Web Services for Big Earth Observation Data

Lubia Vinhas[1], **Gilberto Ribeiro de Queiroz**[1],
Karine Reis Ferreira[1], **Gilberto Camara**[1]

[1]Instituto Nacional de Pesquisas Espaciais (INPE)
Av dos Astronautas, 1758, 12227-010, São José dos Campos, SP, Brazil

{lubia.vinhas, gilberto.queiroz}@inpe.br
{karine.ferreira, gilberto.camara}@inpe.br

Abstract. *The aim of geospatial web services is to enable users to share and process geospatial data on the Web, no matter what platform or protocol is used. In this paper, we investigate what are the design decisions needed to implement web services for big Earth observation (EO) data. The focus of the work is discussing what is unique about big EO data and why current standards are unsuitable for big EO data analytics. Instead, simpler APIs can be more effective for accessing big EO data than generic services such as WMS and WCS, specially for data analytics. We support this viewpoint by describing the WTSS Web Time Series Service that offers time series of remote sensing data using a simple API and is suited for big data analytics.*

1. Introduction

Earth observation (EO) satellites produce vast amounts of geospatial data. As an example, the archive of medium-resolution Landsat satellite at the US Geological Survey holds more than five million images of the Earth's surface, with about 1 PB of data. 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 for Geoinformatics research: *How to design and build technologies that allow the EO community to use big data sets?*

EO satellites provide a continuous and consistent set of information about the Earth's land and oceans. Using big EO data sets, we can detect long-term changes and trends in the environment, and measure the impacts of climate change, urban and ocean pollution and land expansion for food production. To transform data in information, we need to analyse terabytes of data, with different spectral, temporal, and spatial resolutions, acquired by different sensors and stored at different places. Given the broad application areas of EO data, researchers need to consider how to store and organise them in a way that will facilitate data interoperability and data analytics.

The issue of handing big geospatial data has attracted much attention in recent literature. Vitolo et al. (2015) reviewed the web technologies dealing with big environmental data, concluding that domain-specific projects often require tailored solutions. Li et al. (2016) revisited the existing geospatial data handling methods and theories to determine if they are still capable of handling emerging geospatial big data. They concluded that traditional data handling approaches and methods are inadequate and new developments are needed. However, their conclusions are focused on the processing of discrete geographical data, as data captured by mobile devices and GPS based sensors. Amirian et al. (2014)

evaluated typical and modern database systems used for the management of big geospatial data, concluding that there is no single solution that meets all the user needs, therefore geospatial data handling is an open issue. Guo et al. (2014) presented an on-demand computing schema for remote sensing images based on a processing chain model that runs on a private cloud computing platform. However, their motivation is on the processing of timestamps of imagery, not fully exploiting the processing of time series of data.

There also have been recent technological developments for handling big geospatial data. Alternatives include MapReduce-based solutions such as Google Earth Engine (Gorelick, 2012), distributed multidimensional array databases such as SciDB (Stonebraker et al., 2013) and object-relational DBMS extensions such as RasDaMan (Baumann et al., 1998). Unlike the current state of object-relational databases for geospatial data, which have been standardised, each of these solutions is unique and incompatible with the others.

Given the incompatibility of the available architectures for big geospatial data, the question that naturally arises is whether we can still enjoy the benefits of interoperability provided by web services. Geospatial web services are important for interoperability, integration with applications and data sharing. In other words, *is it possible and viable to extend the current generation of geospatial web services to work with big geospatial data*?

To answer this question, we need to consider the standards proposed by the Open Geospatial Consortium (OGC). The OGC has played a crucial role in geospatial data interoperability by proposing web services standards for visualising, disseminating and processing geospatial data. These include the Web Map Service (WMS), Web Coverage Service (WCS), Sensor Observation Service (SOS) and Web Processing Service (WPS) standards. Some space agencies have adopted some OGC web services to deliver their EO products. However, OGC standards have not yet proved to be a consensus solution when it comes to process the big EO data sets available in the agencies repositories, to extract information in a timely and robust way. In this paper, we investigate if the implicit decisions taken by OGC when designing services such as WMS, WCS and WFS allow them to be used for big EO data. We also consider whether new kinds of web services are required to handle big EO data efficiently.

