PythonicDISORT: A Python reimplementation of the Discrete Ordinate Radiative Transfer package DISORT

Dion J. X. Ho [1]

1 Columbia University, Department of Applied Physics and Applied Mathematics, USA

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Sophie Beck

Reviewers:

- @arjunsavel
- @simonrp84
- @pscicluna

Submitted: 11 February 2024
Published: unpublished

Summary

The Radiative Transfer Equation (RTE) models the processes of absorption, scattering and emission as electromagnetic radiation propagates through a medium. We address the 1D RTE (1) in a plane-parallel atmosphere and consider three sources: blackbody emission from the atmosphere $s(\tau)$, scattering from sunlight $\frac{\omega I_0}{4\pi} p\left(\mu, \phi; -\mu_0, \phi_0\right) \exp\left(-\mu_0^{-1}\tau\right)$, and incoming radiation from other atmospheric layers or the Earth's surface modeled by Dirichlet boundary conditions.

$$
\begin{aligned}
\mu \frac{\partial u(\tau, \mu, \phi)}{\partial \tau} = u(\tau, \mu, \phi) &- \frac{\omega}{4\pi} \int_{-1}^{1} \int_{0}^{2\pi} p\left(\mu, \phi; \mu', \phi'\right) u\left(\tau, \mu', \phi'\right) \mathrm{d}\phi' \mathrm{d}\mu' \\
&- \frac{\omega I_0}{4\pi} p\left(\mu, \phi; -\mu_0, \phi_0\right) \exp\left(-\mu_0^{-1}\tau\right) - s(\tau)
\end{aligned}
\tag{1}
$$

The RTE is important in many fields of science and engineering. The gold standard for numerically solving the 1D RTE is the Discrete Ordinate Radiative Transfer package `DISORT` which was coded in FORTRAN 77 and first released in 1988 (Stamnes et al., 1988). It has been widely used, for example by `MODTRAN` (Berk et al., 2014), `Streamer` (Key & Schweiger, 1998), and `SBDART` (Ricchiazzi et al., 1998), all of which are comprehensive radiative transfer models that are themselves widely used in atmospheric science. `DISORT` implements the Discrete Ordinates Method which has two key steps. First, the diffuse intensity function u is expanded as the Fourier cosine series:

$$
u\left(\tau, \mu, \phi\right) = \sum_{m=0} u^m\left(\tau, \mu\right) \cos\left(m\left(\phi_0 - \phi\right)\right)
$$

This addresses the ϕ' integral in (1) and decomposes the problem into solving

$$
\mu \frac{du^m(\tau, \mu)}{d\tau} = u^m(\tau, \mu) - \int_{-1}^{1} D^m\left(\mu, \mu'\right) u^m\left(\tau, \mu'\right) \mathrm{d}\mu' - Q^m(\tau, \mu) - \delta_{0m} s(\tau)
$$

for each Fourier mode of u. The second key step is to discretize the μ' integral using some quadrature scheme; `DISORT` uses the double-Gauss quadrature scheme from Sykes (1951). This results in a system of ODEs that can be solved using standard methods.

Our package `PythonicDISORT` is a Python 3 reimplementation of `DISORT` that replicates most of its functionality while being easier to install, use, and modify, though at the cost of computational speed. It has `DISORT`'s main features: multi-layer solving, delta-M scaling, Nakajima-Tanaka (NT) corrections, only flux option, isotropic internal sources (thermal sources), Dirichlet boundary conditions (diffuse flux boundary sources), Bi-Directional Reflectance

Function (BDRF) for surface reflection, and more. In addition, `PythonicDISORT` has been tested against `DISORT` on `DISORT`'s own test problems. As far as we know, all prior attempts at creating Python interfaces for `DISORT` have focused on creating wrappers and `PythonicDISORT` is the first true Python reimplementation.

