1. Comparison of the occurring satellites

The satellite data used in the script are based on the four Landsat satellites -48[1]- , the two Sentinel-2 satellites (Sentinel-2A &B) and selected MODIS datasets. These satellite families were selected to cover the widest possible range of applications. The different characteristic resolutions of the respective satellites are of importance here. The characteristics used in the script are summarised in Table 1: Sentinel-2 data have existed since 2015, but only as processed script data since 2017 (as *surface reflection* data, cf. Chap. 2.1.1).

Table 1: Overview of satellite resolutions and data availability.

	Landsat	Sentinel-2	MODIS
Data availability since	August 1982	March 2017	February 2000
Temporal resolution	16 days	5 days	1 day
Spatial resolution	30 metres	10, 20 metres	500, 1,000 metres
Spectral resolution	6 tapes	10 tapes	7 bands

The selection of a satellite for a scientific analysis is a fundamental step, because it decisively determines the result. Depending on the question, different satellites are used, and both the temporal and spatial levels must be taken into account, among other things. For example, Landsat and Sentinel-2 data are preferred for small-scale analyses because of their high spatial resolution. If an event that occurred even further back in time is to be analysed, the Landsat time series is suitable. With Sentinel-2 data, more recent events (from 2017) can be analysed with a very high spatial and temporal resolution. MODIS, on the other hand, is particularly suitable for detecting large-scale changes, which are covered by daily data. Examples for the application of the different satellite data are discussed in chapter 4.

[1] Landsat 6 is not represented in this series as the satellite did not reach orbit (YOUNG et al. 2017).

2. Presentation of the script

The script can be divided into a total of six sections; the first block consists of the parameter input and the display of the study area. The second part includes all the functions in the script, which can be assigned to different categories. Another section (third block) contains the visualisation parameters, which are necessary for the display of the results in GEE. The processing of the data is found in the fourth section and represents the largest part of the script. The export of the results takes place afterwards and can be activated or deactivated by means of the parameter input. The presentation of the results in GEE itself concludes the script.

2.1. Parameter input

The parameter input is the block of the script that should primarily be changed by the user. A table with all available parameter settings and their application is shown below:

Table 2: Explanation of all possible parameter entries in the script.

Parameter [var X]	Description	Input; Type
satellite	Selection of the desired satellite and thus the respective associated *ImageCollection*	Landsat', 'Sentinel' or 'Modis'; **String**
geometry	Determination of the study area	Draw in a surface via the *Draw Toolbox* or enter the corner coordinates of a surface in the *Code Editor* with the following sequence: [xMin, yMin, xMax, yMax]; **Vector data or list of coordinates**
month_start & month_end year_start & year_end Date_start & Date_end	Determination of the study period; It must be noted here that the month and year filtering is carried out before the filtering according to the exact date limits (cf. Chap. 2.1.2)	Month and year entry as **integer type** and exact date entry as **string type** in the format 'YYYY-MM-DD'.
max_cloud_cover	The maximum cloud cover of the individual **Landsat** and **Sentinel scenes**	Between 0 and 100 **Integer/Float**
index	The index to be calculated; The list of integrated indices can be extended as desired in the functions block	'NDVI', 'NDWI', 'NDSI', 'NBR' (all satellites); 'LST' (Modes); **String**
metric	The statistical characteristics to be calculated	'mean', 'median', 'min', 'max', 'stdDev', 'count'; **String**
Differenced	This If loop determines whether a differentiated index is to be calculated	Entering 'Yes' activates this loop. With any other input, this block is not executed; **String**
month_start & month_end year_start & year_end Date_start & Date_end **Postevent**	Determination of the investigation period for the *postal event* (cf. chapter 2.1.2)	Month and year entry as **integer type** and exact date entry as **string type** in the format 'YYYY-MM-DD'.
export to drive	This if loop determines whether the result should be exported to Google Drive	Entering 'Yes' activates this loop. With any other input, this block is not executed; **String**
epsg	*EPSG Code of* the desired output coordinate system	'EPSG:XXXXX'; **String**
export singleimage	This if-loop determines whether the result of a single scene should be exported to Google Drive	Entering 'Yes' activates this loop. With any other input, this block is not executed; **String**
imageID	The ImageID of the individual scene within the selected *ImageCollection.*	X; **Integer**
region	A text which is in the export name of the result	*Example*: 'Germany'; **String**
time_period	A text which is in the export name of the result	*Example*: 'June-July 2018'; **String**

2.1.1. Image Collections

The first parameter input (**var satellite**) determines the desired satellite family and the corresponding *image collection.* This *image collection* is defined by several satellite images, which were all taken by the same sensor. This allows the user to quickly filter and/or sort according to individual criteria in order to obtain one or more specific scenes. Spatial as well as temporal filtering can be carried out, but also filtering according to the degree of cloud cover of the respective, individual scenes as well as further user-specific filtering is possible. [2]

In total, several *image collections of* a satellite are available; for the Landsat series used, three *collections are* available. Each of these was processed differently and is available as a separate data set; these are: *Surface Reflection*, *Top of Atmosphere* and *Raw Images*.[3] With Sentinel-2, on the other hand, there are only two *collections,* namely *Surface Reflection* and *Top of Atmosphere.* A fundamental difference here lies not only in the type of processing, but also in the availability of the data: While the *Top of Atmosphere-processed* data have been available since June 2015 (and thus since the beginning of the Sentinel-2 mission), the *Surface Reflection-processed* data have only been available since March 2017. [4]

Surface Reflection data was used in the script, but other processed data can also be used. For this, only the so-called *Earth Engine snippet* in the script has to be changed (example for the Landsat 8 Surface Reflection *snippet*: *ee.ImageCollection("LANDSAT/LC08/C01/T1_SR");* cf. also Fig. 7 in Chapter 2.4). These can be found on the platform's data catalogue under the following link: https://developers.google.com/earth-engine/datasets.

