Revista Brasileira de Cartografia
RBC | Rev. Bras. Cartografia

DOI: 10.14393/rbcv70n5-45036

Reproducible geospatial data science: Exploratory data analysis using collaborative analysis environments

Ciência especial reproduzível: Análise exploratória de dados usando ambientes colaborativos

Alber Sánchez [1]
Lubia Vinhas [1]
Gilberto Ribeiro de Queiroz [1]
Rolf Simoes [1]
Vitor Gomes [1]
Luiz Fernando F. G. de Assis [1]
Eduardo Llapa [1]
Gilberto Camara [1]

Recebido em abril de 2018.
Aprovado em dezembro de 2018.

ABSTRACT
The answers to planetary problems could be hidden in gigabytes of satellite imagery from the last 40 years. Unfortunately, scientists lack the means for processing such amount of data as they are used to work over small quantities of satellite images. To amend this issue, we propose the use of web services from Big Earth data platforms along collaborative analysis environments. Both Web services and collaborative analysis environments fit the hypothesis-test workflow followed by researchers while writing analysis routines. Besides, the early use of Big Earth data structures eases the subsequent process of scaling analysis up to larger extensions. To test our proposal, we use our own Big Earth observation data platform, on which decades of satellite images are arranged into data cubes. By using our Web services platform, we integrate those data cubes into our collaborative analysis environment (a Jupyter notebook). Since our analysis routines consume the same data structure of the whole data sets, it is easier to scale up the analysis.

KEYWORDS: Reproducible science, data analysis, time series.

[1]Instituto Nacional de Pesquisas Espaciais - INPE. Av. dos Astronautas, 1.758 - Jardim da Granja – CEP 12227-010 - São José dos Campos – SP, Brasil {alber.ipia, lubia.vinhas, gilberto.queiroz, rolf.simoes, gilberto.camara}@inpe.br; vitor@ieav.cta.br; {luizffga, edullapa}@dpi.inpe.br

RESUMO
As respostas aos problemas planetários podem estar ocultas em gigabytes de imagens de satélites adquiridas nos últimos 40 anos. Mas nem sempre os cientistas têm os meios para processar esse volume de dados uma vez que costumam trabalhar com pequenas quantidades de imagens de satélite. Para responder a esse problema, propomos o uso de serviços Web de plataformas de Big Data em ambientes colaborativos de análise. Acreditamos que os serviços Web e os ambientes colaborativos de análise encaixam com o padrão de hipótese-teste seguido pelos pesquisadores enquanto escrevem suas rotinas de análise. Além disso, o uso inicial de estruturas de dados de Big Data facilita o processo de análise em escala para uma maior extensão de dados. Para testar a nossa proposta, usamos nossa própria plataforma de Big Data de observação da Terra na qual décadas de imagens de satélite são organizadas em cubos de dados. Ao usar nossa plataforma de serviços Web, integramos esses cubos de dados em nosso ambiente colaborativo de análise (um Jupyter notebook). Uma vez que nossas rotinas de análise consomem a mesma estrutura de dados que todo o conjunto de dados, é fácil escalar a análise para toda a base.

PALAVRAS-CHAVE: Ciência reprodutível, análise de dados, séries temporais.

* * *

Introduction

The process of analyzing Earth observation data is a combination of science and art. It requires knowledge, perseverance and some resignation for the effort put on failed tests which never reach the final publications. To advance their research, scientists rely on a hypothesis-test cycle and diaries – or notebooks – to keep record of their findings. This process also relies on computer code, which scientists write their own way, following the same hypothesis-test cycle over small data sets. Nowadays, computers also help scientists to manage their digital notebooks based on concepts such as Literate Programming and Overlay Journals. Furthermore, these notebooks are being taken to the web in the form collaborative analysis environments, which are on-line documents that mix code, data, descriptions, and tables to summarize the results of scientific research. This electronic approach to analysis fits well the current data distribution model based on files (KNUTH, 1984; GRAY, 2009; PEREZ and GRANGER, 2007).

However, this approach is unsuitable to the analysis of large regions of space and time. Besides, a file-based model —as the one used to distribute satellite imagery— fosters problems such as data duplication and lack of traceability. On the other hand, global data sets are either unavailable or just too large for independent result validation. Both scenarios worsen the current scientific reproducibility crisis (BAKER, 2016; NATURE, 2016).