In what follows we will argue that OGC services such as WMS, WFS and WMS rely implicitly on a backend that stores EO data as as a stack of 2D images. We consider that their design is inadequate for scientists working with large sets of remote sensing time series. We also describe a new proposal for a Web Time Series Services (WTSS), which is better suited for accessing large sets of EO time series. We illustrate how WTSS works by presenting a case study. We conclude by looking ahead at the improvements required in WMS and WCS for efficient support for big EO data retrieval and processing.

2. Why current Web Coverage Services are not fit for big data

When designing an architecture for big EO data, one needs to consider the needs of the users. Researchers typically use web services such as WCS and WFS as data sources; they access them by an application, usually a desktop GIS. The typical access query is to retrieve a vector or raster 2D coverage. After retrieving the coverage, users will then apply spatial analysis or image processing algorithm to the data. This *modus operandi* still follows the traditional cartographic abstractions of maps, where each coverage is processed separately.

This working model breaks down when one wants to get access to a large EO data set. For example, the MODIS vegetation index archive for Brazil from 2002 to 2014 has 12.000 independent files. Using WCS to retrieve such a data set for analysis, one has to retrieve each file, run the desired algorithm and then move to the next file. Obviously, this way of working is not adequate. Researchers need to address the set of data as whole to run their algorithms.

To better understand the requirements of big EO data analytics, consider a conceptual view of the problem. Earth observation satellites revisit the same place at regular intervals. Thus measures can be calibrated so that observations of the same place in different times are comparable. These calibrated observations can be organised in regular intervals, so that each measure from sensor is mapped into a three dimensional multivariate array in space-time (see Figure 1). Let $S = \{s_1, s_2, \ldots, s_n\}$ be a set of remote sensing images which shows the same region at *n* consecutive times $T = \{t_1, t_2, \ldots, t_n\}$. Each location *[x, y, t]* of a pixel in an image (latitude, longitude, time) maps to a *[i, j, k]* position in a 3D array. Each array position *[i, j, k]* will be associated to a set of attributes values $A = \{a_1, a_2, \ldots, a_m\}$ which are the sensor measurements at each location in space-time (see Figure 1). For optical sensors, these observations are proportional to Earth's reflexion of the incoming solar radiation at different wavelengths of the electromagnetic spectrum. Therefore, a 3D array is a better appropriate conceptual model for big EO data than the 2D map metaphor used in GIS and in most OGC web services.

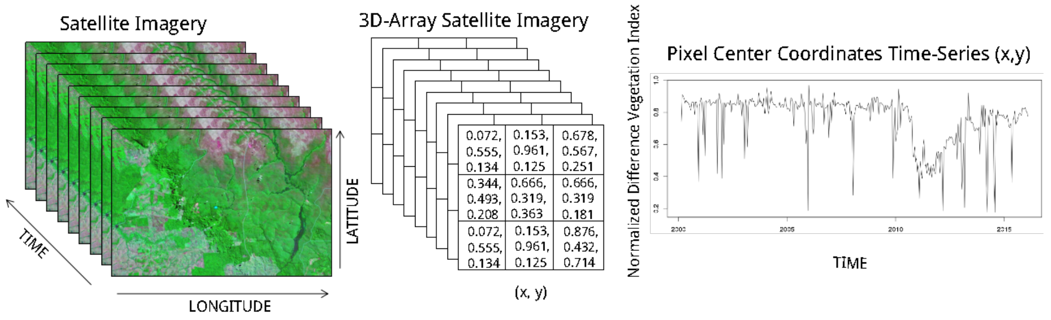


Figure 1. A Normalized Difference Vegetation Index (NDVI) time series.

