SITS: Data Analysis and Machine Learning using Satellite Image Time Series

Rolf Simoes *National Institute for Space Research (INPE), Brazil*
Gilberto Camara *National Institute for Space Research (INPE), Brazil*
Alexandre Carvalho *Institute for Applied Economics Research (IPEA), Brazil*
Victor Maus *International Institute for Applied System Analysis (IIASA), Switzerland*
Gilberto Queiroz *National Institute for Space Research (INPE), Brazil*

Using time series derived from big Earth Observation data sets is one of the leading research trends in Land Use Science and Remote Sensing. One of the more promising uses of satellite time series is its application for classification of land use and land cover, since our growing demand for natural resources has caused major environmental impacts. The SITS package provides support on how to use statistical learning techniques with image time series. These methods include linear and quadratic discrimination analysis, support vector machines, random forests and neural networks.

Introduction

Earth observation satellites provide a continuous and consistent set of information about the Earth's land and oceans. Most space agencies have adopted an open data policy, making unprecedented amounts of satellite data available for research and operational use. This data deluge has brought about a major challenge: *How to design and build technologies that allow the Earth observation community to analyse big data sets?*

The approach taken in the current package is to develop data analysis methods that work with satellite image time series. The time series are obtained by taking calibrated and comparable measures of the same location in Earth at different times. These measures can be obtained by a single sensor (e.g., MODIS) or by combining different sensors (e.g., LANDSAT-8 and SENTINEL-2). If obtained by frequent revisits, the temporal resolution of these data sets can capture the most important land use changes.

Time series of remote sensing data show that land cover changes do not always occur in a progressive and gradual way, but they may also show periods of rapid and abrupt change followed either by a quick recovery [Lambin et al., 2003]. Analyses of multiyear time series of land surface attributes, their fine-scale spatial pattern, and their seasonal evolution leads to a broader view of land-cover change. Satellite image time series have already been applied to applications such as mapping for detecting forest disturbance [Kennedy et al., 2010], ecology dynamics [Pasquarella et al., 2016], agricultural intensification [Galford et al., 2008] and its impacts on deforestation [Arvor et al., 2012].

The SITS package provides support on how to use statistical learning techniques with image time series. In a broad sense, statistical learning refers to a class of algorithms for classification and regression analysis [Hastie et al., 2009]. These methods include linear and quadratic discrimination analysis, support vector machines, random forests and neural networks. In a typical classification problem, we have measures that capture class attributes. Based on these measures, referred as training data, one's task is to select a predictive model that allows inferring classes of a larger data set.

Current approaches to image time series analysis still use limited number of attributes. A common approach is deriving a small set of phenological parameters from vegetation indexes, like beginning, peak, and length of growing season [Brown et al., 2013] [Kastens et al., 2017] [Estel et al., 2015] [Pelletier et al., 2016]. These phenological parameters are then fed in specialised classifiers such as TIMESAT [Jönsson and Eklundh, 2004]. These approaches do not use the power of advanced statistical learning techniques to work on high-dimensional spaces and with big training data sets [James et al., 2013].

The SITS package uses the full depth of satellite image time series to create larger dimensional spaces. We tested different methods of extracting attributes from time series data, including those reported by Pelletier et al. [2016] and Kastens et al. [2017]. Our conclusion is that part of the information in raw time series is lost after filtering or statistical approximation. Thus, the method we developed has a deceptive simplicity: *use all the data available in the time series samples.* The idea is to have as many temporal attributes as possible, increasing the dimension of the classification space. Our experiments found out that modern statistical models such as support vector machines, and random forests perform better in high-dimensional spaces than in lower dimensional ones.

In what follows, we describe the main characteristics of the SITS package. The first part describes the basic data structures used in SITS and the tools used for visualisation and data exploration. Then we show how to do data acquisition from external sources, with an emphasis on the WTSS ("web time series service"). The next sections describe filtering and clustering techniques. We then discuss machine learning techniques for satellite image time series data and how to apply them to image time series. Finally, we present validation methods.

Data Handling and Visualisation Basics in SITS

The basic data unit in the SITS package is the SITS tibble, which is a way of organizing a set of time series data with associated spatial information. In R, a "tibble" differs from the traditional data frame, insofar as a tibble can contain lists embedded as column arguments. Tibbles are part of the "tidyverse", a collection of R package designed to work together in data manipulation. The "tidyverse" includes packages such as "ggplot2", "dplyr" and "purrr" [Wickham and Grolemund, 2017]. The SITS package makes extensive use of the "tidyverse".

For a better explanation of how the "SITS tibble" works, we will read a data set containing 2,115 labelled samples of land cover in Mato Grosso state of Brazil. This state has 903,357 km^2 of extension, being the third largest state of Brazil. It includes three of Brazil's biomes: Amazonia, Cerrado and Pantanal. It is the most important agricultural frontier of Brazil and is Brazil's largest producer of soybeans, corn and cotton.

The samples contain time series extracted from the MODIS MOD13Q1 product from NASA from 2000 to 2016, provided every 16 days at 250-meter spatial resolution in the Sinusoidal projection. Based on ground surveys and high resolution imagery, we selected 2,115 samples of nine classes: forest, cerrado, pasture, soybean-fallow, fallow-cotton, soybean-cotton, soybean-corn, soybean-millet, and soybean-sunflower. Crop and pasture ground data was collected by researchers Alexandre Coutinho, Julio Esquerdo and Joao Antunes from the Brazilian Agricultural Research Agency (EMBRAPA) through farmer interviews in October 2009 and in October 2013. Samples for cerrado and forest classes were provided by Rodrigo Bergotti from INPE. Ground samples for soybean-fallow class were provided by Damien Arvor[Arvor et al., 2012].

```
# retrieve a set of samples from an RDS file
samples.tb <- samples_MT_9classes
samples.tb
```

```
## # A tibble: 2,115 x 7
##    longitude latitude start_date end_date   label   coverage    time_seri~
##        <dbl>    <dbl> <date>     <date>     <chr>   <chr>       <list>
##  1     -55.2   -10.8  2013-09-14 2014-08-29 Pasture mod13q1_512 <tibble [~
##  2     -57.8   - 9.76 2006-09-14 2007-08-29 Pasture mod13q1_512 <tibble [~
##  3     -51.9   -13.4  2014-09-14 2015-08-29 Pasture mod13q1_512 <tibble [~
##  4     -56.0   -10.1  2005-09-14 2006-08-29 Pasture mod13q1_512 <tibble [~
##  5     -54.6   -10.4  2013-09-14 2014-08-29 Pasture mod13q1_512 <tibble [~
##  6     -52.5   -11.0  2013-09-14 2014-08-29 Pasture mod13q1_512 <tibble [~
##  7     -52.1   -14.0  2013-09-14 2014-08-29 Pasture mod13q1_512 <tibble [~
##  8     -57.7   -13.3  2015-09-14 2016-08-28 Pasture mod13q1_512 <tibble [~
##  9     -54.7   -16.4  2015-09-14 2016-08-28 Pasture mod13q1_512 <tibble [~
## 10     -53.7   -15.7  2014-09-14 2015-08-29 Pasture mod13q1_512 <tibble [~
## # ... with 2,105 more rows
```

