CLOUD-DESKTOP REMOTE SENSING DATA MANAGEMENT TO ENSURE TIME SERIES ANALYSIS, INTEGRATION OF QGIS AND GOOGLE EARTH ENGINE

E. Panidi [1, *], I. Rykin [1], P. Kikin [2], A. Kolesnikov [3]

[1] Department of Cartography and Geoinformatics, Institute of Earth Sciences, Saint Petersburg State University, St. Petersburg, Russia - panidi@ya.ru, e.panidi@spbu.ru
[2] Peter the Great St. Petersburg Polytechnic University (SPbPU), St. Petersburg, Russia
[3] Siberian State University of Geosystems and Technologies, Novosibirsk, Russia

KEY WORDS: Google Earth Engine, QGIS, Remote Sensing Data Processing

ABSTRACT:

Our context research is conducted to investigate the possibility of common application of the remote sensing and ground-based monitoring data to detection and observation of the dynamics and change in climate and vegetation cover parameters. We applied the analysis of the annual graphs of Normalized Difference Water Index to estimate the length and time frames of growing seasons. Basing on previously gained results, we concluded that we can use the Index-based monitoring of growing season parameters as a relevant technique. We are working on automation of computations that can be applied to processing satellite imagery, computing Normalized Difference Water Index time series (in the forms of maps and annual graphs), and estimation of growing season parameters. As currently used data amounts are big (or up-to-big) geospatial data, we use the Google Earth Engine platform to process initial datasets. Our currently described experimental work incorporates investigation of the possibilities for integration of cloud computing data storage and processing with client-side data representation in universal desktop GISs. To ensure our study needs we developed a prototype of a QGIS plugin capable to run processing in GEE and represent results in QGIS.

1. INTRODUCTION

Global change in the natural environment is caused by different factors. It is needed to detect and explore dependencies between the environment components to understand and forecast change trends. Study of global change and particularly change in climate and vegetation cover involves the analysis of retrospective data collected using both ground-based observation networks and remote sensing (satellite) measurements over the past 30-40 years and more (if available).

Our research is conducted to investigate the possibilities for common application of remote sensing and ground-based monitoring data, and aimed on production of some synergy by fusion of ground-based data accuracy with spatial resolution of satellite observations when studying dynamics and change in climate and vegetation cover parameters.

Studies performed earlier by other authors (Medvedeva et al., 2008; Miklashevich, Bartalev, 2016), as well as some results of our previously conducted research (Panidi et al, 2017; 2018; 2019), demonstrate a feasibility of integrating approach. This integration assumes investigation of change in the climate-vegetation system instead of separated analysis of climate and(or) vegetation cover. The approach is based upon the idea of indirect exploration when the analysis of change detected in one of the system components in a number of cases allows to detect, confirm or clarify the parameters of change occurred in another component.

Particularly, our previous work was devoted to consideration of the applicability of vegetation indices derived from satellite imagery to tracking the dynamics of vegetation cover, and to indirect assessment of the climate dynamics. Remote sensing is a well-recognized tool for vegetation monitoring, especially in a case of high spatial and temporal resolution of collected imagery. In the regions where the ground network of meteorological stations is sparse, vegetation monitoring can help to detail and interpolate spatially the estimations of climate parameters change, as the climate change is one of most significant factors of the vegetation cover change.

Accordingly to our previously conducted studies (Panidi et al, 2017; 2018; 2019) and to a number of the studies of other authors (Delbart et al, 2005; Sekhon et al, 2010) the Normalized Difference Water Index (NDWI) is a relevant instrument for growing seasons monitoring, while growing season length, onset and ending reflect change in surface air temperature. Analysis of the NDWI annual graphs can be applied to determine growing season parameters (Panidi, Tsepelev, 2017). Full growing season can be separated by detection spring and autumn minimums on the graph, while summer growing season can be detected as a plateau at the central part of full growing season segment of NDWI graph (Fig. 1).

