UNIVERSITY OF CENTRAL FLORIDA

BARTTest Users Manual

A TEST SUITE FOR RADIATIVE-TRANSFER AND RETRIEVAL CODES

Author:
Michael D. HIMES

Supervisor:
Dr. Joseph HARRINGTON

June 8, 2023

Contents

1 Team Members

- Michael D. Himes, University of Central Florida (mhimes@knights.ucf.edu).
- Joseph Harrington, University of Central Florida (jh@physics.ucf.edu).
- Ryan Challener, University of Central Florida (rchallen@knights.ucf.edu).
- Patricio Cubillos, Space Research Institute, Austrian Academy of Sciences (patricio.cubillos@oeaw.ac.at)
- Jasmina Blecic, New York University Abu Dhabi (jasmina@nyu.edu).
- M. R. Green, University of Central Florida (mraechelgreen@knights.ucf.edu)

2 Introduction

This document describes BARTTest, a collection of tests for radiative-transfer (RT) and exoplanet retrieval codes. The tests include simple forward models with analytically-verifiable answers, realistic forward models for an HD 189733b-like planet with various types of temperature profiles, and retrievals on synthetic and real data. The simple models test the presence, shape, and strength of lines and other basic aspects of RT codes. The realistic forward models and retrievals provide results that can be compared with other codes.

BARTTest is configured by default to test BART, the Bayesian Atmospheric Radiative Transfer code. The configuration and input files are provided as reference for users to see the exact setup for each test so that it can be reproduced by other codes. This manual provides a description of each test as well as each individual file for further clarity.

This manual describes the BARTTest program usage for the user (Sections 3 through 5) and potential contributors (Section 6). Section 3 indicates the system requirements and how to obtain the code. It also details the commands to run the program. Section 4 describes the directories and files contained in BARTTest. Section 5 shows how to execute BARTTest from start to finish for BART. Section 6 details how a user would contribute the results of their code to BARTTest.

2.1 BARTTest Overview

BARTTest is organized as follows:

```
BARTTest
 ├─ code-output
 │   ├─ 01BART
 │   └─ <other codes>
 ├─ doc
 ├─ lib
 ├─ results
 │   ├─ 01BART
 │   ├─ <other codes>
 │   └─ plots
 └─ tests
     ├─ 00inputs
     │   ├─ filters
     │   ├─ par
     │   └─ TLI
     ├─ c01hjcleariso
     ├─ c02hjclearnoinv
     ├─ c03hjclearinv
     ├─ f01oneline
     ├─ f02fewline
     ├─ f03multiline
     ├─ f04broadening
     ├─ f05abundance
     │   └─ atm
     ├─ f06blending
     ├─ f07multicia
     ├─ f08isothermal
     ├─ f09energycons
     ├─ r01hd189733b .3 s01hjcleariso
     ├─ s02hjclearnoinv
     └─ s03hjclearinv
```

Each directory and the files within will be discussed in detail in Section 4.

2.2 License

3 Setup

3.1 System Requirements

BARTTest is configured to run on Linux; there are terminal commands executed that are Linux-specific, so users would have to modify this to run on other operating systems. The hardware requirements to run BARTTest are dependent on which tests the user wishes to execute. The isothermal (f08), energy conservation (f09), comparison (c01 - c03), and retrieval (r01, s01-s03) tests all have greater requirements for storage and RAM. The retrievals are by default set to use 10 CPU cores. To use less, the user must alter the configuration files; at least 3 cores are required to ensure proper sampling of the parameter space.

Requirements (excluding isothermal, comparison, and retrieval tests):

- 1-core CPU
- 1 GB storage
- 6 GB RAM

Requirements (all tests):

- 3-core CPU
- 40 GB storage
- 16 GB RAM

3.2 Execution Methods

The user has a number of ways to run the tests depending on where they are at in the process. The general process is as follows when running BARTTest for BART:

- Pull BART from Github and compile
- Pull line lists (note: only necessary for isothermal, energy conservation, comparison, and retrieval tests)
- Run the tests
- Produce plots of the results

The program is controlled by a Makefile in the top-level directory; all commands to execute BARTTest thus take the form of `make <command>`, executed from the top-level directory. The configured commands are listed in Table 1.