The “SITS tibble” contains data and metadata. The first six columns contain the metadata: spatial and temporal location, label assigned to the sample, and coverage from where the data has been extracted. The spatial location is given in longitude and latitude coordinates for the “WGS84” ellipsoid. For example, the first sample has been labelled “Pasture”, at location (-55.1852, -10.8387), and is considered valid for the period (2013-09-14, 2014-08-29). Informing the dates where the label is valid is crucial for correct classification. In this case, the researchers involved in labelling the samples chose to use the agricultural calendar in Brazil, where the spring crop is planted in the months of September and October, and the autumn crop is planted in the months of February and March. For other applications and other countries, the relevant dates will most likely be different from those used in the example.

The “SITS tibble” also contains the time series data for each spatiotemporal location. The timeseries data is also organized as a tibble, with a column with the dates and the other columns with the values for each spectral band.

```r
# print the first time series
samples.tb$time_series[[1]]
```

```
## # A tibble: 23 x 7
##     Index       ndvi   evi   nir    mir   blue    red
##   * <date>     <dbl> <dbl> <dbl>  <dbl>  <dbl>  <dbl>
##   1 2013-09-14 0.424 0.280 0.288 0.244  0.0605 0.116
##   2 2013-09-30 0.467 0.264 0.257 0.167  0.0357 0.0933
##   3 2013-10-16 0.504 0.299 0.266 0.202  0.0405 0.0877
##   4 2013-11-01 0.649 0.407 0.276 0.103  0.0266 0.0588
##   5 2013-11-17 0.796 0.669 0.455 0.0875 0.0271 0.0518
##   6 2013-12-03 0.725 0.540 0.366 0.0799 0.0516 0.0584
##   7 2013-12-19 0.797 0.557 0.349 0.0478 0.0171 0.0394
##   8 2014-01-01 0.805 0.723 0.526 0.103  0.0327 0.0568
##   9 2014-01-17 0.736 0.687 0.516 0.145  0.0629 0.0784
##  10 2014-02-02 0.787 0.734 0.532 0.132  0.0452 0.0634
## # ... with 13 more rows
```

The SITS package provides functions for data manipulation and displaying information of a "SITS tibble". For example, the function `sits_bands()` lists the available bands.

```r
sits_bands(samples.tb)
```

```
## [1] "ndvi" "evi"  "nir"  "mir"  "blue" "red"
```

Another useful command is `sits_labels()` that shows the labels of the sample set and their frequencies.

```r
sits_labels(samples.tb)
```

```
## # A tibble: 9 x 3
##   label          count   freq
##   <chr>          <int>  <dbl>
## 1 Cerrado          400 0.189
## 2 Fallow_Cotton     34 0.0161
## 3 Forest           138 0.0652
## 4 Pasture          370 0.175
## 5 Soy_Corn         398 0.188
## 6 Soy_Cotton       399 0.189
## 7 Soy_Fallow        88 0.0416
## 8 Soy_Millet       235 0.111
## 9 Soy_Sunflower     53 0.0251
```

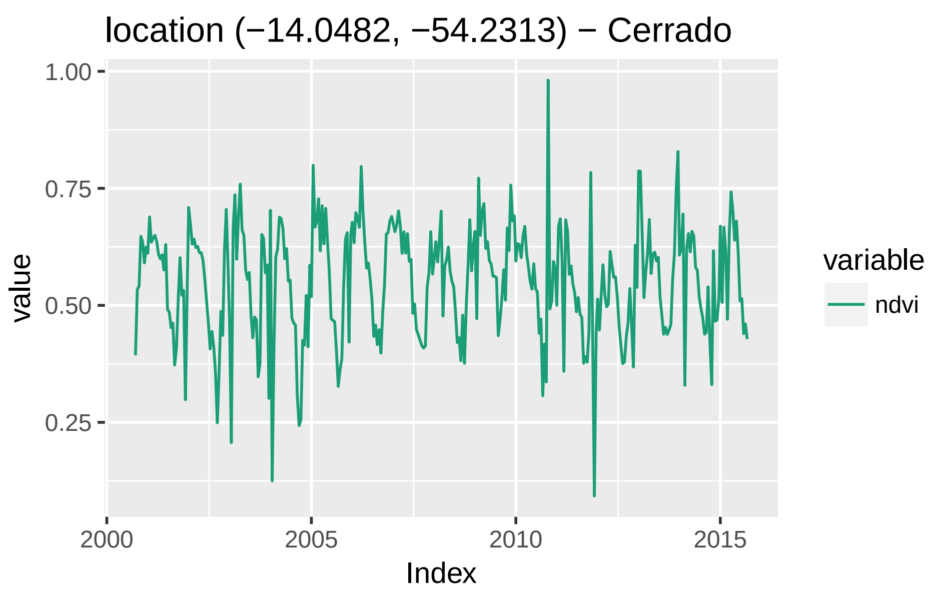
In many cases, it is useful to relabel the data set. For examples, there may be situations when one wants to use a smaller set of labels, since samples in one label on the original set may not be distinguishable from samples with other labels. We then should use the `sits_relabel()` function. This function requires a conversion list, as shown in the example below.

```
# a list for relabelling the samples
new_labels <- list("Cerrado"       = "Savanna",
                   "Pasture"       = "Grasslands",
                   "Soy_Corn"      = "Double_Cropping",
                   "Soy_Cotton"    = "Double_Cropping",
                   "Soy_Sunflower" = "Double_Cropping",
                   "Soy_Fallow"    = "Single_Cropping",
                   "Soy_Millet"    = "Single_Cropping",
                   "Fallow_Cotton" = "Single_Cropping")
# apply the sits_relabel function
samples2.tb <- sits_relabel(samples_MT_9classes, new_labels)
# view the result
sits_labels(samples2.tb)
```

```
## # A tibble: 5 x 3
##   label           count   freq
##   <chr>           <int>  <dbl>
## 1 Double_Cropping   850 0.402
## 2 Forest            138 0.0652
## 3 Grasslands        370 0.175
## 4 Savanna           400 0.189
## 5 Single_Cropping   357 0.169
```