Using the 3D array metaphor, scientists can approach the classification problem in different ways. They can first classify each image separately and then compare the resulting classifications in time (the *space-first, time-later* approach). Alternatively, they can first classify each time series and then join the results in space (the *time-first, space-later* approach). Combinations of these approaches are also possible. To enable researchers to develop innovative analytical methods that make good use of big EO data, the system architecture needs to support both approaches. The main question that arises is whether a standard such as WCS supports the 3D array metaphor.

In principle, the concept of WCS could apply both to 2D and 3D arrays. In practice, the current WCS 3D coverage specification is not fully supported by the implementations available, that serves only 2D coverages. The OGC WCS can easily handle multivariate coverages, but it is not easy to specify a multivariate, multitemporal coverage in WCS. Furthermore, coverages are still retrieved one by one, thus creating a stack of 2D maps on the client machine. In this way, WCS does not free the user from the burden of dealing with large numbers of individual files. Instead of dealing with a big EO data set as a sequence of maps, as is implicit in WFS and WCS, researchers need new tools. In this perspective, good geospatial web services for big data should be designed to serve the needs of information extraction algorithms. These methods will in general be different of those available in a typical GIS.

3. Web Time Series Service

In the previous sections, we argued that the current generation of OGC standards has been designed to match the cartographic metaphors used by today's GIS. We also argued that the 3D array metaphor is better suited for big EO data than the map metaphor. We now consider how to support an important kind of algorithms that extract information from remote sensing time series.

To get a remote sensing time series, one first organises a large set of EO data as a 3D array. From each pixel location in the array, one can extract a time series of one or more variables for a temporal interval. The benefits of remote sensing time series analysis arise when the temporal resolution of the big data set is sufficient to capture the most important changes. In this case, the temporal autocorrelation of the data can be stronger than the spatial autocorrelation. In other words, given data with adequate repeatability, a pixel will be more related to its temporal neighbours rather its spatial ones. In this case, *time-first, space-later* methods will give better results than the *space-first, time-later* approach.

Recent results in the literature show that analysis of satellite image times series enables extracting long-term trends of land change (Olsson et al., 2005; Adami et al., 2012; Sakamoto et al., 2005). Such information which would be harder to obtain by processing 2D images separately. These analysis are supported by algorithms such as TWDTW (Maus et al., 2016), TIMESTAT (Jönsson and Eklundh, 2004) and BFAST (Verbesselt et al., 2010). These algorithms process individual time-series and combine the results for selected periods to generate classified maps.

Motivated by the need to retrieve satellite image time series from large 3D arrays, we have designed and implemented the Web Time Series Service (WTSS). A WTSS server takes as input a 3D array. Each array has a spatial and a temporal dimension and can be multidimensional in terms of its attributes. For example, the dataset showed in Figure 1 represents a time series of the NDVI index extracted from a large 3D MODIS array.

The WTSS service is independent of the actual data architecture used for 3D array store. It can work with solutions such as flat files, MapReduce distributed datasets, array databases or object-relational databases. We have implemented the service using both a set of flat files and the SciDB array database management system (Stonebraker et al., 2013), with the same external interface. The WTSS interface provides three operations:

1. *list_coverages*: this operation allows clients to retrieve the capabilities provided by any server that implements WTSS. Or simply put, it returns a list of coverage names available in a server instance. The server response is a JSON[1] document. The names returned by this operation can be used in subsequent operations.

```
// request:
www.dpi.inpe.br/wtss/list_coverages

// response
 "coverages": [ "mod09q1", "mod13q1"]
```