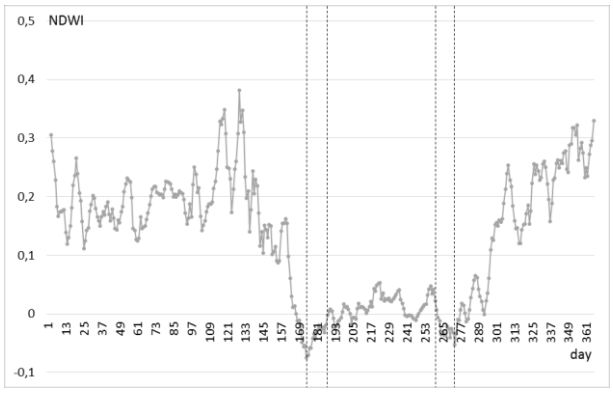

Figure 1. Example of 1-day temporal resolution annual graph of NDWI; Dashed vertical lines separates (from left to right) spring, summer and autumn growing seasons

* Corresponding author

We applied analysis of the annual NDWI graphs to estimate the length and time frames of growing seasons. Additionally, involvement of vegetation indices (particularly NDWI) makes it possible to detect regional-scale differentiation (i.e., spatial heterogeneity) of growing season characteristics, while the ground observation network can be sparse and hardly applicable to solve this problem.

2. APPROACHES

At the different stages of the study we used for analysis 8-day MOD09A1 (Vermote, 2015) and 1-day MOD09GA (Vermote, Wolfe, 2015) Moderate Resolution Imaging Spectroradiometer (MODIS) composites. 1-day resolution datasets potentially give a possibility to ensure monitoring of vegetation cover dynamics and indirect estimation of climate (weather) parameters change with a ground observations discreetness. However, increasing of the temporal resolution to 1 day brought us to hardness in local processing of the data due to its close-to-big-data volume.

We attracted Google Earth Engine (GEE) platform to process big satellite data we use. GEE is a public cloud-based computing and storage platform that gives access to multiple global time series of satellite imagery and imagery processing algorithms through a Web browser (Gorelick et al, 2017). Users may process data using algorithms (program code) already published on the GEE Web site, or using a program code newly developed by themselves. In general, a code can be developed in JavaScript language directly in the code editor embedded into GEE Web site.

Originally, use of the GEE is not available from interfaces of desktop Geographic Information Systems (GISs). So direct combining of GEE scripting[1] in a browser with powerful toolsets of desktop GISs is not possible. However, Python Application Programming Interface[2] (API) of the GEE makes it possible to develop custom (external) applications, whether stand along or built in desktop GIS. Our currently described study is focussed onto investigation of the integration possibilities for the GEE cloud computing facilities with data harvesting and post-processing made directly at a client side in a desktop GIS. We use QGIS as it is open source and tool-rich desktop GIS.

We are developing a prototype of a QGIS plugin in Python language that is used for generating GEE processing requests, defining the results of processing, and representing defined results in QGIS in a form convenient for further visual and quantitative analysis.

3. STUDY AREA

The area of interest in our research is the Komi Republic that is one of the Russian regions located in the northeast of European Russia. Selection of this area is based upon its location within three natural zones (i.e., taiga, forest-tundra, tundra) and altitudinal zonality in Ural mountains, diversity of climate-forming factors, long-term observation series, and sparseness of ground-based observation network (Fig. 2, 3). There are 10 meteorological stations per 415,900 sq. km. Incorporation of the northern Ural mountains into study area brings an opportunity to discover a barrier effect of the mountain ridge, as most of the territory of Komi Republic is located to the west of the ridge and appears to be located in a zone of western air mass transfer.

Significant feature of the study area is its location in a region with a large number of cloud days. It bring to the presence of multiple and sometimes long gaps in collected and processed data and appears not good for study process, but having methodological interest (as spatial and time gap-filling data interpolation techniques have to be developed, tested and applied in this area).