This situation shows the issues of scaling up software routines for data analysis. Putting aside those related to computing power —they are already addressed in the literature on the data deluge or big data — we focus on the transit from small to large datasets. It is important for scientists to keep fast and short iterations of think-code-test and to minimize the amount of re-work incurred while scaling up analysis (BELL, HEY, and SZALAY, 2009; BOYD, 2012; LI, 2016).

We addressed this problem by setting up collaborative analysis environments along big Earth data web services. The former enables fast iterations of the hypothesis-test cycle while the latter enables scientist to analyze increasingly larger data sets. In other words, the earlier scientist use with Big Earth observation data structures, the easier to scale analysis to larger extensions.

In this paper, we examine how Web services provided by big data platforms can be integrated into the analysis workflow of Earth observation data. To achieve this, we briefly introduce a computing platform – developed by us— and its web services (Sections 2 and 3). Then, we describe analysis environments and how they fit into the scientists' workflow (Section 4). Finally, we test our approach by setting up Jupyter notebook – a collaborative analysis environment – in which we mixture the web services provided by our platform and the analysis analytical tools provided by the Python programming language.

This paper is an extended version of Sanchez et al. (2017), presented in XVII Brazilian Symposium on GeoInformatics (GEOINFO 2017). For this paper, we split the analysis section in two, one with extended descriptions of

the analysis methods, and the other with the methods applications in our selected collaborative environment (A Jupyter notebook).

2 The e-sensing platform

The Brazilian National Institute for Space Research (INPE) runs the e-sensing project. This project is building a platform for scientist to research Land Use and Land Cover Change (LUCC). The platform sorts decades of satellite images into multidimensional space-time arrays.

The main requirements to these platforms are analytical scaling, software reuse, collaborative work, and replication. Analytical scaling is about moving data and code among computing platforms with little or no modifications at all. Software reuse refers to the ability to run code from different origins. Collaborative work and replication are about sharing and replicating analysis results. We address software reuse, collaborative work, and replication by using open source and open access software and data. Our platform only hosts open source software and open access data such as MODIS and LANDSAT images (CAMARA et al., 2016; STONEBRAKER et al., 2009).

We have been using our platform to classify time series of vegetation indexes of the Amazon and *Cerrado* biomes into LUCC classes. Later, during post-processing stages, we analyze the LUCC trajectories over time. But the data workflow inside our platform relies on a mixture of technologies such as scripting languages (R, Python, Bash), distributed storage (SciDB, Hadoop), and operating system tools. As a result, the scientific reproducibility of our results is compromised. Therefore, we chose web services as the way to expose our platform computing capabilities while hiding its internal complexities (ASSIS et al., 2016; CAMARA et al., 2016a; LU et al., 2016; MACIEL et al., 2017; MAUS et al., 2016).

On the other hand, the CEOS Data Cube Platform (CEOS-ODC) handles storing, accessing, and managing metadata of remotely sensed data.

CEOS-ODC is built on top of the Australian Geoscience Data Cube. Just as e-sensing, the CEOS-ODC platform can process large amounts of satellite imagery using open source tools. However, they employ different analysis and architectures. While e-sensing is focused on time series analysis, CEOS-ODC puts spatial before temporal analysis. Regarding architectures, e-sensing is built on top of array databases while CEOS-ODC is built around the programming language Python and data files; this difference is subtle but important since databases are independent of programming languages. As a consequence, the e-sensing platform is able to run analysis written in different languages while CEOS-ODC is constrained to Python scripts (CEOS, 2016).

3 A Web Service for retrieving time series

Sharing and re-using computer resources has been important since the 90s because writing software is error-prone and high performance hardware is expensive. Nowadays, Web services are a common way to address this matter. Web services are the standardized way to access software and data over the World Wide Web independently of operating systems and programming languages. Through them, scientists can access the data and algorithms available in our platform. At the same time, web services hide complexities – such as mixed technologies, and distributed storage – behind a uniform interface.

