OMEGA-PY: PYTHON TOOLS FOR OMEGA DATA – v2.3.

A. Stcherbinine[1], J. Carter[2,3], Y. Langevin[2], M. Vincendon[2], Y. Leseigneur[2], O. Barraud[4] and J-P. Bibring[2].
[1]Department of Astronomy and Planetary Science, Northern Arizona University, Flagstaff, AZ, USA (aurelien.stcherbinine@nau.edu), [2]IAS, Université Paris-Saclay, CNRS, Orsay, France, [3]LAM, Université Aix-Marseille, CNRS, CNES, Marseille, France, [4]German Aerospace Center (DLR), Institute of Planetary Research, Berlin, Germany.

Introduction: *OMEGA-Py* [1] is a Python 3 module dedicated to the scientific use of data provided by the Observatoire pour la Minéralogie, l'Eau, les Glaces et l'Activité (OMEGA) instrument onboard the ESA Mars Express (MEx) orbiter [2]. It has been developed as an alternative to the historical *SOFT 10* IDL routines of the official OMEGA software provided by the instrument team (ftp://psa.esac.esa.int/pub/mirror/MARS-EXPRESS/OMEGA/MEX-M-OMEGA-2-EDR-FLIGHT-EXT7-V1.0/SOFTWARE/). The module notably includes a re-implementation of the most recent release of the IDL OMEGA software, but also contains several additional data reduction functions such as build-in atmospheric and thermal corrections (using previously published methods) and graphics tools including interactive visualization of the data or generation of composite OMEGA maps.

Figure 1 *– The OMEGA-Py logo.*

Data handling: Similarly to the *readomega.pro* routine of the SOFT 10, OMEGA-Py read the binary .QUB & .NAV files that can be downloaded from the ESA PSA to generate level 1B data with reflectance spectra (Note: OMEGA-Py assumes filenames with uppercase letters and not lowercase, so pay attention if you download the data from the PDS instead of the PSA). Unlike the IDL software, the module has been implemented with Object Oriented Programming (OOP), which makes easier the simultaneous handling of different OMEGA observations. The loading of an OMEGA observation is done by calling the `__init__` method of the `OMEGAdata` class (e.g., `omega = OMEGAdata("0967_3")` to load the cube 0967_3). The user can then access all the data (reflectance, geometry, wavelengths…) from the attributes of the newly created `omega` object.

The wavelengths are also automatically reordered in ascending order and the spectra are "cleaned" to remove the overlaps between the three channels of the instrument (V, C, L) along with the corrupted spectels as identified in the `ic` array of `readomega`.

It is also possible to save/load directly `OMEGAdata` objects using dedicated functions of the module in order to save computation time by avoiding recomputing every time the importation and/or correction steps. The `OMEGAdata` objects keep a record of the atmospheric/thermal corrections applied to them.

Data correction: In addition to the raw data importation, which is done similarly to the IDL `readomega`, OMEGA-Py also includes build-in functions to perform the atmospheric and thermal corrections on a previously loaded `OMEGAdata` object, using methods described in [3, 4].

Thermal correction. The thermal correction can be performed using two methods. The preferred one assumes the theoretical reflectance at 5 µm from the one at 2.4 µm and typical spectra [3], but a second one has been implemented for cubes with no data from the C-channel [5]. In addition, to reduce the computation time of the thermal correction, parallel processing has been implemented using the *multiprocessing* module for the first method. For instance, one can perform the thermal correction of a data cube using 15 simultaneous processes with `omega_corr = corr_therm(omega, npool=15)`.

Atmospheric correction. The atmospheric correction of a data cube is performed using the volcano-scan technique, by scaling a typical atmospheric spectrum from the 2 µm CO_2 atmospheric band [4]. Upcoming versions will allow the user to use

a custom atmospheric spectrum instead of the one currently provided. The atmospheric correction is performed by using `omega_corr = corr_atm(omega)`.

Note that if both corrections are needed, one can simply call the `corr_therm_atm()` function that will perform both corrections: `omega_corr = corr_therm_atm(omega, npool=15)`.

Visualization: OMEGA-Py also comes with a series of visualization functions specifically developed. All of them are packed in `omegapy.omega_plots`, the user can choose to project the data on a lat/lon grid or a polar view, or display the image without projection.