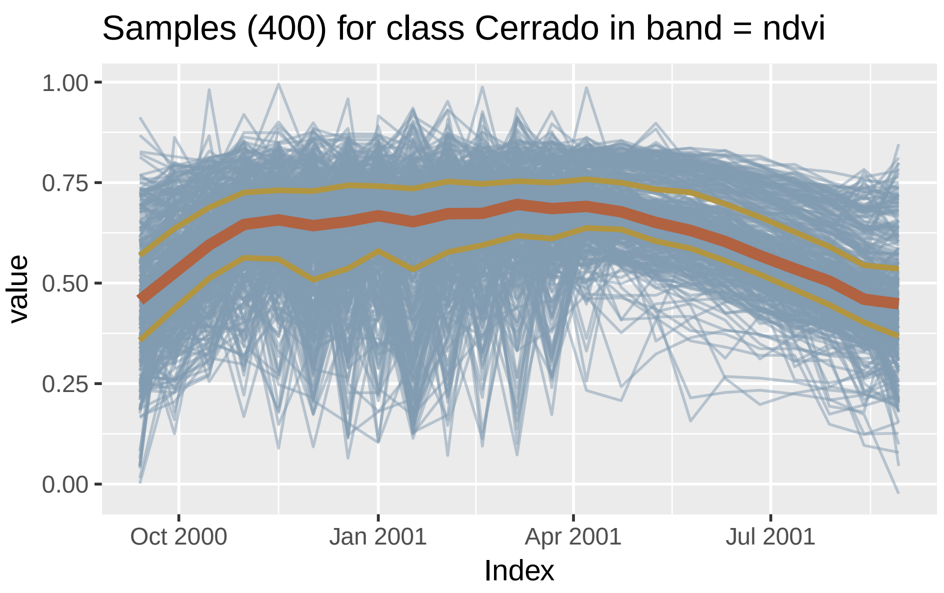
Given that we have used the tibble data format for the metadata and and the embedded time series, one can use the functions of the `dplyr`, `tidyr` and `purrr` packages of the "tidyverse" [Wickham and Grolemund, 2017] to process the data. For example, the following code uses the `sits_select()` function to get a subset of the sample data set with two bands ("ndvi" and "evi") and then uses the `dplyr::filter()` function to select the samples labelled either as "Cerrado" or "Pasture". We can then use the `sits_plot()` function to display the time series. Given a small number of samples to display, the `sits_plot()` function tries to group as many spatial locations together. In the following example, the first 15 samples of the "Cerrado" class all refer to the same spatial location in consecutive time periods. For this reason, these samples are plotted together.

```
# select the "ndvi" bands
samples_ndvi.tb <- sits_select(samples_MT_9classes, bands = c("ndvi"))
# select only the samples with the cerrado label
samples_cerrado.tb <- dplyr::filter(samples_ndvi.tb, label == "Cerrado")
# plot the first 15 samples (different dates for the same points)
sits_plot(samples_cerrado.tb[1:15,])
```



For a large number of samples, where the amount of individual plots would be substantial, the default visualisation combines all samples together in a single temporal interval (even if they are valid for different years). Therefore, all samples of the same band and the same label are aligned to a common interval. This plot is useful to show the spread of values for the time series of each band. The strong red line in the plot shows the median of the values, and the two orange lines are the first and third interquartile ranges. The sits_plot() function has different ways of working. Please refer to the documentation for more details.

```
# plot all cerrado samples together (shows the distribution)
sits_plot(samples_cerrado.tb)
```



Importing data into SITS

The SITS package allows different methods of data input, including: (a) obtain data from a WTSS (Web Series Time Service); (b) obtain data from the SATVEG service developed by EMBRAPA (Brazil's Agriculture Research Agency). (x) read data stored in a time series in the ZOO format [Zeileis and Grothendieck, 2005]; (d) read a time series from a RasterBrick [Hijmans, 2015]. Option (d) will be described in the section were we describe raster processing. The WTSS service is a light-weight service, designed to retrieve time series for selected locations and periods [Vinhas et al., 2016]. This service has been implemented by the research team of the National Institute for Space Research to allow remote access to time series data. To view service details, the user needs to call the function `sits_infoWTSS()` that provides information on the coverages available on the server.

```
sits_infoWTSS()
```

```
## -----------------------------------------------------------
## The WTSS server URL is http://www.dpi.inpe.br/tws/wtss
## Available coverages:
## MOD13Q1
## itobi
## merge
## mod13q1_512
## ------------------------------------------------------------
```

After finding out which coverages are available at the WTSS service, one may request specific information on each coverage by using the function `sits_coverageWTSS()` which lists the contents of the data set, including source, bands, spatial extent and resolution, time range, and temporal resolution. This information is then stored in a tibble for later use.

```
# get information about a specific coverage
coverage.tb <- sits_coverageWTSS("mod13q1_512")
```

```
## WTSS service not available at URL http://www.dpi.inpe.br/tws/wtss
```

```
coverage.tb
```

```
## # A tibble: 1 x 14
##   wtss.~ servi~ name  band_~ start_date end_date   time~  xmin  xmax  ymin
##   <list> <chr>  <chr> <list> <date>     <date>     <lis> <dbl> <dbl> <dbl>
## 1 <S4: ~ WTSS   mod1~ <tibb~ 2000-02-18 2017-02-18 <chr~  -180   180 -90.0
## # ... with 4 more variables: ymax <dbl>, xres <dbl>, yres <dbl>, crs <chr>
```

The user can then request one or more points using the `sits_getdata()` function. This function provides a general means of access to image time series. In its simplest fashion, the user provides the latitude and longitude of the desired location, the coverage name, the bands, and the start date and end date of the time series. If the start and end dates are not provided, all available period is retrived. The result is a "SITS tibble" that can be visualised using `sits_plot()`.

```
# a point in the transition forest pasture in Northern MT
long <- -55.57320
lat <- -11.50566
# obtain a time series from the WTSS server for this point
series.tb <-
    sits_getdata(longitude = long, latitude = lat, service = "WTSS",
                 coverage = "mod13q1_512", bands = c("ndvi", "evi"),
                 start_date = "2001-01-01", end_date = "2016-12-31")
```

```
## WTSS service not available at URL http://www.dpi.inpe.br/tws/wtss
```

```
## WTSS - unable to retrieve point - see log file for details
```

```
# plot the series
sits_plot(series.tb)
```