2.1.2. Temporal filtering

Temporal filtering is an essential part of the script because it determines the desired period of investigation. Two different types of temporal filtering are possible in the script: filtering by months, in which only selected months are represented in a specific time span of at least one year (cf. lines 10-14 in Fig. 1), and gapless filtering, which is defined by a time span between a start and end point (cf. lines 16-17 in Fig. 2). The month filtering is useful, for example, for analyses in arctic regions with a short vegetation period, as irrelevant data or months can be specifically masked out here. Another reason for filtering according to months can also be a high degree of cloud cover due to the season, for example in tropical regions, which makes further analysis more difficult.

In order to carry out a month filtering in the script, it is necessary that the variables *month_start* and *month_end* as well as *year_start* and *year_end* lie within the variables *Date_start* and *Date_end in terms* of time (cf. Fig. 1). The reason for this is the order of the filtering, because later in the script the filtering by year and month is carried out first and then

---

2 GORELICK et al. 2017.
3 GOOGLE EARTH ENGINE; GORELICK et al. 2017; KUMAR & MUTANGA 2018.
4 GOOGLE EARTH ENGINE.

the filtering by the exact dates (cf. also Fig. 7 in Chapter 2.4). Analogous to this is the gapless filtering of the exact date boundaries; here the period under investigation must lie within the month and year variables.

In the example shown in Figure 1, the parameter settings for a monthly filtering are shown, with the investigation period covering the months of June and July in the years from 2003 to 2006.

```
10  var month_start = 6;                    //Startmonth of the temporal integration
11  var month_end = 7;                      //Endmonth of the temporal integration
12
13  var year_start = 2003;                  //Startyear of the temporal integration.
14  var year_end = 2006;                    //Endyear of the temporal integration.
15
16  var Date_start = '2003-06-01';          //'YYYY-MM-DD'; Care about the starting year of the satellite
17  var Date_end = '2006-07-31';            //'YYYY-MM-DD'
```

Figure 1: Parameter inputs for a month filtering.

Figure 2 shows an example of filtering according to exact date boundaries; here the investigation period runs without gaps from 01.06.2003 to 31.07.2006. The time interval of the variables *Date_start* and *Date_end* thus lies within the variables in rows 10-14.

```
10  var month_start = 1;                    //Startmonth of the temporal integration
11  var month_end = 12;                      //Endmonth of the temporal integration
12
13  var year_start = 2003;                  //Startyear of the temporal integration.
14  var year_end = 2006;                    //Endyear of the temporal integration.
15
16  var Date_start = '2003-06-01';          //'YYYY-MM-DD'; Care about the starting year of the satellite
17  var Date_end = '2006-07-31';            //'YYYY-MM-DD'
```

Figure 2: Parameter inputs for gapless filtering between two data.

In this case, a larger time window than required was initially selected by means of the month and year filtering. However, with the next filtering, i.e. according to the exact date limits, this time window is narrowed down and finally corresponds to the desired study period.

In summary, the following two points are important for temporal filtering:

- Month and year filtering is done before filtering the exact date boundaries.
- If a monthly filtering is to be carried out, the filtering after an uninterrupted time span must cover a larger time window than that of the monthly filtering. This is to be observed analogously for filtering after an uninterrupted time span.

2.2. Functions

The functions used in the script can be divided into a total of four categories, these include the renaming of the bands of the respective satellites, the cloud masking, the calculation of the desired indices as well as the selection of the *reducer* for the respective statistical features.

2.2.1. Renaming the bands

The renaming of the respective bands was carried out in this script to enable a uniform procedure with regard to the calculation of the indices. The reason for this lies in the different

names of the bands of the respective satellites; there are even differences between individual satellite families (in the Landsat series between Landsat 4-7 and Landsat 8, cf. Tab. 3).

Table 3: Functions for renaming the individual bands.