Table 1: Makefile Commands

Command	Description
bart	Downloads BART from Github to BARTTest/../BART/ and compiles.
linelists	Downloads necessary line lists for the tests, except STDS's CH4 list.

Table 1: Makefile Commands

Command	Description
oneline	Executes the f01oneline test (see Section 4.6.2).
fewline	Executes the f02fewline test (see Section 4.6.3).
multiline	Executes the f03multiline test (see Section 4.6.4).
broadening	Executes the f04broadening test (see Section 4.6.5).
abundance	Executes the f05abundance test (see Section 4.6.6).
blending	Executes the f06blending test (see Section 4.6.7).
multicia	Executes the f07multicia test (see Section 4.6.8).
isothermal	Executes the f08isothermal test (see Section 4.6.9).
energycons	Executes the f09energycons test (see Section 4.6.10).
comparison_tli	Generates the TLI file for the c01hjcleariso, c02hjclearnoinv, and c03hjclearinv tests (see Sections 4.6.11 – 4.6.13).
comparison_iso	Runs the isothermal emission and transmission cases for the c01hjcleariso test. Note: This may take a couple hours.
comparison_noinv	Runs the noninverted emission and transmission cases for the c02hjclearnoinv test. Note: This may take a couple hours.
comparison_inv	Runs the inverted emission and transmission cases for the c03hjclearinv test. Note: This may take a couple hours.
comparison_BarstowEtal	Runs the selected Barstow et al. (2020) cases: Model 0, Model 1, 1000 K w/ 1e-4 CO, 1500 K w/ 1e-4 CO, 1500 K w/ 1e-5 CO (see Sections 4.6.14 and 4.6.15).
BarstowEtal_OSF	Downloads the Barstow et al. (2020) OSF archive, which contains their spectra.
comparison_plots	Produces the plots that compare Transit's results to the Barstow et al. (2020) cases.
comparisontests	Generates TLI file, runs all of the cases for the c01hjcleariso, c02hjclearnoinv, and c03hjclearinv tests, and produces plots (see see Sections 4.6.11 – 4.6.13). Note: this takes on the order of hours.
synthretrievals	Runs all of the synthetic retrievals (s01 – s05; see Section 4.6.16). Note: this may take on the order of weeks.
BarstowEtalretrievals	Runs all of the Barstow et al. (2020) synthetic retrievals (s04 and s05; see Sections 4.6.19 and 4.6.20). Note: this may take on the order of a week.
retrieval_tli	Generates the TLI file for the s01 – s03 tests. Note: equivalent to comparison_tli.
retrieval_iso_e	Runs emission retrieval for the s01hjcleariso test. Note: this takes on the order of days.
retrieval_iso_t	Runs transmission retrieval for the s01hjcleariso test. Note: this takes on the order of days.