A useful case is when users have a set of labelled samples, that are to be used as a training data set. In this case, one usually has trusted observations which are labelled and commonly stored in plain text CSV files. The `sits_getdata()` function can receive a CSV file path as an argument. In this case, the CSV must provide some required fields. Each location information must have its latitude and longitude, the start and end dates of an yearly time series, and the label, as showm in the example bellow.

```
# open CSV file with trusted samples
read.csv(system.file("extdata/samples/samples_import.csv",
                     package = "sits"))
```

```
## # A tibble: 12 x 6
##        id longitude latitude start_date end_date   label
##     <int>     <dbl>    <dbl> <fctr>     <fctr>     <fctr>
##  1      1     -55.0    -15.2 2015-09-14 2016-08-28 Pasture
##  2      2     -55.0    -15.2 2015-09-14 2016-08-28 Pasture
##  3      3     -55.0    -15.2 2015-09-14 2016-08-28 Pasture
##  4      4     -55.2    -15.4 2015-09-14 2016-08-28 Soy_Cotton
##  5      5     -55.2    -15.4 2015-09-14 2016-08-28 Soy_Cotton
##  6      6     -55.2    -15.4 2015-09-14 2016-08-28 Soy_Cotton
##  7      7     -55.2    -15.5 2015-09-14 2016-08-28 Soy_Corn
##  8      8     -55.3    -15.6 2015-09-14 2016-08-28 Soy_Corn
##  9     9     -55.3    -15.6 2015-09-14 2016-08-28 Soy_Corn
## 10    10     -46.6    -10.4 2005-09-14 2006-08-28 Cerrado
## 11    11     -46.6    -10.4 2004-09-14 2005-08-28 Cerrado
## 12    12     -46.4    -10.9 2007-09-14 2008-08-28 Cerrado
```

CSV files enable `sits_getdata()` function to retrieve all samples at once. Groud truth samples are used to train machine learning algorithms to classify images. However it may be useful to explore some of its characteristics before generate the classification model. These characteristics can enable the specialist to verify the quality of the sample, its representativeness regarding the land use and land cover universe and, its self consistency. To support these tasks SITS package provides some functionalities such as filtering and clustering. These subjects are discussed in the next section.

Filtering techniques

Optical remotely sensed data are affected by cloud cover that introduces a large amount of noise in satellite image time series. Inter-annual climate variability also changes the phenological cycles of the vegetation, resulting in time series whose periods and intensities do not match on an year to year basis [Atkinson et al., 2012]. To associate intervals of a satellite image time series with land-use and land-cover classes, we need methods suitable for noisy and out-of-phase time series.

Several techniques have been used to correct and smooth such time-series vegetation index data, and to support the estimation of phenological parameters. These methods include curve fitting [Bradley et al., 2007], asymmetric Gaussian functions [Jonsson and Eklundh, 2002-08], wavelet decomposition [Sakamoto et al., 2005], the Savitzky–Golay filter [Chen et al., 2004], and the Whittaker smoother [Atzberger and Eilers, 2011]. In this section, we present all time series filtering routines (also known as *smoothing* routines) supported by SITS package: Savitzky–Golay filter (`sits_sgolay()`), Whitakker filter (`sits_whittaker()`), envelope filter (`sits_envelope()`) and, autoregressive integrated moving average (ARIMA) filter (`sits_cloud_filter()`).

For this purpose, we use a single time series data derived from the very same coordinate point used to retrieve WTSS data. To best visualize the effects of filtering we select only NDVI band of the first two years of the time series.

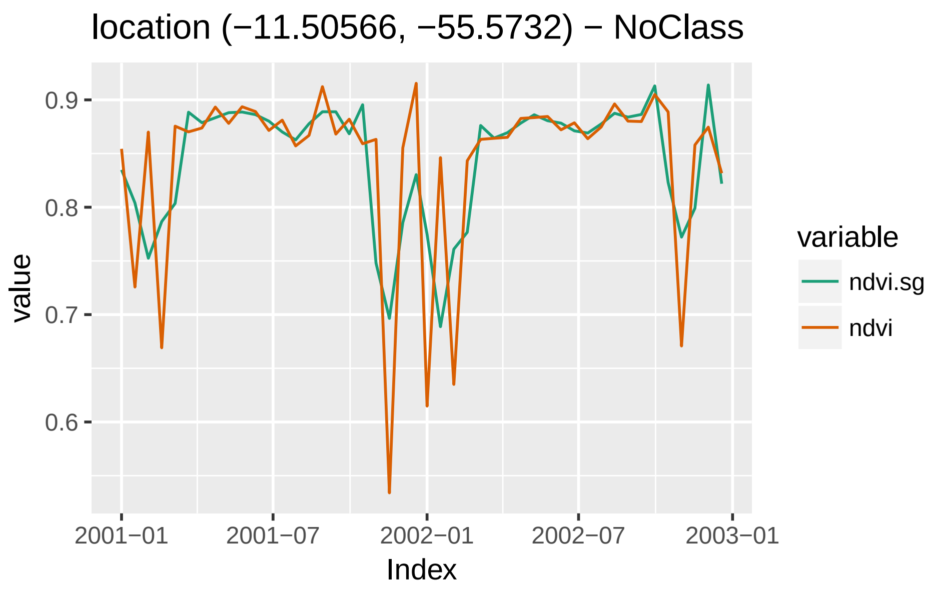
```r
# obtain two years NDVI time series from the WTSS server for the last used coordinates
series.tb <-
    sits_getdata(longitude = long, latitude = lat, service = "WTSS",
                 coverage = "mod13q1_512", bands = c("ndvi"),
                 start_date = "2001-01-01", end_date = "2002-12-31")
```

```
## WTSS service not available at URL http://www.dpi.inpe.br/tws/wtss
```

Savitzky–Golay filter

The Savitsky-Golay filter works by fitting successive sub-sets of adjacent data points with a low-degree polynomial by the method of linear least squares. The difference between an original and a filtered time series using the Savitsky-Golay filter is shown in example bellow.