2. *describe_coverage*: this operation returns the metadata for a given coverage identified by its name. It includes its range in the spatial and temporal dimensions. It also receives a JSON document as a response.

```
// request:
www.dpi.inpe.br/wtss/describe_coverage?name=mod09q1

// response:
{"name": "mod09q1",
  "description": "Surface Reflectance 8-Day L3 Global 250m",
  "detail": "https://lpdaac.usgs.gov/dataset_discovery/
             modis/modis_products_table/mod09q1",
  "dimensions":
  [ {"name": "col_id", "description": "column",
     "min_idx": 0, "max_idx": 172799, "pos": 0 },
    {"name": "row_id", "description": "row",
     "min_idx": 0, "max_idx": 86399, "pos": 1},
    {"name": "time_id", "description": "time",
     "min_idx": 0, "max_idx": 1024, "pos": 2  } ],
  "attributes":
  [ {"name": "red",
     "description": "250m Surface Reflectance Band 1",
     "datatype": "16-bit signed integer",
     "valid_range": {"min": -100, "max": 16000},
     "scale_factor": 0.0001, "missing_value": -28672} ]
    "geo_extent": {
     "spatial": {
       "extent": {"xmin": -180.0, "ymin": -90.0,
                  "xmax":  180.0, "ymax":  90.0},
       "resolution": {"x": 0.0020833333, "y": 0.0020833333},
       "srid": 4326},
     "temporal": {
       "interval": {"start": "2000-02-18", "end": "2014-08-13"},
       "resolution": 8,
       "unit": "day"}
```

[1]JavaScript Object Notation (JSON) is a lightweight data-interchange format (see `json.org`).

3. *time_series*: this operation requests the time series of values of a coverage attribute at a given location.

```
// request:
http://www.dpi.inpe.br/wtss/time_series?coverage=mod0q1&
    attributes=red&latitude=-12&longitude=-54&
    start=2000-02-18&end=2000-03-21

// response:
  "result": {
    "attributes": [
      {
        "attribute": "red",
        "values": [ 3726, 2834, 4886, 231, 1264 ]},
     ],
    "timeline": [ "2000-02-18", "2000-02-26",
                  "2000-03-05", "2000-03-13", "2000-03-21" ],
    "center_coordinates": {
      "latitude": -4.9989583328814176,
      "longitude": -54.000193143463676 } },
  "query": {
    "coverage": "mod09q1",
    "attributes": [ "red"],
    "latitude": -5,
    "longitude": -54}
```

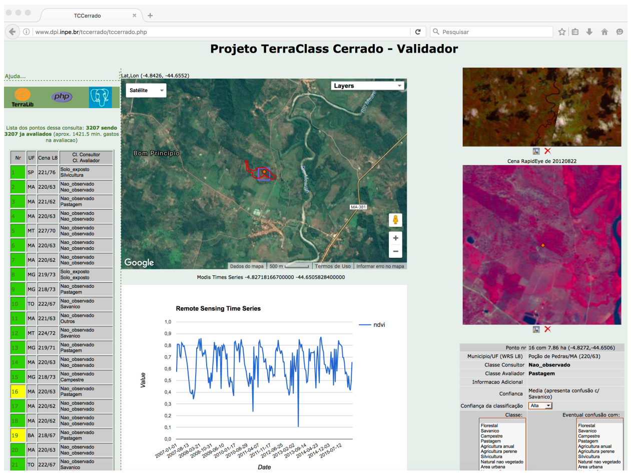
WTSS saves time when dealing with huge volumes of data because it closes the gap between big EO data and applications in a flexible and simple API. Syntactically, WTSS is different from OGC standards, since it uses the JSON format instead of XML to deliver complex responses. Writing lightweight web applications for data visualisation and retrieval is simpler with JSON. Its more cohesive and short notation helps users of the API to understand how the service works and simplifies decoding by client applications. OGC is considering using JSON based formats, as shown by the draft implementation of the OM-JSON service for Observation and Measurement Data (OGC, 2015a). This shows that OGC recognises the benefits of the JSON format for lightweight services.

WTSS time series response is also easy to be consumed by statistical computing languages such as **R**. We have developed an **R** package that implements the WTSS client to allows researchers to use **R** methods such as TWDTW (Maus et al., 2016) and BFAST (Verbesselt et al., 2010) to analyse large sets of satellite image time series.

4. Using the WTSS service

As an example of using the WTSS, consider an application that aims at validating thematic maps created from remote sensing imagery. Jensen (2009) points out that there are different types of errors introduced during the mapping process and stress the need for tools to assess the map accuracy, increasing the usefulness and credibility of the data. In a typical web-based validation tool, experts have to analyse different data, such as images with different spatial resolutions, and also the temporal dynamics at the locations represented by the EO time-series values.

