent
Reclassify the protected areas:
0 (unprotected): -1
1 (protected): 1

```
// Import the data.
var LandOwnership =
ee.FeatureCollection("users/sarahwyj97/LandOwnership");

// High opportunity for restoration: publicly/privately owned conserved land; low opportunity for restoration: non-conserved land
var conserved_reclassified =
floodplain_conserved.remap([0, 1], [-1, 1]);
```

Public/Private Protected Area

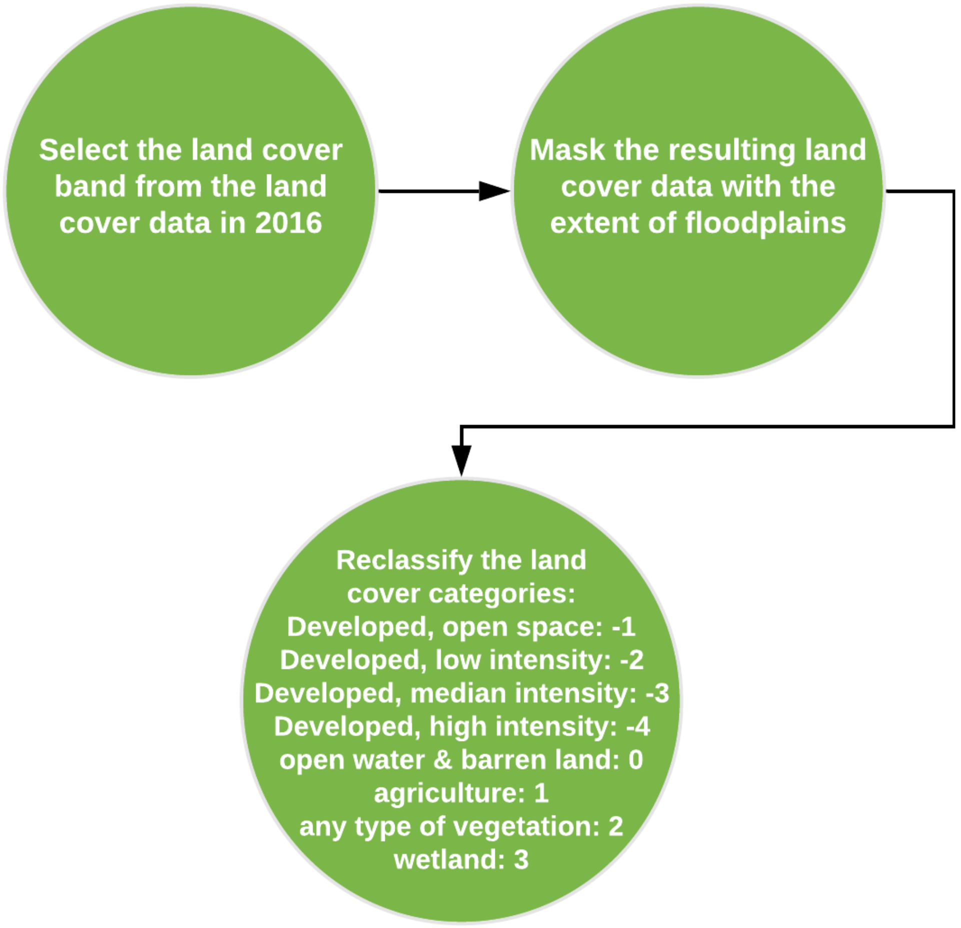


Reclassified Protected Area

🟥 non-protected area

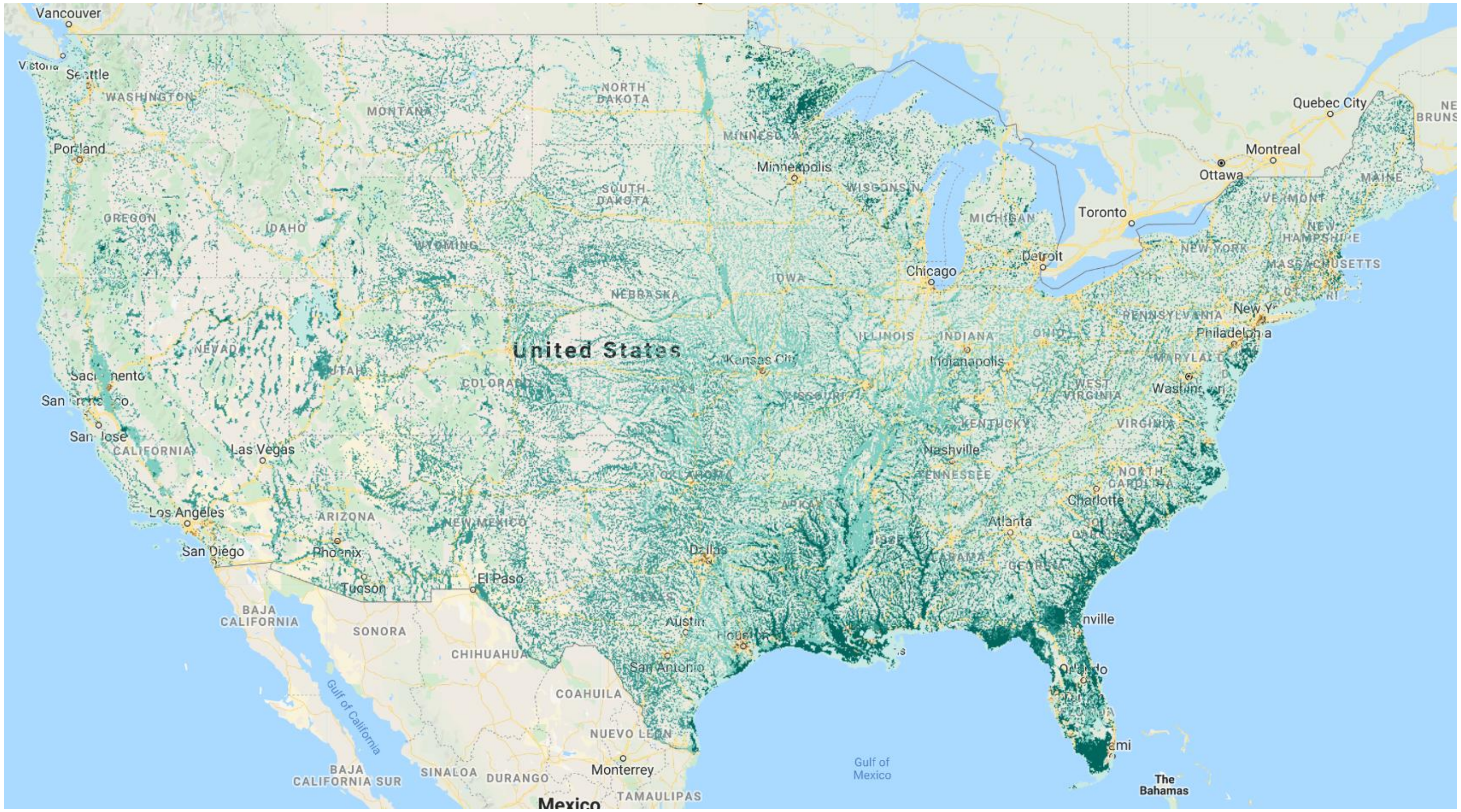
🟩 protected area

Land Cover


Select the land cover band from the land cover data in 2016
Mask the resulting land cover data with the extent of floodplains
Reclassify the land cover categories:
Developed, open space: -1
Developed, low intensity: -2
Developed, median intensity: -3
Developed, high intensity: -4
open water & barren land: 0
agriculture: 1
any type of vegetation: 2
wetland: 3

```
// Import the image collection and filter the data to get land cover in 2016.
var Landcover = ee.ImageCollection("USGS/NLCD").filterDate
('2016-01-01', '2016-12-31;

// Select the land cover band.
var lc = Landcover.select('landcover');
var lc_floodplains = lc.first().mask(Extent);

// High opportunity for restoration: vegetated; low opportunity for restoration: developed
var lc_reclassified =
lc_floodplains.remap([11,12,21,22,23,24,31,41,42,43,52,71,81,8
2,90,95],
[0,0,-1,-2,-3,-4,0,2,2,2,2,2,1,1,3,3]);
```