Table 1: Makefile Commands

Command	Description
retrieval_noinv_e	Runs emission retrieval for the s02hjclearnoinv test. Note: this takes on the order of days.
retrieval_noinv_t	Runs transmission retrieval for the s02hjclearnoinv test. Note: this takes on the order of days.
retrieval_inv_e	Runs emission retrieval for the s03hjclearinv test. Note: this takes on the order of days.
retrieval_inv_t	Runs transmission retrieval for the s03hjclearinv test. Note: this takes on the order of days.
retrieval_BarstowEtal_NEMESIS_clear	Runs transmission retrieval on NEMESIS spectrum for Barstow et al. (2020) Model 0 (see Sections 4.6.19 and 4.6.20).
retrieval_BarstowEtal_NEMESIS_cloud	Runs transmission retrieval on NEMESIS spectrum for Barstow et al. (2020) Model 1 (see Sections 4.6.19 and 4.6.20).
retrieval_BarstowEtal_CHIMERA_clear	Runs transmission retrieval on CHIMERA spectrum for Barstow et al. (2020) Model 0 (see Sections 4.6.19 and 4.6.20).
retrieval_BarstowEtal_CHIMERA_cloud	Runs transmission retrieval on CHIMERA spectrum for Barstow et al. (2020) Model 1 (see Sections 4.6.19 and 4.6.20).
retrieval_BarstowEtal_TauREx_clear	Runs transmission retrieval on Tau-REx spectrum for Barstow et al. (2020) Model 0 (see Sections 4.6.19 and 4.6.20).
retrieval_BarstowEtal_TauREx_cloud	Runs transmission retrieval on Tau-REx spectrum for Barstow et al. (2020) Model 1 (see Sections 4.6.19 and 4.6.20).
hd189_tli	Generates the TLI file for the HD 189733b retrieval (see Section 4.6.21).
hd189_retrieval	Runs a retrieval using HD 189733b secondary eclipse data. Note: this takes on the order of days (see Section 4.6.21).
hd189	Equivalent to executing hd189_tli and hd189_retrieval Note: this takes on the order of days (see Section 4.6.21).
quicktests	Runs f01 – f07.
forwardtests	Runs f01 – f09.
all	Downloads BART and HITRAN/HITEMP/ExoMol line lists, and runs f01 – f09 and c01 – c05. Note: does not execute the retrievals due to the time involved.
plots	Produces plots of the code output for f01 – f09, c01 – c03, and s01 – s03.
retrievalplots	Produces plots that compare the retrieved results to the known ground-truth for the synthetic retrieval cases.

Table 1: Makefile Commands

Command	Description
clean	Deletes everything in BARTTest/code-output/01BART/ and BARTTest/results/01BART/.

These are the built-in methods of executing BARTTest. If the user wishes to run the package in some other manner, it must be manually configured.

4 Descriptions of Directories and Files

This section describes the files and subdirectories in BARTTest.

4.1 Top-level Files

The BARTTest directory contains some files:

- environment.yml: a conda environment that can run the software.
- HOW_TO_FIND_WHAT_YOU_ARE_LOOKING_FOR: a quickguide to the directories and what they contain.
- LICENSE: identical to Section 2.2. It contains the usage license.
- Makefile: the main driver of the program; it handles commands used to execute tests, download line lists, and so on. See Section 3.2 for more information.
- README: a very brief version of this manual.

4.2 code-output

The code-output directory contains subdirectories for the output of each RT/retrieval code. Each subdirectory is numbered according to the time that the output was received. For example, there is a directory “01BART” which contains BART’s output for each of the tests as it was the first code to be tested.

4.3 doc

The doc directory contains documentation related to BARTTest, which you are currently reading.

4.4 lib

The lib directory contains various functions and scripts used in the execution of BARTTest.
Files in the directory:

- abuncomp.py: contains a function to plot the results of the f05abundance test (see Section 4.6.6). Produces the plot when executed.
- comparison.py: contains a function to plot the results of the c01 – c03 tests for multiple codes. When executed, it produces plots to compare BART and RHD.

