Archive with instructions for use

To run the codes it will be necessary to install Python or Anaconda software (windows or Linux) with the Jupyter Notebook editor for files of type: .ipynb or another editor for .py files. The execution of the .py files must be via terminal (if the editor does not have execution support).

Step 1:

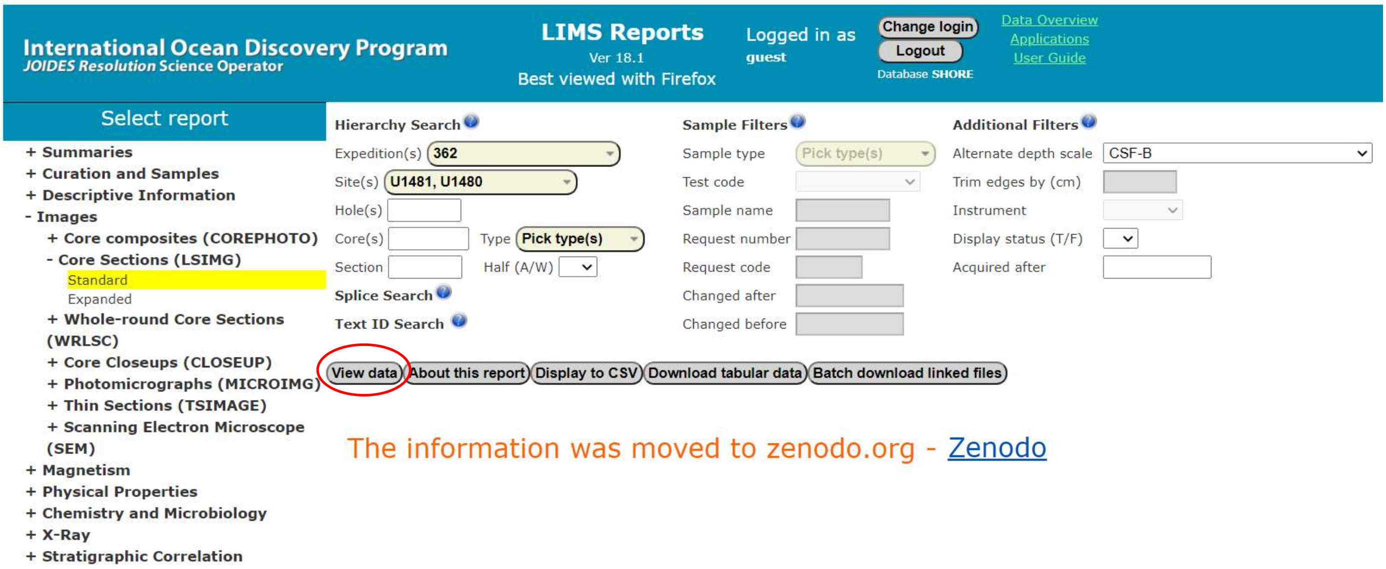
- Create the following directory structure:

```
/LithoPy-image:
      /IODP_images
             /U1480
            /U1481
      /annotated_images
            /U1480
            /U1481
      /json
     /geo_prop_original
             /U1480
            /U1481
     /geo_prop_interp
             /U1480
                  /dataset1
                 /dataset2
            /U1481
                 /dataset1
                /dataset2
     /python_codes
     /logs
           /dataset0
           /dataset1
                 /U1480
                /U1481
           /dataset2
                 /U1480
                /U1481
    /results
          /graphics
               /dataset0
              /dataset1
```

```
        /dataset2
    /datasets_RF
        /dataset0
        /dataset1
            /U1480
                /group1
                /group2
            /U1481
                /group1
                /group2
        /dataset2
            /U1480
                /group1
                /group2
            /U1481
                /group1
                /group2
```

Note: Changes in the directories adjust the configuration file: config.py.

- Download the images on the website: http://web.iodp.tamu.edu/LORE/ and select the menus as shown in the image below:



Expedition(s) 362

Site(s) U1481, U1480

Type Pick type(s)

Alternate depth scale CSF-B

The information was moved to zenodo.org - Zenodo

- You will be directed to Zenodo

- Download according to the selected site

- Extract the images from the site U1480 all in the same directory: /IODP_images/U1480

- Extract the images from the site U1481 all in the same directory: /IODP_images/U1481

-

Step 2:

- Copy the files from the json directory in supplementary material to the directory: /LithoPy-image/json

Step 3:

- Copy the files from the Python code directory in supplementary material to the directory: /LithoPy-image/python_codes

Step 4:

- Download the geophysical properties file by website at the link: http://web.iodp.tamu.edu/LORE/, Physical Properties menu, according to the properties used in the article (GRA, MAD, MS, NGR, PWL, RGB and RSC). For the site U1480 save in /geo_prop_original/U1480. For the site U1481 save int /geo_prop_original/U1481.

Step 5:

- Copy the file practical_arrangement_to-dataset2.csv to /LithoPy-image/python_codes. The file can be found in the supplementary material.