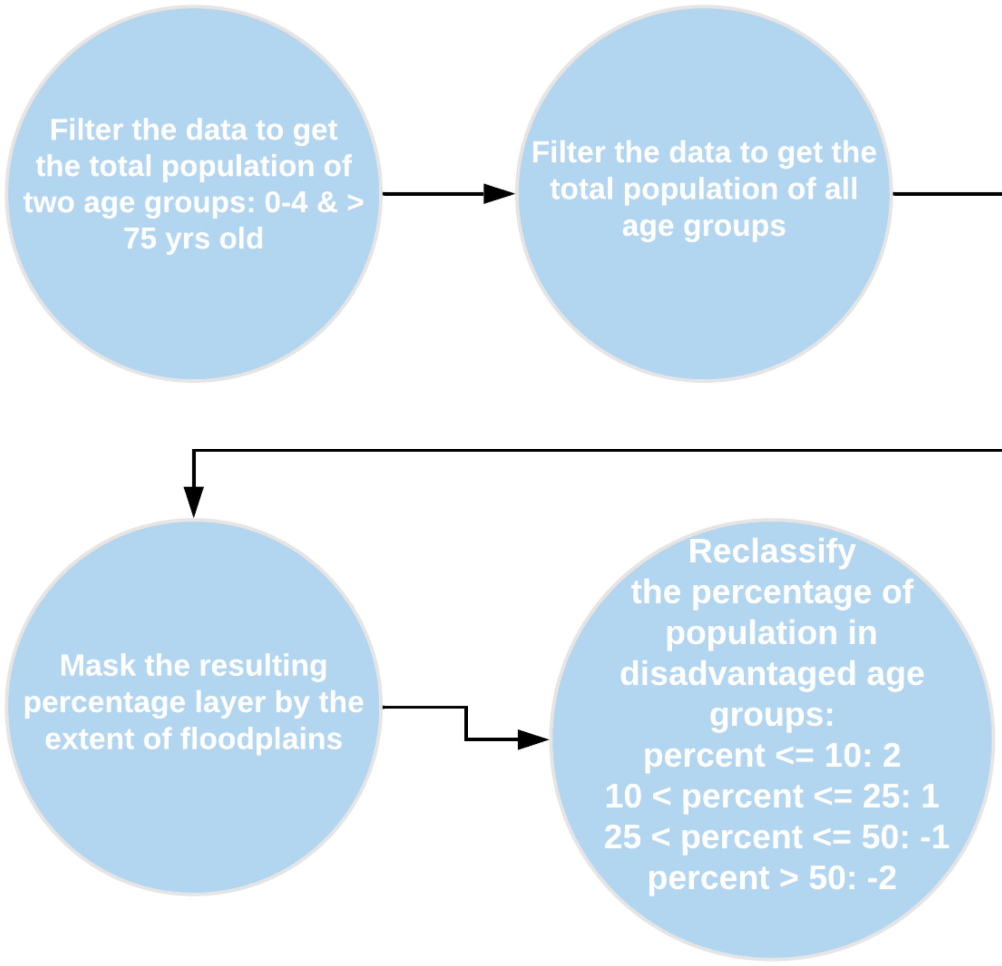
Land Cover



Reclassified Landcover

- 🟫 high intensity development
- 🟧 medium intensity development
- 🟨 low intensity development
- ⬜ development open space
- ⬜ water / barren land
- 🟩 agriculture
- 🟩 vegetation
- 🟩 wetland

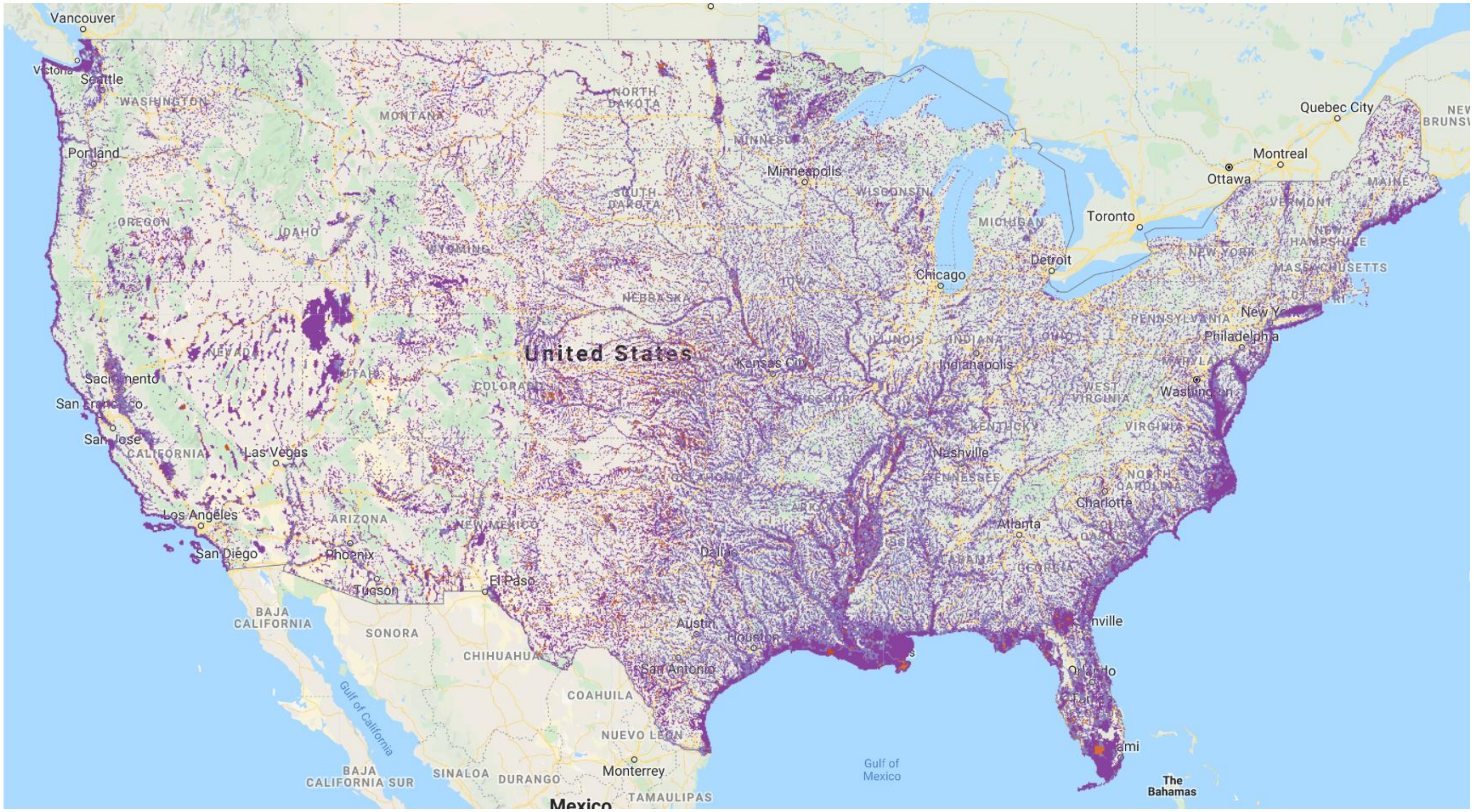
Percentage of Population in Disadvantaged Age Groups


Filter the data to get the total population of two age groups: 0-4 & > 75 yrs old
Filter the data to get the total population of all age groups
Calculate the percentage by dividing the disadvantaged population by the total population
Mask the resulting percentage layer by the extent of floodplains
Reclassify the percentage of population in disadvantaged age groups:
percent <= 10: 2
10 < percent <= 25: 1
25 < percent <= 50: -1
percent > 50: -2

Percentage of Population in Disadvantaged Age Groups

```
// Import the data.
var Age =
ee.ImageCollection("CIESIN/GPWv411/GPW_Basic_Demographic_Characteristics");

// Find the percentage of population of
disadvantaged age groups: 0-4 and > 75 years old.
var Age_filter =
ee.Filter.and(ee.Filter.or(ee.Filter.eq('Age_Group', '0-4'),
ee.Filter.eq('Age_Group', '75-')), ee.Filter.eq('Sex', 'b'));
var Age_vulnerable = Age.filter(Age_filter).sum();
var Pop_filter = ee.Filter.and(ee.Filter.eq('Age_Group', 'Total
Population'), ee.Filter.eq('Sex', 'b'));
var Total_pop = Age.filter(Pop_filter).first();
var Age_pctvulnerable =
Age_vulnerable.divide(Total_pop).multiply(100);
var Age_pctvulnerable_fp =
Age_pctvulnerable.mask(Extent);

// High opportunity for restoration: lower than or
equal to 25%
var suitable_mask1 = Age_pctvulnerable_fp.lte(10);
var Age_suitable1 =
Age_pctvulnerable_fp.where(suitable_mask1,
2).mask(suitable_mask1).toInt();
var suitable_mask2 =
Age_pctvulnerable_fp.gt(10).and(Age_pctvulnerable_fp.lte(
25));
var Age_suitable2 =
Age_pctvulnerable_fp.where(suitable_mask2,
1).mask(suitable_mask2).toInt();

// Low opportunity for restoration: higher than
25%
var unsuitable_mask1 =
Age_pctvulnerable_fp.gt(25).and(Age_pctvulnerable_fp.lte(
50));
var Age_unsuitable1 =
Age_pctvulnerable_fp.where(unsuitable_mask1, -
1).mask(unsuitable_mask1).toInt();

var unsuitable_mask2 = Age_pctvulnerable_fp.gte(50);
var Age_unsuitable2 =
Age_pctvulnerable_fp.where(unsuitable_mask2, -
2).mask(unsuitable_mask2).toInt();

var Age_pct_reclassed = ee.ImageCollection([Age_suitable1,
Age_suitable2, Age_unsuitable1,
Age_unsuitable2]).reduce(ee.Reducer.max());
```

