**GOOGLE EARTH ENGINE CODE:**

// Define the area of interest (AOI) south of Cremona.

var aoi = ee.Geometry.Polygon([

[10.0174, 45.1056],

[10.1062, 45.1056],

[10.1062, 44.9858],

[10.0174, 44.9858]

]);

// Function to load Sentinel-2 surface reflectance data and calculate NDWI.

var getImageCollection = function(start, end) {

return ee.ImageCollection('COPERNICUS/S2\_SR\_HARMONIZED')

.filterBounds(aoi) // Filter by AOI.

.filterDate(start, end) // Filter by date range.

.filter(ee.Filter.lt('CLOUDY\_PIXEL\_PERCENTAGE', 20)) // Filter out cloudy images.

.map(function(image) {

var ndwi = image.normalizedDifference(['B3', 'B8']).rename('NDWI'); // Compute NDWI.

var water = ndwi.gt(0.3).rename('Water'); // Classify water using NDWI > 0.3.

return water.copyProperties(image, ['system:time\_start']);

});

};

// Load and process data for 2022 and 2023.

var s2\_2022 = getImageCollection('2022-04-01', '2022-09-30');

var s2\_2023 = getImageCollection('2023-04-01', '2023-09-30');

// Function to calculate water area for each image.

var calculateWaterArea = function(image) {

var waterArea = image.multiply(ee.Image.pixelArea())

.reduceRegion({

reducer: ee.Reducer.sum(), // Sum of pixel areas.

geometry: aoi, // Over the AOI.

scale: 10, // Pixel resolution.

maxPixels: 1e10

}).get('Water');

return ee.Feature(null, {'date': ee.Date(image.get('system:time\_start')).format('YYYY-MM-dd'), 'waterArea': waterArea});

};

// Calculate water area time series for 2022 and 2023.

var waterAreaSeries\_2022 = ee.FeatureCollection(s2\_2022.map(calculateWaterArea));

var waterAreaSeries\_2023 = ee.FeatureCollection(s2\_2023.map(calculateWaterArea));

// Function to compute detailed statistics.

var computeStats = function(imageCollection) {

var waterAreaImage = imageCollection.map(function(image) {

return image.multiply(ee.Image.pixelArea());

}).sum();

var stats = waterAreaImage.reduceRegion({

reducer: ee.Reducer.mean()

.combine(ee.Reducer.variance(), null, true)

.combine(ee.Reducer.minMax(), null, true)

.combine(ee.Reducer.histogram({maxBuckets: 50, minBucketWidth: 10000}), null, true), // Histogram settings.

geometry: aoi,

scale: 10,

maxPixels: 1e10

});

return stats;

};

// Compute statistics for 2022 and 2023.

var stats\_2022 = computeStats(s2\_2022);

var stats\_2023 = computeStats(s2\_2023);

// Print statistics to the console.

print('Statistics for 2022:', stats\_2022);

print('Statistics for 2023:', stats\_2023);

// Function to normalize water area to fit within the range 1 to 100.

var normalizeWaterArea = function(image, maxArea) {

return image.multiply(100).divide(ee.Image.constant(maxArea));

};

// Function to compute and plot histograms.

var computeAndPlotHistograms = function(imageCollection, year, color) {

// Sum water masks over the entire period.

var waterSum = imageCollection.map(function(image) {

return image.multiply(ee.Image.pixelArea());

}).sum();

// Get the maximum water area for normalization.

var maxArea = waterSum.reduceRegion({

reducer: ee.Reducer.max(),

geometry: aoi,

scale: 10,

maxPixels: 1e10

}).get('Water');

// Convert maxArea to an ee.Image to use in the divide operation.

var maxAreaImage = ee.Image.constant(maxArea);

// Normalize the water area.

var normalizedWaterSum = waterSum.multiply(100).divide(maxAreaImage);

// Compute total water area.

var totalWaterArea = waterSum.reduceRegion({

reducer: ee.Reducer.sum(),

geometry: aoi,

scale: 10,

maxPixels: 1e10

}).get('Water');

print('Total Water Area in ' + year + ' (m²):', totalWaterArea);

// Create histogram with normalized water area.

var histogram = ui.Chart.image.histogram({

image: normalizedWaterSum.updateMask(normalizedWaterSum.gt(1)), // Mask out values less than 1.

region: aoi,

scale: 10,

minBucketWidth: 1 // Bucket width to fit in the range 1 to 100.

})

.setSeriesNames(['Water Area'])

.setOptions({

title: 'Normalized Histogram of Water Area in ' + year,

vAxis: {title: 'Frequency'},

hAxis: {

title: 'Water Area (Normalized)',

viewWindow: {min: 1, max: 100} // Setting range to 1 - 100.

},

colors: [color]

});

print(histogram);

};

// Compute and plot histograms for 2022 and 2023.

computeAndPlotHistograms(s2\_2022, '2022', 'blue');

computeAndPlotHistograms(s2\_2023, '2023', 'cyan');

// Plot the water area time series.

var chart2022 = ui.Chart.feature.byFeature(waterAreaSeries\_2022, 'date', 'waterArea')

.setChartType('ColumnChart')

.setOptions({

title: 'Water Area in 2022',

vAxis: {title: 'Water Area (m²)'},

hAxis: {title: 'Date'},

colors: ['blue']

});

var chart2023 = ui.Chart.feature.byFeature(waterAreaSeries\_2023, 'date', 'waterArea')

.setChartType('ColumnChart')

.setOptions({

title: 'Water Area in 2023',

vAxis: {title: 'Water Area (m²)'},

hAxis: {title: 'Date'},

colors: ['cyan']

});

// Combine both years in a single chart for comparison.

var combinedSeries = waterAreaSeries\_2022.merge(waterAreaSeries\_2023);

var chartCombined = ui.Chart.feature.byFeature(combinedSeries, 'date', 'waterArea')

.setChartType('LineChart')

.setOptions({

title: 'Comparison of Water Area (2022 vs 2023)',

vAxis: {title: 'Water Area (m²)'},

hAxis: {title: 'Date'},

series: {

0: {color: 'blue', lineWidth: 2},

1: {color: 'cyan', lineWidth: 2}

},

legend: {position: 'bottom'}

});

// Print the charts to the console.

print(chart2022);

print(chart2023);

print(chartCombined);

// Visualize results on the map.

Map.centerObject(aoi, 10); // Center map on AOI.

var waterVisParams = {palette: ['000000', '00FFFF'], min: 0, max: 1}; // Visualization parameters.

Map.addLayer(s2\_2022.median().select('Water'), waterVisParams, 'Water Extent 2022'); // Add 2022 water extent.

Map.addLayer(s2\_2023.median().select('Water'), waterVisParams, 'Water Extent 2023'); // Add 2023 water extent.