Estimation of Carbon Sink Using CASA Model

Overview

This supplementary document provides details on the CASA (Carnegie-Ames-Stanford Approach) model used to estimate carbon sinks for the year 2022. The study integrates Sentinel-2, Landsat, MODIS, and TerraClimate datasets using Google Earth Engine (GEE) for remote sensing and geospatial analysis. The CASA model estimates the Net Primary Productivity (NPP) and evaluates carbon sink capacity by processing vegetation indices, land cover types, and meteorological data.

Data Sources

- **Sentinel-2**: High-resolution imagery used for vegetation analysis.
- **Landsat 7-9**: Moderate-resolution data utilized for NDVI calculations.
- **MODIS (MCD12Q1)**: Provides vegetation type and land cover information.
- **TerraClimate**: Monthly climate data including precipitation, temperature, and solar radiation.

Data Preprocessing

Data preprocessing included:

1. **Geometric Correction**: Ensuring spatial alignment across datasets.
2. **Atmospheric Correction**: Applied scaling factors for reflectance.
3. **Cloud Removal**: Using QA bands and thresholds to exclude clouds and shadows.
4. **Reprojection**: All datasets were resampled to the WGS84 (EPSG:4326) coordinate system.

Code Implementation in Google Earth Engine

Section 1: Study Area Initialization and Basic Functions

```
var geom = table;
Map.addLayer(geom, {}, 'study area');
Map.centerObject(geom);

// Applies scaling factors.
function applyScaleFactors(image) {
  var opticalBands = image.select('SR_B.').multiply(0.0000275).add(-0.2);
  return image.addBands(opticalBands, null, true);
}

// Remove cloud for Landsat5-9
function cloudRemoval(image) {
  var cloudShadowBitMask = (1 << 4);
  var cloudsBitMask = (1 << 3);
  var qa = image.select('QA_PIXEL');
  var mask = qa.bitwiseAnd(cloudShadowBitMask).eq(0)
               .and(qa.bitwiseAnd(cloudsBitMask).eq(0));
  var mask2 = image.select("blue").gt(0.2);
  return image.updateMask(mask).updateMask(mask2.not()).toDouble()
              .copyProperties(image)
              .copyProperties(image, ["system:time_start"]);
}

function maskS2clouds(image) {
  var qa = image.select('QA60');
  var cloudBitMask = 1 << 10;
  var cirrusBitMask = 1 << 11;
  var mask = qa.bitwiseAnd(cloudBitMask).eq(0)
           .and(qa.bitwiseAnd(cirrusBitMask).eq(0));
  var cloudProb = image.select('MSK_CLDPRB');
  var scl = image.select('SCL');
  var cloud = cloudProb.lte(30);
  var shadow = scl.eq(3);
  var cirrus = scl.eq(10);
  var mask_scl = cloud.and(cirrus.neq(1)).and(shadow.neq(1));

  var opticalBands = image.select("B.*").multiply(0.0001);
  return image.addBands(opticalBands, null, true)
           .updateMask(mask)
           .updateMask(mask_scl)
```

```
            .copyProperties(image)
            .copyProperties(image, ["system:time_start"]);
}

// Assign a common name to the sensor-specific bands.
var LC9_BANDS = ['SR_B2', 'SR_B3', 'SR_B4', 'SR_B5', 'SR_B6', 'SR_B7', 'QA_PIXEL'];
var LC8_BANDS = ['SR_B2', 'SR_B3', 'SR_B4', 'SR_B5', 'SR_B6', 'SR_B7', 'QA_PIXEL'];
var LC7_BANDS = ['SR_B1', 'SR_B2', 'SR_B3', 'SR_B4', 'SR_B5', 'SR_B7', 'QA_PIXEL'];
var S2_BANDS = ['B2', 'B3', 'B4', 'B8', 'B11', 'B12'];
var STD_NAMES = ['blue', 'green', 'red', 'nir', 'swir1', 'swir2', 'QA_PIXEL'];

function get_imageC(year, region) {
   var date_start = ee.Date.fromYMD(year, 1, 1);
   var date_end = date_start.advance(1, 'year');
   var S2Col = ee.ImageCollection("COPERNICUS/S2_SR_HARMONIZED")
            .filterBounds(region)
            .filter(ee.Filter.date(date_start, date_end))
            .map(maskS2clouds)
            .select(['B4', 'B8'], ['red', 'nir']);
   var L9Col = ee.ImageCollection('LANDSAT/LC09/C02/T1_L2')
            .filterBounds(region)
            .filter(ee.Filter.date(date_start, date_end))
            .map(applyScaleFactors)
            .select(LC9_BANDS, STD_NAMES)
            .map(cloudRemoval);
   var L8Col = ee.ImageCollection('LANDSAT/LC08/C02/T1_L2')
            .filterBounds(region)
            .filter(ee.Filter.date(date_start, date_end))
            .map(applyScaleFactors)
            .select(LC8_BANDS, STD_NAMES)
            .map(cloudRemoval);
   var L7Col = ee.ImageCollection('LANDSAT/LE07/C02/T1_L2')
            .filterBounds(region)
            .filter(ee.Filter.date(date_start, date_end))
            .map(applyScaleFactors)
            .select(LC7_BANDS, STD_NAMES)
            .map(cloudRemoval);
   var LandsatCol = ee.ImageCollection(L9Col.merge(L8Col).merge(L7Col))
            .select(['red', 'nir'])
            .merge(S2Col)
            .sort("system:time_start");
   return LandsatCol;
}

// Data Preparation
var Climate_collection = ee.ImageCollection('IDAHO_EPSCOR/TERRACLIMATE')
   .filter(ee.Filter.date('2022-01-01', '2023-01-01'))
   .select(['pr', 'tmmn', 'tmmx', 'srad']);
var Landsat_collection = get_imageC(2022, geom);
var Landcover_collection = ee.ImageCollection('MODIS/061/MCD12Q1').select('LC_Type1');

function reprojectandclip(image) {
  return image.reproject('EPSG:4326', null, 500).clip(geom);
}

function solar_radiation_qc(image) {
  var srad = image.select('srad').multiply(0.1).multiply(2.592);
  var tmmn = image.select('tmmn').multiply(0.1);
  var tmmx = image.select('tmmx').multiply(0.1);
  var pr = image.select('pr');
```

```
  var tmean = tmmn.add(tmmx).divide(2).rename('tmean');
  return image.addBands(srad).addBands(tmean).addBands(pr);
}

Climate_collection = Climate_collection.map(solar_radiation_qc).map(reprojectandclip);

// Calculate RH using Zhuang et al. (2014)
function calculateRH(image) {
    var tmean = image.select('tmean');
    var pr = image.select('pr');
    var RH = ee.Image().expression(
        '0.22 * (exp(0.0913 * T) + log(0.3145 * R + 1)) * 30 * 0.465', {
          T: tmean,
          R: pr
        }
    ).rename('RH');
    return image.addBands(RH);
}

var Climate_with_RH = Climate_collection.map(calculateRH);

// Combine NPP, RH to Calculate NCEI
var carbonSink = Landsat_SR.map(function(image) {
    var RH = Climate_with_RH.select('RH');
    var NPP = image.select('NPP');
    var NCEI = NPP.subtract(RH).rename('NCEI'); // Net Carbon Emission Intensity
    return image.addBands(NCEI);
});

// Visualization and Export
var nceiVis = {
    min: -10,
    max: 10,
    palette: ['red', 'white', 'green']
};

Map.addLayer(carbonSink.select('NCEI').mean(), nceiVis, 'NCEI');

// Export NCEI
Export.image.toDrive({
    image: carbonSink.select('NCEI').mean(),
    description: 'NCEI_2022',
    fileNamePrefix: 'NCEI_2022',
    scale: 30,
    crs: 'EPSG:4326',
    region: geom,
    maxPixels: 1e13
});
```

