ERUO
User Manual

Version 1.0, date: 29/11/2021

User manual written by: Alfonso Ferrone

Based on the ERUO algorithm [1]*, developed by:*

Alfonso Ferrone, Anne-Claire Billault-Roux and Alexis Berne

1. Introduction

This manual contains instruction on how to process (and postprocess) the data collected by the MRR-PRO using the ERUO library. Few example measurements accompany the library, and the user manual uses these data files to illustrate the functioning of the library.

The ERUO library is presented in [1] and its code is available at:
https://github.com/alfonso-ferrone/ERUO.git
All the content of this repository needs to be downloaded in order to follow all the steps presented in this manual.

1.1 Software requirements

All the code in the ERUO library has been written in Python 3.8, and we expect it to function correctly for different Python 3.X versions.

In order to use the library, the user needs to have access to Python 3.X with the following packages installed:

- numpy
- scipy
- xarray
- astropy
- netCDF4
- joblib
- matplotlib

The library has been tested on Linux (Ubuntu 20.04.3 LTS) with the following packages versions:

Name	numpy	scipy	xarray	netCDF4	astropy	joblib	matplotlib
Version	1.19.2	1.5.2	0.17.0	1.5.3	4.0.2	0.17.0	3.3.2

ERUO has also been tested on Windows (Windows 10 Pro, Version 20H2) with the following packages versions:

Name	numpy	scipy	xarray	netCDF4	astropy	joblib	matplotlib
Version	1.21.2	1.2.1	0.19.0	1.5.7	3.0.5	1.0.1	3.4.3

We expect the library to function correctly with different version numbers of the libraries.

1.2 Notes on the measurement setup

This guide uses the data provided in the ERUO GitHub repository to illustrate the functioning of the library. These measurements have been collected in the vicinity of Princess Elisabeth Antarctica by an MRR-PRO[1] which was saving the "spectrum_raw" in the data files.

The library is designed to use this "spectrum_raw" as starting point of the processing, and it will not function correctly if used on a dataset collected by an MRR-PRO that has been configured to collect "spectrum_reflectivity" instead of "spectrum_raw".

Therefore, if the user plans to use ERUO on a dataset, they must ensure that in the configuration page of the MRR-PRO the spectrum recording has been set to "spectrum_raw".

2. How to use ERUO

This section explains the basic functioning of the library, using the default configuration for the processing.

All the instructions available in this sections have been also detailed in the following video-tutorial: https://youtu.be/rUTj2xojSqo

2.1 Organizing the directories for data and products

Download the ERUO library the GitHub repository. In case the user decides to download it as ".zip" file, it should be extracted in a location of the user's choice on their computer.

Inside the "examples" folder, create 5 sub-folders:

- "Archives", to store the output of the pre-processing script,
- "Proc_files", to store the processed files,
- "Postproc_files", to store the postprocessed files,
- "Quickplots_proc", to store the plots of the processed files,
- "Quickplots_postproc", to store the plots of the postprocessed files.

These names are just suggestions, and each folder can be named differently if the user prefers so.

In this guide we will produce plots for a subset of the variables (quickplots thereafter) for both the processed and postprocessed files. When processing other datasets, the user may decide to generate quickplots for just one of the two series of files (or neither), and in that case the creation of the respective folder can be omitted.

1 More information on the dataset, the measurement campaign and the configuration of the MRR-PRO are available in [1].

2.2 Indicating the folder paths in the configuration files

Once the folder for all the ERUO products have been created, open the “config.ini” file (located inside the ERUO folder) with a text editor of the user’s choice.

For an easier visualization of the structure of this file, we suggest to set the syntax highlighting to “Python”, so that the lines starting with the character “#” will be correctly recognized as comments.

The first section of this file is named [PATHS]. For the basic processing, this is the only section that needs to be modified. In particular, the only entries that need to be modified are:

- dir_input_netcdf,
- dir_npy,
- dir_proc_netcdf,
- dir_postproc_netcdf,
- dir_quickplots,
- dir_quickplots_postproc.

The user will need to assign the full path to the example MRR-PRO files (provided in the repository, located under “examples/Raw_data”) and to the five folders just created (see section 2.1) to their respective entries in the input files.

For example, let’s suppose that the user is working on Linux and that they saved the content of the ERUO repository on their Desktop at the location:

/home/your_username/Desktop/ERUO-main

(The string “your_username” in the file path should be replaced with the user’s actual username).

The user will have to modify the line in the “config.ini” files containing the value of the variable “dir_input_netcdf”, which will become:

dir_input_netcdf = /home/your_username/Desktop/ERUO-main/examples/Raw_data

If the user followed the naming convention of Section 2.1, the line containing the variable “dir_npy” in the “config.ini” files will have to modified in the following manner:

dir_npy = /home/your_username/Desktop/ERUO-main/examples/Archives

The same procedure will have to be repeated for all the remaining four entries listed before.

