MANUAL GUIDELINE

"RETRIEVAL OF FULL VERTICAL PROFILE OF CIRRUS MICROPHYSICAL PROPERTIES"

Author : Trismono Candra Krisna
Institution : SRON Netherlands Institute for Space Research
Version : V1.0
Release date : 02 June 2020
Email : t.c.krisna@sron.nl

CHANGE LOG

Version	Date	Author	Reason for changes
1.0	02 May 2020	Trismono C. Krisna	Initial document

INTRODUCTION

This document is written as a brief guideline to the end-to-end (E2E) cloud retrieval software. An optimal estimation (OE) approach is applied to obtain cloud parameters, namely total optical thickness *tau*, cloud top particle effective radius *reff_top,* cloud base particle effective radius *reff_top*, and shape *k*, which are further applied to infer the full vertical profile of *reff* as a function of *tau*. Basic idea of this technique is described in Krisna (2018).

CONTENT

The software is delivered as a full package consisting of several modules. Each is represented by a folder as follows,

1. CREATE CASE
   To generate the configuration of measurement condition.

2. SIM_FWD
   To simulate line-by-line spectra (forward model) given the output of [1] as the input.

3. L1B
   To generate synthetic measurement given the output of [2] as the input. This module generates spectra at instrument wavelengths by a convolution with noise applied on top of it.

4. SIM_RET
   To retrieve cloud parameters: *tau*, *reff_top, reff_base*, and *k*. The measurement input is taken from the output of [3]. An OE with Levenberg-Marquardt (LM) algorithm is implemented in the retrieval. Forward model and spectra convolution module are also carried out within the retrieval.

5. ERROR_ANALYSIS
   To calculate error using outputs of [4] as the input. The analysis is carried out by taking Jacobian matrix and error covariance from the last iteration assuming a linear problem.

Additionally, DATA dir is aimed to store auxiliary data such as the spectral albedo (sea water) and the wavelength grid. Those folders must **be placed in parallel** because a relative path is commonly used by each module to determine the location of others. Currently, the modules work consecutively (end-to-end), that makes convenient for sensitivity studies. It is possible to pull out an individual module, such as [4] and [5] if the user wants to work on the retrieval of real measurement data.

HARDWARE AND SOFTWARE REQUIREMENTS

The codes were built and tested in a LINUX environment with 4 cores (CPU) and 8 GB RAM. This is the standard requirement. The number of core is important for running modules with multi process. Before running the module, *libRadtran* should have been installed. The software is built and tested using *libRadtran* version 2.0.2. Older or newer versions might be applicable.

Please refer to *libRadtran* website http://www.libradtran.org/doc/libRadtran.pdf for more details.

The software is coded in Python3.7 with standard libraries. Please check the availability of the required libraries in your local machine prior running. A Python3 compiler is necessary. All versions of Python3 compiler should be relevant, but using version 3.7 is recommended. The codes should also work on Python 2 with some adjustments but it is **not recommended** considering its remaining lifetime.

HOW TO RUN MODULES

Brief steps to run each module is explained here. For more detailed information, please check in the codes. Descriptions have been made to explain each task, so the user can follow the workflow easily. Each module has corresponding MODULES dir, which is essential to store the routines. To make changes, please make a copy dir so it won't risk the master.

1. CREATE_CASE
   Please go to dir MODULES. The main code is called main_create.py. For this standard example, only *tau*, *reff_top, reff_base, and k* are varied. The other parameters, e.g., surface albedo, measurement geometry, cloud height, etc, remain constant but it's possible to be adjusted depending on the purpose. The code can be executed from the terminal,

   >> python3 main_create.py

   where python3 is the alias of Python3 compiler in your machine. Once the code has been executed, the outputs are stored in the dir OUTPUT. Please make sure this folder exists prior execution (recommended), although the code is capable to create. If it is already exist and (or) has outputs from previous executions, it will not be overwritten.

2. SIM_FWD
   The main code is located in the MODULES dir, namely main_fwd_sp.py for serial-processing or main_fwd_mp.py for multi-processing. The use of multi-processing module is recommended. The number of process employed, namely n_proc, must be adjusted in main_fwd_mp.py. In this example n_proc is set to 3, that can be extended as long as it does not exceed the number of core (CPU). Otherwise, the maximum performance won't be gained. Prior execution, change the path of *libRadtran* in fwd_wrapper.py (variable lib_dir) based on the condition in your local machine and use an absolute path. The code can be executed from the terminal,

   >> python3 main_fwd_<sp/mp>.py

   The outputs, such as input-output of forward simulation and transformed radiative quantities, will be stored in the dir INOUT. The dir CLOUD is dedicated to store the cloud profiles. The main code may generate those dirs, but it is recommended to create them before execution.