Figure 2 shows a web-based validation tool, that includes a graphical display of MODIS/NDVI series. The user interacts with the application by clicking on the map, this will generate a location that will be used in a request to a WTSS server, using a JavaScript WTSS client (see Listing 1).


TCCerrado
www.dpi.inpe.br/tccerrado/tccerrado.php
Ajuda...
Lat,Lon (-4.8426, -44.6552)
Satélite
Layers
Bom Principio
Google
Modis Times Series -4.82718166700000 -44.6505828400000
Remote Sensing Time Series
ndvi
Value
Date
Cena RapidEye de 20120822

Projeto TerraClass Cerrado - Validador

Lista dos pontos dessa consulta: **3207 sendo 3207 ja avaliados** (aprox. 1421.5 min. gastos na avaliacao)

Nr	UF	Cena L8	Cl. Consultor Cl. Avaliador
1	SP	221/76	Solo_exposto Silvicultura
2	MA	220/63	Nao_observado Nao_observado
3	MA	221/62	Nao_observado Pastagem
4	MA	220/63	Nao_observado Nao_observado
5	MT	227/70	Nao_observado Nao_observado
6	MA	220/63	Nao_observado Nao_observado
7	MA	220/62	Nao_observado Nao_observado
8	MG	219/73	Solo_exposto Solo_exposto
9	MG	218/73	Nao_observado Pastagem
10	TO	222/67	Nao_observado Savanico
11	MA	221/63	Nao_observado Outros
12	MT	224/72	Nao_observado Savanico
13	MG	219/71	Nao_observado Pastagem
14	MA	220/63	Nao_observado Nao_observado
15	MG	218/73	Nao_observado Campestre
16	MA	220/63	Nao_observado Pastagem
17	MA	220/62	Nao_observado Nao_observado
18	MA	220/62	Nao_observado Nao_observado
19	BA	218/67	Nao_observado Pastagem
20	MA	220/63	Nao_observado Nao_observado
21	TO	222/67	Nao_observado Savanico

Ponto nr	16 com 7.86 ha (-4.8272,-44.6506)
Municipio/UF (WRS L8)	Poção de Pedras/MA (220/63)
Classe Consultor	**Nao_observado**
Classe Avaliador	**Pastagem**
Informacao Adicional	
Confianca	Media (apresenta confusão c/ Savanico)
Confiança da classificação	Alta
Classe:	Eventual confusão com:
Florestal Savanico Campestre Pastagem Agricultura anual Agricultura perene Silvicultura Natural nao vegetado Area urbana	Florestal Savanico Campestre Pastagem Agricultura anual Agricultura perene Silvicultura Natural nao vegetado Area urbana

Figure 2. The TerraClass cerrado validating tool.

```javascript
// connect to the WTSS server
var wtss_server = new wtss('http://www.dpi.inpe.br/wtss');

// for a given point (lat,long) return the series
wtss_server.time_series(
    {
        "coverage": "mod09q1",
        "attributes": ["red", "nir"],
        "longitude": long,
        "latitude": lat,
        "start"="2004-01-01",
        "end"="2004-05-01"
    }, fill_time_series);
```

Listing 1: The WTSS JavaScript client embbeded in the validation tool.

Figure 3 shows another web-based application that uses WTSS. In this application, again selecting a location with a click on the map, a user obtain graphical display of a MODIS/NDVI time series. But it also obtains the result of the algorithm TWDTW method for land use and land cover mapping (Maus et al., 2016) using a sequence of multi-band satellite images. This tool allows scientists to experiment the method in a single location before applying it to a given area. A **R** implementation of the TWDTW is available at `https://github.com/vwmaus/dtwSat`. Listing 2 shows a **R** code that consumes the MODIS/NDVI time series otained from a WTSS server, using a WTSS **R**-client.