Percentage of Population in Disadvantaged Age Groups



Reclassified Percentage of Population in Disadvantaged Age Groups

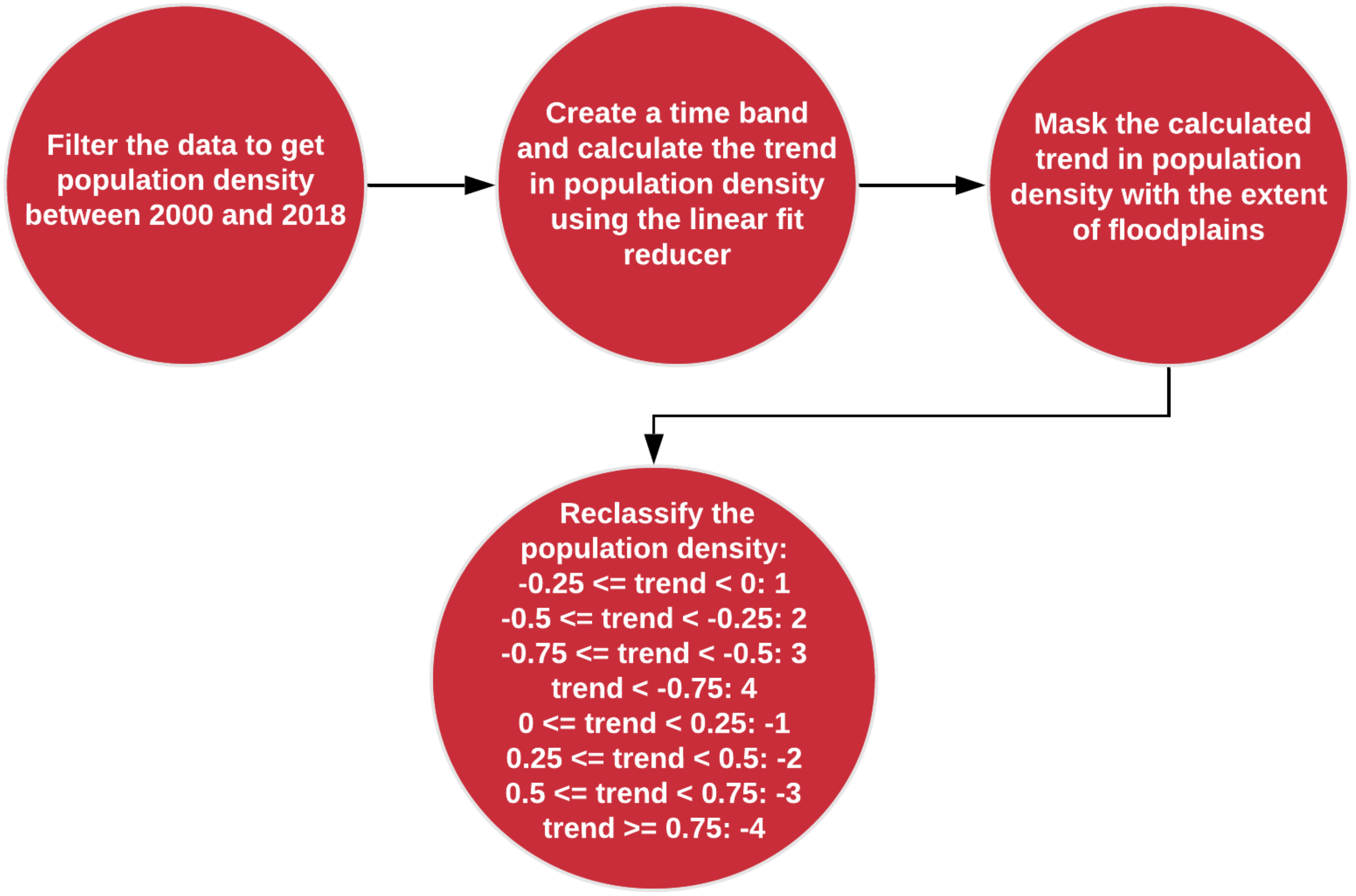
🟥 50% <= percent

🟧 25% <= percent < 50%

🟦 10% <= percent < 25%

🟪 percent < 10%

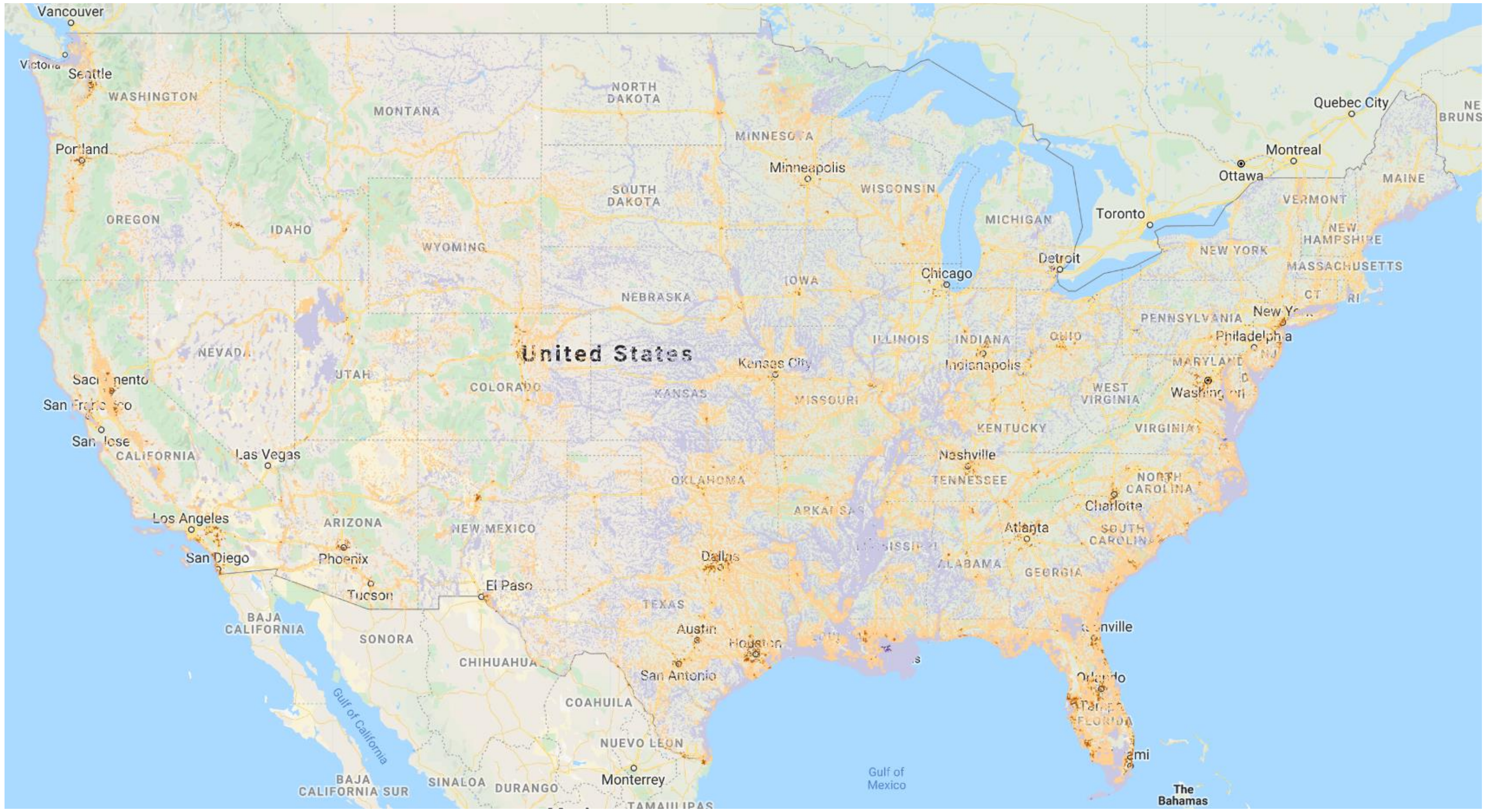
Trend in Population Density


Filter the data to get population density between 2000 and 2018
Create a time band and calculate the trend in population density using the linear fit reducer
Mask the calculated trend in population density with the extent of floodplains
Reclassify the population density:
-0.25 <= trend < 0: 1
-0.5 <= trend < -0.25: 2
-0.75 <= trend < -0.5: 3
trend < -0.75: 4
0 <= trend < 0.25: -1
0.25 <= trend < 0.5: -2
0.5 <= trend < 0.75: -3
trend >= 0.75: -4

Trend in Population Density

```
// Import the data and filter it according to time.
var Popdensity = ee.ImageCollection("CIESIN/GPWv411/GPW_UNWPP-
Adjusted_Population_Count").filterDate ('2000-01-01', '2018-12-31’);

// Calculate trend in population density
var collection_pop = Popdensity.select('unwpp-
adjusted_population_count').map(createTimeBand);
var fit_pop = collection_pop.reduce(ee.Reducer.linearFit());
var fit_pop_floodplains = fit_pop.mask(Extent);
var trend_popdensity = fit_pop_floodplains.select(['scale']);

// High opportunity for restoration: lower than or equal to 0
var suitable_mask1 = trend_popdensity.gte(-
0.25).and(trend_popdensity.lte(0));
var pop_suitable1 = trend_popdensity.where(suitable_mask1,
1).mask(suitable_mask1).toInt();

var suitable_mask2 = trend_popdensity.gte(-
0.5).and(trend_popdensity.lt(-0.25));
var pop_suitable2 = trend_popdensity.where(suitable_mask2,
2).mask(suitable_mask2).toInt();

var suitable_mask3 = trend_popdensity.gte(-
0.75).and(trend_popdensity.lt(-0.5));
var pop_suitable3 = trend_popdensity.where(suitable_mask3,
3).mask(suitable_mask3).toInt();

var suitable_mask4 = trend_popdensity.lt(-0.75);
var pop_suitable4 = trend_popdensity.where(suitable_mask4,
4).mask(suitable_mask4).toInt();

// Low opportunity for restoration: higher than 0
var unsuitable_mask1 =
trend_popdensity.gt(0).and(trend_popdensity.lt(0.25));
var pop_unsuitable1 = trend_popdensity.where(unsuitable_mask1, -
1).mask(unsuitable_mask1).toInt();

var unsuitable_mask2 =
trend_popdensity.gte(0.25).and(trend_popdensity.lt(0.5));
var pop_unsuitable2 = trend_popdensity.where(unsuitable_mask2, -
2).mask(unsuitable_mask2).toInt();

var unsuitable_mask3 =
trend_popdensity.gte(0.5).and(trend_popdensity.lt(0.75));
var pop_unsuitable3 = trend_popdensity.where(unsuitable_mask3, -
3).mask(unsuitable_mask3).toInt();

var unsuitable_mask4 = trend_popdensity.gte(0.75);
var pop_unsuitable4 = trend_popdensity.where(unsuitable_mask4, -
4).mask(unsuitable_mask4).toInt();

var popdensity_reclassed = ee.ImageCollection([pop_suitable1,
pop_suitable2, pop_suitable3, pop_suitable4, pop_unsuitable1,
pop_unsuitable2, pop_unsuitable3,
pop_unsuitable4]).reduce(ee.Reducer.max());
```