Interactive display. One of the very useful features for the exploration of OMEGA data is the interactive display, similar to what can be done with ENVI for IDL users. By clicking with the mouse on the OMEGA image (reflectance or derived map), the user can explore and extract the spectra from every pixel of the map using the `show_omega_interactif_v2()` function (see figure 2).

Composite maps. OMEGA-Py provides functions to generate composite maps from multiple OMEGA observations, with either the reflectance or previously processed high-level maps (such as band depth). For instance, the maps in figures 3 to 5 of [6] have been generated using OMEGA-Py functions (`show_omega_list_v2()`).

Conclusion and perspectives: The OMEGA-Py module is a new powerful set of tools developed to facilitate the scientific exploitation of OMEGA observations. The Python implementation and the built-in correction and visualization functions will make this amazing dataset more accessible, especially to the younger generation of planetary scientists that are more used to this language than IDL.

Even if the software is not yet "official", we are currently conducting a validation process with the instrument team to be able to provide it as an official alternative to the SOFT 10 IDL software in a near future, which will be released as version 3.

Acknowledgments: The OMEGA/MEx data are freely available on the ESA PSA at https://archives.esac.esa.int/psa/#!Table%20View/OMEGA=instrument. The source code of the OMEGA-Py Python module is freely available on GitHub at https://github.com/AStcherbinine/omegapy.

References: [1] Stcherbinine, A (2023) *Zenodo*, https://doi.org/10.5281/zenodo.7818829. [2] Bibring, J-P. et al. (2004) *ESA Publication Division*, *1240*, 37-49. [3] Jouglet, D. et al. (2007) *JGR*, *112*, E08S06. [4] Langevin, Y et al. (2005) *Science*, *307*, 1584-1586. [5] Audouard, J (2014) PhD thesis, Université Paris XI. [6] Stcherbinine, A. et al. (2021) *Icarus*, *369*, 114627.

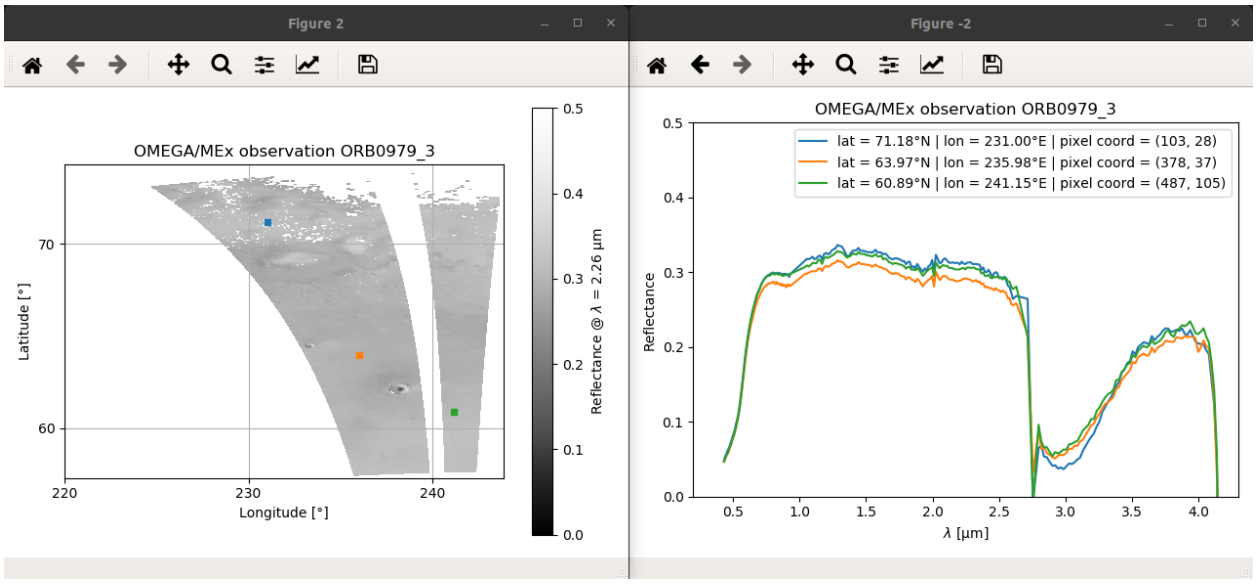

Figure 2 *– Interactive visualization of an OMEGA observation with* `show_omega_interactif_v2()`.