Figure 2. Location of the study area; Komi Republic is marked by red contour; Satellite imagery courtesy of the ESRI ArcGIS Online

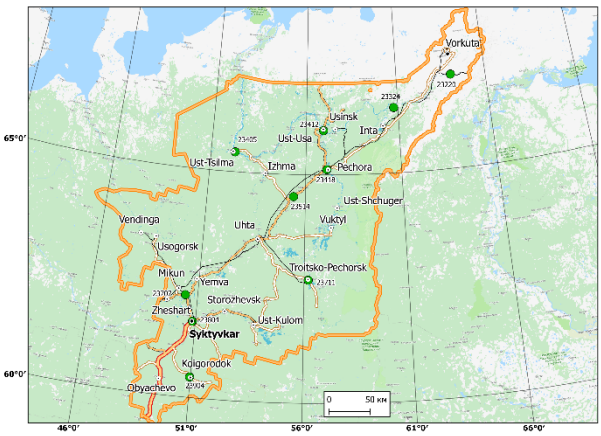

Figure 3. Meteorological stations (green points) of the Komi Republic

4. DATA

Implementation of data processing with a higher temporal resolution has led to the need of cloud-based storing and processing of remote sensing data instead of the traditional way (storing and processing on a desktop computer). Such a re-engineering of data processing chains is a result of higher data amounts needed to be reworked in comparison to the 8-day temporal resolution case.

1-day MOD09GA product is distributed in a sinusoidal map projection with a spatial resolution of ~500 m/pix. Global coverage is divided into tiles 2400 by 2400 pixels each (i.e., 1200 by 1200 kilometers approximately). Thus, initial imagery volume of approximately ~500 GB have to be downloaded and preprocessed to form the time series from 2000 to the present (for the maximum available period), if the study area is located within one tile. If the study area crosses borders of two or more tiles, the data volume is multiplied accordingly.

[1] *https://code.earthengine.google.com*

[2] *https://github.com/google/earthengine-api/tree/master/python*

NDWI gridded maps are computed in our study using MOD09A1 and MOD09GA datasets for the territory of Komi Republic. The first one was used mostly at previous stages of the study, while currently we investigate 1-day computations based on the second mentioned dataset. Computations are made accordingly to the Gao's formula (Gao, 1996) for the period of 20 years since 2000:

$$NDWI = \frac{\rho_{0.86} - \rho_{1.24}}{\rho_{0.86} + \rho_{1.24}}$$

where $\rho_{0.86}$ = near-infrared reflectance
$\rho_{1.24}$ = short wave infrared reflectance

5. METHODS

A number of authors conclude effectiveness of GEE implementation when processing satellite imagery time series (Warren et al., 2015; Agapiou, 2017; Vos et al., 2019). In our study, server-side (in-the-cloud) data storage and preprocessing, allow to increase significantly the speed and efficiency of computations, through elimination of full initial data sets downloading to the client computer.

Significant feature of the data processing in our case is that significant part of data is not used directly in computations. Image channels other than two needed to produce NDWI gridded map incorporated into original dataset is a first exapple of these not-needed data. Second example is the image data located outside of the study area (area of interest), in which computations are performed. Due to the number of channels in initial dataset (that is 22 including data quality and technical data channels[3]), and to the geometry of Komi Republic administrative boundaries, more than 90 % of initial data amount remains wasted in the meaning of processing needs. In the meaning of downloaded data amount it is ~450 GB per tile of wasted data traffic.