Trend in Population Density



Reclassified Trend in Population Density

🟫 trend >= 0.75

🟧 0.75 > trend >= 0.5

🟧 0.5 > trend >= 0.25

🟨 0.25 > trend >= 0

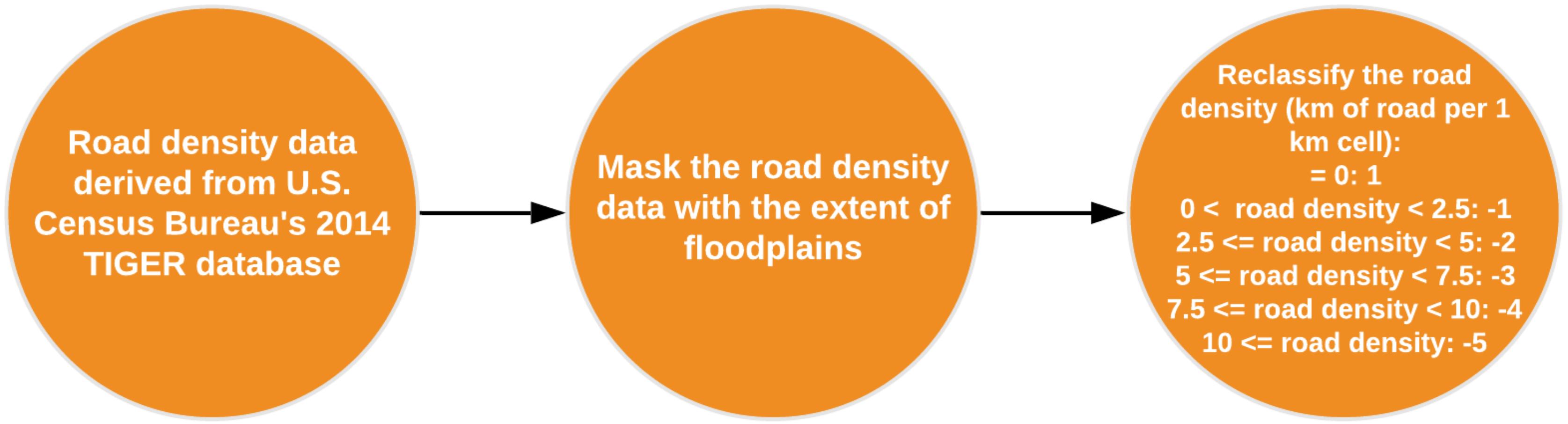
⬜ 0> trend >= -0.25

🟪 -0.25 > trend >= -0.5

🟪 - 0.5 > trend >= -0.75

🟪 - 0.75 > trend

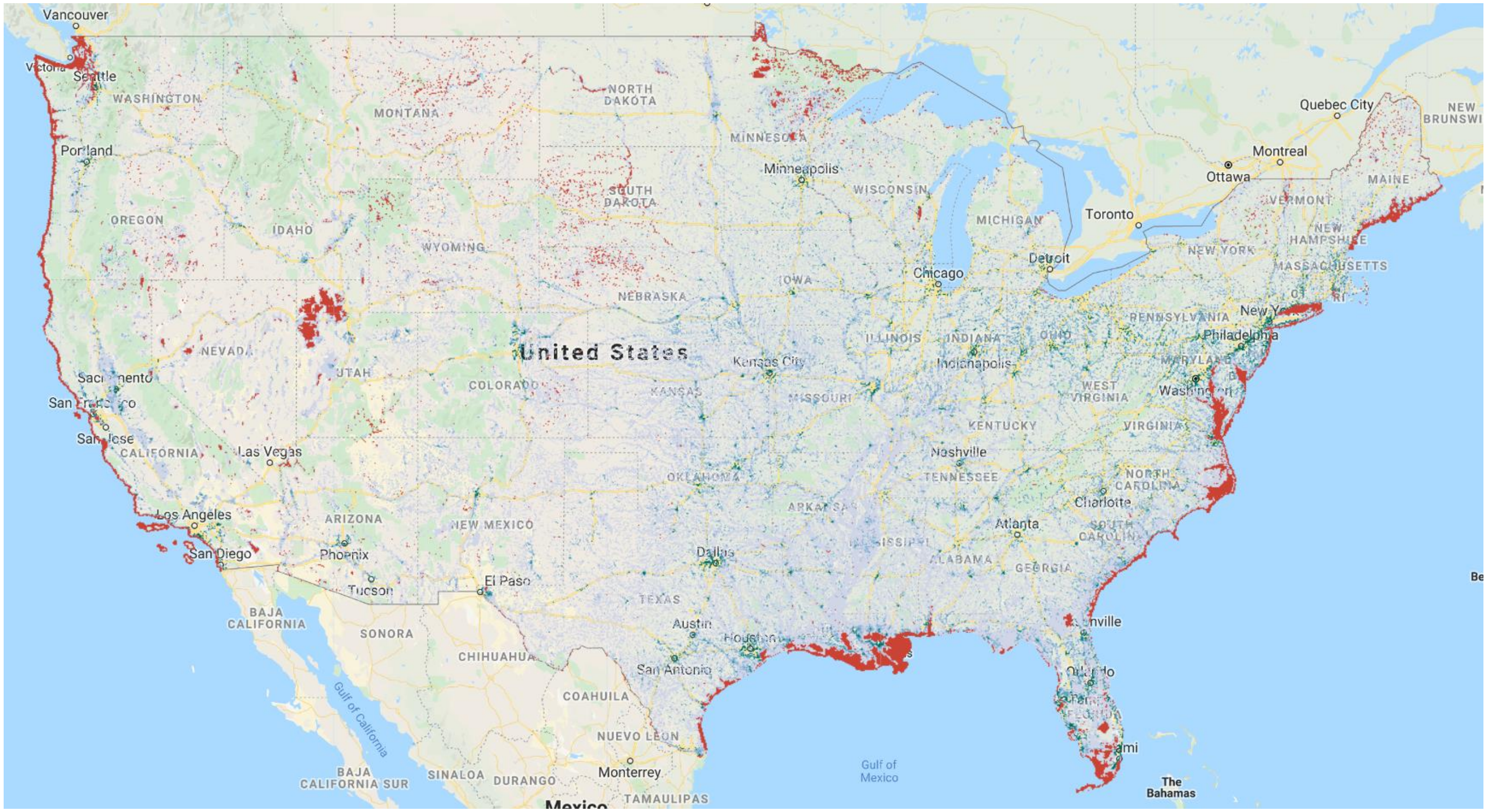
Road Density


Road density data derived from U.S. Census Bureau's 2014 TIGER database
Mask the road density data with the extent of floodplains
Reclassify the road density (km of road per 1 km cell):
= 0: 1
0 < road density < 2.5: -1
2.5 <= road density < 5: -2
5 <= road density < 7.5: -3
7.5 <= road density < 10: -4
10 <= road density: -5

Road Density

```
// Import data.
var RoadDensity = ee.Image("users/sarahwyj97/RoadDensity");

// Mask the road density with floodplain extent.
var RoadDensity_fp = RoadDensity.mask(Extent);

// High opportunity for restoration: equal to 0
var suitable_mask = RoadDensity_fp.eq(0);
var Road_suitable = RoadDensity_fp.where(suitable_mask,
1).mask(suitable_mask).toInt();

// Low opportunity for restoration: higher than 0
var unsuitable_mask1 =
RoadDensity_fp.gt(0).and(RoadDensity_fp.lt(2.5));
var Road_unsuitable1 = RoadDensity_fp.where(unsuitable_mask1, -
1).mask(unsuitable_mask1).toInt();

var unsuitable_mask2 =
RoadDensity_fp.gte(2.5).and(RoadDensity_fp.lt(5));
var Road_unsuitable2 = RoadDensity_fp.where(unsuitable_mask2, -
2).mask(unsuitable_mask2).toInt();

var unsuitable_mask3 =
RoadDensity_fp.gte(5).and(RoadDensity_fp.lt(7.5));
var Road_unsuitable3 = RoadDensity_fp.where(unsuitable_mask3, -
3).mask(unsuitable_mask3).toInt();

var unsuitable_mask4 =
RoadDensity_fp.gte(7.5).and(RoadDensity_fp.lt(10));
var Road_unsuitable4 = RoadDensity_fp.where(unsuitable_mask4, -
4).mask(unsuitable_mask4).toInt();

var unsuitable_mask5 = RoadDensity_fp.gte(10);
var Road_unsuitable5 = RoadDensity_fp.where(unsuitable_mask5, -
5).mask(unsuitable_mask5).toInt();

var Road_reclassed = ee.ImageCollection([Road_suitable,
Road_unsuitable1, Road_unsuitable2, Road_unsuitable3,
Road_unsuitable4, Road_unsuitable5]).reduce(ee.Reducer.max());
```