Function	Description
renameBandsTM_ETM	Renaming of bands **'B1'**, '**B2'**, '**B3'**, '**B4'**, '**B5'**, '**B7'** to **'B'**, '**G'**, '**R', 'NIR'**, '**SWIR1'**, '**SWIR2'** at Landsat 4-7
renameBandsOLI	Renaming of bands **'B2'**, '**B3'**, '**B4'**, '**B5'**, '**B6'**, '**B7'** to **'B'**, '**G'**, '**R', 'NIR'**, '**SWIR1'**, '**SWIR2'** for Landsat 8
renameBandsSentinel	Renaming of bands **'B2'**, '**B3'**, '**B4', 'B8'**, '**B11'**, '**B12'** to **'B'**, '**G'**, '**R', 'NIR'**, '**SWIR1'**, '**SWIR2'** for Sentinel-2
renameBandsModis	Rename the tapes **'Nadir_Reflectance_Band1'**, **'Nadir_Reflectance_Band2'**, **'Nadir_Reflectance_Band3', 'Nadir_Reflectance_Band4'**, **'Nadir_Reflectance_Band5'**, **'Nadir_Reflectance_Band6', 'Nadir_Reflectance_Band6'** to **'R'**, **'NIR1'**, **'B'**, **'G'**, **'NIR2', 'SWIR1', 'SWIR2'** for MODIS (Nadir BRDF-Adjusted Reflectance)
renameBandLST	Renaming of the band **'LST_Day_1km'** to **'LST'** at MODIS

A function in the script for renaming the individual bands is shown in Figure 3 using Landsat 8 as an example. Information about the individual bands (designation, resolution, wavelength range, etc.) is provided by GEE in the description of the respective *image collection* on the GEE platform. [5]

```
99  // Define functions to select and rename bands for Landsat-8
100 function renameBandsOLI(image) {
101     var bands = ['B2', 'B3', 'B4', 'B5', 'B6', 'B7', 'pixel_qa'];
102     var new_bands = ['B', 'G', 'R', 'NIR', 'SWIR1', 'SWIR2', 'pixel_qa'];
103     return image.select(bands).rename(new_bands);
104 }
```

Figure 3: Function for renaming the necessary bands in Landsat 8.

The ***pixel_qa band*** at Landsat (cf. Fig. 3) is used for processing the data and is neither listed in Table 3 nor renamed in the script, nevertheless it is included in the function. The reason for this is that the ***pixel_qa band*** is required for cloud masking later on (cf. Chap. 2.2.2).

2.2.2. Cloud masking

Clouds and their shadows are a common problem in optical remote sensing because they obscure the earth's surface. [6]Especially for analyses with a high number of satellite images, cloud masking is essential, as otherwise clouds would influence or distort the result when calculating statistical features, such as when creating a median NDVI satellite image. This cloud masking is therefore carried out for the Landsat series and for Sentinel-2 with the aid of functions. For the MODIS datasets, no cloud masking is used by means of a function, since the cloud masking of the data was already carried out in *preprocessing* (cf. Chap. 2.1.1).

Figure 4 shows the cloud masking function for Landsat series 4-8, which is based on the ***pixel_qa*** band (spatial resolution: 30 m). This band contains pixel-based information on,

5 https://developers.google.com/earth-engine/datasets

6 ZHU & HELMER 2018; Schmitt ET al. 2019.

among other things, the occurrence of clouds and their shadow fall as well as snow, ice and water[7]. These values were calculated using a *CFMask* algorithm, which is best suited for the detection of clouds. [8]

In this way, selected pixels, such as cloud and cloud shadow pixels, can be specifically masked (cf. Fig. 4, right). For this, the *bitmask* or the respective *bit numbers are* required, which can be found in the *image collection* information of the *data catalogue* (cf. Fig. 4, left). The following figure illustrates the cloud masking function.

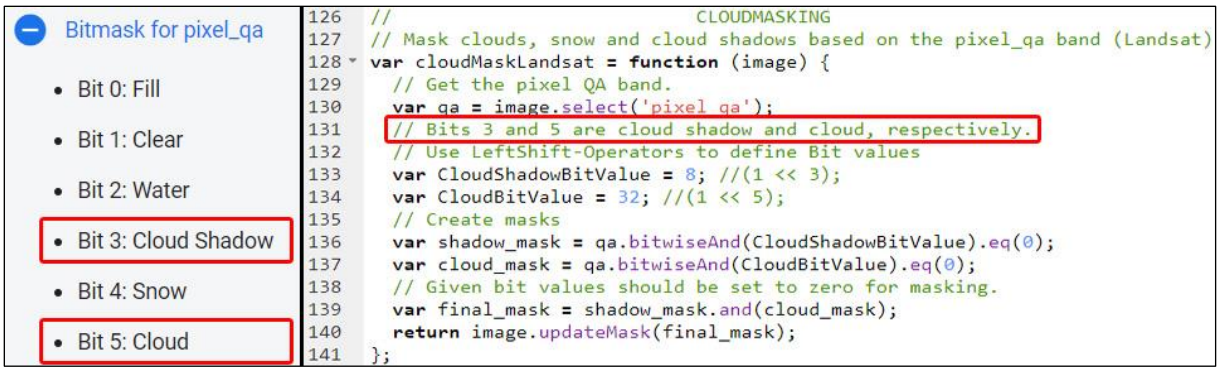

Figure 4: The cloud masking function for Landsat 4-8 (right) and the corresponding bitmask (left). Source: modified after GOOGLE EARTH ENGINE.

This function thus masks clouds as well as their shadow fall from the individual Landsat scenes so that cloud-free scenes can be analysed.