Section 2: Load Satellite Data

```
// Function to load and process satellite data
function get_imageC(year, region) {
    var date_start = ee.Date.fromYMD(year, 1, 1);
    var date_end = date_start.advance(1, 'year');

    // Sentinel-2 Collection
    var S2Col = ee.ImageCollection("COPERNICUS/S2_SR_HARMONIZED")
                .filterBounds(region)
                .filter(ee.Filter.date(date_start, date_end))
                .map(maskS2clouds)
                .select(['B4', 'B8'], ['red', 'nir']);
```

```
    // Landsat 9 Collection
    var L9Col = ee.ImageCollection('LANDSAT/LC09/C02/T1_L2')
            .filterBounds(region)
            .filter(ee.Filter.date(date_start, date_end))
            .map(applyScaleFactors)
            .select(LC9_BANDS, STD_NAMES)
            .map(cloudRemoval);

    // Landsat 8 Collection
    var L8Col = ee.ImageCollection('LANDSAT/LC08/C02/T1_L2')
            .filterBounds(region)
            .filter(ee.Filter.date(date_start, date_end))
            .map(applyScaleFactors)
            .select(LC8_BANDS, STD_NAMES)
            .map(cloudRemoval);

    // Landsat 7 Collection
    var L7Col = ee.ImageCollection('LANDSAT/LE07/C02/T1_L2')
            .filterBounds(region)
            .filter(ee.Filter.date(date_start, date_end))
            .map(applyScaleFactors)
            .select(LC7_BANDS, STD_NAMES)
            .map(cloudRemoval);

    // Merge Landsat and Sentinel collections
    var LandsatCol = ee.ImageCollection(L9Col.merge(L8Col).merge(L7Col))
            .select(['red', 'nir'])
            .merge(S2Col)
            .sort("system:time_start");
    return LandsatCol;
}

// Load data for 2022
var Landsat_collection = get_imageC(2022, geom);
```

