

Code of Vector Comparison in GEE

Please view the app through <https://taoshiqi.users.earthengine.app/view/wind-vector-comparison>

To access the code, please set up a GEE account and click the link below
https://code.earthengine.google.com/?accept_repo=users/taoshiqi/ShiqiCodeShare
Or clone the git repository <https://earthengine.googlesource.com/users/taoshiqi/ShiqiCodeShare>

A full copy of the code is written below.

```
////////////////////////////////////
// App for comparing vector variable
// view the app through:
// https://taoshiqi.users.earthengine.app/view/wind-vector-comparison
//
// Concepts and equations from Dr. Pontius' Book - Metrics that Makes a Difference
// Code by Shiqi Tao. Checked by Arman Abajracharya and Evan Collins.
// Final project for Course GEOG379, Clark University
//
// Notes: (1) The book METRICS THAT MAKE A DIFFERENCE gives equations that apply
//        to a flat plane or to geographic projections that preserve direction.
//        (2) The book METRICS THAT MAKE A DIFFERENCE gives equations where vector
//        X has a positive component e when X points east and a positive component
//        n when X points north. Other conventions portray wind as the direction
//        from which the wind comes, in which case wind has a positive component u
//        when the wind blows from east to west and a positive component v when the
//        wind blows from north to south.
////////////////////////////////////

// input data
var ERA5_month = ee.ImageCollection("ECMWF/ERA5/MONTHLY");
var countries = ee.FeatureCollection('projects/google/examples/population-explorer/LSIB_SIMPLE-with-GHSL_POP');

////////////////////////////////////
//functions

//function 1: convert U and V component to speed(magnitude) and direction
var windStrenDirect = function(wind_u,wind_v) {
  // u = x, v = y
  var speed = wind_v.pow(2).add(wind_u.pow(2)).sqrt();
```

```

var direction_fromeast = wind_u.atan2(wind_v).divide(Math.PI).multiply(180); //stan([x,y])
//convert direction to degree from north
var direction = direction_fromeast.add(direction_fromeast.lt(-90).multiply(360)).multiply(-1).add(90);
direction = direction.mask(speed.neq(0));
return speed.rename('speed').addBands(direction.rename('direction'));
};

```

```

//function 2: calculate mean absolute deviation using speed(magnitude) and direction [vector data]

```

```

var calculateMAD = function(wind_t1_vector, wind_t2_vector, region_geom){
// input 2 vector data consists of two bands, for two time respectively
var mabsoluteDev = wind_t2_vector.subtract(wind_t1_vector).abs()
var convertDirection = mabsoluteDev.select('direction').min(
    mabsoluteDev.select('direction').multiply(-1).add(360))
mabsoluteDev = mabsoluteDev.select('speed').addBands(convertDirection)

```

```

//calculate the mean value for a region

```

```

var MAD_Num = mabsoluteDev.reduceRegion({
    reducer: ee.Reducer.mean(),
    geometry: region_geom,
    scale:30000})

```

```

return MAD_Num

```

```

}

```

```

//function 3.1: equation 11.14

```

```

var direction_Subtraction = function(direction_t1,direction_t2) {
    var diff = direction_t2.subtract(direction_t1)
    var c1 = diff.multiply(diff.abs().lt(180))
    var c2 = diff.subtract(360).multiply(diff.gt(180))
    var c3 = diff.add(360).multiply(diff.lt(-180))
    var c4 = diff.abs().multiply(diff.abs().eq(180))
    //var c5 = ee.Number(0).multiply(direction_1.eq(0).max(direction_2.eq(0)))
    return c1.add(c2).add(c3).add(c4)//.add(c5)
}

```

```

//function 3.2: same equation, but for number, not a map (raster)

```

```

var windStrenDirect_number = function(value_u,value_v) {
    // u = x, v = y
    var speed = value_v.pow(2).add(value_u.pow(2)).sqrt()
    var direction_fromeast = value_u.atan2(value_v).divide(Math.PI).multiply(180) //stan([x,y])
    var direction = direction_fromeast.add(direction_fromeast.lt(-90).multiply(360)).multiply(-1).add(90) //direction from north