This was also carried out with Sentinel-2, but the ***QA60*** band was used, which is characterised by a spatial resolution of 60 metres. This cloud masking sometimes delivers inadequate results, especially under critical atmospheric conditions. There are several reasons for this, which will not be discussed further in this paper. Improved cloud masking in Sentinel-2 is [9]explained, for example, in the work of NGUYEN et al. (2020), for which the authors used a *random forest classifier* and the Normalised Difference Snow Index (NDSI) to distinguish snow areas from clouds. They found that their approach achieved a much more comprehensive cloud masking.

The self-created script is only based on cloud masking using the ***QA60*** tape, a more detailed masking of the clouds or the implementation of the function of NGUYEN et al. would be possible at this point, but has not (yet) been added.

2.2.3. Calculation of the indices

A total of five different indices are represented in the script; these are shown in Table 4, together with the respective calculation and description. Furthermore, the satellites are listed

7 UNITED STATES GEOLOGICAL SURVEY.
8 FOGA et al. 2017.
9 NGUYEN et al. 2020.

with which the respective indices can be calculated. Finally, the table is supplemented by a selected exemplary journal article that has already used the respective indices in GEE:

Table 4: The indices occurring in the script.

Index	Calculation	Description	Satellites	Example
Normalised difference vegetation index (**NDVI**)	$\frac{NIR - RED}{NIR + RED}$	Unit of measurement for the vitality of vegetation	Landsat, Sentinel-2, Modes	XIONG et al. 2017
Normalised difference water index (**NDWI**)	$\frac{NIR - SWIR1}{NIR + SWIR1}$ [10]	Unit of measurement for the water content of vegetation, detection of water areas	Landsat, Sentinel-2, Modes	GUO et al. 2020
Normalised difference snow index (**NDSI**)	$\frac{Green - SWIR1}{Green + SWIR1}$	Detection of snow and ice surfaces	Landsat, Sentinel-2, Modes	GU et al. 2021
Normalised Burn Ratio (**NBR**)	$\frac{NIR - SWIR2}{NIR + SWIR2}$	Fire area detection	Landsat, Sentinel-2	PARKS et al. 2018
Land surface temperature (**LST**)	Calculation by GEE	Analysis of the daily surface temperature	Modes	EBRAHIMY et al. 2021

The calculation of the indices is done on the basis of functions; this is exemplified with Fig. 5: Here, first the formula of the respective index, in this case the NDVI, is defined in the file format *string by* means of an *expression in* line 159. Subsequently, the bands described therein must be assigned to the corresponding bands; this is done by means of the *. select()* function (cf. lines 161-162 in Fig. 5). This calculation creates a new band, which must be added to the *image collection* (. *addBands()* function; cf. line 165). Finally, the band is given a new name via the *. rename()* function (cf. line 166) and the newly created index is changed to the file format of a floating point number via the *. float()* function (cf. line 166). The reason for this is that the index has a value range from -1 to 0 to +1.

```
155  //                  INDICES
156  // Create function for NDVI
157  function my_ndvi(image) {
158    var ndvi_var = image.expression(
159      "(NIR - RED)/(NIR + RED)",
160      {
161      NIR: image.select("NIR"),
162      RED: image.select("R"),
163      });
164    return image
165      .addBands(ndvi_var
166      .rename('NDVI'))
167      .float();
168  }
```

Figure 5: Function for calculating the Normalised Difference Vegetation Index.

[10] For Landsat and Sentinel according to GAO 1996; for Modis the NDWI was calculated by the following formula: $\frac{Green - NIR}{Green + NIR}$.

2.2.4. Statistical features

One aim of the script is to analyse *image collections,* which are (usually) defined by a large number of individual scenes. In order to carry out this analysis, it makes sense to calculate statistical characteristics. For this purpose, for example, several scenes that overlap spatially but were recorded at different times are reduced to a single scene. There are various so-called *reducers, which are* summarised in Table 5.

Table 5: Statistical characteristics.

Parameter input [var X]	***Reducer* in GEE**	**Description**
median	ee.Reducer.median();	Calculation of the median
mean	ee.Reducer.mean();	Calculation of the (weighted) arithmetic mean
min	ee.Reducer.min();	Calculation of the minimum values
max	ee.Reducer.max();	Calculation of the minimum values
stdDev	ee.Reducer.stdDev();	Calculation of the standard deviation
count	ee.Reducer.count();	Calculation of *the Number of Observations*

2.3. Visualisation parameters

The visualisation parameters are used to display the results in the map area of the script. Selected bands, their value range as well as the respective colour palette and other settings such as transparency can be selected and displayed. The colour scales can be inserted[11]via hexagonal code, among other things. Figure 6 shows exemplary visualisation parameters; the parameters for the display of the NDVI were defined in lines 213-215. Although this index has a value range between -1 and +1, negative values are not taken into account with these settings. The colour palette was taken from GEE. Further representations also originate from the Earth Engine and are exemplarily represented in Figure 6.

```
212 //====================================PARAMS=============================================
213 var ndviparams = {min: 0.0, max: 1.0, palette: [
214     'FFFFFF', 'CE7E45', 'DF923D', 'F1B555', 'FCD163', '99B718', '74A901', '66A000', '529400',
215     '3E8601', '207401', '056201', '004C00', '023B01', '012E01', '011D01', '011301']};
216 var lstParams = {min: 13000.0, max: 16500.0, palette: ['040274', '040281', '0502a3', '0502b8',
217 '0502ce', '0502e6', '0602ff', '235cb1', '307ef3', '269db1', '30c8e2', '32d3ef','3be285', '3ff38f',
218 '86e26f', '3ae237', 'b5e22e', 'd6e21f', 'fff705', 'ffd611', 'ffb613', 'ff8b13', 'ff6e08', 'ff500d',
219 'ff0000', 'de0101', 'c21301', 'a71001', '911003']};
220 var Landsat_median_RGB_Params = {bands: ['R', 'G', 'B'], min: 0, max: 4000, gamma: 1.5};
```