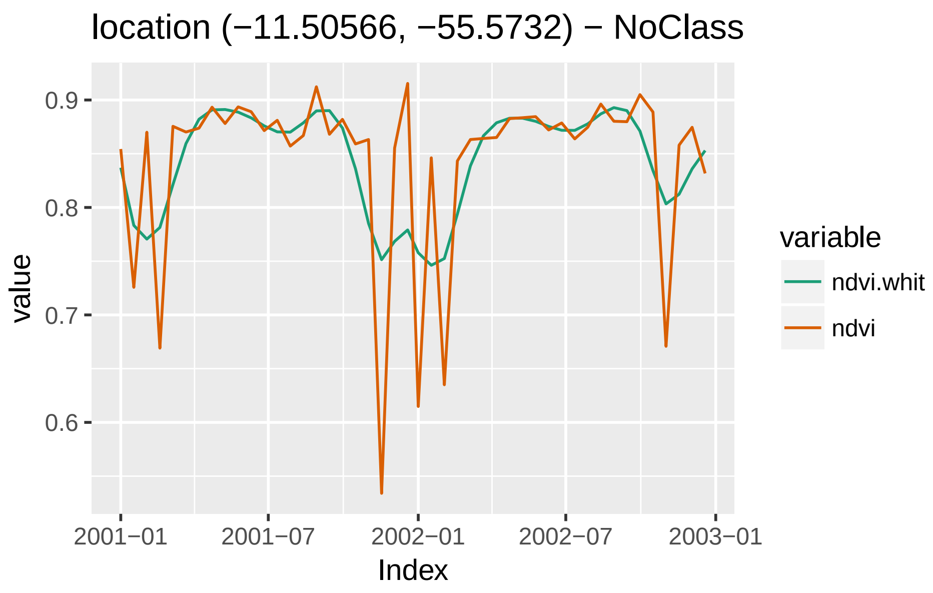
```
# apply SavitzkyGolay filter
series_sg.tb <- sits_sgolay(series.tb)
# plot the series
sits_plot(sits_merge(series_sg.tb, series.tb))
```



Whitakker filter

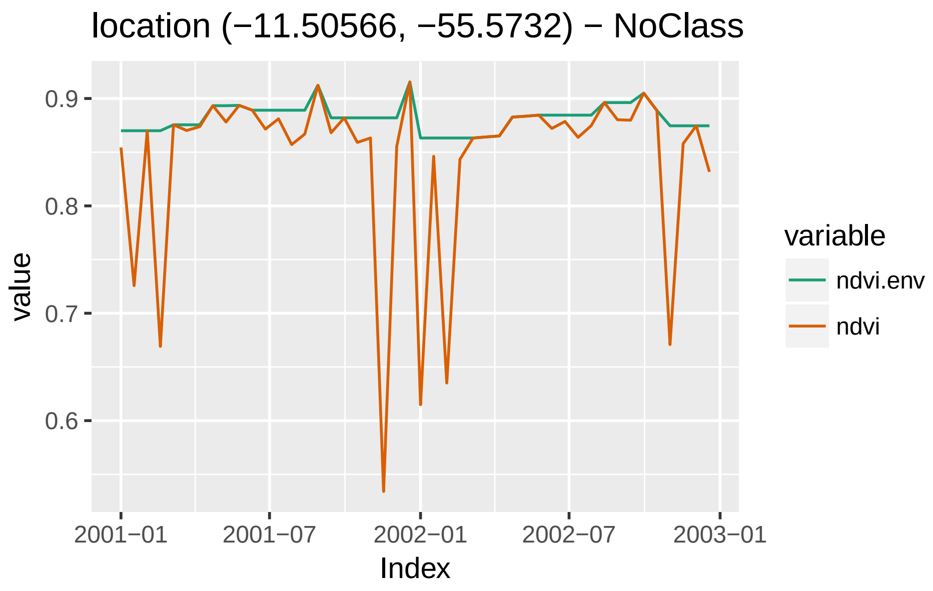
The Whittaker smoother attempts to fit a curve that represents the raw data, but is penalized if subsequent points vary too much [Atzberger and Eilers, 2011]. Mathematically it is a large, but sparse optimisation problem. The difference between an original and a filtered time series using the Whitakker filter is shown in example bellow.

```
# apply Whitakker filter
series_whit.tb <- sits_whittaker(series.tb)
# plot the series
sits_plot(sits_merge(series_whit.tb, series.tb))
```



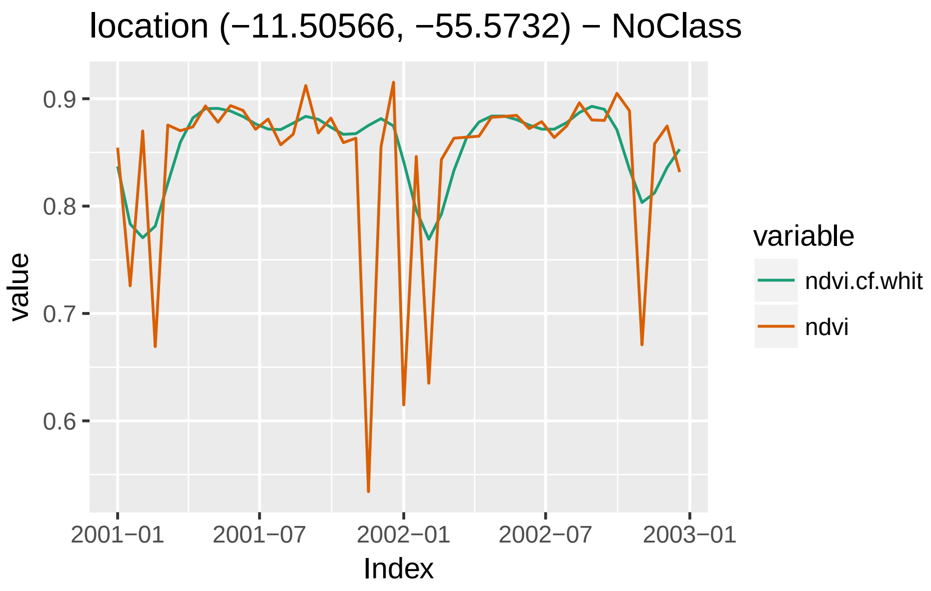
Envelope filter

```
# apply envelope filter
series_env.tb <- sits_envelope(series.tb)
# plot the series
sits_plot(sits_merge(series_env.tb, series.tb))
```



ARIMA filter

```
# apply ARIMA filter
series_cf.tb <- sits_cloud_filter(series.tb)
# plot the series
sits_plot(sits_merge(series_cf.tb, series.tb))
```



Time series filtering can be useful to work with noisy data.