- energycons.py: contains a function to process the results of the f09energycons test (see Section 4.6.10). Produces a text file with the results when executed.
- makedepths.py: contains a function to take the outputs from the c01 – c03 tests and produce the synthetic eclipse/transit depths for the s01 – s03 tests. Note: uncertainties are exclusively due to photon noise.
- makeplots.py: contains functions used to plot the results of the f01 – f09 tests, and produces those plots when executed.
- opacityconv.py: contains a function to read a (binary) Transit opacity file, convert it to Numpy arrays, and save out a .npz file.
- plotPTs.py: a script to plot the T(p) profiles used in the c01 – c03 and s01 – s03 tests.
- retrievalplots.py: contains functions to plot the results of the s01 – s03 tests against the known ground-truth answers.
- retrievalplots_BarstowEtal.py: as above, but for the Barstow et al. (2020) cases of s04 and s05.
- spec_grid_comp.py: a script to produce plots related to Appendix B of Harrington et al. (2021).
- voigtcomp.py: contains a function to compare the line profile from the broadening test with the normalized theoretical Voigt profile corresponding to the setup (temperature, pressure, molecular properties, layer abundances). Used in the f04broadening test (see Section 4.6.5).
- voigt.py: contains functions to compute a Gaussian, Lorentzian, and Voigt profile. Used by voigtcomp.py to compute the normalized theoretical Voigt profile.

4.5 results

The results directory contains results derived from the output(s) of tests. This includes plots (e.g., comparing spectra) as well as numbers (e.g., the factor from test f04). It is similar in structure to code-output: each subdirectory is associated with the plots of the output of a particular RT/retrieval code, and these are numbered according to the time that the output was received.

The plots subdirectory contains plots of the output of multiple codes for the comparison and retrieval tests. Subdirectories within the plots subdirectory are named according to the codes that are being compared and what is being compared ('comparison-iso' or 'retrieval-inv'). For example, a comparison of 'BART', 'code', and 'othercode' for the c01hjcleariso comparison test would be located at BARTTest/results/plots/BART_code_othercode_c01hjcleariso/

4.6 tests

The tests/ directory contains the setups for each test. The subdirectory 00inputs/ contains the inputs used for tests. The remaining subdirectories are the tests; subdirectories named f##name are forward model tests, c##name are forward model comparison tests, s##name are synthetic retrieval tests, and r##name are real-data retrieval tests. Each subdirectory is detailed below. Note that paths in this section are given with respect to BARTTest/tests/.

The forward model tests are configured for Transit, the RT package of BART. The retrieval tests are configured for BART. These input and configuration files are provided so that the user may reference these for clarity on the exact setup for each test, allowing easy replication in other

RT packages. The following sections describe each file in each subdirectory. Note that other RT or retrieval codes may handle inputs differently, such as in the case of collision-induced absorptions.

In general, each forward model test will have a Transit configuration file (e.g., broadening_emission.trc), a Pylineread configuration file (e.g., broadening.plc), and an atmospheric model (e.g., broadening.atm). Some tests have multiple configuration files or atmospheric models, such as f05. For the synthetic retrieval tests, each directory will have two BART configuration files for transit and eclipse geometry (e.g., iso_uni_emission.brt), four text files containing the eclipse and transit depths and uncertainties (e.g., iso_ecldepths.txt and iso_ecluncs.txt), and a text file containing the paths to the filter files (e.g., iso_uni_filters.txt), with respect to the test directory.

Each test is executed over a specific wavenumber grid. Rather than list those here, we direct the Transit/BART configuration files. The parameters of interest are ‘wnlow‘ (or ‘wlhigh‘), ‘wnhigh‘ (or ‘wllow‘), and ‘wndelt‘; ‘wnlow‘ (or ‘wlhigh‘) sets the starting point of the grid, ‘wndelt‘ determines the step size between grid points, and ‘wnhigh‘ (or ‘wllow‘) sets the maximum possible grid value. Note that if ‘wnhigh‘ (or ‘wllow‘) does not correspond to the ‘wndelt‘ spacing, the last value in the grid will be the next-smallest value that does correspond to the spacing. For example, if ‘wnlow‘ is 100, ‘wnhigh‘ is 199.9, and ‘wndelt‘ is 0.25, then the grid would cover wavenumbers 100 – 199.75, inclusive.

4.6.1 00inputs

00inputs contains the input files used for Transit (the radiative transfer package of BART) to execute the tests.