Figure 6: Selected visualisation parameters.

2.4. Data processing

The processing of the data represents the largest block in the script and includes, among other things, the filtering processes, the application of the functions (renaming the bands, cloud masking, adding the indices) and the calculation of the differentiated indices. Figure 7 illustrates these processes using the Landsat 8 satellite as an example. First, the *image collection is* selected using the *Earth Engine snippet* (*Tier **1***, *Surface Reflectance*; **T1**_**SR**) (cf. line 289 in Figure 7). This loads the entire Landsat 8 archive. In order to now obtain the specific, desired scenes, the filtering processes are carried out: First, the spatial filtering is done with

[11] The codes can be determined under this URL link: https://htmlcolorcodes.com/.

the command . *filterBounds(geometry)* (cf. line 290). This sorts out the scenes that do not cover the study area. Next, all scenes are to be kept in the *image collection that do not* exceed the maximum specified cloud cover (cf. line 291). Using the . *map()* function in lines 292 and 293, the selected functions (band renaming and cloud masking) are applied to each individual scene in the *Image Collection.* Finally, the temporal filtering is done, first filtering by year and month (cf. lines 294-295) and then filtering by the exact date boundaries (cf. line 296 and also chapter 2.1.2). Finally, the individual scenes are tailored to the study area (cf. line 297).

```
288 // Load an Landsat-8 ImageCollection.
289   var L8_col = ee.ImageCollection('LANDSAT/LC08/C01/T1_SR')
290   .filterBounds(geometry)
291   .filter(ee.Filter.lt('CLOUD_COVER_LAND', max_cloud_cover))
292   .map(renameBandsOLI)
293   .map(cloudMaskLandsat)
294   .filter(ee.Filter.calendarRange(year_start, year_end, 'year'))
295   .filter(ee.Filter.calendarRange(month_start, month_end, 'month'))
296   .filterDate(Date_start, Date_end)
297   .map(function(image){return image.clip(geometry)});
298
299 // Merge all collections
300   var Merge_col_Landsat = L4_col.merge(L5_col).merge(L7_col).merge(L8_col);
301 // Sort by date
302   var Merge_col_Landsat = Merge_col_Landsat.sort("system:time_start");
303   print('All Images', Merge_col_Landsat);
304 // Add indices to ImageCollection
305   var temp = Merge_col_Landsat.map(my_ndvi)
306                               .map(my_ndwi)
307                                 .map(my_nbr)
308                                   .map(my_ndsi);
309   print('All Images (with all indices)', temp);
310 //============================================================================
```

Earth Engine Snippet

Figure 7: Data processing using the Landsat 8 satellite as an example.

Since there are now a total of four different processed *image collections of* the Landsat series (Landsat 4-8), they are merged into a single *image collection with* the . *merge()* function (cf. line 300). Finally, this is sorted temporally (cf. line 302), since temporal overlaps of the different Landsat satellites exist[12]. The *print()* function is used to output the processed *image collection* or the individual scenes it contains with their metadata and the available bands in the *console* in GEE. Finally, the indices are added to the *image collection* (cf. lines 305-308).

```
415     //Reduce the two collections with the selected metric
416     var temp_PREEVENT = temp.reduce(metric);
417     var temp_POSTEVENT_FOR_DIFFERENCED = temp_POSTEVENT.reduce(metric);
418     var temp_differenced = temp_PREEVENT.subtract(temp_POSTEVENT_FOR_DIFFERENCED);
419     print('Difference Image:', temp_differenced);
420     var temp_differenced = temp_differenced.rename(['B', 'G', 'R', 'NIR', 'SWIR1', 'SWIR2',
421                                 'NDVI', 'NDWI', 'NBR', 'NDSI'])
422 }
```

Figure 8: Calculation of the differentiated indices.

12 YOUNG et al. 2017.

The calculation of differentiated indices as well as the bands are also made possible with the script; for this purpose, the steps taken in Figure 7 are first carried out again with other temporal variables. This *image collection is* called a ***postevent*** and contains the same parameter entries as the ***preevent*** *image collection* (the *image collection* calculated first)*,* but with the exception of the variables *month_start* and *month_end* as well as *year_start* and *year_end* or *Date_start* and *Date_end*. In the case of the ***Postevent*** *Image Collection,* these variables are marked with the suffix *Postevent* (cf. also Table 1 in Chapter 2.1).