Data processing in GEE is operated through the GEE Code Editor (Web-based Integrated Development Environment – IDE). GIS applications, can use client libraries (JavaScript API, Python API) for pushing queries through the Web (Gorelick et al., 2017). Our experimental work with the GEE has shown that the standard (recommended[4] by Google) operating regime (through the Code Editor) is not enough flexible. It assumes data processing through writing an appropriate program code in Code Editor, directly on GEE website. This manner of the code development can make debugging little bit difficult, while some computations may be performed in a black-box-like regime, and processed data is hardly controlled on the pixel scale (for example, when it is needed to control which pixels are used to average a value in a certain neighbourhood). This feature is significant as the MOD09GA dataset is stored in Sinusoidal projection, while data in Code Editor is visualized in Web Mercator Projection and operated in geographical coordinate space. Data projecting in such a case remains our or our control.

In addition, in-the-cloud set of ready-to-use analytical, computational, processing, and visualization tools is significantly narrower than in the case of operating with universal desktop GISs, such as QGIS and ArcGIS. So when processing in GEE Code Editor we need to develop almost all processing logic from scratch.

When prototyping we used the GEE plugin[5] for QGIS. It is developed since 2017[6]. The plugin was placed in the official QGIS plugin repository only in the December of 2019[7], thus its functionality still remains basic only but enough to support requests and responses when interchanging data between the GEE and desktop using GEE Python API.

Python API use involves deploying the API library on a user's computer. In our study, we used Python also to build extension (plugin prototype) for QGIS over the GEE plugin and to provide graphical user interface as an instrument of formation of the Web requests to GEE.

In the context of our prototype, GEE plugin plays infrastructure role. It is involved to solve the issue of GEE Python API library deployment, and to provide user authentication when connecting with GEE server. After passing authentication in GEE via the GEE plugin we can run our own code to perform computations on the GEE side using the Python API. It is needed also to connect corresponding library when developing data processing code in the Python console (Fig. 4).

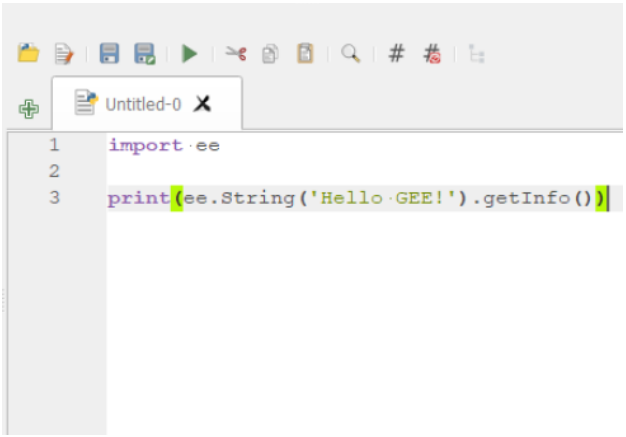

Figure 4. Initialisation of GEE connection in the Python console

The output generated by GEE is loaded in QGIS as an RGB-image by default. After this, it is and visualized through the functionality of developed plugin (Fig. 5). Currently available functionality of GEE Python API and (correspondingly) developed plugin prototype remains not enough to operate with interchanged data in the scientific form (form of meaning values, acceptable for computations).

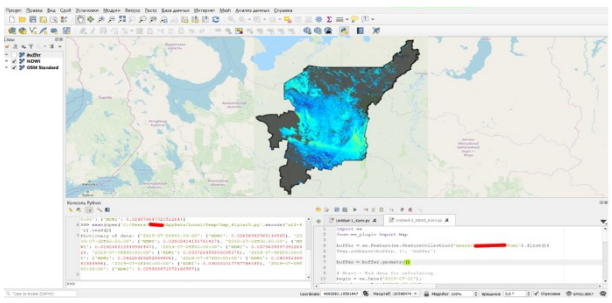

Figure 5. NDWI map for the Komi Republic (March 1, 2000) loaded directly into QGIS; clouds and water are masked

Scientific data (NDWI values in our case) can be received currently through the console writing/reading only. However, it makes possible to compile annual graphs for selected point locations or areas (this is ongoing work). Next aim is the