Statement of need

We clarify that `PythonicDISORT` is not meant to replace `DISORT`. Due to fundamental differences between Python and FORTRAN, `PythonicDISORT`, though optimized, remains about an order of magnitude slower than `DISORT`. Thus, projects which prioritize computational speed should still use `DISORT`. Moreover, `PythonicDISORT` lacks `DISORT`'s latest features, most notably its pseudo-spherical correction.

`PythonicDISORT` is instead designed with three goals in mind. First, it is meant to be a pedagogical and exploratory tool. `PythonicDISORT`'s ease of installation and use makes it a low-barrier introduction to Radiative Transfer and Discrete Ordinates Solvers. Even researchers who are experienced in the field may find it useful to experiment with `PythonicDISORT` before deciding whether and how to upscale with `DISORT`. Installation of `PythonicDISORT` through `pip` should be system agnostic as `PythonicDISORT`'s core dependencies are only `NumPy` and `SciPy`. We also intend to implement conda installation. In addition, using `PythonicDISORT` is as simple as calling the Python function `pydisort`. In contrast, `DISORT` requires FORTRAN compilers, has a lengthy and system dependent installation process, and each call requires shell script for compilation and execution.

Second, `PythonicDISORT` is designed to be modified by users to suit their needs. Given that Python is a widely used high-level language, `PythonicDISORT`'s code should be understandable, at least more so than `DISORT`'s FORTRAN code. Moreover, `PythonicDISORT` comes with a Jupyter Notebook (our *Comprehensive Documentation*) that breaks down both the mathematics and code behind the solver. Users can in theory follow the Notebook to recode `PythonicDISORT` from scratch; it should at least help them make modifications.

Third, we intend for `PythonicDISORT` to be a testbed. For the same reasons given above, we expect that it is easier to implement and test experimental features in `PythonicDISORT` than in `DISORT`. This should expedite research and development for `DISORT` and similar algorithms.

`PythonicDISORT` was first released on PyPI and GitHub on May 30, 2023. We know of its use in at least three ongoing projects: on the Two-Stream Approximations, on atmospheric photolysis, and on the topographic mapping of Mars through photoclinometry. We will continue to maintain and upgrade `PythonicDISORT`. Our latest version: `PythonicDISORT v0.4.2` was released on Nov 28, 2023.

Acknowledgements

I acknowledge funding from NSF through the Learning the Earth with Artificial intelligence and Physics (LEAP) Science and Technology Center (STC) (Award #2019625). I am also grateful to my Columbia University PhD advisor Dr. Robert Pincus and co-advisor Dr. Kui Ren for their advice and contributions.

References

Berk, A., Conforti, P., Kennett, R., Perkins, T., Hawes, F., & Bosch, J. van den. (2014). MODTRAN® 6: A major upgrade of the MODTRAN® radiative transfer code. *2014 6th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, 1–4. https://doi.org/10.1109/WHISPERS.2014.8077573

Key, J. R., & Schweiger, A. J. (1998). Tools for atmospheric radiative transfer: Streamer and FluxNet. *Computers & Geosciences*, *24*(5), 443–451. https://doi.org/https://doi.org/10.1016/S0098-3004(97)00130-1

Ricchiazzi, P., Yang, S., Gautier, C., & Sowle, D. (1998). SBDART: A research and teaching software tool for plane-parallel radiative transfer in the earth's atmosphere. *Bulletin of the American Meteorological Society*, *79*(10), 2101–2114. https://doi.org/https://doi.org/10.1175/1520-0477(1998)079%3C2101:SARATS%3E2.0.CO;2

Stamnes, K., Tsay, S.-C., Wiscombe, W., & Jayaweera, K. (1988). Numerically stable algorithm for discrete-ordinate-method radiative transfer in multiple scattering and emitting layered media. *Appl. Opt.*, *27*(12), 2502–2509. https://doi.org/10.1364/AO.27.002502

Sykes, J. B. (1951). Approximate Integration of the Equation of Transfer. *Monthly Notices of the Royal Astronomical Society*, *111*(4), 377–386. https://doi.org/10.1093/mnras/111.4.377