For the calculation of the differentiated indices, both *image collections* are first reduced to one scene with the selected statistical feature and then, in the sense of a *change detection*, the changes of the two time periods are highlighted, this is done by subtracting them using the *. subtract()* function*.*

2.5. Exporting the results

Exporting the results is an essential part of the script, as it allows the data to be further processed in geographical information systems. A Google Drive account is required for downloading the results, because a folder called "GEE_Export" is created there after executing the script, in which all results are saved. Generally, however, the results are only downloaded if this has been activated in the parameter input (**var export_to_drive** = **'Yes'**). The data is played out in the file format *GeoTIFF*, all exported tapes must be of the same file type, i.e. as *float.* Please note that the individual bands are still available as unscaled data, the respective scaling factors can be found in the *Data Catalog.*

2.5.1. Export Case A: Standard case

The standard case occurs when the differentiated index has not been calculated and no individual scene has been selected from the image collection. In the standard case, not only the export of the selected index with the respective statistical feature is enabled, but also the export of the corresponding median RGB images. Figure 9 shows the export process, which is done using the *Export.image.toDrive()* function. First, the respective *image collection* (marked in the script with the variable ***temp***) is selected and reduced to the specified statistical characteristic with the *. reduce(metric)* function (cf. line 554 in Fig. 9). The names of the files were composed of various factors such as the satellite, the index and user-selected labels (cf. lines 555-556). The spatial resolution (referred to as *scale in* the script) is already determined in the previous process and depends on the input of the satellite.

The output coordinate system (*crs*; cf. line 559) is determined via the EPSG code, which was specified by the user in the parameter input. Finally, the study area is selected (cf. line 560).

```
550  //                              EXPORTING DATA
551  //Exporting the selected index with the selected metric
552  if (export_to_drive == 'Yes' && Differenced != 'Yes') {
553    Export.image.toDrive({
554    image: temp.reduce(metric),
555    description: pl + '_All_bands_and_indices_'+ metric + '_' + Date_start + '_' + Date_end + '_'
556    + region + '_' + time_period,
557    folder: 'GEE_Export',
558    scale: scale,
559    crs: epsg,
560    region: rect_geodesic
561  })
562
563    //RGB Median Image for Landsat & Sentinel
564    if (satellite == 'Sentinel') {
565      //Sentiel-2 RGB Bands = 10m Spatial Resolution
566      var scale = 10;
567      }
568    var rgb_image = temp.median().select(['R', 'G', 'B']);
569    Export.image.toDrive({
570    image: rgb_image,
571    description: pl + '_' + 'RGB_median_' + Date_start + '_' + Date_end +
572    '_' + region + '_' + time_period,
573    folder: 'GEE_Export',
574    scale: scale,
575    crs: epsg,
576    region: rect_geodesic
577  })}
```

Figure 9: Exporting the results.

Exporting the RGB median images is described with lines 564-576, here there are two differences to the operations shown above: First, when using Sentinel-2 data, the spatial resolution is set to 10 metres, as the Red, Green and Blue bands are labelled with the appropriate resolution. [13]Furthermore, instead of the selected statistical feature, the bands R, G and B are selected and the medians of the bands are calculated (cf. lines 568 & 570).

2.5.2. Export Case B: Differentiated index

The export of differentiated indices is the smallest block in the export area and is controlled with the variable ***var Differenced in the*** parameter input. Since the two *image collections* were already reduced to one scene during the calculation of the differentiated indices by means of the selected statistical feature (cf. chapter 2.4), it is no longer necessary to use a *reducer* when exporting. Figure 10 illustrates the export of the differentiated indices. It would be possible at this point to export the respective reduced ***pre-*** and ***postevent*** data; however, this has not (yet) been integrated into the script.

```
579  //                          EXPORTING DIFFERENCED INDEX
580  if (export_to_drive == 'Yes' && Differenced == 'Yes') {
581    Export.image.toDrive({
582    image: temp_differenced,
583    description: pl + '_differenced_' + index + '_' + metric + '_' + Date_start + '_' + Date_end + '-'
584    + Date_start_Postevent + '_' + Date_end_Postevent  + '_' + region + '_' + time_period,
585    folder: 'GEE_Export',
586    scale: scale,
587    crs: epsg,
588    region: rect_geodesic
589  })}
```

Figure 10: Exporting differentiated indices.

13 EUROPEAN SPACE AGENCY (ESA).

2.5.3. Export Case C: Individual Scenes

The script not only enables the export of reduced *image collections*, but also the export of individual scenes that are available in the respective *image collection.* This is controlled by the two variables ***var export_singleimage*** and ***var imageID***, whereby the former is only used to activate the *if-loop in* the export area. The second variable must be entered in the file type *integer* and corresponds to the position of the desired scene in the *image collection*. This position is explained by Figure 11: An *Image Collection* with all the scenes it contains is output in the *Console* by the *print()* function when the script is executed. Both the individual tapes and the metadata of the scenes are included. The metadata is particularly important because it provides information about the date of recording, the degree of cloud cover, the geographical location (cf. Fig. 11, red boxes) and other characteristics of the scenes. If, for example, the scene of 20.06.2018 is of importance to the user, it can also be downloaded by entering the *image ID.* To do this, the number of the position (in the example, the number 2) must be inserted in the parameter input.