[3] *https://lpdaac.usgs.gov/products/mod09gav006/*
[4] *https://developers.google.com/earth-engine/python_install_manual#coding-in-the-python-api*
[5] *https://gee-community.github.io/qgis-earthengine-plugin/*
[6] *https://github.com/gee-community/qgis-earthengine-plugin/graphs/code-frequency*
[7] *https://github.com/gee-community/qgis-earthengine-plugin/releases*

extension of the developed prototype with scientific data formation (writing) functionality, to store loaded values on the client computer as the scientific rasters.
For testing (benchmarking) purposes, we developed also equivalent NDWI-computing script in JavaScript language directly in GEE's Code Editor. The script can be used to compute NDWI maps, to apply water and cloud masks to the maps, and to cut the maps with the area of interest (point, buffer zone, of area object – Fig. 6). Series of at-the-point or area-averaged NDWI values can be derived from computed and cut maps and printed in JavaScript console.

Figure 6. NDWI gridded map masked by water and cloud masks and cut with buffer zone in the GEE Code Editor

This script has the functionality that is almost equal (at the current stage) to the developed plugin prototype. It helps to control results gained by desktop application through the GEE Python API, while we face a lack of guiding information on Python API usage.

6. RESULTS AND DISCUSSION

Currently available functionality of GEE plugin for QGIS and our built-over plugin involves data visualization in QGIS in the form of rendered (in accordance with a certain colour palette) raster that is not scientific dataset and not suitable for further automated analysis. The dataset stores colours of the pixels, but not the NDWI values. However, such data can be used for visual analysis, while scientific data can be uploaded in parallel as an array of needed values. In this way, some additional work is needed to assimilate this array in QGIS. At current stage, we do this manually.
Our currently described experimental work incorporates investigation of the possibilities for integration of cloud computing data storage and processing with client-side data representation in universal desktop GIS. To ensure needs of our context study we developed a prototype of a QGIS plugin that is used to generate GEE processing requests, harvest processing results, and represent harvested results in a form convenient for further visual and quantitative analysis in QGIS.
We discovered also some additional problems of the GEE Python API applicability:

1. Poor documenting and lack of training materials
2. The need of setting up of the environment (related program libraries) for the GEE Python API, while it is difficult to restore the program environment equal to that was used by API developers (and the developers themselves recognize this)
3. Complexity and a number of errors in the GEE Python API, in fact not all the functions are available for use

Nevertheless, GEE Python API demonstrates good result and great potential as a GEE-QGIS interactor. In our experiments time economy for data processing and representation in a needed editable form on the client computer, appears to be transformed from days (needed for download gigabytes of data) to seconds (needed to load and draw map or graph for selected territory).
Obviously, a deeper elaboration of data operation functionality is needed to satisfy requirements of our project that is a future work. Additional work have to be performed to form understanding of graphical user interface model applicable to replace program initialisation of GEE connection and processing requests formation.

ACKNOWLEDGEMENTS

The MOD09GA Version 6 product was retrieved through the Google Earth Engine platform: courtesy of the NASA EOSDIS Land Processes Distributed Active Archive Center (LP DAAC), USGS/Earth Resources Observation and Science (EROS) Center (*https://lpdaac.usgs.gov/products/mod09gav006/*).

Ground observation data were retrieved from Waisori Web interface, courtesy of the RIHMI-WDC of Roshydromet, Veselov V.M., Pribylskaya I.R., Mirzeabasov O.A. (*http://aisori-m.meteo.ru/waisori/*).

Google Earth Engine platform: courtesy of Google™ (*https://earthengine.google.com/platform/*).

REFERENCES

Agapiou, A., 2017: Remote sensing heritage in a petabyte-scale: satellite data and heritage Earth Engine© applications. *International Journal of Digital Earth,* 10(1), 85-102. doi:10.1080/17538947.2016.1250829

Delbart, N. J-P., Kergoats, L., Le Toan, T., Lhermitte, J., Picard, G., 2005: Determination of Phenological Dates in Boreal Regions Using Normalized Difference Water Index. *Remote Sens. of Env.*, 97(1), 26-38. doi:10.1016/j.rse.2005.03.011

Gao, B.C., 1996: NDWI – A Normalized Difference Water Index for Remote Sensing of Vegetation Liquid Water From Space. *Remote Sens. of Env.,* 58 (3), 257-266.