New (fake) molecules are defined to be used in the f01 – f06 tests:

- CG1, similar to H2
- CG2, similar to He
- CG3, similar to N2
- LG1, similar to H2O
- LG2, similar to CH4
- LG3, similar to CO
- LG4, similar to CO2

Note that due to the similarity between these molecules, the user may use the real molecule in place of the fake one as far as calculations are concerned (mass, isotopes, isotope ratios, etc.). However, it is important to note that the line lists for these fake molecules do not match the real line lists of the real molecules; be sure to use the supplied .par line lists.

Files in 00inputs/:

- CIA_HITRAN_H2He_0200-9900K_0.5-500um.dat: An opacity table for collision-induced absorptions between H2 and He molecules.
- hd189733b-fp00k2odfnew.pck: Kurucz stellar model file for HD 189733-like star.
- HD189733b.tep: Transiting ExoPlanet file for HD 189733b. Contains system parameters.
- litran.dat: A modified version of the HITRAN definitions file used by the Pylineread module (reads line lists to build a file of line transitions) of Transit. Defines the fake molecules listed above.

- molecules.dat: A modified version of the molecule definitions file used by Transit. Defines the fake molecules listed above.

Subdirectories in 00inputs/:

- filters/: contains filter files used for the retrieval tests.
- par/: contains line lists for the fake molecules, and files used to pull HITRAN/HITEMP linelists from the HITRAN website.
- TLI/: contains the Transition Line Information files that Transit uses as input. Note that this is initially empty.

4.6.2 f01oneline

This test uses a model atmosphere that has one layer of 100% LG1, and all remaining layers 85% CG1 and 15% CG2. LG1 has only one line at 2.29 um. The test is executed in eclipse geometry.

Files in f01oneline/:

- oneline.atm: Atmospheric model used by Transit.
- oneline.plc: Configuration file for pylineread. Creates line list input file for Transit.
- oneline_emission.trc: Configuration file for Transit. Produces emission spectrum.

4.6.3 f02fewline

This test uses a model atmosphere that has one layer of 100% LG2, and all remaining layers 85% CG1 and 15% CG2. LG2 has three lines: 2.15 um, 2.29 um, 3.20 um. The test is executed in both eclipse and transit geometries.

Files in f02fewline/:

- fewline.atm: Atmospheric model used by Transit.
- fewline.plc: Configuration file for pylineread. Creates line list input file for Transit.
- fewline_emission.trc: Configuration file for Transit. Produces emission spectrum.
- fewline_transmission.trc: Configuration file for Transit. Produces transmission spectrum.

4.6.4 f03multiline

This test uses a model atmosphere that has one layer of 100% LG2, one layer of 100% LG3, one layer of 100% LG4, and all remaining layers 85% CG1 and 15% CG2. LG2 has lines as in f02fewline; LG3 has lines at 2.38 um, 2.50 um, and 2.54 um; LG4 has lines at 2.86 um, 3.02 um, and 3.78 um. The test is executed in both eclipse and transit geometries.

Files in f03multiline/:

- multiline.atm: Atmospheric model used by Transit.
- multiline.plc: Configuration file for pylineread. Creates line list input file for Transit.
- multiline_emission.trc: Configuration file for Transit. Produces emission spectrum.
- multiline_transmission.trc: Configuration file for Transit. Produces transmission spectrum.

4.6.5 f04broadening

This test looks at the accuracy of the Voigt profile shape for one line. One layer of the model atmosphere has 0.01% LG1 and 99.99% CG3, and all remaining layers 100% CG3. The layer with LG1 is at a pressure of 0.33516 bars and a temperature of 1442.58 K. The normalized theoretical Voigt profile is computed for this situation by calculating the half width at half max (HWHM) of a Gaussian due to thermal broadening and the HWHM of a Lorentzian due to pressure broadening. These are used as parameters for the Faddeeva function; the real part of this function is related to the Voigt profile. This is compared to the normalized opacity spectrum of the line as computed by Transit.