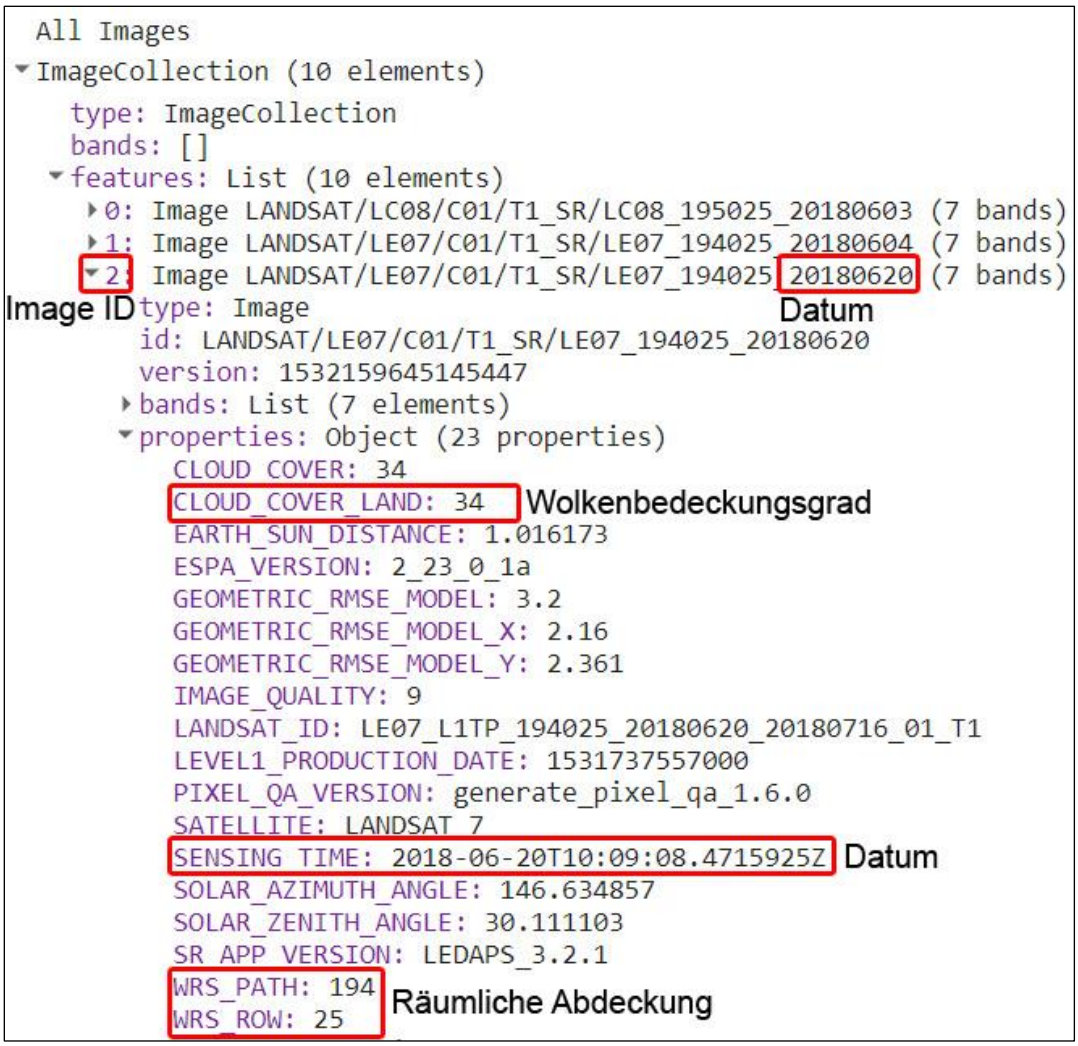

Figure 11: Image ID of a scene in the Image Collection and a section of its metadata.

Figure 12 shows the export of a single scene, where all available bands, including all calculated indices, of the scene are selected and exported. In addition, information from the respective metadata is used by the . *getInfo()* functions and integrated into the export name. This creates a unique, recognisable name for the result, in the case of the implementation of Figure 12, this would be as follows: **Landsat_7_Image_from_2018-06-20T10-09-08.**

A point of criticism here lies in the designation of the bands in the continuing GIS programmes, these are not displayed in some programmes as *B*, *G*, *R* etc., but only as *Band_1*, *Band_2*, *Band_3* etc.

```
591  //                          EXPORTING SINGLE IMAGES
592  // Landsat single Image Export
593  if (satellite == 'Landsat' && export_singleimage == 'Yes' && Differenced != 'Yes') {
594    var listOfImages = temp.toList(temp.size());
595    var singleimage = ee.Image(listOfImages.get(imageID));
596    var singleimage = singleimage.select(['B', 'G', 'R', 'NIR', 'SWIR1', 'SWIR2',
597                                          'NDVI', 'NDWI', 'NBR', 'NDSI']);
598
599    var date = singleimage.date().format().getInfo();
600    var date = date.replace(':', '-');
601    var date = date.replace(':', '-');
602    var landsat_satellite = singleimage.get('SATELLITE').getInfo().toLowerCase().slice(1);
603
604    Export.image.toDrive({
605    image: singleimage,
606    description: 'L' + landsat_satellite + '_Single_Image_from_' + date  + '_' + region,
607    folder: 'GEE_Export',
608    scale: scale,
609    crs: epsg,
610    region: rect_geodesic
611  })}
```

Figure 12: Export of individual Landsat and Sentinel-2 scenes.