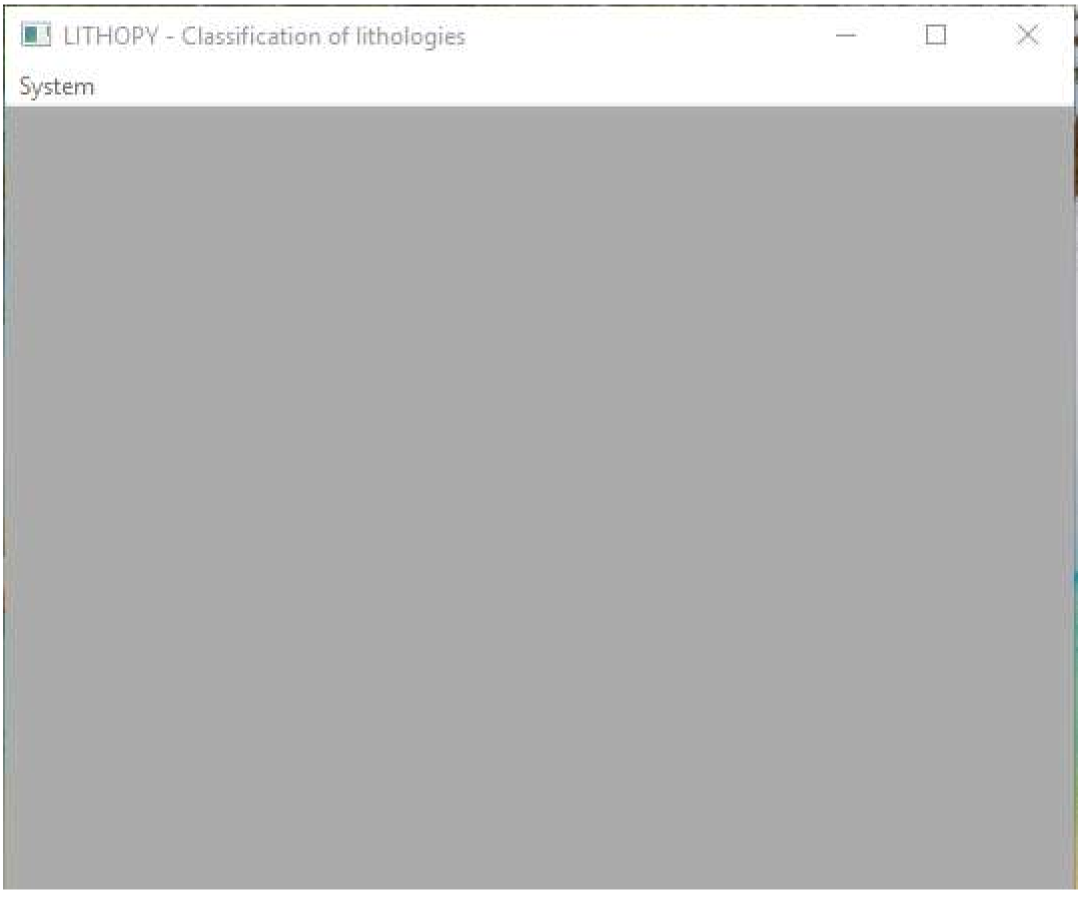
RUN CODES

There are two ways to run the codes: Graphical User Interface (GUI) or Terminal

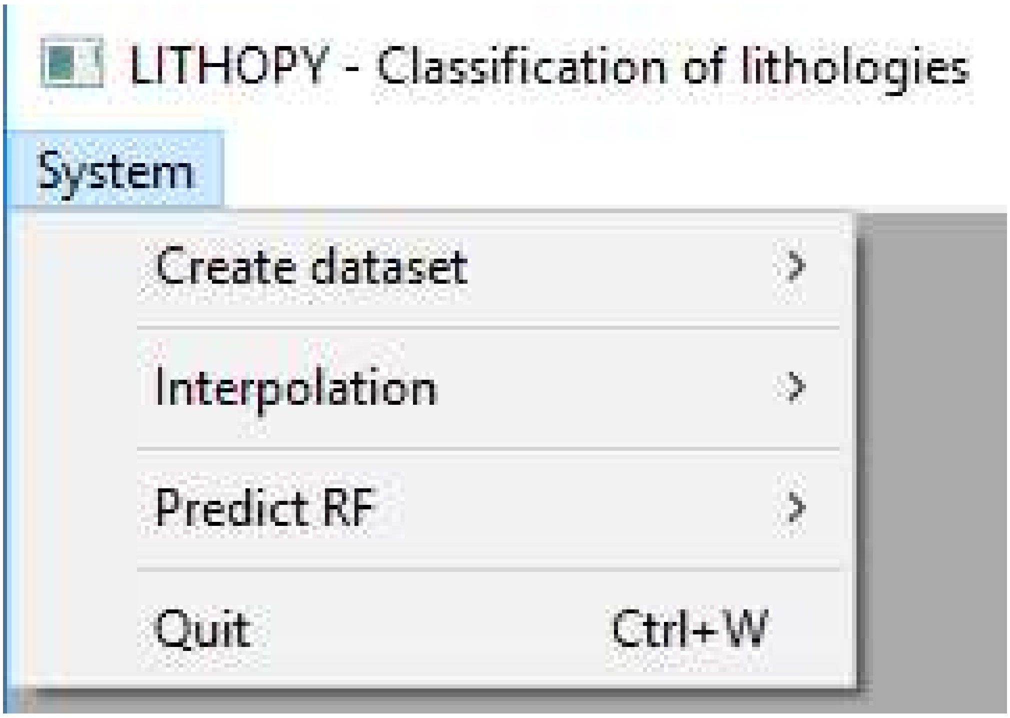
Run codes in the GUI:

- Open the terminal (command prompt), locate the directory with the python files (.py) and execute:

```
python lithopy_v.1.py
```



- In the System menu there is access to the application



- The execution sequence:

1. System / Create dataset / Start 1
2. System / Interpolation / Start
3. System / Create dataset / Start 2
4. System / Create dataset / Start 3
5. System / Predict RF / Start

Note:

- Only step 5 can be performed when the datasets already exist. In this case, the directories must be checked according to Step 1 (Create the following directory structure).

- Execute the codes via GUI, you should check the settings directly in the .py code.

Run codes on the terminal (in this order):

- cut_image_poly_create_csv_U1480.py

- cut_image_poly_create_csv_U1481.py

Creates the images annotated by lithology.

Creates .csv file with all the original geophysical properties per lithology interval and segmented region – json.

Example:

362-u1480h-1h-1-a;region-0;Calcareous-clay_claystone.csv

- interpolation_dataset0.py

- interpolation_dataset1_U1480.py

- interpolation_dataset1_U1481.py

Interpolates the values according to defined settings.

To dataset0

creates the following files, example:

- 362_U1480_E_1H_interp_temp.csv (file temporary without interpolation, with empty fields), one for each hole.

- full_interp_csv_362_U1480_E_1H_xxx.csv (one for each hole with interpolated values according to interpolator)

To dataset1 it follows the same standard, only using Akima interpolator.

- join_interpolation_dataset0.py

- join_interpolation_dataset1_U1480.py

- join_interpolation_dataset1_U1481.py

Groups interpolated files into a single file.

File example:

- linear_full_362_U1480_E.csv (combining interpolator with site, hole, core)

To dataset1 it uses the Akima interpolator.

- create_dataset2_U1480.py

- create_dataset2_U1481.py

Create dataset2 complete for RF, separated by groups and lithology.

- create_datasets.py

Create dataset0, dataset1 and dataset2 complete for RF, separated by groups. Save files in the directory datasets_RF used by RF.

- RF_classification_dataset0.py

- RF_classification_dataset1.py

- RF_classification_dataset2.py

Run the algorithm using the RF method with its settings for each dataset. The files are found in the directory datasets_RF according site and group

To dataset0

creates the following files, example to interpolator piecewise polynomial:

- result_RF_predict_piecewise_polynomial_dataset0.csv columns used by RF + cod_lit column + cod_lit_test column, per interpolator

- join_RF_predict_piecewise_polynomial_dataset0.csv

All dataset0 + cod_lit column + cod_lit_test column, per interpolator

To dataset1

creates the following files, example: site: U1480, group:1, test: 0.1 (train:0.9), number of trees (50), maximum of trees (2000):

- result_RF1_U1480_group1_0.1_50.csv

RF output with accuracy values, number of trees, maximum number of trees and cross-validation for each site and group

- result_RF_predict_group1_0.1_50_2000.csv

columns used by RF + cod_lit column + cod_lit_test column

- join_RF_predict_test_group1_0.1_50_2000.csv

All dataset1 + cod_lit column + cod_lit_test column

To dataset2

creates the following files, example: group:1, ID: 1, test: 0.1 (train:0.9), number of trees (10), maximum of trees (50):

- result_RF_texture_acc__1_1_10_0.1.csv

RF output with accuracy values, number of trees, maximum number of trees, group and cross-validation for each ID (practical arrangement).

- result_RF_texture_1_1_50_10_0.1.csv

columns used by RF + cod_lit_test column per ID (practical arrangement).

- result_RF_predict_1_1_50_10_0.1.csv

columns used by RF + cod_lit column + cod_lit_test column per ID (practical arrangement).

- join_RF_texture_test_1_1_50_10_0.1.csv

All dataset2 + cod_lit column + cod_lit_test column

- view_segments_test_graphics.py

Perform the classification of the segmented region as described in the article (Fig. 30 and table 8, 9). Adjust in the code (line 50) the link to the file extracted from RF_classification_dataset2.py.

Other settings:

- To generate the ROC curve, use a file starting with: result_RF_predict *. *

- File requirements.txt contains the python libraries used.

- The supplementary material already has the datasets_RF directory with the datasets created. Ready datasets can be used. Copy to the local datasets_RF folder and extract the files that are compressed. After running the files: RF_classification_dataset0.py, RF_classification_dataset1.py or RF_classification_dataset2.py