Files in f04broadening/:

- broadening.atm: Atmospheric model used by Transit.
- broadening.plc: Configuration file for pylineread. Creates line list input file for Transit.
- broadening_emission.trc: Configuration file for Transit. Produces emission spectrum.

4.6.6 f05abundance

This test explores the effect of varying the abundance of a molecule in an optically-thin situation. If the atmosphere is optically thin and the line is weak, then doubling the abundance should result in a line that is twice as intense.

This test uses a uniform atmosphere composed of x% LG1 and 100-x% CG1 at all layers. X ranges from 0% to 0.1% in steps of 0.01%. All 11 cases are executed in eclipse geometry.

Files in f05abundance/:

- abundance_line.plc: Configuration file for pylineread. Creates line list input file for Transit.
- abundance_*_emission.trc: Configuration files for Transit. * ranges from 0 to 1e-3 in steps of 1e-4.
- README: Brief readme about the test.

Subdirectories in f05abundance/:

- atm/: Directory containing the atmospheric models. Naming convention is, i.e. 1e-4_uniform.atm, where 1e-4 corresponds to the abundance of LG1.

4.6.7 f06blending

This test uses two lines in close proximity to look at the result of two lines blending together. There is a LG1 line at 2.28919 um and a LG2 line at 2.28921 um. LG1 and LG2 are only present in 1 layer of the atmospheric model.

Files in f06blending/:

- blending.atm: Atmospheric model used by Transit.
- blending.plc: Configuration file for pylineread. Creates line list input file for Transit.
- blending_emission.trc: Configuration file for Transit. Produces emission spectrum.

4.6.8 f07multicia

Transit incorporates collision-induced absorptions (CIAs) via a text file opacity table in a certain format. Other codes may also handle CIAs in a similar manner. This test uses an atmosphere uniformly composed of 85% H_2 and 15% He at all altitudes. The forward model is generated for three cases: no CIAs, H_2-He CIAs, and H_2-H_2 and H_2-He CIAs.

Files in f07multicia/:

- multicia.atm: Atmospheric model used by Transit.
- multicia.plc: Configuration file for pylineread. Creates line list input file for Transit.
- noCIA_emission.trc: Configuration file for Transit. Produces emission spectrum. No CIAs.
- oneCIA_emission.trc: Configuration file for Transit. Produces emission spectrum. H_2-He CIAs.
- twoCIA_emission.trc: Configuration file for Transit. Produces emission spectrum. H_2-H_2 and H_2-He CIAs.

4.6.9 f08isothermal

This test checks the case of isothermal emission. For isothermal emission, the result should be a blackbody spectrum corresponding to the temperature of the atmosphere. Full line lists are used for multiple molecules Note that real molecules and line lists are used for this test, not the fake molecules and line lists used in previous tests.

Files in f08isothermal/:

- isothermal.atm: Atmospheric model used by Transit.
- isothermal.plc: Configuration file for pylineread. Creates line list input file for Transit.
- isothermal_emission.trc: Configuration file for Transit. Produces emission spectrum.

4.6.10 f09energycons

This test runs forward models for a 60% H_2, 10% CO, 10% CO_2, 10% CH_4, and 10% H_2O non-inverted atmosphere at different resolutions, integrates each spectrum, and compares the values. Since the spectra are not sampled at true line-by-line resolution, there should be minor ($<$1%) differences.

Files in f09energycons/:

- energycons.atm: Atmospheric model used by Transit.
- energycons.plc: Configuration file for pylineread. Creates line list input file for Transit. Note: exactly matches isothermal.plc.
- energycons_*_emission.trc: Configuration files for Transit. Produces emission spectra. * indicates the number of steps made within a 1 cm^{-1} interval (default: 1, 2, 5, and 10 steps).

4.6.11 c01hjcleariso

This test generates forward models for an HD 189733b-like planet with an isothermal PT profile in both eclipse and transit geometry. The atmosphere contains H, He, C, N, O, H_2, CO, CO_2, CH_4, H_2O, and NH_3. Line lists are included for H_2, CO, CO_2, CH_4, H_2O, and NH_3. CIAs are included for H_2-H_2 and H_2-He.