```

```

return ee.List([speed,direction])
}

//function 3.3: calculate mean deviation (quantity component)
var calculateMeanDev = function(wind_u_t1, wind_v_t1, wind_u_t2, wind_v_t2, region_geom){

var mean_u_t1 = wind_u_t1.rename('value').reduceRegion({
  reducer: ee.Reducer.mean(),
  geometry: region_geom,
  scale:30000})
var mean_v_t1 = wind_v_t1.rename('value').reduceRegion({
  reducer: ee.Reducer.mean(),
  geometry: region_geom,
  scale:30000})
var mean_u_t2 = wind_u_t2.rename('value').reduceRegion({
  reducer: ee.Reducer.mean(),
  geometry: region_geom,
  scale:30000})
var mean_v_t2 = wind_v_t2.rename('value').reduceRegion({
  reducer: ee.Reducer.mean(),
  geometry: region_geom,
  scale:30000})

var wind_vecNum_t1 = windStrenDirect_number(ee.Number(mean_u_t1.get('value')),
  ee.Number(mean_v_t1.get('value')))

var wind_vecNum_t2 = windStrenDirect_number(ee.Number(mean_u_t2.get('value')),
  ee.Number(mean_v_t2.get('value')))

var meanDev_speed =
ee.Number(wind_vecNum_t2.get(0)).subtract(ee.Number(wind_vecNum_t1.get(0)))
var meanDev_direction =
direction_Subtraction(ee.Number(wind_vecNum_t1.get(1)),ee.Number(wind_vecNum_t2.get(1
)))

return ee.List([meanDev_speed,meanDev_direction])
}

```

```

////////////////////
//start of the app functions //
////////////////////

```

```

var drawingTools = Map.drawingTools();

drawingTools.setShown(false);

while (drawingTools.layers().length() > 0) {
  var layer = drawingTools.layers().get(0);
  drawingTools.layers().remove(layer);
}

var dummyGeometry =
  ui.Map.GeometryLayer({geometries: null, name: 'geometry', color: '23cba7'});

drawingTools.layers().add(dummyGeometry);

function clearGeometry() {
  var layers = drawingTools.layers();
  layers.get(0).geometries().remove(layers.get(0).geometries().get(0));
}

function drawRectangle() {
  clearGeometry();
  drawingTools.setShape('rectangle');
  drawingTools.draw();
}

function drawPolygon() {
  clearGeometry();
  drawingTools.setShape('polygon');
  drawingTools.draw();
}

var chartPanel = ui.Panel({
  style:
    {height: '250px', width: '400px', position: 'bottom-right', shown: false}
});

Map.add(chartPanel);

function VectorComparison_calculation() {
  // Make the chart panel visible the first time a geometry is drawn.
  if (!chartPanel.style().get('shown')) {
    chartPanel.style().set('shown', true);
  }
}

```

```

// Get the drawn geometry; it will define the reduction region.
var aoi = drawingTools.layers().get(0).getEeObject();

// Set the drawing mode back to null; turns drawing off.
drawingTools.setShape(null);

// Reduction scale is based on map scale to avoid memory/timeout errors.
var mapScale = Map.getScale();
var scale = mapScale > 5000 ? mapScale * 2 : 5000;

// start calculating
// Set filter variables.
var time1 = filters.Time1.getValue();
if (time1) var time1_start = ee.Date(time1+'-01'),
    time1_end = ee.Date(time1+'-27');

var wind_t1 = ee.Image(ERA5_month.filterDate(time1_start,time1_end).first())
var wind_u_t1 = wind_t1.select('u_component_of_wind_10m') // x
var wind_v_t1 = wind_t1.select('v_component_of_wind_10m') // y

var wind_t1_vector = windStrenDirect(wind_u_t1, wind_v_t1)

//set time 2
var time2 = filters.Time2.getValue();
if (time2) var time2_start = ee.Date(time2+'-01'),
    time2_end = ee.Date(time2+'-27');

var wind_t2 = ee.Image(ERA5_month.filterDate(time2_start,time2_end).first())
var wind_u_t2 = wind_t2.select('u_component_of_wind_10m') // x
var wind_v_t2 = wind_t2.select('v_component_of_wind_10m') // y

var wind_t2_vector = windStrenDirect(wind_u_t2, wind_v_t2)

///// mean deviation
var meanDev_results = calculateMeanDev(wind_u_t1, wind_v_t1, wind_u_t2, wind_v_t2, aoi)
var meanDev_speed = ee.Number(meanDev_results.get(0)).multiply(100).round().divide(100)
var meanDev_direction =
ee.Number(meanDev_results.get(1)).multiply(100).round().divide(100)

//////// mean absolute deviation
var mabsoluteDev_Num = calculateMAD(wind_t1_vector, wind_t2_vector, aoi)
var mabsoluteDev_speed =
ee.Number(mabsoluteDev_Num.get('speed')).multiply(100).round().divide(100)