The Web Time Series Service (WTSS) retrieves time series of Earth Observation data for specific locations on Earth. WTSS reduces the gap between data and remote-sensing time-series clients through simple text representations using JSON (a standard file format). Traditionally, assembling time series of Earth Observation imagery is a time-consuming task because users need to sequentially open several image files, extract some pixels, and then store them. Instead, WTSS connects to a multidimensional array database and makes temporal queries on behalf of the client. WTSS

exposes three main operations **list_coverages**, **describe_coverage**, and **time_series**. **list_coverages** returns a JSON list of the available coverages in the service. **describe_coverage** retrieves metadata of a specific coverage. Finally, the **time_series** operation retrieves specific time series. WTSS implementation is publicly available on-line (VINHAS et al., 2016).

Moreover, WTSS has clients for the QGIS software and for the scripting languages R and Python. These WTSS clients enable scientists to access our data from on-line analysis environments.

4 Interactive and collaborative analysis environments

Literate programming is a style of coding software in which programs are treated as pieces of literature. That is, natural and machine languages are weaved together into a document where thought order prevails over code optimizations. Its goal is to create programs easier to understand and maintain and to achieve this, literate programming makes explicit the reasoning behind the code (KNUTH, 1984).

Note how literate programming fits the way scientists analyses their data. Once data is collected, scientists make research questions, and then formulate hypotheses for later testing them on the data. The question making and hypothesis formulating is better described using natural language while data processing and hypothesis testing are automated using code.

The modern realization of literate programming is the on-line analysis environments. They add collaboration and interactivity to the traditional scientific notebooks and laboratory journals. Two implementation of these analysis environments are available for R and Python. R is a computing environment designed for statistical analysis while Python is a general purpose programming language focused on readability and extensibility. Both support numerical processing, statistical data structures. Both R and Python are supported by large communities of users coming from either the field of

statistics or computer science. In this paper we preferred Python because most of the authors come from computer science field (IHAKA, 1998; JONES et al., 2001; OGRADY, 2016).

IPython adds facilities to Python for scientific computing. IPython has an interactive command with tailor-made features for scientists, such as code completion, plotting, and parallel and distributed processing. These characteristics are taken to the web in the form of Jupyter notebooks. For example, the data and algorithms regarding the recent astronomic discovery of gravitational waves are available as Jupyter notebooks (KLUYVER et al., 2016; CANTON et al., 2014; USMAN et al., 2016).

5 Analysis of time series of vegetation indexes

Vegetation indexes are simple estimates of vegetation activity derived from satellite imagery. They are independent of measurement units and for this reason they are well suited for Land cover identification. However, satellite imagery is subject to noise which induces variance on the time series of vegetation indexes (HUETE, 1985; JIANG, 2008). Statistical analysis provides several tools for time series analysis; some of them are of common usage for image analysis (e.g. line fitting, Fourier decomposition, Whitaker smoother, and the Kalman filter) and classification (e.g. Dynamic Time Warping), particularly for noise removal and classification (ATKINSON et al., 2012). In this section we provide a trivial summary of analysis techniques because a complete discussion is beyond the scope of this paper.

Line fitting is the process of finding the straight line which minimizes the differences to the points in the time series. Line fitting is useful to find global trends in the data and it is the starting point for more complex fittings.

Fourier decomposition is a smoothing technique which is based on the Discrete Fourier Transform (DFT) and its inverse function. Assuming that time series are originally defined in the time domain, DFT converts time series data to the frequency domain while the inverse DFT convert from back

from the frequency to the time domain. In the frequency domain, a time series is the sum of sinusoids characterized by a frequency. Higher frequencies correspond to noise. Smoothing is achieved by removing these high-frequency sinusoids and then reconstructing the time series using the inverse DFT (HEIDEMAN, 1984; JAKUBAUSKAS, 2001).

The Whitaker smoother computes smoothed values for each observation using least squares over the linear combination of nearest observations, while penalizing the roughness of the smoothed results (ATZBERGER, 2011; EILERS, 2003).

The Kalman filter is an algorithm for estimating an unobserved quantity from a set of noise observations. As new observations are available, the Kalman filter improves its estimation, and due to its simplicity and speed, it is suitable for applications in engineering, econometrics, and more recently, remote sensing (GREWAL and ANDREWS, 2010; KLEYNHANS, 2011).