Road Density



Reclassified Road Density

🟩 road density >= 10

🟦 10 > road density >= 7.5

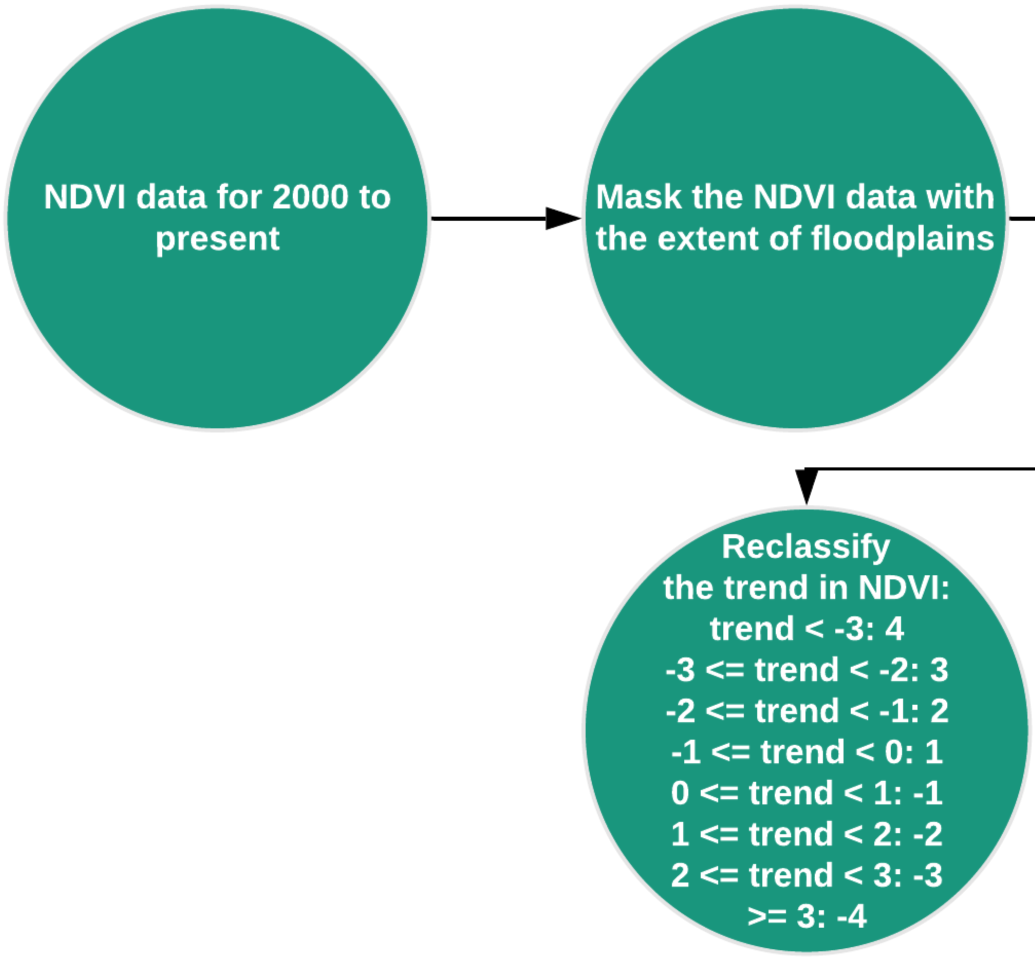
🟦 7.5 > road density >= 5

⬜ 5 > road density >= 2.5

⬜ 2.5 > road density >= 0

🟥 2.5 > road density

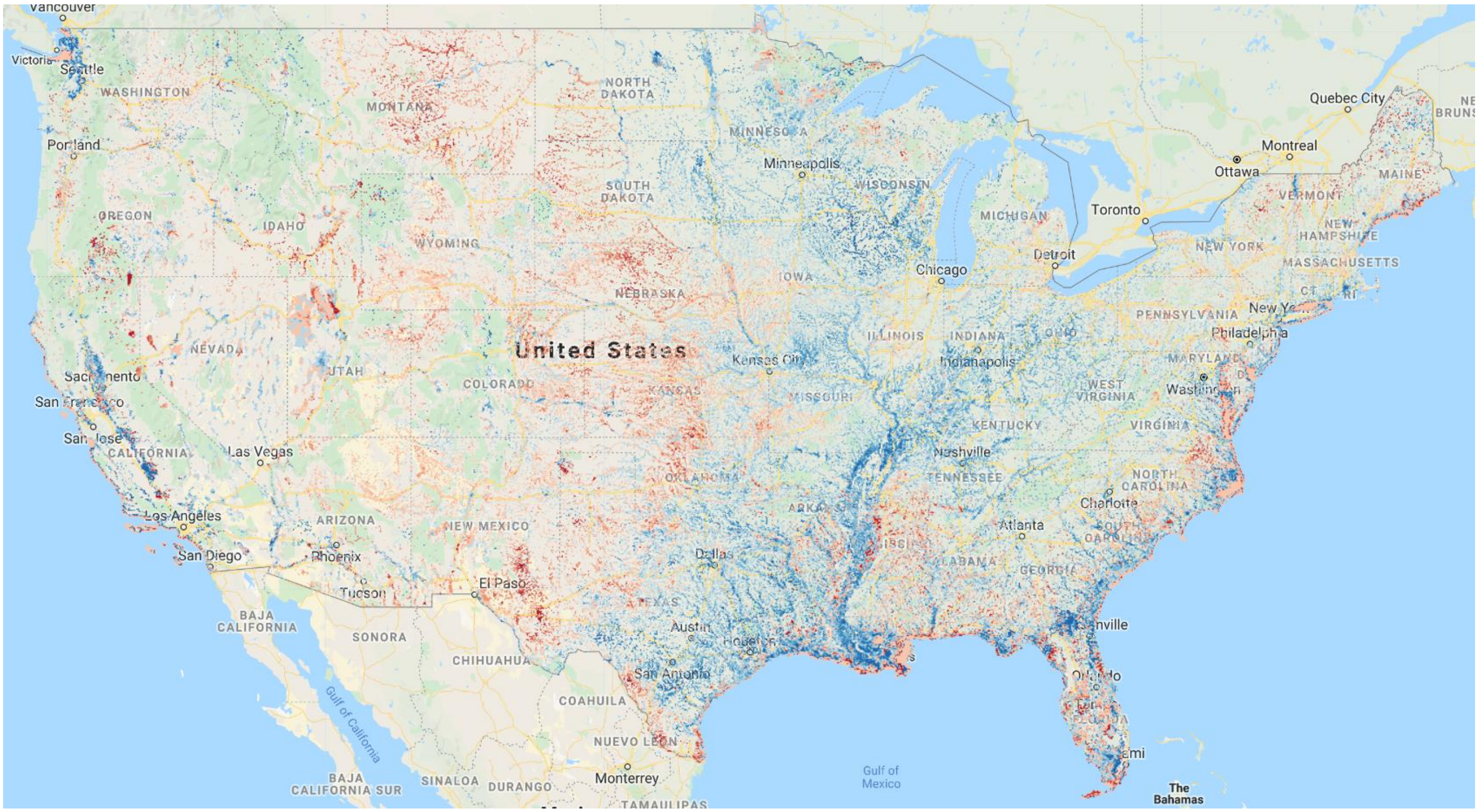
Trend in Normalized Difference Vegetation Index


NDVI data for 2000 to present
Mask the NDVI data with the extent of floodplains
Create a time band for the NDVI data and calculate the trend in NDVI using the linear fit reducer
Reclassify the trend in NDVI:
trend < -3: 4
-3 <= trend < -2: 3
-2 <= trend < -1: 2
-1 <= trend < 0: 1
0 <= trend < 1: -1
1 <= trend < 2: -2
2 <= trend < 3: -3
>= 3: -4

Trend in Normalized Difference Vegetation Index

```
// Import data.
var MODIS_NDVI = ee.ImageCollection("MODIS/MOD13Q1");

// Calculate trend in the NDVI.
function createTimeBand(img) {
  var month = img.date().difference(ee.Date('2000-01-01'), 'month');
  return ee.Image(month).float().addBands(img);
}
var collection_NDVI =
MODIS_NDVI.select('NDVI').map(createTimeBand);
var fit_NDVI = collection_NDVI.reduce(ee.Reducer.linearFit());
var fit_NDVI_floodplains = fit_NDVI.mask(Extent);
var trend_NDVI = fit_NDVI_floodplains.select(['scale']);

// Great restoration potential: below 0
var suitable_mask1 = trend_NDVI.gte(-1).and(trend_NDVI.lt(0));
var NDVI_suitable1 = trend_NDVI.where(suitable_mask1,
1).mask(suitable_mask1).toInt();

var suitable_mask2 = trend_NDVI.gte(-2).and(trend_NDVI.lt(-1));
var NDVI_suitable2 = trend_NDVI.where(suitable_mask2,
2).mask(suitable_mask2).toInt();

var suitable_mask3 = trend_NDVI.gte(-3).and(trend_NDVI.lt(-2));
var NDVI_suitable3 = trend_NDVI.where(suitable_mask3,
3).mask(suitable_mask3).toInt();

var suitable_mask4 = trend_NDVI.lt(-3);
var NDVI_suitable4 = trend_NDVI.where(suitable_mask4,
4).mask(suitable_mask4).toInt();

// Low restoration potential: higher than or equal to 0
var unsuitable_mask1 = trend_NDVI.gte(0).and(trend_NDVI.lt(1));
var NDVI_unsuitable1 = trend_NDVI.where(unsuitable_mask1, -
1).mask(unsuitable_mask1).toInt();

var unsuitable_mask2 = trend_NDVI.gte(1).and(trend_NDVI.lt(2));
var NDVI_unsuitable2 = trend_NDVI.where(unsuitable_mask2, -
2).mask(unsuitable_mask2).toInt();

var unsuitable_mask3 = trend_NDVI.gte(2).and(trend_NDVI.lt(3));
var NDVI_unsuitable3 = trend_NDVI.where(unsuitable_mask3, -
3).mask(unsuitable_mask3).toInt();

var unsuitable_mask4 = trend_NDVI.gte(3);
var NDVI_unsuitable4 = trend_NDVI.where(unsuitable_mask4, -
4).mask(unsuitable_mask4).toInt();

var NDVI_reclassed = ee.ImageCollection([NDVI_suitable1,
NDVI_suitable2, NDVI_suitable3, NDVI_suitable4, NDVI_unsuitable1,
NDVI_unsuitable2, NDVI_unsuitable3,
NDVI_unsuitable4]).reduce(ee.Reducer.max());
```