3. L1B
   The main code is in the MODULES dir, namely main_l1b.py. The noise realization is adapted from the Moderate Resolution Imaging Spectroradiometer (MODIS) Signal-to-Noise Ratio (SNR). This is a simple assumption, thus it is possible to be modified by the user. Please read the description given in the code for more detail. The code can be executed from the terminal,

   >> python3 main_l1b.py

   The outputs are stored in the dir OUTPUT. For each measurement, the spectra information is split into the visible to near-infrared range (VNIR) and the short wavelength infrared (SWIR). To define the spectra at a particular wavelength, a convolution is applied. The instrument spectral response function (ISRF) is assumed as a Gaussian function with Full Width at Half Maximum (FWHM) of 3 nm and 201 sampling grid.

4. SIM_RET
   The main code is placed in the MODULES dir and it is called main_l2m_sp.py for serial-processing or main_l2m_mp.py for multi-processing (recommended). The number of process employed, namely n_proc, must be adjusted in main_fwd_mp.py. In this example n_proc is set to 3, that can be extended as long as it does not exceed the number of core (CPU). Otherwise, the maximum performance won't be gained. Please change the *libRadtran* path to your local machine. This can be done by changing variable lib_dir in l2m_wrapper.py in MODULES.

   Before running retrievals, you must create the retrieval setting. A python code retrieval_setting.py is available in the dir INPUT. Do not change or remove this dir. Albeit, a standard setting has been defined, users have the freedom to change the setting. It must be reasonable, otherwise deprecated retrieval performance may occur. When the setting has been fixed, it can be executed as follows,

   >> python3 retrieval_setting.py

   which creates the retrieval_setting.ini (retrieval setting) in the respective dir. The a priori is currently set to 0.5 for *tau*, 10 µm for *reff_top,* 30 µm for *reff_base,* and 4 for *k*. The prior error is intentionally set to a relatively large number (twice as a priori) to minimize possible constrains in the retrieval. The maximum number of iteration is currently set to 15. However, retrievals will most likely converge before 8 iterations (3-6 in average). The noise flag = 1 is for retrieval with noise and 0 for without noise applied on the measurement. The retrieval is performed using wavelengths between 630 and 670 nm in the visible to near infrared (VNIR), between 1180 and 1300 nm, and between 1500 and 1700 nm in the short wavelength infrared (SWIR) range. Using this configuration, the number of measurements used in the retrieval would be 109 (over-constrained problem). Finally, the retrieval code can be executed from terminal as follows,

   >> python3 main_l2m_<sp/mp>.py

   The dir DUMMY is aimed to store input-output of forward simulation, such as cloud profile and line-by-line spectra, as well as convolved spectra. Retrieval outputs are

stored in the dir OUTPUT. Those two dirs can be created by the code, but it is recommended to create them before execution.

The retrieval module generates three main outputs: Jacobian matrix, Gain matrix, and retrieval statistic which summarizes results (e.g., optimum solution and cost function) and retrieval diagnostics. conv_id 1 = converged, 2 = not converged (exceed maximum number of iteration) , and 3 = not converged (boundary condition hit). Only retrievals with index 1 provide meaningful information. Jacobian matrix kmat_<pixel_id>_<iter_id>.dat is comprised of six columns: wavelength, measured radiance, measurement covariance (error), modeled radiance, Jacobian *tau*, *reff_top, reff_base,* and *k*. The Gain gain_<pixel_id>.dat has three columns: wavelength, Gain *tau*, *reff_top, reff_base,* and *k*. Those two matrices are meaningful to analyze retrieval errors.

5. ERROR_ANALYSIS
   The main code is located in the MODULES dir and it is called main_l2m_error.py. Note that variable weighting (prior error) must be consistent with those defined in the retrieval setting [4]. Jacobian matrix from the last iteration [4] is used in this analysis. The code can be executed from the terminal as follows,

   >> python3 main_l1m_error.py

   Retrieval errors are stored in the dir OUTPUT. Four different errors are provided: prior_error = error estimate before measurement is made, noise_error = error due to instrument noise, smoothing_error = error due to the lack of prior information, and total_error = error due to noise and smoothing error.

KNOWN LIMITATION

The use of this retrieval technique is limited for cirrus clouds with thin–moderate optical thickness. For optically thick clouds, the retrieval may fail due to decreasing sensitivity towards the cloud base. The ice particle habit is currently fixed to the severely rough column_8elements by Yang et. al. (2013). This is opted to be consistent with the current habit applied by MODIS for cirrus retrievals. It is possible to change that from the uvspec_input.py located in the SIM_FWD/MODULES and SIM_RET/MODULES. Make sure that both are consistent.