Dynamic Time Warping (DTW) is an algorithm that computes a similarity measure – a distance – between two time series. Given a set of time series of known land coverages (the patterns), we compute the DTW distances to a time series of an unknown land cover (the samples). The samples are assigned to the labels of the patterns with the shortest DTW distance (BERNDT and CLIFFORD, 1994).

These analysis methods are applied to time series of vegetation indexes in the following section.

6 A collaborative environment - Jupyter notebook

We setup a Jupyter notebook for the exploratory analysis of time series of vegetation indexes. It mixes the web services provided by our platform and the analytical tools provided by the Python programming language. This notebook presents three common jobs regarding time series of vegetation indexes: Exploratory analysis, filtering or smoothing, and classification.

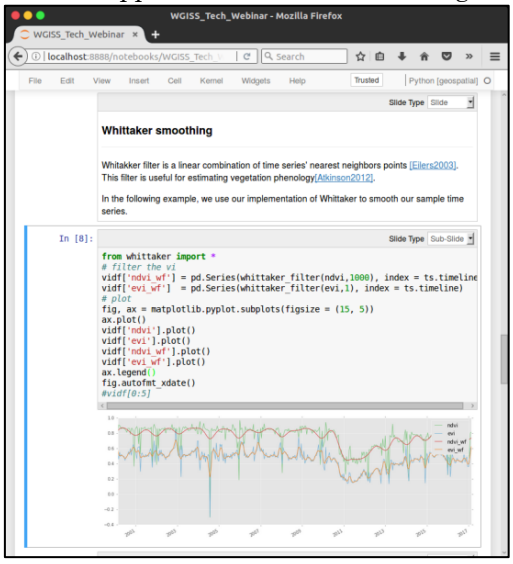
Figure 1 – Get a time series into a Python pandas data frame.

```
1.  import pandas as pd
2.  from wtss import wtss
3.  from tsmap import *
4.  w = wtss("http://www.dpi.inpe.br/tws")
5.  latitude  = -14.919100049
6.  longitude = -59.11781088
7.  ts = w.time_series("mod13q1_512", ("ndvi", "evi"), \
8.      latitude, longitude)
9.  ndvi = pd.Series(ts["ndvi"], index = ts.timeline) * \
10.     cv_scheme['attributes']['ndvi']['scale_factor']
11. evi  = pd.Series(ts["evi"],  index = ts.timeline) * \
12.     cv_scheme['attributes']['evi']['scale_factor']
13. vidf = pd.DataFrame({'ndvi': ndvi, 'evi': evi})
```

Source: Elaborated by the authors.

In the exploratory analysis, we get the data and then plot the time series and its location on a map. Figure 1 shows how to retrieve MODIS data into a data frame which is a table-like data structure. Lines 1 to 3 load the existing libraries, while lines 4 to 6 establish a point on Earth, some vegetation indexes, and where to find the Web Service. Line 7 retrieves time series from the Web Service, and finally, lines 9 to 13 arrange the data into a data structure called *data frame*.

Figure 2 – An on-line analysis environment for time series of Earth observation data. This environment displays a description of the Whitaker smoother, its Python implementation, and its results when applied to a time series of vegetation indexes.



Source: Elaborated by the authors.

Once the time series is formatted as a data frame, it is possible to apply on it functions that receive and return data frame's columns as parameters. In this way, we smoothed our time series using the Whittaker smoother (Figure 2), the Kalman filter, and the Fourier decomposition.

The code used to apply filters on the data is illustrated in Figure 3. Line 1 imports the filter which is applied to vegetation indexes (lines 2 and 3). The remaining lines of code print the filtered vegetation indexes along with the original data (lines 5 to 11). This code pattern is repeated for applying the Kalman filter and the Fourier decomposition (Figure 4).

Figure 3 – Filter a time series using the Whitaker smoother.

```
1.  from whittaker import *
2.  vidf['ndvi_wf'] = pd.Series(whittaker_filter(ndvi,1000),index = ts.timeline)

3.  vidf['evi_wf']  = pd.Series(whittaker_filter(evi,1), index = ts.timeline)
4.  fig, ax = matplotlib.pyplot.subplots(figsize = (15, 5))
5.  ax.plot()
6.  vidf['ndvi'].plot()
7.  vidf['evi'].plot()
8.  vidf['ndvi_wf'].plot()
9.  vidf['evi_wf'].plot()
10. ax.legend()
11. fig.autofmt_xdate()
```

Source: Elaborated by the authors.