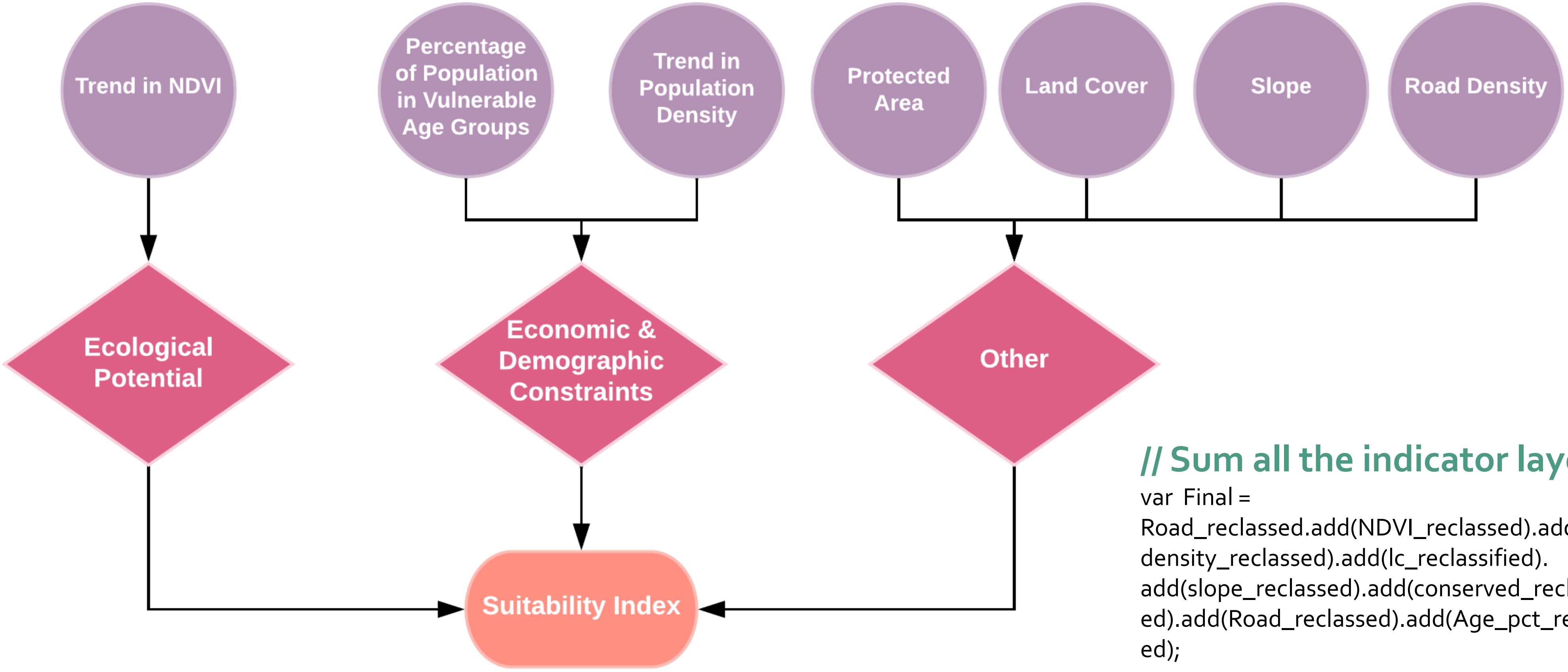
Trend in Normalized Difference Vegetation Index



Reclassified Trend in NDVI

🟥	trend < -3
🟥	-3 <= trend < -2
🟧	-2 <= trend < -1
🟫	-1 <= trend < 0
⬜	0 <= trend < 1
🟦	1 <= trend < 2
🟦	2 <= trend < 3
🟦	3 <= trend

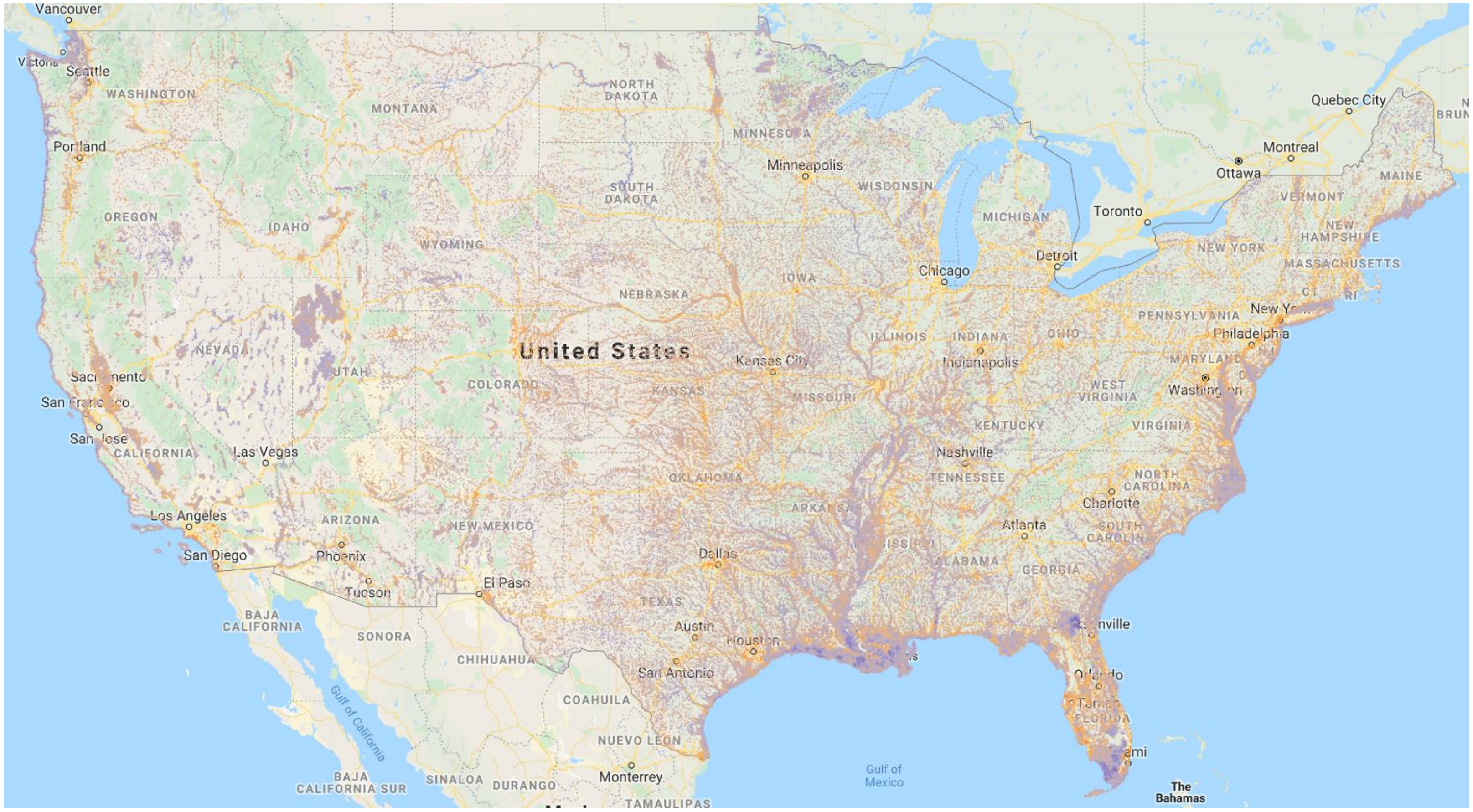
Suitability Index


Trend in NDVI
Percentage of Population in Vulnerable Age Groups
Trend in Population Density
Protected Area
Land Cover
Slope
Road Density
Ecological Potential
Economic & Demographic Constraints
Other
Suitability Index

// Sum all the indicator layers.

```
var  Final =
Road_reclassed.add(NDVI_reclassed).add(pop
density_reclassed).add(lc_reclassified).
add(slope_reclassed).add(conserved_reclassifi
ed).add(Road_reclassed).add(Age_pct_reclass
ed);
```

