Jet Propulsion Laboratory
California Institute of Technology

ReFRACtor: Towards a Re-Usable Multi-Instrument Atmospheric Composition Retrieval Framework

Sebastian Val, James McDuffie, Kevin Bowman, Jonathon Hobbs, Vijay Natraj, Matthew Thill

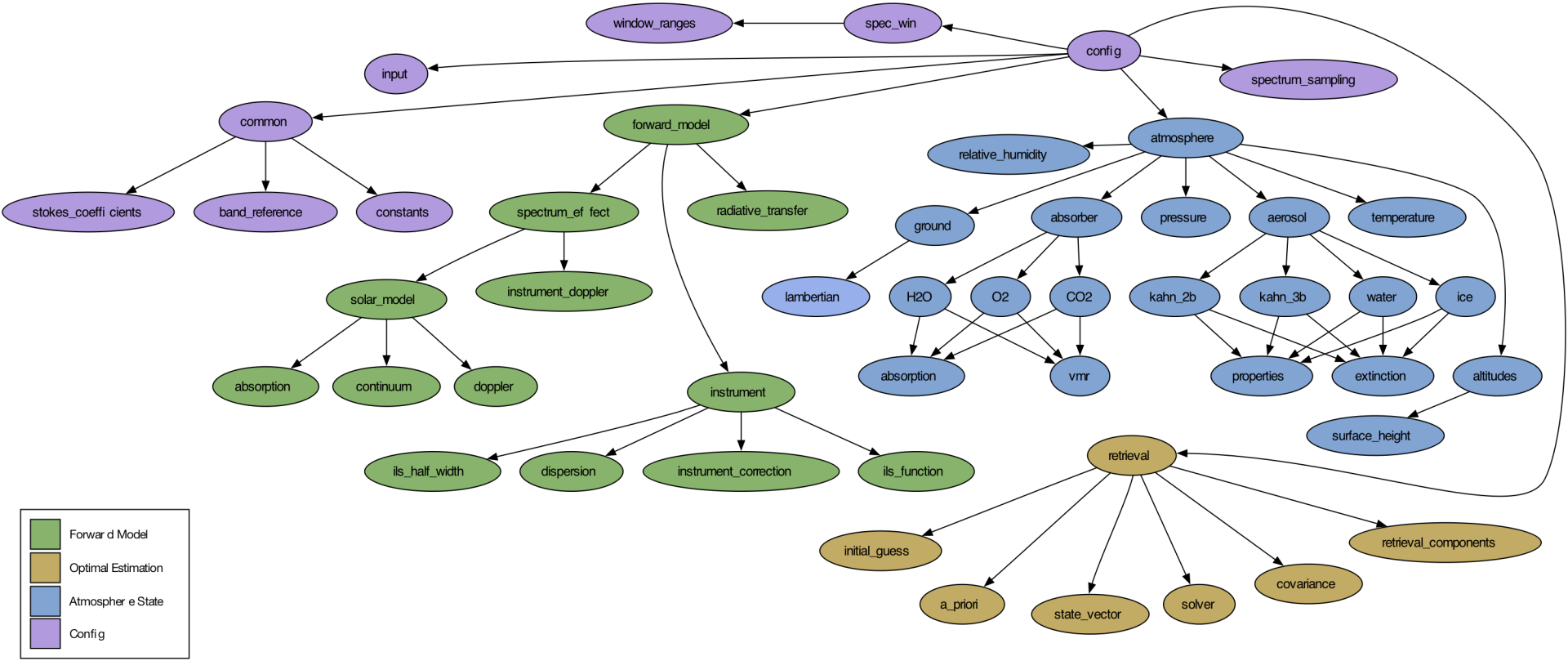
Introduction

- Open-source multi-instrument radiative transfer (RT) and atmospheric composition retrieval framework
- Fork of the operational Orbiting Carbon Observatory-2 (OCO-2) retrieval software package
- Supports broad spectral range from thermal infrared (TIR) to ultraviolet (UV)
- Combines radiance measurements from multiple instruments to enable joint retrievals and data fusion products

Features

- Bundled Pan Spectral ABSCO Generation Software
- Pluggable Radiative Transfer Modules (First-Order, 2-Stream, Linearized Discrete Ordinate)
- Pluggable Solver Modules (Connor, Non-Linear Least Squares)
- Modular Configuration (Data Readers, Atmosphere Definition, and Instrument Models)
- Extensive Set of Atmospheric, Surface, and Instrument Parameters For Retrieval
- Testable With The Inclusion of Over 180 Automated Unit Tests
- Runnable On Both Linux and Mac OS X