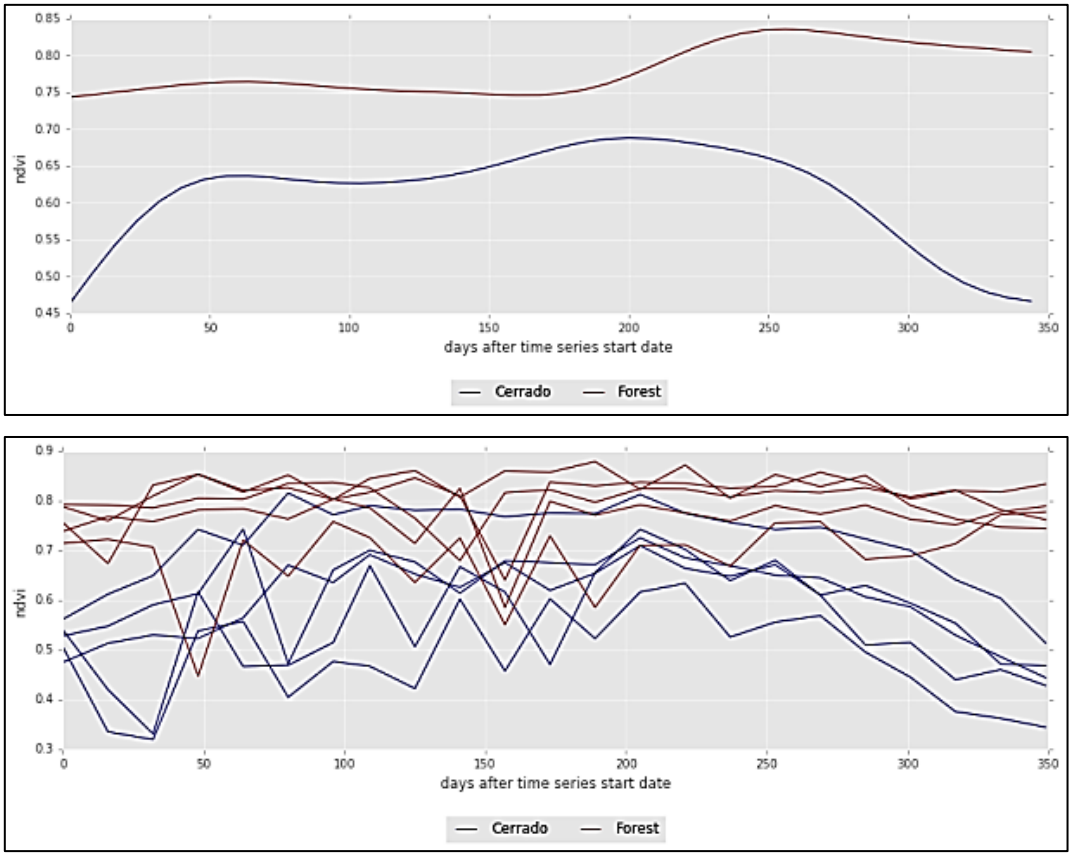
Figure 4 – Fourier decomposition of time series of vegetation indexes.

Source: Elaborated by the authors.

The last example in our Jupyter notebook is classification. We used Dynamic Time Warping (DTW) to classify time series of vegetation indexes. We prepared a set of pattern time series corresponding to the land covers

cerrado and forest. We also collected a set of sample points from which we know the latitude, the longitude and the land cover over a specific time interval; then we retrieved the time series of these points using WTSS. Figure 4 shows the time series of both patterns and samples. Figure 5 shows the code required to read the prepared files, retrieve the time series and to do the classification: Lines 1 and 2 load libraries while lines 3 and 5 load patterns of vegetation indexes and samples points from text files. Line 6 retrieves the time series corresponding to the samples. Finally, line 7 calls the classifier on the samples using the patterns.

Figure 5 – Patterns (top) and samples (bottom) of NDVI time series for classification.



Source: Elaborated by the authors.