```

```

var mabsoluteDev_direction =
ee.Number(mabsoluteDev_Num.get('direction')).multiply(100).round().divide(100)

//end calculating

//make chart 1
var quantity_speed = meanDev_speed.abs()
var MADcompared_speed = mabsoluteDev_speed.max(meanDev_speed.abs())
var allocation_speed = MADcompared_speed.subtract(quantity_speed)

var dataTable_speed = [
  ['Speed', { role: 'annotation' }, 'Quantity', 'Allocation'],
  ['Wind speed', 'Mean deviation = '+ meanDev_speed.getInfo(),
  quantity_speed.getInfo(), allocation_speed.getInfo()]
];

var options_speed = {
  width: 200,
  height: 100,
  hAxis: {textPosition: 'in', title: 'Mean absolute deviation = '+ MADcompared_speed.getInfo()
+' m/s'},
  legend: { position: 'top' },
  isStacked: true
};

var chart_speed = new ui.Chart(dataTable_speed, 'BarChart', options_speed);

//make chart 2
var quantity_dir = meanDev_direction.abs()
var MADcompared_dir = mabsoluteDev_direction.max(meanDev_direction.abs())
var allocation_dir = MADcompared_dir.subtract(quantity_dir)

var dataTable_dir = [
  ['Direction', { role: 'annotation' }, 'Quantity', 'Allocation'],
  ['Wind direction', 'Mean deviation = '+ meanDev_direction.getInfo(),
  quantity_dir.getInfo(), allocation_dir.getInfo()]
];

var options_direction = {
  width: 200,
  height: 100,
  hAxis: {textPosition: 'in', title: 'Mean absolute deviation = '+ MADcompared_dir.getInfo() +'
degree'},
  legend: { position: 'top'},

```

```

    isStacked: true
  };

var chart_direct = new ui.Chart(dataTable_direct, 'BarChart', options_direction);

// Replace the existing chart in the chart panel with the new chart.
chartPanel.widgets().set(0, chart_speed)
chartPanel.widgets().set(1, chart_direct)
}

drawingTools.onDraw(ui.util.debounce(VectorComparison_calculation, 500));
drawingTools.onEdit(ui.util.debounce(VectorComparison_calculation, 500));

var symbol = {
  rectangle: '■',
  polygon: '▲'
};

var addLayerToMap = function() {

var maplist = Map.layers();
if (maplist.length() != 0) Map.remove(maplist.get(4)),
    Map.remove(maplist.get(3)),
    Map.remove(maplist.get(2)),
    Map.remove(maplist.get(1)),
    Map.remove(maplist.get(0));

// Set filter variables.
var time1 = filters.Time1.getValue();
if (time1) var time1_start = ee.Date(time1+'-01'),
    time1_end = ee.Date(time1+'-27');

var wind_t1 = ee.Image(ERA5_month.filterDate(time1_start,time1_end).first())
var wind_u_t1 = wind_t1.select('u_component_of_wind_10m') // x
var wind_v_t1 = wind_t1.select('v_component_of_wind_10m') // y

var wind_t1_vector = windStrenDirect(wind_u_t1, wind_v_t1)

//set time 2
var time2 = filters.Time2.getValue();
if (time2) var time2_start = ee.Date(time2+'-01'),
    time2_end = ee.Date(time2+'-27');

var wind_t2 = ee.Image(ERA5_month.filterDate(time2_start,time2_end).first())