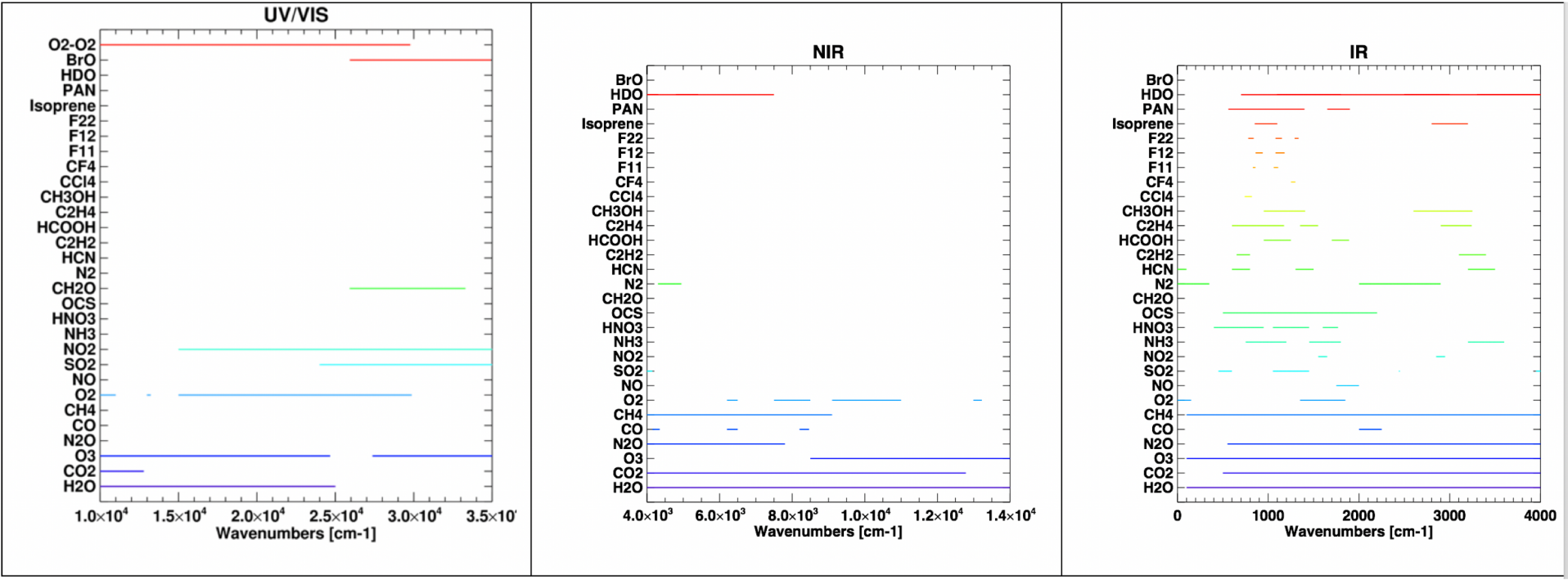
Modular Configuration Architecture



Pan Spectral ABSCO Generation Software

- Able to generate a set of absorption coefficient (ABSCO) tables that span the thermal IR to UV spectral range
- Handles a wide range of molecules including principal absorbers as well as molecules that play important roles in atmospheric chemistry (isoprene), impact air quality (NH_3), or provide information on atmospheric circulation (HDO)
- Uses Line-By-Line Radiative Transfer Model (LBLRTM) powered by AER maintained line parameter database (LPD) as the basis for the line parameters used in LBLRTM
- For molecules that have both line parameters and cross-section parameters, HITRAN recommendations are followed

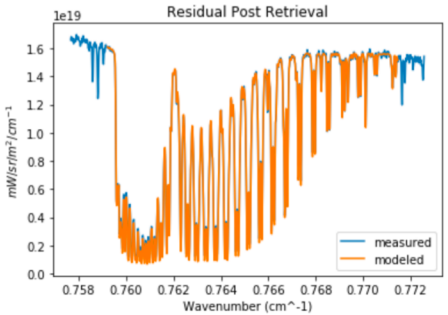
Pan Spectral ABSCO Generation Software



Spectral ranges adopted for all selected molecules in the UV/VIS (left), NIR (center), IR (right)

Python Usage

```
In [4]: from refractor.factory import process_config
        from refractor import framework as rf

In [5]: from config import oco_config

In [6]: data_dir = os.path.realpath('../test/in')
        l1b_file = os.path.join(data_dir, "oco2_L1bScND_16094a_170711_B7302r_171102090317-selected_ids.h5")
        met_file = os.path.join(data_dir, "oco2_L2MetND_16094a_170711_B8000r_171017214714-selected_ids.h5")

        sounding_id = "2017071110541471"

In [7]: config_def = oco_config.retrieval_config_definition(l1b_file, met_file, sounding_id)
        config_inst = process_config(config_def)

In [8]: fm = config_inst.forward_model
        solver = config_inst.retrieval.solver.solve()

In [10]: plot_fm(0)
         title("Residual Post Retrieval");
```



Contact & Contribute

refractor.github.io/framework/

github.com/ReFRACtor

Sebastian.Val@jpl.nasa.gov

Jet Propulsion Laboratory
California Institute of Technology

jpl.nasa.gov

© 2018. All rights reserved, California Institute of Technology
A portion of this work was performed at the Jet Propulsion Laboratory, California Institute of Technology, under contract with NASA.
Funded by NASA Advanced Information Systems Technology (AIST) Program. AIST16-0019