Clustering techniques

One of the main uses of clustering for satellite image time series is dealing with data with high-variability and incomplete information. Since observations of same land cover can differ from year to year, clustering techniques can assist in an exploratory analysis of such data. These methods help us to determine what is the separability of the training samples.

The subject of time series clustering has attracted much interest on the literature. Two important reviews are provided by Aghabozorgi et al. [2015] and Liao [2005]. Aghabozorgi et al. [2015] divides these methods in *shape-based*, *feature-based* and *model-based*. In the shape-based approach, shapes of two time series are matched as well as possible, by a non-linear stretching and contracting of the time axes. This is the case of the dynamic time-warping (DTW) techique. In the feature-based approach, the raw time series is treated as a feature vector pertaining of a multi-dimensional space. In model-based methods, a raw time-series is transformed into model parameters and then a suitable model distance and then a clustering algorithm is applied to these parameters. SITS package provide support for the first two methods by informing an appropriate distance function.

In SITS, we have chosen to explore agglomerative hierarchical clustering. The main advantage of this method is that it does not require a predefined number of clusters as an initial parameter. This is an important feature in satellite image time series clustering since it is not easy to define the number of clusters present in a set of multi-attribute time series [Aghabozorgi et al., 2015]. In what follows, we discuss agglomerative hierarchical clustering method and present how to use it in SITS.

Agglomerative hierarchical clustering

Hierarchical clustering is a family of methods that groups elements using a distance function to associate a real value to a pair of elements. From this distance measure, we can compute the dissimilarity between any two elements from the data set. Using a linkage criterion, the algorithm decides which two clusters are merged or split at a given iteration. There are many variations of this technique since we can use different distance functions and linkage criteria. It is one of the most used clustering techniques, given its visualisation power and ease of use [Keogh et al., 2003].

There are two kinds of hierarchical clustering processes: *agglomerative* and *divisive*. The first one is also known as "bottom up strategy" in which every initial element of the data set is itself a cluster. This process starts by merging them according to the linkage criterion. It can continue up until only one cluster remains in the data set. On the contrary, the second kind is a "top down strategy" in which initial elements start as an unique cluster and at each iteration, the clusters are partitioned. This process can proceed until each element becomes itself a cluster. Algorithms for agglomerative hierarchical clustering (AHC) are much more common than the divisive algorithms counterparts [Berkhin et al., 2006]. SITS focuses on the AHC method, since it allows us to use the dendrogram to decide on the most appropriate cluster partition [Hennig, 2015]. Dendrograms are quite useful to decide on the number of cluster and one the metric and linkage criterion. In SITS, the procedure to generate clusters is divided in two steps: first, we produce a dendrogram from the data; second, we generate different clusters by cutting the dendrogram tree at some level. The first step is realized by `sits_dendrogram()` function and, the second step by `sits_cluster()`.

In AHC, different linkage criteria produce distinct results. The most common methods are single-linkage, complete-linkage, average-linkage, and Ward-linkage. Complete-linkage prioritises the within-cluster dissimilarities, producing clusters with shorter distance samples. Complete-linkage clustering can be sensitive to outliers, that can increase the resulting intracluster data variance. As an alternative, Ward proposes a criteria to minimise the data variance by means of either *sum-of-squares* or *sum-of-squares-error* [Ward, 1963]. Ward's intuition is that clusters of multivariate observations, such as time series, should be approximately elliptical in shape. The linkage creteria is used to generate the dendrogram tree and hence is passed as an argument of `sits_dendrogram()` function.

In our subsequent examples we use the relabeled set created above and stored in `samples2.tb` variable. It consists of 2,115 samples distributed in five labels: "Savanna", "Grasslands", "Double_Cropping", "Single_Cropping" and, "Forest".

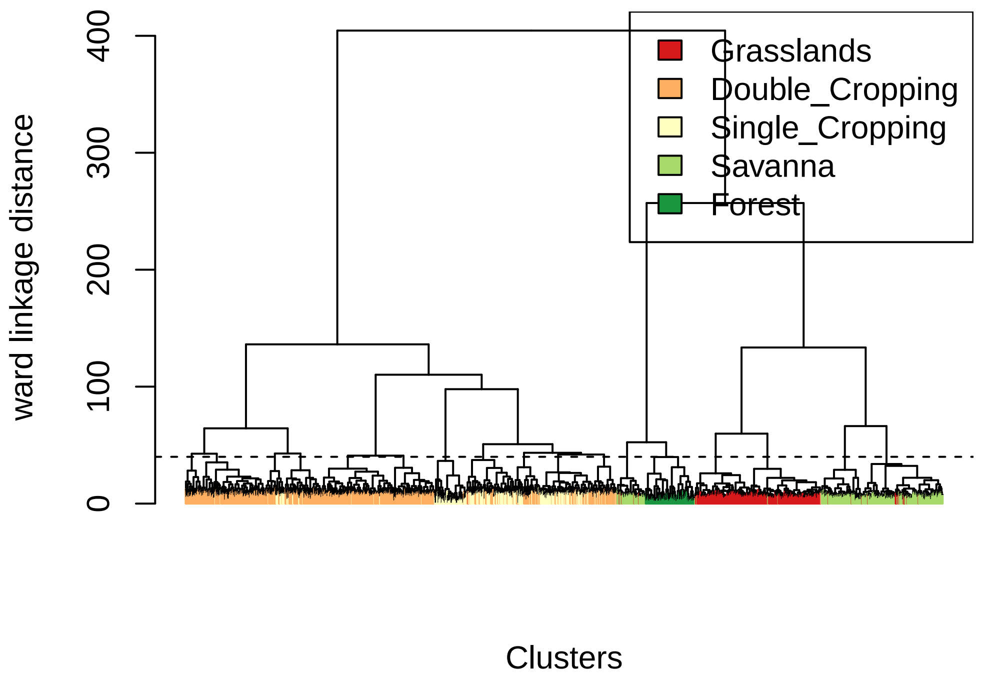
Dendrogram

A dendrogram is a tree defined by an ultrametric distance [Hennig, 2015]. It is used to visualise the successive merges in the hierarchical process. The dendrogram shows the height where each merge happened, which corresponds to the minimum distance between two clusters defined by a linkage criterion. The dendrogram can be used to evaluate cluster separability in a qualitative fashion, and helps the practitioner to determine where to cut the tree to obtain the final clusters [Liao, 2005]. In the following code, we create a dendrogram object with default clustering parameters for linkage criterion (Ward) and distance function (DTW).

```
# create a dendrogram object with default clustering parameters
dendro <- sits_dendrogram(samples2.tb)
```

`sits_dendrogram()` has an expensive computation and hence the procedure takes several seconds to finish. Once a dendrogram is computed it is easy to create the clusters by cutting the dendrogram tree at some linkage distance. We can see the dendrogram by calling `sits_plot_dendrogram()`.

```
# plot the resulting dendrogram
sits_plot_dendrogram(samples2.tb, dendro, cutree_height = 40)
```



The position a sample takes on the dendrogram, i.e. its linked edges, gives an general view how similar/dissimilar the land use classes are to each other. We can see that most of samples for each class are grouped together. We can note in the detail (at the bottom of the above figure) the existence of some outliers, those thin blocks fragments of different colour from its surroundings. Probably, these are a strong source of confusion.