Figure 6 – Python code for classifying time series using Dynamic Time Warping.

```
from dtw import *
from tools import *
patterns_ts = pd.read_json("examples/patterns.json", orient='records')
patterns_ts["timeline"] = pd.to_datetime(patterns_ts["timeline"])
samples = pd.read_csv("examples/samples.csv")
samples_ts = wtss_get_time_series(samples)
classification = classifier_1nn(patterns_ts, samples_ts)
```

Source: Elaborated by the authors.

In summary, we joined data and analysis environments in order to plot, filter, and classify time series of Earth observation data by means of Jupyter notebooks and web services. This approach is flexible as users can use the same data and web services over different programming languages and analysis environments. For example, we setup another notebook using R, which is a statistical programming language. We do not describe this R notebook here because of lack of room, but the code is available on-line[2].

7 Conclusions

In this paper, we discussed how literate programming is being taking to the Web as interactive and collaborative analysis environments. We also showed how these environments are enhanced with web services and how both – environments and services – help scientists to prepare their analysis routines. We set up a Jupyter notebook in which we analyzed data retrieved by the Web Time Series Service. In this way, we showed how to display, filter, smooth and classify time series of vegetation indexes. This is a convenient for scientists not only to interact with time series of Earth observation data but also to prepare their analysis routines before running them on big Earth observation data platforms such as e-sensing.

[2] e-Sensing: Big Earth observation data analytics for land use and land cover change information https://github.com/e-sensing/SITS_R_notebook

Web services close the gap between big Earth observation data and analysis tools by means of collaborative environments for small amounts of data. As the amount of data to be processed increases, it is better to send the analysis routine to the data which is an ongoing effort at the e-sensing project.

Finally, we would like to remark that the aforementioned the Jupyter notebook, the Web Time Series Service, and the analysis routine are available on-line to everyone at http://github.com/e-sensing/wgiss-py-webinar.

8 Acknowledgements

The authors are supported by the São Paulo Research Foundation (FAPESP) e-science program (grant 2014-08398-6). Gilberto Câmara is also supported by CNPq (grant 312151-2014-4).

References

ASSIS, L. F. et al. Big data streaming for remote sensing time series analytics using MapReduce. **Proceedings of the XVII Brazilian Symposium on GeoInformatics**, 2016.

ATKINSON, P. M. et al. Inter-comparison of four models for smoothing satellite sensor time-series data to estimate vegetation phenology. **Remote Sensing of Environment**, v. 123, p. 400–417, aug 2012. ISSN 00344257.

ATZBERGER, C.; EILERS, P. H. C. A time series for monitoring vegetation activity and phenology at 10-daily time steps covering large parts of South America. International Journal of Digital Earth, v. 4, n. 5, p. 365–386, set. 2011.

BAKER, M. Is there a reproducibility crisis? **Nature**, v. 533, n. 7604, p. 452–454, may 2016. ISSN 0028-0836.

BELL, G.; HEY, T.; SZALAY, A. Computer science. Beyond the data deluge. **Science** (New York, N.Y.), v. 323, n. 5919, p. 1297–1298, 2009. ISSN 0036-8075.

BERNDT, D. J.; CLIFFORD, J. Using dynamic time warping to find patterns in time series. In: FAYYAD, U. M.; UTHURUSAMY, R. (Ed.). **KDD Workshop**. [S.l.]: AAAI Press, 1994. p. 359–370. ISBN 0-929280-73-3.

BOYD, D.; CRAWFORD, K. Critical Questions for Big Data. **Information, Communication & Society**, v. 15, n. 5, p. 662–679, 2012. ISSN 1369-118X.

CAMARA, G. et al. Big earth observation data analytics: matching requirements to system architectures. In: ACM. **Proceedings of the 5th ACM SIGSPATIAL International Workshop on Analytics for Big Geospatial Data**. Burlingname, CA, USA, 2016. p. 1–6.

CANTON, T. D. et al. Implementing a search for aligned-spin neutron star-black hole systems with advanced ground based gravitational wave detectors. **Phys. Rev.**, D90, n. 8, p. 082004, 2014.

CEOS. The CEOS Data Cube**. Three-year work plan 2016-2018**. Committee on Earth Observation Satellites, 2016.

EILERS, P. H. C. A Perfect Smoother. **Analytical Chemistry**, v. 75, n. 14, p. 3631-3636, jul. 2003.