The computation of Jacobian matrix is considerably expensive. To calculate Jacobian, the forward model need to be run five (5) times. As the forward model opts for tau550 set, the entire spectra range including lambda = 550 nm must be calculated. Those factors makes one iteration needs 7 minutes more or less, depending on the machine. Assuming one retrieval needs maximum 6 iteration, around 42 minutes are required for one pixel retrieval. To handle the computational cost, a multiprocessing module is embedded. In the future, an alternative method to derive Jacobian should be implemented.

SOURCE CODE AVAILABILITY

Codes are available on the version control
https://github.com/trismono/CLOUD_RETRIEVAL_PROFILE_FULL.git.

REFERENCE

Krisna, T.C., 2018: Airborne Passive Remote Sensing of Optical Thickness and Particle Effective Radius of Cirrus and Deep Convective Clouds, PhD Thesis, University of Leipzig.

Rodgers, C. D., 2000: Inverse Methods for Atmospheric Sounding: Theory and Practice. World Scientific Publishing Company, 240 pp.

Yang, P., L. Bi, B.A. Baum, K. Liou, G.W. Kattawar, M.I. Mishchenko, and B. Cole, 2013: Spectrally Consistent Scattering, Absorption, and Polarization Properties of Atmospheric Ice Crystals at Wavelengths from 0.2 to 100 μm. J. Atmos. Sci., 70, 330–347, https://doi.org/10.1175/JAS-D-12-039.1

APPENDIX

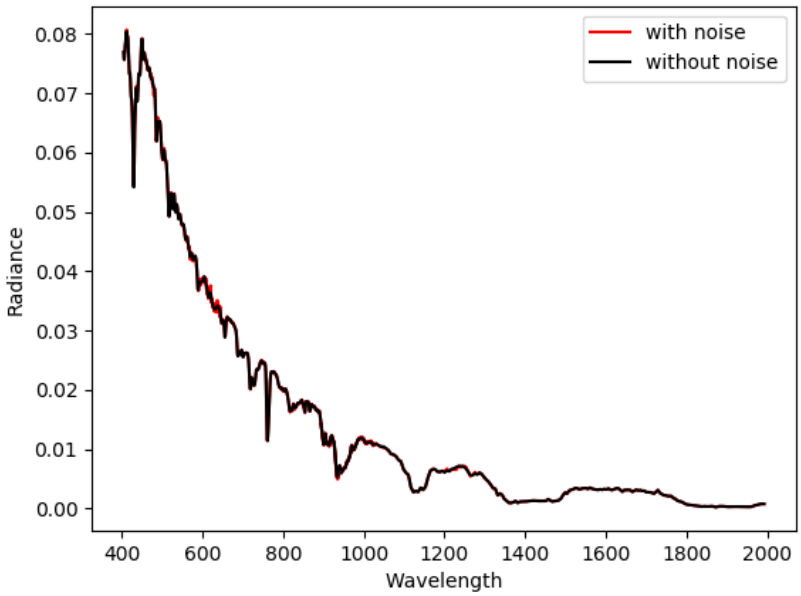

Figure 1. Spectra radiance with and without noise for cloudy condition. Cirrus *tau* = 0.3, *reff_top* = 8 µm, *reff_base* = 20 µm, *k* = 4, surface albedo = sea water, sza = 30°, phi0 = 100°.

```
Info     | State tau = 0.592 reff_top = 8.027 reff_base = 30.531 and shape = 4.011
Info     | Cost function chi2 = 600.709
Info     | Pixel ID 50 :: Iteration 2
Info     | Perturbation coefficient p_tau = 0.025 p_reff_top = 0.400 p_reff_base = -1.500 and p_shape = 0.200
Info     | State tau = 0.591 reff_top = 8.028 reff_base = 30.497 and shape = 4.010
Info     | Cost function chi2 = 15.765
Info     | Pixel ID 50 :: Iteration 3
Info     | Perturbation coefficient p_tau = -0.006 p_reff_top = 0.080 p_reff_base = 0.305 and p_shape = 0.040
Info     | State tau = 0.594 reff_top = 8.072 reff_base = 31.485 and shape = 4.033
Info     | Cost function chi2 = 614.450
Info     | Pixel ID 51 :: Iteration 2
Info     | Perturbation coefficient p_tau = 0.025 p_reff_top = 0.400 p_reff_base = -1.500 and p_shape = -0.200
Info     | State tau = 0.594 reff_top = 8.084 reff_base = 31.412 and shape = 4.031
Info     | Cost function chi2 = 17.814
Info     | Pixel ID 51 :: Iteration 3
Info     | Perturbation coefficient p_tau = 0.006 p_reff_top = 0.081 p_reff_base = 0.315 and p_shape = -0.040
Info     | State tau = 0.591 reff_top = 8.028 reff_base = 30.497 and shape = 4.010
Info     | Cost function chi2 = 15.859
Info     | Convergence ID = 1 :: Retrieval succeed!
Info     | Retrieval done in 1183.65 sec
Info     | Elapsed time per iteration = 591.83 sec
Info     | Pixel ID 52 :: Iteration 1
Info     | Perturbation coefficient p_tau = 0.025 p_reff_top = 0.400 p_reff_base = 1.500 and p_shape = -0.200
Info     | State tau = 0.594 reff_top = 8.084 reff_base = 31.412 and shape = 4.031
Info     | Cost function chi2 = 17.916
Info     | Convergence ID = 1 :: Retrieval succeed!
Info     | Retrieval done in 1187.53 sec
Info     | Elapsed time per iteration = 593.77 sec
Info     | Pixel ID 53 :: Iteration 1
Info     | Perturbation coefficient p_tau = -0.025 p_reff_top = 0.400 p_reff_base = 1.500 and p_shape = 0.200
Info     | State tau = 0.592 reff_top = 8.318 reff_base = 36.819 and shape = 4.155
Info     | Cost function chi2 = 529.030
Info     | Pixel ID 52 :: Iteration 2
Info     | Perturbation coefficient p_tau = 0.025 p_reff_top = 0.400 p_reff_base = 1.500 and p_shape = -0.200
```