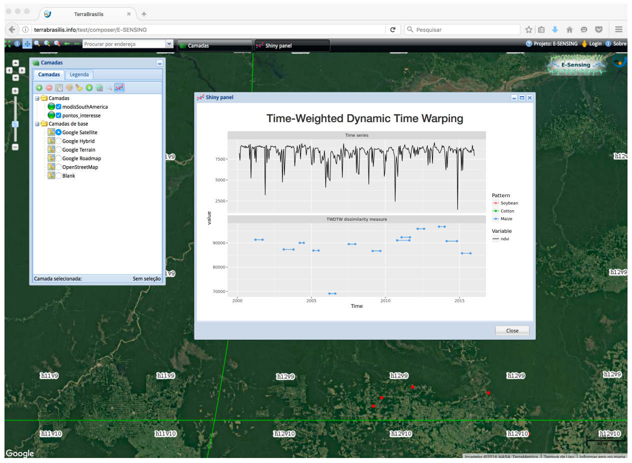

TerraBrasilis
terrabrasilis.info/test/composer/E-SENSING
Camadas
Shiny panel
Time-Weighted Dynamic Time Warping
Time series
TWDTW dissimilarity measure
Pattern
Soybean
Cotton
Maize
Variable
ndvi
Time
value
Close
Google

Figure 3. A web-based application with TWDTW algorithm.

```r
# installing and loading packages
library(dtwSat)
library(wtss.R)

# connect to the WTSS server and retrieve the time series
server = WTSS("http://www.dpi.inpe.br/tws/wtss")
coverages = listCoverages(server)
cv = describeCoverage(server, coverages[2])
attr <- c("ndvi", "evi",  "red",  "nir",  "blue", "mir" )

# get a time series
spatio_temporal = timeSeries(server, names(cv),
                  attributes=attr,
                  latitude=-10.408, longitude=-53.495,
                  start="2000-02-18", end="2016-01-01")

ts = twdtwTimeSeries(spatio_temporal[[
    names(spatio_temporal)]]$attributes)
plot(ts, type="timeseries")

patt = twdtwTimeSeries(patterns.list)

log_fun = logisticWeight(alpha=-0.1, beta=100)
matches = twdtwApply(x=ts, y=patt, weight.fun=log_fun, keep=TRUE)

plot(x = matches, type = "alignments")
```

Listing 2: The WTSS R client called in **R** implementation of the TWDTW algorithm.

5. Future Directions on Web Services for Big Data

In the previous sections, we have argued that the current design of WCS and WMS limits the use of these standards for dealing with big EO data sets. We have also presented WTSS, an example of a service designed to work with big data. In this section, we examine some future directions for web services suited for big EO data.

Given the burden of transferring large data sets over the internet, web services for big data should be concerned with performing server-side processing, rather than supporting client-side retrieval. Instead of downloading large data files, such web services would enable algorithms to be run on a remote server. This points to an emphasis on web processing services.

There are two OGC web services designed for data processing: the WPS Web Processing Service (OGC, 2015b) and the WCPS Web Coverage Processing Service (Baumann, 2010). WPS standardizes how clients can invoke the execution of a process. Thus it focus on the description of requests and reponses to inputs and outputs. It is not concerned on how to express the processing in the server side. Müller et al. (2010) build an architecture using WPS where client applications can move the code to the server in order to perform operations. However, WPS requires that the web server provides the needed operations, which is not currently the case with big EO data. Researchers are still in the stage of experimenting with new methods. Rather than providing a pre-defined set of algorithms, the server-side processing service should be flexible to allow researchers to run their own methods on big data sets.

The WCPS standard proposes a coverage processing language in an attempt to allow clients to express their computations for evaluation by the server. Although endorsed by OGC, this proposal has not been widely adopted. To the best of our knowledge there is only one implementation supporting it as a component of the RasDaMan project (Baumann et al., 1998). Furthermore, the WCPS standard is based on a sequence of pixel-by-pixel operations over a set of 2D coverages. The basic request structure of WCPS consists of a loop over a list of coverages, an optional filter predicate, and an expression indicating the desired processing of each coverage (Baumann, 2010). This way of working is not flexible enough to include algorithms that work with time-series and for spatio-temporal processing. This is illustrated by the work of Karantzalos et al. (2015), where the WCPS service was only used for data retrieval, and the classification algorithm was run in the client machine using open source libraries.