Gorelick, N., Hancher, M., Dixon, M., Ilyushchenko, S., Thau, D., Moore, R., 2017: Google Earth Engine: Planetary-scale geospatial analysis for everyone. *Remote Sens. of Env.,* 202, 18-27. doi:10.1016/j.rse.2017.06.031

Medvedeva, M.A., Bartalev, S.A., Lupyan, E.A., Matveev, A.M., Tolpin, V.A., Poida, A.A., 2008: The Possibility of Estimation of the Growing Season Onset Basing on Satellite and Meteorological Data. *Sovremennye Problemy Distantsionnogo Zondirovaniya Zemli iz Kosmosa [Contemp. Probl. of Remote Sens. of the Earth from Space]*, 5(2), 313-321. (in Russian)

Miklashevich, T.S., Bartalev, S.A., 2016: Method for estimating vegetation cover phenological characteristics. *Sovremennye Problemy Distantsionnogo Zondirovaniya Zemli iz Kosmosa [Contemp. Probl. of Remote Sens. of the Earth from Space]*, 13(1), 9-24. doi:10.21046/2070-7401-2016-13-1-9-24 (in Russian)

Panidi, E., Rykin, I., Nico, G., Tsepelev, V., 2019: Toward Satellite-based Estimation of Growing Season Framing Dates in Conditions of Unstable Weather. Adv. in Remote Sens. and Geo Inf. App., Adv. in Sci., Tech. & Innov., Springer, 131-133. doi:10.1007/978-3-030-01440-7_31

Panidi, E.A., Rykin, I.S., Tsepelev, V.Yu., 2018: Towards the issue of allocation of the time frames for growing seasons using ground observations and remote sensing data. *Proceedings of the International conference "InterCarto/InterGIS"*. 24(2), 129-140. doi.org/10.24057/2414-9179-2018-2-24-129-140 (In Russian)

Panidi, E., Tsepelev, V., 2017: NDWI-based Technique for Detection of Change Dates of the Growing Seasons in Russian Subarctic. *Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci.* XLII-3/W2, 179-182. doi:10.5194/isprs-archives-XLII-3-W2-179-2017

Sekhon, N.S., Hassan, Q.K., Sleep, R.W., 2010: A Remote Sensing Based System to Predict Early Spring Phenology Over Boreal Forest. *Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci*., XXXVIII(1), 5 p.

Vermote, E., 2015: MOD09A1 MODIS/Terra Surface Reflectance 8-Day L3 Global 500m SIN Grid V006. NASA EOSDIS Land Processes DAAC [Data set]. doi:10.5067/MODIS/MOD09A1.006

Vermote, E., Wolfe, R., 2015: MOD09GA MODIS/Terra Surface Reflectance Daily L2G Global 1kmand 500m SIN Grid V006 [Data set]. doi:10.5067/MODIS/MOD09GA.006

Vos, K., Splinter, K.D., Harley, M.D., Simmons, J.A., Turner, I.L., 2019: CoastSat: a Google Earth Engine-enabled Python toolkit to extract shorelines from publicly available satellite imagery. *Environmental Modelling and Software*, 122, 104528. doi:10.1016/j.envsoft.2019.104528

Warren, M.S., Brumby, S.P., Skillman, S.W., Kelton, T., Wohlberg, B., Mathis, M., Chartrand, R., Keisler, R., Johnson, M., 2015: Seeing the Earth in the Cloud: processing one petabyte of satellite imagery in one day. IEEE Applied Imagery Pattern Recognition Workshop, AIPR 2015, 7444536. doi:10.1109/AIPR.2015.7444536