GRAY, J. Jim gray on escience: A transformed scientific method. **The fourth paradigm: Data-intensive scientific discovery**, Microsoft Research Redmond, WA, v. 1, 2009.

GREWAL, M.; ANDREWS, A. Applications of Kalman Filtering in Aerospace 1960 to the Present. **IEEE Control Systems Magazine**, v. 30, n. 3, p. 69–78, jun 2010. ISSN 0272-1708.

HEIDEMAN, M.; JOHNSON, D.; BURRUS, C. Gauss and the history of the fast fourier transform. **IEEE ASSP Magazine**, v. 1, n. 4, p. 14–21, out. 1984.

HUETE, A. R.; JACKSON, R. D.; POST, D. F. Spectral response of a plant canopy with different soil backgrounds. **Remote Sensing of Environment**, v. 17, n. 1, p. 37–53, feb. 1985.

IHAKA, R. R: Past and future history. **Computing Science and Statistics**, v. 392396, 1998.

JAKUBAUSKAS., M. E.; LEGATES., D. R.; KASTENS., J. H. Time-Series, Harmonic Analysis of Sensing., AVHRR NDVI Data. **Photogrammetric Engineering & Remote Sensing**, v. 67, n. 4, p. 461–470., 2001.

JIANG, Z. et al. Development of a two-band enhanced vegetation index without a blue band. **Remote Sensing of Environment**, v. 112, n. 10, p. 3833–3845, 2008.

JONES, E. et al. SciPy: Open source scientific tools for Python. 2001. Site <http://www.scipy.org/> Accessed on September 2018

KLEYNHANS, W. et al. Detecting Land Cover Change Using an Extended Kalman Filter on MODIS NDVI Time-Series Data. **IEEE Geoscience and Remote Sensing Letters**, v. 8, n. 3, p. 507–511, 2011.

KLUYVER, T. et al. Jupyter Notebooks—a publishing format for reproducible computational workflows. **Positioning and Power in Academic Publishing: Players, Agents and Agendas**, p. 87–90, 2016.

KNUTH, D. E. Literate programming. **The Computer Journal**, v. 27, n. 2, p. 97–111, 1984.

LI, S. et al. Geospatial big data handling theory and methods: A review and research challenges. **ISPRS Journal of Photogrammetry and Remote Sensing**, International Society for Photogrammetry and Remote Sensing, Inc. (ISPRS), v. 115, p. 119–133, 2016. ISSN 09242716.

LU, M. et al. Spatio-temporal change detection from multidimensional arrays: Detecting deforestation from MODIS time series. **ISPRS Journal of Photogrammetry and Remote Sensing**, v. 117, p. 227–236, jul 2016. ISSN 09242716.

MACIEL, A. M. et al. STILF - A spatiotemporal interval logic formalism for reasoning about events in remote sensing data. **Brazilian symposium on remote sensing**, 18. (SBSR). Proceedings. São José dos Campos: National Institute for Space Research (INPE), 2017. p. 4558–4565.

MAUS, V. et al. A time-weighted dynamic time warping method for land-use and land-cover mapping. **IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing** v. 9, n. 8, p. 3729 – 3739, 2016.

NATURE. Reality check on reproducibility. **Nature**, v. 533, n. 7604, p. 437–437, may 2016. ISSN 0028-0836.

OGRADY, S. The redmonk programming language rankings: January 2016. Site <http://redmonk.com/sogrady/2015/07/01/language-rankings-6-15> Accessed on November 2017.

SANCHEZ, A. et al. Reproducible geospatial data science: Exploratory Data Analysis using collaborative analysis environments. **Brazilian Symposium on Geoinformatics**, Salvador, BA, Brazil, December 6-8, 2017, Online proceedings, 2017.

STONEBRAKER, M. et al. Requirements for Science Data Bases and SciDB. **Fourth Biennial Conference on Innovative Data Systems Research,** Asilomar, CA, USA, January 4-7, 2009, Online Proceedings, 2009.

USMAN, S. A. et al. The PyCBC search for gravitational waves from compact binary coalescence. **Classical and Quantum Gravity**., v. 33, n. 21, p. 215004, 2016.

VINHAS, L. et al. Web services for big earth observation data. **GeoInfo**. [S.l.: s.n.], 2016. p. 166–177.