One of the challenges of designing a new language for coverage processing is the diversity of algorithms and applications already existing and the practices of the research community. Researchers are most productive when working on familiar computing environments. Scientists like to test new ideas on small and well-known data sets. Only after they are satisfied with the experiments, they are willing to work with big data. Therefore, a standard for big Earth observation data analytics should meet important needs of the research community:

1. Analytical scaling: support for the full cycle of research, allowing algorithms developed at the desktop to be run on big databases with minor or no modifications.
2. Software reuse: researchers should be able to easily adapt existing methods for big data, with minimal reworking.

3. Collaborative work: the results should be shareable with the scientific community, and different research teams should be able to build their own infrastructure.

Data scientists are conservative in their choice of tools. They prefer to work on tools with a simple software kernel where they can easily add new packages that encapsulate new analytical methods. A prime example is the **R** suite of statistical tools (R Core Team, 2015). **R** provides a wide variety of statistical and graphical tools, including spatial analysis, time-series analysis, classification, clustering, and data mining for many disciplines (e.g. hydrology, ecology, soil science, agronomy). It is extensible through high quality packages. It provides methods and tools in an open source environment and allows research reproducibility. Using **R** as their primary tool for big data analytics, researchers can thus scale up their methods, reuse previous work, and easily collaborate with their peers. Therefore, we consider that tailor-made, simpler, APIs that support **R** programming can be more effective for processing big EO data than requiring scientists to re-implement their methods on a new coverage processing language.

For future work, we envision a web service for processing big EO data as a simple and standardised interface that can be used with a few commands. It would let users encapsulate their algorithms for server-side processing. Such a service would go beyond what is currently offered by OGC standards such as WPS and WCPS to allow progress on big EO data analytics. Researchers would be able to develop and share new methods, working on a familiar **R** environment. They would be able to test their methods on their desktop before running experiments on big data sets. Such a service that would allow significant progress on big EO data analytics.

6. Conclusions

This paper addressed the problem of how to design and build technologies that allow the EO community to explore the unprecendented amounts of satellite data available for research and operational use. We investigate if the implicit decisions taken by OGC when designing services such as WMS, WCS and WFS allow them to be used for big EO data. We also consider whether new kinds of web services are required to handle big EO data efficiently.

We conclude that the current OGC standards, and their implementations, can not be considered to be the most appropriate to handle big data EO data sets since they are inspired by the traditional cartographic abstractions of maps, where each coverage is processed separately. The most innovative analytical methods that make good use of big EO data also require a conceptual view of ig EO data sets as 3D arrays of data, where each Each array position *[i, j, k]* will be associated to a set of attributes values $A = \{a_1, a_2, \ldots, a_m\}$ which are the sensor measurements at each location in space-time. This conceptual view allows the two approaches for data processing *space-first, time-later* and *time-first, space-later* shown in the paper.

We argue that tailor-made, simpler, APIs can be more effective than generic services such as WMS and WCS for accessing big EO data, specially aimed at data analytics on 3D arrays, where the temporal component is of crucial importance. As an example of such tailor-made service we describe the Web Time Series Service (WTSS). We show how WTSS closes the gap between big EO data and applications consuming time-series of data in a flexible and simple API that uses the JSON format to delivery complex responses.

This is the first step to realise a vision of a web service for processing big EO data as a simple and standardised interface that can be used with a few commands and let users encapsulate their algorithms for server-side processing.

Acknowledgements

This work is supported by FAPESP e-science program (grant 2014-08398-6). Gilberto Camara is also supported by CNPq (grant 3121512014-4).