For a Windows-based example, refer to the video-tutorial (https://youtu.be/rUTj2xojSqo).

2.3 Preprocessing

The ERUO library is designed as a series of fours scripts, stored in the ERUO folder that the user will find inside the GitHub repository.

The scripts are numbered, with the digits 01 to 04 at the beginning of the name of each file. The preprocessing is the first of them, “01_preprocess_dataset.py”.

To execute it, open a terminal (on Linux) or a Command Prompt/Anaconda Prompt (on Windows) and navigate to the location of the script.

If the user installed the Python packages listed in Section 1.1 inside a specific Anaconda environment, such environment should be activated now.

The preprocessing can be executed by typing in the terminal:

python 01_preprocess_dataset.py

Once the execution is finished, inside the “Archives” folder (created in Section 2.1) the user will be able to find all the output of the preprocessing.

The user should check if the following files have appeared inside the folder:

- all_spectra_quantile.py
- alternative_interference_mask.py
- border_correction.npy
- interference_mask.npy
- reconstructed_median.npy

These files will be accompanied by some figures, displaying some of the intermediate steps and products of the preprocessing.

In particular, by comparing the figure “anomalies.png” and “corrected_anomalies.png”, the user may verify if the strongest peaks in the raw spectrum have been masked (grayed out) in the leftomst panel of “corrected_anomalies.png”.

When processing the examples, the two main peaks (approximately at range gate index 220) should be correctly masked, together with some fainter interference regions.

2.4 Processing and postprocessing

The remaining scripts are executed in the same way as the preprocessing one.

In the same terminal/prompt, execute the processing by typing:

python 02_process_dataset.py

Note that the script may print several warning messages at screen, but their presence can be safely ignored in most cases.

After the execution has been terminated, the processed files will be available in the “Proc_files” folder. The user should check that the files have been created.

We suggest to visualize the files immediately. To do so, the user should open the script “04_generate_quickplots.py” with an editor of their choice, and change the values of the two flags at the top of the script to:

QUICKPLOT_PROCESSED = 1

QUICKPLOT_POSTPROCESSED = 0

Then, the script can be executed by typing in the same terminal/prompt as before the instruction:

python 04_generate_quickplots.py

The resulting quickplots will appear inside the “Quickplots_proc” directory.

Once the user has checked the quickplots, they can proceed with the postprocessing. In the same terminal/prompt as before, type:

python 03_postprocess_dataset.py

The files will be generated inside the directory “Postproc_files”. To visualize them, open once again the script “04_generate_quickplots.py” with the editor and change the two flags at the top to:

QUICKPLOT_PROCESSED = 0

QUICKPLOT_POSTPROCESSED = 1

And execute it again by typing in the terminal/prompt:

python 04_generate_quickplots.py

The quickplots will appear in the “Quickplots_postproc” directory, that the user should access to check that the output products have been generated correctly.

In the case of the examples provided in the GitHub repository, the files in the “20191222” folder contain only some random noise in the low levels, while the one in the “20191223” contains snowfall measurements.

Note that the quickplots for the processed and postprocessed files can also be generated at the same time, by running the script “04_generate_quickplots.py” only at the end, and setting the values of the first two flags to:

QUICKPLOT_PROCESSED = 1

QUICKPLOT_POSTPROCESSED = 1

3. Datasets dominated by precipitation signals

The preprocessing of different datasets is likely to produce different masks, with interference detected at different location in the spectrum.

If a large section in the bottom part of the spectrum has been masked, this may indicate that the dataset has a large fraction of precipitation data. This will negatively impact the processing of the dataset, with false-positive in the flagging of interferences and the removal of precipitation data.

Therefore, in this case the preprocessing procedure should be repeated with some modifications:

1. isolate some clear-sky events throught the dataset;
2. copy the MRR-PRO files for those events to a different location;
3. open the “config.ini” file and assign the path to such location to the “dir_input_netcdf” variable;
4. run the preprocessing script as indicated in section 2.3;
5. re-open the “config.ini” file and re-assign the path to the whole dataset of MRR-PRO files to be processed (the one called “Raw_data” in the example described previously);
6. continue normally with the processing.

This set of operations with create an interference mask using files without any precipitation data, removing the risk of removing valid precipitation signal from the processed products.

4. Additional material

A video tutorial on how to process the example files available in the repository is available at: https://youtu.be/rUTj2xojSqo

Detailed information on the algorithm are available in [1].

5. References

[1] Ferrone, A., Billault-Roux, A.-C. M., and Berne, A.: ERUO: a spectral processing routine for the MRR-PRO, Atmos. Meas. Tech. Discuss. [preprint], https://doi.org/10.5194/amt-2021-294, in review, 2021.