In our example above, we can separate our samples by cutting the tree at some level to get two groups. At left, we have the "yellow" and "orange" samples that correspond to "Double" and "Single" croping; at right, we see "greens" and "red" samples corresponding to "Savanna", "Forest" and, "Grasslands".

```
# create clusters by cutting the dendrogram at the linkage distance 300
clusters.tb <- sits_cluster(samples2.tb, dendro, height = 300)
# show clusters samples frequency
sits_cluster_frequency(clusters.tb)
```

```
##
##                       1    2 Total
##    Double_Cropping    1  849   850
##    Forest           138    0   138
##    Grasslands       369    1   370
##    Savanna          400    0   400
##    Single_Cropping    1  356   357
##    Total            909 1206  2115
```

This division makes sense by separating "Cropping" and "Non-Cropping" land use class. It is possible to relabel samples according to its corresponding cluster by calling `sits_cluster_relabel()` function. Before we proceed the labeling its is important to remove the outliers samples. This can be accomplished with the function `sits_cluster_cleaner()` that removes all samples whose label (cluster) counts less than a given percentual relative to its cluster (label).

```
# clear sample outliers relative to clusters (those with less than 1% in a cluster)
clusters2.tb <- sits_cluster_cleaner(clusters.tb, min_clu_perc = 0.01)
# show clusters samples frequency
sits_cluster_frequency(clusters2.tb)
```

```
##
##                       1    2 Total
##    Double_Cropping    0  849   849
##    Forest           138    0   138
##    Grasslands       369    0   369
##    Savanna          400    0   400
##    Single_Cropping    0  356   356
##    Total            907 1205  2112
```

Now, we can proceed by renaming the samples labels according to its respective cluster.

```
# proceed relabeling procedure
clusters2.tb <-
```

```
    sits_cluster_relabel(clusters2.tb,
                         c("Non-Cropping_Land", "Cropping_Land"))
# show clusters samples frequency
sits_cluster_frequency(clusters2.tb)
```

```
##
##                         1    2 Total
##   Cropping_Land         0 1205  1205
##   Non-Cropping_Land   907    0   907
##   Total               907 1205  2112
```

This function must be used with caution as it renames the original label by a corresponding cluster name passed as parameter without any verification. We could proceed by cutting the dendrogram at lower linkage distances to obtain other set of clusters. We can note that at some point it is possible to achieve a reasonable separability between classes (in our example, this is indicated by the dashed line drawed according to `cutree_height` parameter). The dendrogram thus provides users with a powerful visual aid to understand data separability. Depending on the level of dendrogram cut, we will find more or less confusion inside the clusters.

Land use and land cover classification using satelite image time series

The main advantage using satellite image time series in land use studies is that the time series is methodologically consistent with the very nature of the land covers. Using this kind of data allows focusing on land changes through time. Currently, most studies that use satellite image time series for land classification still use variations of the classical remote sensing image classification methods. Given a series of images, researchers use methods that produce a single composite for the whole series @[Gomez2016]. In their review on this subject, Gomez et al. [2016] discuss 12 papers that use satellite image time series to derive image composites that are later used for classification. Câmara et al. [2016] denote these works as taking a *space-first, time-later* approach.

An example of *space-first, time-later* work on big EO data analysis is the work by Hansen et al. [2013]. Using more than 650,000 LANDSAT images and processing more than 140 billion pixels, the authors compared data from 2000 to 2010 to produce maps of global forest loss during the decade. A pixel-based classification algorithm was used to process each image to detect forest cover. The method classifies each 2D image one by one.

In our view, these methods do not use the full potential of satellite image time series. The benefits of remote sensing time series analysis arise when the temporal resolution of the big data set is able to capture the most important changes. Here, the temporal autocorrelation of the data can be stronger than the spatial autocorrelation. Given data with adequate repeatability, a pixel will be more related to its temporal neighbours than to its spatial ones. In this case, *time-first, space-later* methods lead to better results than the *space-first, time-later* approach [Câmara et al., 2016].

SITS package provides functionality to explore the full depth of satellite image time series data. It treat time series as a feature vector. To be consistent, the procedure aligns all time series from different years by its time proximity considering an given cropping schedule. Once aligned, the feature vector is formed by all pixel “bands”. The idea is to have as many temporal attributes as possible, increasing the dimension of the classification space. In this scenario, statistical learning models are the natural candidates to deal with high-dimensional data: learning to distinguish all land cover and land use classes from trusted samples exemplars, also known as training data, to infer classes of a larger data set. In the next section we discuss about machine learning techniques supported in SITS package with more detail.

Machine Learning classification models

In the training stage, Additional “bands” can be computed to increase the distinction between classes. This method has a deceptive simplicity

Support Vector Machine

Random Forest

Validation techniques

Our experiment consists of the comparison of different methods to obtain the time series prototypes for each class. After obtaining the prototypes, we classified the data using the TWDTW method and used a cross-validation procedure to evaluate the results.

Final remarks

References

Saeed Aghabozorgi, Ali Seyed Shirkhorshidi, and Teh Ying Wah. Time-series clustering–a decade review. *Information Systems*, 53:16–38, 2015.

D. Arvor, M. Meirelles, V. Dubreuil, A. Bégué, and Y. E. Shimabukuro. Analyzing the agricultural transition in Mato Grosso, Brazil, using satellite-derived indices. *Applied Geography*, 32(2):702–713, 2012.

Peter M Atkinson, C Jeganathan, Jadu Dash, and Clement Atzberger. Inter-comparison of four models for smoothing satellite sensor time-series data to estimate vegetation phenology. *Remote sensing of environment*, 123:400–417, 2012.

Clement Atzberger and Paul HC Eilers. Evaluating the effectiveness of smoothing algorithms in the absence of ground reference measurements. *International Journal of Remote Sensing*, 32(13):3689–3709, 2011.