3. List of sources

EBRAHIMY, H, AGHIGHI, H., AZADBAKHT, M., AMANI, M., MAHDAVI, S., MATKAN, A.-A. (2021): "Downscaling MODIS Land Surface Temperature Product Using an Adaptive Random Forest Regression Method and Google Earth Engine for a 19-Years Spatiotemporal Trend Analysis over Iran". In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 14, pp. 2103-2111.

EUROPEAN SPACE AGENCY (ESA) (n.d.): Sentinel Online - Resolution and Swath. Available online at: https://sentinel.esa.int/web/sentinel/missions/sentinel-2/instrument-payload/resolution-and-swath, [Last accessed: 03.03.2021].

FOGA, S, SCARAMUZZA, P.-L., GUO, S., ZHU, Z., DILLEY, R.-D., BECKMANN, T., SCHMIDT, G.-L., DWYER, J.-L., JOSEPH HUGHES, M., LAUE, B. (2017): 'Cloud detection algorithm comparison and validation for operational Landsat data products'. In: *Remote Sensing of Environment*, 194, pp. 379-390.

GAO, B (1996): "NDWI-A normalized difference water index for remote sensing of vegetation liquid water from space". In: *Remote Sensing of Environment*, 58 (3), pp. 257-266.

GOOGLE EARTH ENGINE (ed.) (n.d.): Earth Engine Data Catalog. Available online at: https://developers.google.com/earth-engine/datasets, [Last accessed: 14.01.2021].

GORELICK, N., HANCHER, M., DIXON, M., ILYUSHCHENKO, S., THAU, D., MOORE, R.. (2017): 'Google Earth Engine: planetary-scale geospatial analysis for everyone'. In: *Remote Sensing of Environment* 202, pp. 17-28.

GU, C, ZHANG, Y., LIU, L., LI, L., LI, S., ZHANG, B., CUI, B., RAI, M.-K. (2021): "Qualifying Land Use and Land Cover Dynamics and Their Impacts on Ecosystem Service in Central Himalaya Transboundary Landscape Based on Google Earth Engine". In: *Land*, 10 (2), p. 173.

GUO, Y.-T., ZHANG, X.-M., LONG, T.-F. JIAO, W.-L., HE, G.-J., YIN, R.-Y., DONG, Y.-Y. (2020): "China forest cover extraction based on Google Earth Engine". In: *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLII-3/W10, pp. 855-862.

KUMAR, L, MUTANGA, O. (2018): "Google Earth Engine Applications Since Inception: Usage, Trends, and Potential". In: *Remote Sensing*, 10 (10), p. 1509.

NGUYEN, M, BAEZ-VILLANUEVA, O., BUI, D., NGUYEN, P., RIBBE, L. (2020): "Harmonization of Landsat and Sentinel 2 for Crop Monitoring in Drought Prone Areas: Case Studies of Ninh Thuan (Vietnam) and Bekaa (Lebanon)". In: *Remote Sensing*, 12 (2), p. 281.

PARKS, S, HOLSINGER, L., VOSS, M., LOEHMAN, R., ROBINSON, N. (2018): "Mean Composite Fire Severity Metrics Computed with Google Earth Engine Offer Improved Accuracy and Expanded Mapping Potential". In: *Remote Sensing*, 10 (6), p. 879.

SCHMITT, M, HUGHES, L.-H., QIU, C., ZHU, X.-X. (2019): 'Aggregating Cloud-free Sentinel-2 Images with Google Earth Engine'. In: *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-2/W7, pp. 145-152.

UNITED STATES GEOLOGICAL SURVEY (USGS) (n.d.): *Landsat 8 Surface Reflectance Quality Assessment.* Available online at: https://www.usgs.gov/core-science-systems/nli/landsat/landsat-8-surface-reflectance-quality-assessment?qt-

science_support_page_related_con=1#qt-science_support_page_related_con, [Last accessed: 17.02.2021].

YOUNG, N.-E., ANDERSON, R.-S., CHIGNELL, S.-M., VORSTER A.-G., LAWRENCE R., EVANGELISTA, P.-H. (2017): "A survival guide to Landsat preprocessing". In: *Ecology* 98 (4), pp. 920-932.

XIONG, J., THENKABAIL, P.-S., GUMMA, M.-K., TELUGUNTLA, P., POEHNELT, J., CONGALTON, R.-G., YADAV, K., THAU, D. (2017): "Automated cropland mapping of continental Africa using Google Earth Engine cloud computing". In: *ISPRS Journal of Photogrammetry and Remote Sensing*, 126, pp. 225-244.

ZHU, X, HELMER, E.-H. (2018): "An automatic method for screening clouds and cloud shadows in optical satellite image time series in cloudy regions". In: *Remote Sensing of Environment*, 214, pp. 135-153.