Section 3: Load and Process Climate Data

```
// Load TerraClimate data for 2022
var Climate_collection = ee.ImageCollection('IDAHO_EPSCOR/TERRACLIMATE')
                    .filter(ee.Filter.date('2022-01-01', '2023-01-01'))
                    .select(['pr', 'tmmn', 'tmmx', 'srad']);

// Function to preprocess climate data
function preprocessClimate(image) {
    var srad = image.select('srad').multiply(0.1).multiply(2.592); // Solar radiation in MJ/m²
    var tmmn = image.select('tmmn').multiply(0.1); // Minimum temperature in °C
    var tmmx = image.select('tmmx').multiply(0.1); // Maximum temperature in °C
    var pr = image.select('pr'); // Precipitation in mm
    var tmean = tmmn.add(tmmx).divide(2).rename('tmean'); // Mean temperature
    return image.addBands(srad.rename('solar_radiation'))
            .addBands(tmean)
            .addBands(pr.rename('precipitation'));
}

// Preprocess climate data
Climate_collection = Climate_collection.map(preprocessClimate);
```

Section 4: Calculate Zhuang's RH (Heterotrophic Respiration)

```
// Function to calculate RH using Zhuang's formula
function calculateRH(image) {
    var tmean = image.select('tmean'); // Mean temperature
```

```
    var pr = image.select('precipitation'); // Precipitation
    var RH = ee.Image().expression(
        '0.22 * (exp(0.0913 * T) + log(0.3145 * P + 1)) * 30 * 0.465', {
            T: tmean,
            P: pr
        }
    ).rename('RH');
    return image.addBands(RH);
}

// Apply RH calculation to the climate data
var Climate_with_RH = Climate_collection.map(calculateRH);
```

Section 5:Calculate NPP Using the CASA Model

```
// Functions to generate parameters for the CASA model
function generate_LUE(image) {
    return image.where(image.eq(3), 0.485).where(image.eq(1), 0.389)
                .where(image.eq(4), 0.692).where(image.eq(2), 0.985)
                .where(image.eq(5), 0.728).where(image.eq(6), 0.429)
                .where(image.eq(7), 0.429).where(image.eq(8), 0.542)
                .where(image.eq(9), 0.542).where(image.eq(10), 0.542)
                .where(image.eq(11), 0.542).where(image.eq(12), 0.542)
                .where(image.eq(13), 0.196).where(image.eq(14), 0.542)
                .where(image.eq(15), 0.542).where(image.eq(16), 0.217)
                .where(image.eq(17), 0.296).toFloat();
}

function set_NDVImin(image) {
    return image.where(image.gt(0), 0.023).toFloat();
}

function set_NDVImax(image) {
    return image.where(image.eq(1), 0.647).where(image.eq(2), 0.676)
                .where(image.eq(3), 0.738).where(image.eq(4), 0.747)
                .where(image.eq(5), 0.702).where(image.eq(6), 0.636)
                .where(image.eq(7), 0.634).where(image.eq(8), 0.634)
                .where(image.eq(9), 0.634).where(image.eq(10), 0.634)
                .where(image.eq(11), 0.634).where(image.eq(12), 0.634).toFloat();
}

function set_SRmin(image) {
    return image.where(image.gt(0), 1.05).toFloat();
}

function set_SRmax(image) {
    return image.where(image.eq(1), 4.67).where(image.eq(2), 5.17)
                .where(image.eq(3), 6.63).where(image.eq(4), 6.91)
                .where(image.eq(5), 5.845).where(image.eq(6), 4.49)
                .where(image.eq(7), 4.46).where(image.eq(8), 4.46)
                .where(image.eq(9), 4.46).where(image.eq(10), 4.46)
                .where(image.eq(11), 4.46).where(image.eq(12), 4.46).toFloat();
}

// Generate parameters for the CASA model
var Landcover_collection = ee.ImageCollection('MODIS/061/MCD12Q1').select('LC_Type1');
var LUE = Landcover_collection.map(generate_LUE).select(['LC_Type1'], ['LUE']);
var NDVImin = Landcover_collection.map(set_NDVImin).select(['LC_Type1'], ['NDVI_min']);
var NDVImax = Landcover_collection.map(set_NDVImax).select(['LC_Type1'], ['NDVI_max']);
var SRmin = Landcover_collection.map(set_SRmin).select(['LC_Type1'], ['SR_min']);
var SRmax = Landcover_collection.map(set_SRmax).select(['LC_Type1'], ['SR_max']);
```

```
// Calculate NDVI and SR
function calculateIndices(image) {
    var red = image.select('red');
    var nir = image.select('nir');
    var NDVI = nir.subtract(red).divide(nir.add(red)).rename('NDVI');
    var SR = ee.Image(1).add(NDVI).divide(ee.Image(1).subtract(NDVI)).rename('SR');
    return image.addBands(NDVI).addBands(SR);
}

// Apply to Landsat collection
var Landsat_indices = Landsat_collection.map(calculateIndices);

// Combine all CASA parameters
function combineParameters(image) {
    var FPAR1 =
image.select('NDVI').subtract(NDVImin).divide(NDVImax.subtract(NDVImin)).multiply(0.949).add(0.00
1);
    var FPAR2 =
image.select('SR').subtract(SRmin).divide(SRmax.subtract(SRmin)).multiply(0.949).add(0.001);
    var FPAR = FPAR1.add(FPAR2).divide(2).clamp(0.05, 0.95).rename('FPAR');
    return image.addBands(FPAR);
}

var Landsat_SR = Landsat_indices.map(combineParameters);

// Calculate NPP using FPAR and climatic variables
function calculateNPP(image) {
    var FPAR = image.select('FPAR');
    var solarRadiation = image.select('solar_radiation');
    var APAR = FPAR.multiply(solarRadiation).multiply(0.5).rename('APAR');
    var LUE = image.select('LUE');
    var Wstress = image.select('Wstress');
    var Tstress1 = image.select('Tstress1');
    var Tstress2 = image.select('Tstress2');
    var NPP =
APAR.multiply(LUE).multiply(Wstress).multiply(Tstress1).multiply(Tstress2).rename('NPP');
    return image.addBands(NPP);
}

var Landsat_NPP = Landsat_SR.map(calculateNPP);
```