Pavel Berkhin et al. A survey of clustering data mining techniques. *Grouping multidimensional data*, 25:71, 2006.

Bethany A Bradley, Robert W Jacob, John F Hermance, and John F Mustard. A curve fitting procedure to derive inter-annual phenologies from time series of noisy satellite ndvi data. *Remote Sensing of Environment*, 106(2):137–145, 2007.

J. Christopher Brown, Jude H. Kastens, Alexandre Camargo Coutinho, Daniel de Castro Victoria, and Christopher R. Bishop. Classifying multiyear agricultural land use data from Mato Grosso using time-series MODIS vegetation index data. *Remote Sensing of Environment*, 130:39–50, 2013.

Gilberto Câmara, Luiz Fernando Assis, Gilberto Ribeiro, Karine Reis Ferreira, Eduardo Llapa, and Lubia Vinhas. Big earth observation data analytics: matching requirements to system architectures. In *Proceedings of the 5th ACM SIGSPATIAL International Workshop on Analytics for Big Geospatial Data*, pages 1–6, Burlingname, CA, USA, 2016. ACM.

Jin Chen, Per. Jönsson, Masayuki Tamura, Zhihui Gu andBunkei Matsushita, and Lars Eklundh. A simple method for reconstructing a high-quality ndvi time-series data set based on the savitzky–golay filter. *Remote Sensing of Environment*, 91(3-4):332 – 344, 2004.

Stephan Estel, Tobias Kuemmerle, Camilo Alcantara, Christian Levers, Alexander Prishchepov, and Patrick Hostert. Mapping farmland abandonment and recultivation across Europe using MODIS NDVI time series. *Remote Sensing of Environment*, 163: 312–325, 2015.

Gillian L Galford, John F Mustard, Jerry Melillo, Aline Gendrin, Carlos C Cerri, and Carlos E Cerri. Wavelet analysis of MODIS time series to detect expansion and intensification of row-crop agriculture in Brazil. *Remote sensing of environment*, 112(2): 576–587, 2008.

Cristina Gomez, Joanne C. White, and Michael A. Wulder. Optical remotely sensed time series data for land cover classification: A review. *{ISPRS} Journal of Photogrammetry and Remote Sensing*, 116:55 – 72, 2016. ISSN 0924-2716.

M. C. Hansen, P. V. Potapov, R. Moore, M. Hancher, S. A. Turubanova, A. Tyukavina, D. Thau, S. V. Stehman, S. J. Goetz, T. R. Loveland, A. Kommareddy, A. Egorov, L. Chini, C. O. Justice, and J. R. G. Townshend. High-resolution global maps of 21st-century forest cover change. *Science*, 342(6160):850–853, 2013.

T. Hastie, R. Tibshirani, and Friedman J. *The Elements of Statistical Learning. Data Mining, Inference, and Prediction.* Springer, New York, 2009.

Christian Hennig. Clustering strategy and method selection. In Christian Hennig, Marina Meila, Fionn Murtagh, and Roberto Rocci, editors, *Handbook of cluster analysis*. CRC Press, 2015.

Robert J. Hijmans. *raster: Geographic Data Analysis and Modeling*, 2015.

G. James, D. Witten, T. Hastie, and R. Tibshirani. *An Introduction to Statistical Learning: with Applications in R.* Springer, New York, EUA, 2013.

P. Jonsson and L. Eklundh. Seasonality extraction by function fitting to time-series of satellite sensor data. *IEEE Transactions on Geoscience and Remote Sensing*, 40(8): 1824–1832, 2002-08. ISSN 0196-2892. doi: 10.1109/TGRS.2002.802519.

Per Jönsson and Lars Eklundh. TIMESAT—a program for analyzing time-series of satellite sensor data. *Computers & Geosciences*, 30(8):833 – 845, 2004. ISSN 0098-3004.

J. Kastens, J. Brown, A. Coutinho, C. Bishop, and J. Esquerdo. Soy moratorium impacts on soybean and deforestation dynamics in Mato Grosso, Brazil. *PLOS ONE*, 12(4): e0176168, 2017.

Robert E. Kennedy, Zhiqiang Yang, and Warren B. Cohen. Detecting trends in forest disturbance and recovery using yearly Landsat time series. *Remote Sensing of Environment*, 114(12):2897–2910, 2010.

Eamonn Keogh, Jessica Lin, and Wagner Truppel. Clustering of time series subsequences is meaningless: Implications for previous and future research. In *Data Mining, 2003. ICDM 2003. Third IEEE International Conference on*, pages 115–122, 2003.

Eric F Lambin, Helmut J Geist, and Erika Lepers. Dynamics of land-use and land-cover change in tropical regions. *Annual review of environment and resources*, 28(1):205–241, 2003.

T Warren Liao. Clustering of time series data—a survey. *Pattern recognition*, 38(11): 1857–1874, 2005.

Valerie J. Pasquarella, Christopher E. Holden, Les Kaufman, and Curtis E. Woodcock. From imagery to ecology: leveraging time series of all available LANDSAT observations to map and monitor ecosystem state and dynamics. *Remote Sensing in Ecology and Conservation*, 2(3):152–170, 2016. ISSN 2056-3485. doi: 10.1002/rse2.24.

Charlotte Pelletier, Silvia Valero, Jordi Inglada, Nicolas Champion, and Gerard Dedieu. Assessing the robustness of random forests to map land cover with high resolution satellite image time series over large areas. *Remote Sensing of Environment*, 187: 156–168, 2016.

Toshihiro Sakamoto, Masayuki Yokozawa, Hitoshi Toritani, Michio Shibayama, Naoki Ishitsuka, and Hiroyuki Ohno. A crop phenology detection method using time-series MODIS data. *Remote Sensing of Environment*, 96(3-4):366–374, 2005.

Lubia Vinhas, Gilberto Ribeiro, Karine Reis Ferreira, and Gilberto Camara. Web services for big Earth observation data. In *Proceedings of the 17th Brazilian Symposium on GeoInformatics*, pages 26–35, Campos do Jordão, SP, Brazil, 2016. INPE.

Joe H Ward. Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244, 1963.

Hadley Wickham and Garrett Grolemund. *R for Data Science: Import, Tidy, Transform, Visualize, and Model Data.* O'Reilly Media, Inc., 2017.

Achim Zeileis and Gabor Grothendieck. zoo: S3 infrastructure for regular and irregular time series. *Journal of Statistical Software*, 14(6):1–27, 2005.