Figure 2. Screen output of L2M (retrieval module) using multi-processing module.

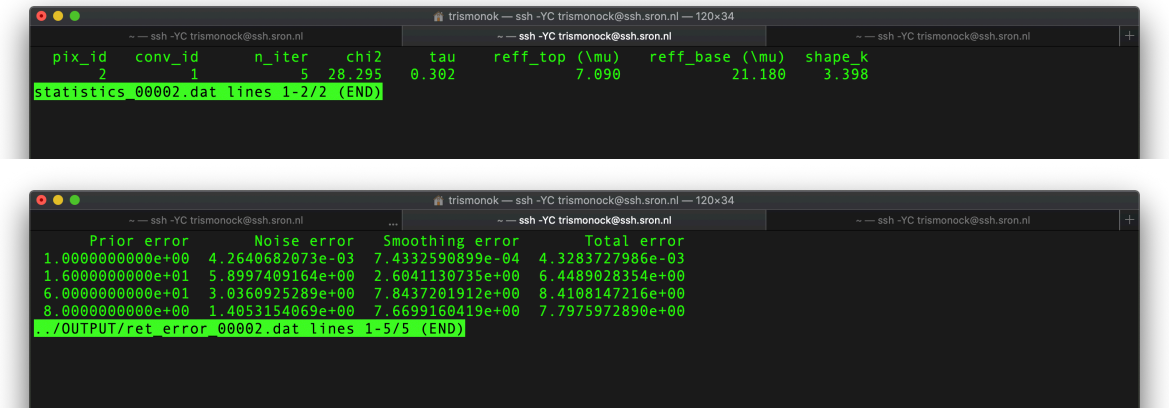

Figure 3. Retrieval statistics (top) and error (bottom). The order of retrieval error (from top to bottom): *tau*, *reff_top* (µm), *reff_base* (µm), and *k*.

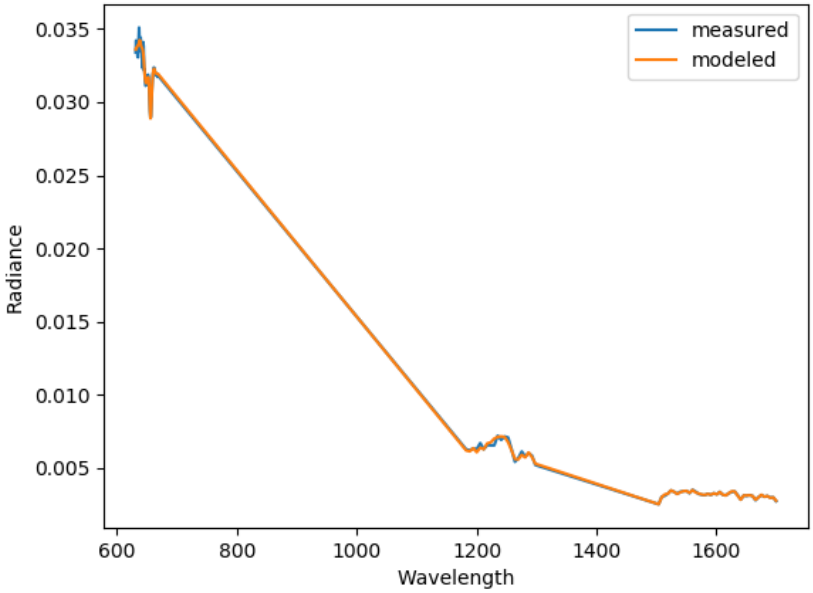

Figure 4. Spectral radiance measured and retrieved (modeled). The retrieval is performed using wavelengths from 630-670 nm, 1180-1300 nm, and 1500-1700. In total the retrieval is performed with 109 wavelengths.