Section 6: Combine NPP with RH and Calculate NCEI

```
// Combine NPP and RH to calculate NCEI
function calculateNCEI(image) {
    var NPP = image.select('NPP');
    var RH = Climate_with_RH.select('RH');
    var NCEI = NPP.subtract(RH).rename('NCEI'); // Net Carbon Emission Intensity
    return image.addBands(NCEI);
}

// Apply NCEI calculation
var Landsat_NCEI = Landsat_NPP.map(calculateNCEI);

// Visualization
var nceiVis = {
    min: -10,
    max: 10,
    palette: ['red', 'white', 'green']
};
```

```
Map.addLayer(Landsat_NCEI.select('NCEI').mean(), nceiVis, 'NCEI');
```

Section 7: Export NPP and NCEI

```
// Export NPP for 2022
Export.image.toDrive({
  image: Landsat_NPP.select('NPP').mean(),
  description: 'NPP_2022',
  fileNamePrefix: 'NPP_2022',
  scale: 30,
  crs: 'EPSG:4326',
  region: geom,
  maxPixels: 1e13
});

// Export NCEI for 2022
Export.image.toDrive({
  image: Landsat_NCEI.select('NCEI').mean(),
  description: 'NCEI_2022',
  fileNamePrefix: 'NCEI_2022',
  scale: 30,
  crs: 'EPSG:4326',
  region: geom,
  maxPixels: 1e13
});
```

Results and Observations

The CASA model outputs for the year 2022 were validated against MODIS NPP products to ensure consistency and accuracy. The key observations include:

1. **Net Primary Productivity (NPP)**: The average NPP for the Zhejiang Province was consistent with expected vegetation productivity rates.
2. **Carbon Sink Capacity**: The carbon sink capacity was derived from Net Ecosystem Productivity (NEP), calculated as the difference between NPP and Heterotrophic Respiration (RH). This method aligns with established biophysical mechanisms and improves accuracy in estimating carbon sinks for temperate forest regions.
3. **Validation**: The CASA model outputs were validated against MODIS NPP products through statistical comparison, including correlation analysis and RMSE evaluation. The results showed high consistency, indicating the robustness of the model.

Limitations and Future Work

1. **Cloud Contamination**: Residual cloud cover in some regions may affect accuracy. Enhanced cloud removal algorithms could be employed.
2. **Parameter Assumptions**: Fixed values for LUE and optimal temperature may introduce errors in specific areas. Future work could explore the use of machine learning models or region-specific calibration techniques to dynamically optimize these parameters, improving the robustness of NPP and NEP estimates.

This concludes the supplementary methods documentation for the CASA model-based carbon sink estimation in 2022.