Files in c01hjcleariso/:

- comparison.plc: Configuration file for Pylineread. Creates line list input file for Transit. Note: requires line lists. See Section 3.2 for information on downloading line lists if the user wishes to download them. Be aware that this will take some time and will use ∼15 GB of storage.
- iso_uni.atm: Atmospheric model used by Transit.
- iso_uni_emission.trc: Configuration file for Transit. Produces emission spectrum.
- iso_uni_transmission.trc: Configuration file for Transit. Produces transmission spectrum.
- pt_iso.png: Plot of the temperature–pressure profile.

4.6.12 c02hjclearnoinv

This test generates forward models for an HD 189733b-like planet with a non-inverted PT profile in both eclipse and transit geometry. The atmosphere contains H, He, C, N, O, H_2, CO, CO_2, CH_4, H_2O, and NH_3. Line lists are included for H_2, CO, CO_2, CH_4, H_2O, and NH_3. CIAs are included for H_2-H_2 and H_2-He.

Files in c02hjclearnoinv/:

- comparison.plc: Configuration file for Pylineread. Creates line list input file for Transit. Note: requires line lists. See Section 3.2 for information on downloading line lists if the user wishes to download them. Be aware that this will take some time and will use ∼15 GB of storage.
- make_noinv_uni.py: Script to produce the atmospheric model.
- noinv_uni.atm: Atmospheric model used by Transit.
- noinv_uni_emission.trc: Configuration file for Transit. Produces emission spectrum.
- noinv_uni_transmission.trc: Configuration file for Transit. Produces transmission spectrum.
- pt_noinv.png: Plot of the temperature–pressure profile.

4.6.13 c03hjclearinv

This test generates forward models for an HD 189733b-like planet with an inverted PT profile in both eclipse and transit geometry. The atmosphere contains H, He, C, N, O, H_2, CO, CO_2, CH_4, H_2O, and NH_3. Line lists are included for H_2, CO, CO_2, CH_4, H_2O, and NH_3. CIAs are included for H_2-H_2 and H_2-He.

Files in c03hjclearinv/:

- comparison.plc: Configuration file for Pylineread. Creates line list input file for Transit. Note: requires line lists. See Section 3.2 for information on downloading line lists if the user wishes to download them. Be aware that this will take some time and will use ∼15 GB of storage.
- make_inv_uni.py: Script to produce the atmospheric model.
- inv_uni.atm: Atmospheric model used by Transit.
- inv_uni_emission.trc: Configuration file for Transit. Produces emission spectrum.
- inv_uni_transmission.trc: Configuration file for Transit. Produces transmission spectrum.
- pt_inv.png: Plot of the temperature–pressure profile.

4.6.14 c04hjclearisoBarstowEtal

This test follows four of the setups described in Barstow et al. (2020). It includes three of the CO-only atmospheres and Model 0, an HD 189733b-like planet, in transit geometry.

Files in c04hjclearisoBarstowEtal/:

- BarstowEtal_CO_1e-4_1000K.trc: Configuration file for Transit, 1000 K atmosphere with CO abundance of 1e-4.
- BarstowEtal_CO_1e-4_1500K.trc: Configuration file for Transit, 1500 K atmosphere with CO abundance of 1e-4.
- BarstowEtal_CO_1e-5_1000K.trc: Configuration file for Transit, 1000 K atmosphere with CO abundance of 1e-5.
- BarstowEtal_model0.trc: Configuration file for Transit, Model 0.
- CO_1e-4_1000K.atm: Atmospheric model used by Transit, 1000 K atmosphere with CO abundance of 1e-4.
- CO_1e-4_1500K.atm: Atmospheric model used by Transit, 1500 K atmosphere with CO abundance of 1e-4.
- CO_1e-5_1000K.atm: Atmospheric model used by Transit, 1000 K atmosphere with CO abundance of 1e-5.
- model0.atm: Atmospheric model used by Transit, Model 0.
- pyline_BarstowEtal_CO.plc: Configuration file for Pylineread, CO-only cases.
- pyline_BarstowEtal_model0_H2O_CO.plc: Configuration file for Pylineread, Model 0.