```

```

var wind_u_t2 = wind_t2.select('u_component_of_wind_10m') // x
var wind_v_t2 = wind_t2.select('v_component_of_wind_10m') // y

var wind_t2_vector = windStrenDirect(wind_u_t2, wind_v_t2)

clearGeometry();
drawingTools.setShape(null);
Map.addLayer(wind_t1_vector.select('speed'),{min:0,max:10,
palette:['#332288','#117733','#44AA99','#88CCEE','#DDCC77','#CC6677','#AA4499','#882255']},'
wind speed - time 1')
Map.addLayer(wind_t1_vector.select('direction'),{min:-180,max:180,
palette:['#0000FF','#FF00FF','#FF0000','#00FFFF','#0000FF']},'wind direction - time 1',false)
Map.addLayer(wind_t2_vector.select('speed'),{min:0,max:10,
palette:['#332288','#117733','#44AA99','#88CCEE','#DDCC77','#CC6677','#AA4499','#882255']},'
wind speed - time 2')
Map.addLayer(wind_t2_vector.select('direction'),{min:-180,max:180,
palette:['#0000FF','#FF00FF','#FF0000','#00FFFF','#0000FF']},'wind direction - time 2',false)
Map.addLayer(countries.style({color: 'white', fillColor: '00000000'}),{'Land boundary'})

};

var filters = {
Time1: ui.Textbox('YYYY-MM', '2010-05'),
Time2: ui.Textbox('YYYY-MM', '2015-05'),
applyButton: ui.Button('Display data', addLayertoMap),
loadingLabel: ui.Label({
value: 'Loading...',
style: {stretch: 'vertical', color: 'gray', shown: false}
})
};

function ColorBar(palette) {
return ui.Thumbnail({
image: ee.Image.pixelLonLat().select(0),
params: {
bbox: [0, 0, 1, 0.1],
dimensions: '100x10',
format: 'png',
min: 0,
max: 1,
palette: palette,
},
style: {stretch: 'horizontal', margin: '0px 8px'},
});
};

```



```

}

function makeLegend1() {
  var labelPanel = ui.Panel(
    [
      ui.Label('0', {margin: '4px 8px'}),
      ui.Label('5', {margin: '4px 8px', textAlign: 'center', stretch: 'horizontal'}),
      ui.Label('10', {margin: '4px 8px'})
    ],
    ui.Panel.Layout.flow('horizontal'));
  return
  ui.Panel([ColorBar(['#332288', '#117733', '#44AA99', '#88CCEE', '#DDCC77', '#CC6677', '#AA4499', '#882255']), labelPanel]);
}

function makeLegend2() {
  var labelPanel = ui.Panel(
    [
      ui.Label('-180', {margin: '4px 8px'}),
      ui.Label('0', {margin: '4px 8px', textAlign: 'center', stretch: 'horizontal'}),
      ui.Label('180', {margin: '4px 8px'})
    ],
    ui.Panel.Layout.flow('horizontal'));
  return ui.Panel([ColorBar(['#0000FF', '#FF00FF', '#FF0000', '#00FFFF', '#0000FF']), labelPanel]);
}

var controlPanel = ui.Panel({
  widgets: [
    ui.Label(
      'Wind - Vector Comparison', {fontWeight: 'bold', fontSize: '24px'}),
    ui.Label('This app helps you compare monthly wind \n(vector data) from two time periods',
      {whiteSpace: 'pre'}),
    ui.Label('Based on equations from Metrics that Make \na Difference by Dr. Robert Gilmore
  Pontius ',
      {whiteSpace: 'pre'}),

    ui.Label('-----'),
    ui.Label('1. Enter two time (YYYY-MM)', {fontWeight: 'bold'}),
    ui.Label('Time 1 (from 1979-01)', filters.Time1),
    ui.Label('Time 2 (until 2020-06)', filters.Time2),

    ui.Panel([
      filters.applyButton,
      filters.loadingLabel

```

```

], ui.Panel.Layout.flow('horizontal')),

ui.Label('2. Select a drawing mode.', {fontWeight: 'bold'}),
ui.Button({
  label: symbol.rectangle + ' Rectangle',
  onClick: drawRectangle,
  style: {stretch: 'horizontal'}
}),
ui.Button({
  label: symbol.polygon + ' Polygon',
  onClick: drawPolygon,
  style: {stretch: 'horizontal'}
}),
ui.Label('3. Draw a geometry and wait for render', {fontWeight: 'bold'}),
ui.Label(
  '4. Repeat 1-3 or edit/move geometry for a \nnew result.',
  {whiteSpace: 'pre',fontWeight: 'bold'}),

ui.Label('-----'),
ui.Label('Legend'),
ui.Label('Wind speed (m/s)'), makeLegend1(),
ui.Label('Wind direction (degree)'), makeLegend2()

],
style: {position: 'bottom-left'},
layout: null,
});

ui.root.insert(0, controlPanel);

```