Final Suitability Map



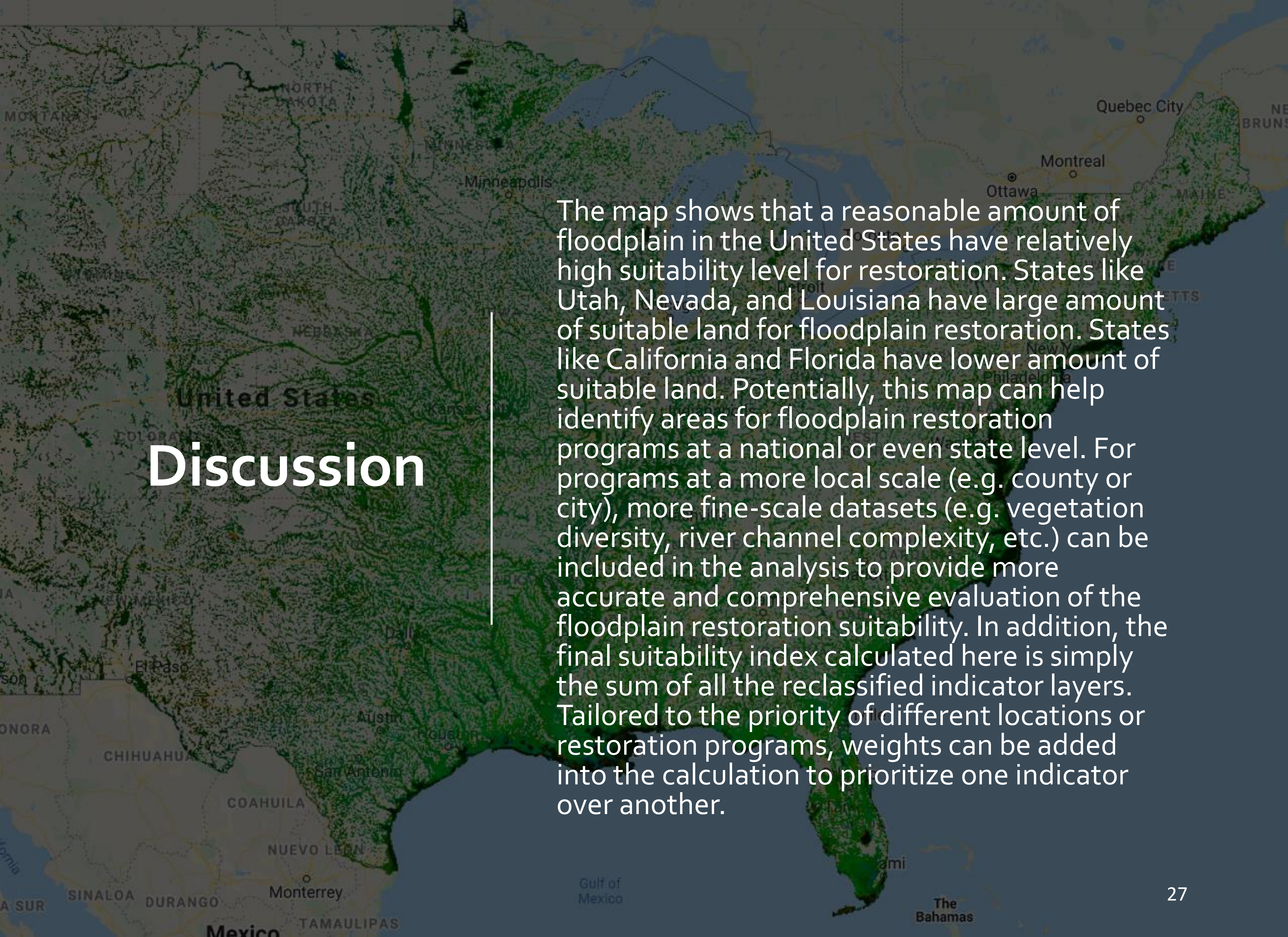
Suitability Index

🟧 very low suitability

🟧 low suitability

🟪 high suitability

🟪 very high suitability



Discussion

The map shows that a reasonable amount of floodplain in the United States have relatively high suitability level for restoration. States like Utah, Nevada, and Louisiana have large amount of suitable land for floodplain restoration. States like California and Florida have lower amount of suitable land. Potentially, this map can help identify areas for floodplain restoration programs at a national or even state level. For programs at a more local scale (e.g. county or city), more fine-scale datasets (e.g. vegetation diversity, river channel complexity, etc.) can be included in the analysis to provide more accurate and comprehensive evaluation of the floodplain restoration suitability. In addition, the final suitability index calculated here is simply the sum of all the reclassified indicator layers. Tailored to the priority of different locations or restoration programs, weights can be added into the calculation to prioritize one indicator over another.

Works Cited

Hulse, D., & Gregory, S. (2004). Integrating resilience into floodplain restoration. Urban Ecosystems, 7(3), 295–314. https://doi.org/10.1023/B:UECO.0000044041.94705.52

Why We Need to Restore Floodplains. (n.d.). Retrieved December 2, 2019, from American Rivers website: https://www.americanrivers.org/threats-solutions/restoring-damaged-rivers/benefits-of-restoring-floodplains/

Rohde, S., Hostmann, M., Peter, A., & Ewald, K. C. (2006). Room for rivers: An integrative search strategy for floodplain restoration. *Landscape and Urban Planning*, *78*(1–2), 50–70. https://doi.org/10.1016/j.landurbplan.2005.05.006

Rosgen, D. L. (1994). A classification of natural rivers. *CATENA*, *22*(3), 169–199. https://doi.org/10.1016/0341-8162(94)90001-9

Woznicki, S. A., Baynes, J., Panlasigui, S., Mehaffey, M., & Neale, A. (2019). Development of a spatially complete floodplain map of the conterminous United States using random forest. Science of The Total Environment, 647, 942–953. https://doi.org/10.1016/j.scitotenv.2018.07.353

Appendix

The link to all the code is
https://code.earthengine.google.com/3f7f4de5350aae081c1019a8ad914292

```
var Extent = ee.Image("users/sarahwyj97/Estimated_floodplain_CONUS"),
  MODIS_NDVI = ee.ImageCollection("MODIS/MOD13Q1"),
  Landcover = ee.ImageCollection("USGS/NLCD").filterDate ('2016-01-01',
'2016-12-31'),
  DEM = ee.Image("JAXA/ALOS/AW3D30_V1_1").select('AVE'),
  LandOwnership =
ee.FeatureCollection("users/sarahwyj97/LandOwnership"),
  Popdensity = ee.ImageCollection("CIESIN/GPWv411/GPW_UNWPP-
Adjusted_Population_Count").filterDate ('2000-01-01', '2018-12-31'),
  RoadDensity = ee.Image("users/sarahwyj97/RoadDensity"),
  Age =
ee.ImageCollection("CIESIN/GPWv411/GPW_Basic_Demographic_Charac
teristics");

Map.setCenter( -98, 39, 4 );
var ColorsForFloodplain = ['#EDBB99', '#27AE60'];
Map.addLayer(Extent, {min:0, max: 1, palette: ColorsForFloodplain},
"Extent of Floodplains");

// Find the percentage of population of disadvantaged age groups: 0-4 and
> 75 years old.
var Age_filter = ee.Filter.and(ee.Filter.or(ee.Filter.eq('Age_Group', '0-4'),
ee.Filter.eq('Age_Group', '75-')), ee.Filter.eq('Sex', 'b'));
var Age_vulnerable = Age.filter(Age_filter).sum();
var Pop_filter = ee.Filter.and(ee.Filter.eq('Age_Group', 'Total Population'),
ee.Filter.eq('Sex', 'b'));
var Total_pop = Age.filter(Pop_filter).first();
var Age_pctvulnerable = Age_vulnerable.divide(Total_pop).multiply(100);
var Age_pctvulnerable_fp = Age_pctvulnerable.mask(Extent);

// High opportunity for restoration: lower than or equal to 25%
var suitable_mask1 = Age_pctvulnerable_fp.lte(10);
var Age_suitable1 = Age_pctvulnerable_fp.where(suitable_mask1,
2).mask(suitable_mask1).toInt();

var suitable_mask2 =
Age_pctvulnerable_fp.gt(10).and(Age_pctvulnerable_fp.lte(25));
var Age_suitable2 = Age_pctvulnerable_fp.where(suitable_mask2,
1).mask(suitable_mask2).toInt();

// Low opportunity for restoration: higher than 25%
var unsuitable_mask1 =
Age_pctvulnerable_fp.gt(25).and(Age_pctvulnerable_fp.lte(50));
var Age_unsuitable1 = Age_pctvulnerable_fp.where(unsuitable_mask1, -
1).mask(unsuitable_mask1).toInt();

var unsuitable_mask2 = Age_pctvulnerable_fp.gte(50);
var Age_unsuitable2 = Age_pctvulnerable_fp.where(unsuitable_mask2, -
2).mask(unsuitable_mask2).toInt();

var Age_pct_reclassed = ee.ImageCollection([Age_suitable1,
Age_suitable2, Age_unsuitable1,
Age_unsuitable2]).reduce(ee.Reducer.max());

Map.addLayer(Age_pctvulnerable_fp, {min:0, max:50, palette:'#CD5C5C,
#3498DB'}, 'Percent of vulnerable age groups in Floodplains');

Map.addLayer(Age_pct_reclassed, {min:-2, max:2, palette:'#D35400,
#DC7633, #8c96c6, #88419d'}, 'Reclassified percentage of vulnerable age
groups');
```

Appendix

```
// Mask the road density with floodplain extent.
var RoadDensity_fp = RoadDensity.mask(Extent);

// High opportunity for restoration: equal to 0
var suitable_mask = RoadDensity_fp.eq(0);
var Road_suitable = RoadDensity_fp.where(suitable_mask,
1).mask(suitable_mask).toInt();

// Low opportunity for restoration: higher than 0
var unsuitable_mask1 =
RoadDensity_fp.gt(0).and(RoadDensity_fp.lt(2.5));
var Road_unsuitable1 = RoadDensity_fp.where(unsuitable_mask1, -
1).mask(unsuitable_mask1).toInt();

var unsuitable_mask2 =
RoadDensity_fp.gte(2.5).and(RoadDensity_fp.lt(5));
var Road_unsuitable2 = RoadDensity_fp.where(unsuitable_mask2, -
2).mask(unsuitable_mask2).toInt();

var unsuitable_mask3 =
RoadDensity_fp.gte(5).and(RoadDensity_fp.lt(7.5));
var Road_unsuitable3 = RoadDensity_fp.where(unsuitable_mask3, -
3).mask(unsuitable_mask3).toInt();

var unsuitable_mask4 =
RoadDensity_fp.gte(7.5).and(RoadDensity_fp.lt(10));
var Road_unsuitable4 = RoadDensity_fp.where(unsuitable_mask4, -
4).mask(unsuitable_mask4).toInt();

var unsuitable_mask5 = RoadDensity_fp.gte(10);
var Road_unsuitable5 = RoadDensity_fp.where(unsuitable_mask5, -
5).mask(unsuitable_mask5).toInt();

var Road_reclassed = ee.ImageCollection([Road_suitable,
Road_unsuitable1, Road_unsuitable2, Road_unsuitable3,
Road_unsuitable4, Road_unsuitable5]).reduce(ee.Reducer.max());

Map.addLayer(RoadDensity_fp, {min:0, max:10, palette:'#F9E79F,
#2874A6'}, 'Road Density in Floodplains');

Map.addLayer(Road_reclassed, {min:-5, max:1, palette:'#016c59,
#1c9099, #67a9cf, #bdc9e1, #f6eff7,#CB4335'}, 'Reclassified road
density’);
// Calculate trend in the NDVI.
function createTimeBand(img) {
  var month = img.date().difference(ee.Date('2000-01-01'), 'month');
  return ee.Image(month).float().addBands(img);
}
var collection_NDVI =
MODIS_NDVI.select('NDVI').map(createTimeBand);
var fit_NDVI = collection_NDVI.reduce(ee.Reducer.linearFit());
var fit_NDVI_floodplains = fit_NDVI.mask(Extent);
var trend_NDVI = fit_NDVI_floodplains.select(['scale']);

// Great restoration potential: below 0
var suitable_mask1 = trend_NDVI.gte(-1).and(trend_NDVI.lt(0));
var NDVI_suitable1 = trend_NDVI.where(suitable_mask1,
1).mask(suitable_mask1).toInt();
```