We thank Ricardo Cartaxo de Souza for his valuable participation in the discussions about WTSS; Alber Sánchez, Luiz Fernando Assis, and Eduardo Llapa Rodríguez for their assistance with the deployment of WTSS server over SciDB and with the WTSS clients evaluation; Andre F. A. Carvalho, Luiz E. P. Maurano and Raul Castanhari for their support with the web-based applications shown in the case studies section.

References

Adami, M., Rudorff, B. F. T., Freitas, R. M., Aguiar, D. A., Sugawara, L. M., and Mello, M. P. (2012). Remote sensing time series to evaluate direct land use change of recent expanded sugarcane crop in brazil. *Sustainability*, 4(4):574–585.

Amirian, P., Basiri, A., and Winstanley, A. (2014). Evaluation of data management systems for geospatial big data. In Murgante, B. and Misra, S., editors, *14th International Conference in Computational Science and Its Applications – ICCSA 2014*, pages 678–690.

Baumann, P. (2010). The ogc web coverage processing service (wcps) standard. *GeoInformatica*, 14(4):447–479.

Baumann, P., Dehmel, A., Furtado, P., Ritsch, R., and Widmann, N. (1998). The multidimensional database system RasDaMan. *ACM SIGMOD Record*, 27(2):575–577.

Gorelick, N. (2012). Google earth engine. In *AGU Fall Meeting Abstracts*, volume 1, page 04.

Guo, W., She, B., and Zhu, X. (2014). Remote sensing image on-demand computing schema for the China ZY-3 satellite private cloud-computing platform. *Transactions in GIS*, 18:53–75.

Jensen, J. R. (2009). *Remote Sensing of the environment: na Earth resource perspective*. Prentice Hall, Chicago.

Jönsson, P. and Eklundh, L. (2004). Timesat–a program for analyzing time-series of satellite sensor data. *Computers & Geosciences*, 30(8):833 – 845.

Karantzalos, K., Bliziotis, D., and Karmas, A. (2015). A scalable geospatial web service for near real-time, high-resolution land cover mapping. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 8(10):4665–4674.

Li, S., Dragicevic, S., Castro, F. A., Sester, M., Winter, S., Coltekin, A., Pettit, C., Jiang, B., Haworth, J., Stein, A., et al. (2016). Geospatial big data handling theory and methods: A review and research challenges. *ISPRS journal of Photogrammetry and Remote Sensing*, 115:119–133.

Maus, V., Câmara, G., Cartaxo, R., Sanchez, A., Ramos, F. M., and de Queiroz, G. R. (2016). 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*, PP(99):1–11.

Müller, M., Bernard, L., and Brauner, J. (2010). Moving code in spatial data infrastructures – web service based deployment of geoprocessing algorithms. *Transactions in GIS*, 14:101–118.

OGC (2015a). OGC Observations and Measurements – JSON implementation. Technical report, Open Geospatial Consortium.

OGC (2015b). OGC WPS 2.0 Interface Standard. Technical report, Open Geospatial Consortium.

Olsson, L., Eklundh, L., and Ardö, J. (2005). A recent greening of the sahel—trends, patterns and potential causes. *journal of Arid Environments*, 63(3):556–566.

R Core Team (2015). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Sakamoto, T., Yokozawa, M., Toritani, H., Shibayama, M., Ishitsuka, N., and Ohno, H. (2005). A crop phenology detection method using time-series MODIS data. *Remote Sensing of Environment*, 96(3-4):366–374.

Stonebraker, M., Brown, P., Zhang, D., and Becla, J. (2013). Scidb: A database management system for applications with complex analytics. *Computing in Science & Engineering*, 15(3):54–62.

Verbesselt, J., Hyndman, R., Newnham, G., and Culvenor, D. (2010). Detecting trend and seasonal changes in satellite image time series. *Remote Sensing of Environment*, 114(1):106–115.

Vitolo, C., Elkhatib, Y., Reusser, D., Macleod, C. J., and Buytaert, W. (2015). Web technologies for environmental big data. *Environmental Modelling & Software*, 63:185 – 198.