Appendix

```
var suitable_mask2 = trend_NDVI.gte(-2).and(trend_NDVI.lt(-1));
var NDVI_suitable2 = trend_NDVI.where(suitable_mask2,
2).mask(suitable_mask2).toInt();

var suitable_mask3 = trend_NDVI.gte(-3).and(trend_NDVI.lt(-2));
var NDVI_suitable3 = trend_NDVI.where(suitable_mask3,
3).mask(suitable_mask3).toInt();

var suitable_mask4 = trend_NDVI.lt(-3);
var NDVI_suitable4 = trend_NDVI.where(suitable_mask4,
4).mask(suitable_mask4).toInt();

// Low restoration potential: higher than or equal to 0
var unsuitable_mask1 = trend_NDVI.gte(0).and(trend_NDVI.lt(1));
var NDVI_unsuitable1 = trend_NDVI.where(unsuitable_mask1, -
1).mask(unsuitable_mask1).toInt();

var unsuitable_mask2 = trend_NDVI.gte(1).and(trend_NDVI.lt(2));
var NDVI_unsuitable2 = trend_NDVI.where(unsuitable_mask2, -
2).mask(unsuitable_mask2).toInt();

var unsuitable_mask3 = trend_NDVI.gte(2).and(trend_NDVI.lt(3));
var NDVI_unsuitable3 = trend_NDVI.where(unsuitable_mask3, -
3).mask(unsuitable_mask3).toInt();

var unsuitable_mask4 = trend_NDVI.gte(3);
var NDVI_unsuitable4 = trend_NDVI.where(unsuitable_mask4, -
4).mask(unsuitable_mask4).toInt();

var NDVI_reclassed = ee.ImageCollection([NDVI_suitable1,
NDVI_suitable2, NDVI_suitable3, NDVI_suitable4, NDVI_unsuitable1,
NDVI_unsuitable2, NDVI_unsuitable3,
NDVI_unsuitable4]).reduce(ee.Reducer.max());

Map.addLayer(fit_NDVI_floodplains,
        {min: [0, -2000, 0], max: [10, 10000, -10], bands: ['scale', 'offset',
'scale']},
        'NDVI');

Map.addLayer(fit_NDVI_floodplains.select(['scale']),
        {min: -1, max: 1, palette: '#D35400, #FCF3CF, #2E86C1'},
        'NDVI trend');

Map.addLayer(NDVI_reclassed,
        {min: -4, max: 4, palette: '#b2182b, #d6604d, #f4a582, #fddbc7,
#d1e5f0, #92c5de, #4393c3, #2166ac'},
        'NDVI reclassified’);
// Calculate trend in population density
var collection_pop = Popdensity.select('unwpp-
adjusted_population_count').map(createTimeBand);
var fit_pop = collection_pop.reduce(ee.Reducer.linearFit());
var fit_pop_floodplains = fit_pop.mask(Extent);
var trend_popdensity = fit_pop_floodplains.select(['scale']);

//High opportunity for restoration: lower than or equal to 0
var suitable_mask1 = trend_popdensity.gte(-
0.25).and(trend_popdensity.lte(0));
var pop_suitable1 = trend_popdensity.where(suitable_mask1,
1).mask(suitable_mask1).toInt();
```

Appendix

```
var suitable_mask2 = trend_popdensity.gte(-
0.5).and(trend_popdensity.lt(-0.25));
var pop_suitable2 = trend_popdensity.where(suitable_mask2,
2).mask(suitable_mask2).toInt();

var suitable_mask3 = trend_popdensity.gte(-
0.75).and(trend_popdensity.lt(-0.5));
var pop_suitable3 = trend_popdensity.where(suitable_mask3,
3).mask(suitable_mask3).toInt();

var suitable_mask4 = trend_popdensity.lt(-0.75);
var pop_suitable4 = trend_popdensity.where(suitable_mask4,
4).mask(suitable_mask4).toInt();

//Low opportunity for restoration: higher than 0
var unsuitable_mask1 =
trend_popdensity.gt(0).and(trend_popdensity.lt(0.25));
var pop_unsuitable1 = trend_popdensity.where(unsuitable_mask1, -
1).mask(unsuitable_mask1).toInt();

var unsuitable_mask2 =
trend_popdensity.gte(0.25).and(trend_popdensity.lt(0.5));
var pop_unsuitable2 = trend_popdensity.where(unsuitable_mask2, -
2).mask(unsuitable_mask2).toInt();

var unsuitable_mask3 =
trend_popdensity.gte(0.5).and(trend_popdensity.lt(0.75));
var pop_unsuitable3 = trend_popdensity.where(unsuitable_mask3, -
3).mask(unsuitable_mask3).toInt();

var unsuitable_mask4 = trend_popdensity.gte(0.75);
var pop_unsuitable4 = trend_popdensity.where(unsuitable_mask4, -
4).mask(unsuitable_mask4).toInt();

var popdensity_reclassed = ee.ImageCollection([pop_suitable1,
pop_suitable2, pop_suitable3, pop_suitable4, pop_unsuitable1,
pop_unsuitable2, pop_unsuitable3,
pop_unsuitable4]).reduce(ee.Reducer.max());

Map.addLayer(fit_pop_floodplains.select(['scale']),
     {min: 0, max:1 , palette: '#FEF5E7, #421E22'},
      'Population trend');

Map.addLayer(popdensity_reclassed, {min: -4, max: 4, palette: '#b35806,
#e08214, #fdb863, #fee0b6, #d8daeb, #b2abd2, #8073ac, #542788'},
'Reclassified population density');

// Land cover.
var lc = Landcover.select('landcover');
var lc_floodplains = lc.first().mask(Extent);

//High opportunity for restoration: vegetated; low opportunity for
restoration:developed
var lc_reclassified =
lc_floodplains.remap([11,12,21,22,23,24,31,41,42,43,52,71,81,82,90,95],
[0,0,-1,-2,-3,-4,0,2,2,2,2,2,1,1,3,3]);
```

Appendix

```
var
landcoverVis =
{
  min: 0.0,
  max: 95.0,
  palette: [
    '000000',
    '000000',
    '000000',
    '000000',
    '000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'466b9f',
'd1def8',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'dec5c5',
'd99282',
'eb0000',
'ab0000',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'b3ac9f',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'68ab5f',
'1c5f2c',
'b5c58f',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'af963c',
'ccb879',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'dfdfc2',
'd1d182',
'a3cc51',
'82ba9e',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'dcd939',
'ab6c28',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'000000',
'b8d9eb',
'000000',
'000000',
'000000',
'000000',
'6c9fb8'
  ],
};
```

```
Map.addLayer(lc_floodplains, landcoverVis, 'Landcover');

Map.addLayer(lc_reclassified, {palette: '#8c510a, #bf812d, #dfc27d,
#f6e8c3, #c7eae5, #80cdc1, #35978f, #01665e'},
  'Reclassified landcover');

// Calculate slope.
var slope = ee.Terrain.slope(DEM);
var slope_floodplains = slope.mask(Extent);

//Good for restoration: below 3.43 degree (6%)
var suitable_mask =
slope_floodplains.gte(0).and(slope_floodplains.lt(3));
var slope_suitable = slope_floodplains.where(suitable_mask,
1).mask(suitable_mask).toInt();

//Bad for restoration: higher than or equal to 3.43 degree (6%)
var unsuitable_mask = slope_floodplains.gte(3);
```

```
var slope_unsuitable = slope_floodplains.where(unsuitable_mask, -
1).mask(unsuitable_mask).toInt();

var slope_reclassed = ee.ImageCollection([slope_suitable,
slope_unsuitable]).reduce(ee.Reducer.max());

Map.addLayer(slope_reclassed,{min: -1, max: 1, palette: ['#F92407',
'#A7CFE9']},'Reclassified Slope');

// Land ownership
var floodplain_conserved =
Extent.clip(LandOwnership).multiply(Extent).mask(Extent);

//High opportunity for restoration: publicly/privately owned conserved
land; low opportunity for restoration: non-conserved land
var conserved_reclassified = floodplain_conserved.remap([0, 1], [-1, 1]);
```

Appendix

```
Map.addLayer(floodplain_conserved, {
  min: 0,
  max: 1,
  palette: ['#EDBB99', '#27AE60']},
  'Public or private conserved land');

Map.addLayer(conserved_reclassified, {
  min: -1,
  max: 1,
  palette: ['#FA8072', '#117A65']},
  'Reclassified land ownership');

//Final composite map
var Final =
Road_reclassed.add(NDVI_reclassed).add(popdensity_reclassed).add(lc_r
eclassified).
add(slope_reclassed).add(conserved_reclassified).add(Road_reclassed).ad
d(Age_pct_reclassed);

Map.addLayer(Final, {
  min: -20,
  max: 20,
  palette:'#e66101,#fdb863,#b2abd2,#5e3c99'},
  'Final Composite Map');
```