4.6.15 c05hjcloudisoBarstowEtal

This test follows the Model 1 setup described in Barstow et al. (2020).

Files in c05hjcloudisoBarstowEtal/:

- BarstowEtal_model1.trc: Configuration file for Transit, Model 1.

4.6.16 s01hjcleariso

For this test, the isothermal emission and transmission spectra generated for c01hjcleariso were binned according to 47 filters spanning 1.97 – 4.95 µm(see BARTTest/tests/00inputs/filters/). The test is to perform a retrieval of atmospheric parameters and abundances using this eclipse/transit depth data and compare the result to the known inputs.

Note that for the isothermal emission retrieval, the retrieved abundances will likely not match the known inputs. This is because, in principle, the abundances do not affect the resulting spectrum. The important result is that the retrieval PT profile is isothermal within 3σ, preferably within 1σ.

Files in s01hjcleariso/:

- iso_uni_ecldepths.txt: Text file containing the eclipse depths corresponding to the 47 filters.
- iso_uni_ecluncs.txt: Text file containing the eclipse depth uncertainties corresponding to the 47 filters.
- iso_uni_emission.brt: BART configuration file to perform the retrieval. Isothermal emission case.
- iso_uni_filters.txt: Text file containing the paths to the filter files corresponding to the depths.
- iso_uni_tradepths.txt: Text file containing the transit depths.
- iso_uni_trauncs.txt: Text file containing the transit depth uncertainties.
- iso_uni_transmission.brt: BART configuration file to perform the retrieval. Isothermal transmission case.
- retrievals.plc: Configuration file for Pylineread. Creatse line list input file for Transit.

4.6.17 s02hjclearnoinv

For this test, the noninverted emission and transmission spectra generated for c02hjclearnoinv were binned according to 47 filters spanning 1.97 – 4.95 µm(see BARTTest/tests/00inputs/filters/). The test is to perform a retrieval of atmospheric parameters and abundances using this eclipse/transit depth data and compare the result to the known inputs.

Files in s02hjclearnoinv/:

- noinv_uni_ecldepths.txt: Text file containing the eclipse depths corresponding to the 47 filters.
- noinv_uni_ecluncs.txt: Text file containing the eclipse depth uncertainties corresponding to the 47 filters.
- noinv_uni_emission.brt: BART configuration file to perform the retrieval. Noninverted emission case.
- noinv_uni_filters.txt: Text file containing the paths to the filter files corresponding to the depths.
- noinv_uni_tradepths.txt: Text file containing the transit depths.
- noinv_uni_trauncs.txt: Text file containing the transit depth uncertainties.

- noinv_uni_transmission.brt: BART configuration file to perform the retrieval. Noninverted transmission case.
- retrievals.plc: Configuration file for Pylineread. Creatse line list input file for Transit.

Besides the above files, there are additional BART configuration files that were used for testing purposes. Rather than describe their differences here, users may compare the files to see how each differs from the 'main' configuration file described above. There are also associated files for the eclipse depths and uncertainties, named similarly to those above. For the outputs of these cases, see the Reproducible Research Compendium for BART1.

4.6.18 s03hjclearinv

For this test, the inverted emission and transmission spectra generated for c03hjclearinv were binned according to 47 filters spanning 1.97 – 4.95 μm(see BARTTest/tests/00inputs/filters/). The test is to perform a retrieval of atmospheric parameters and abundances using this eclipse/transit depth data and compare the result to the known inputs.

Files in s03hjclearinv/:

- inv_uni_ecldepths.txt: Text file containing the eclipse depths corresponding to the 47 filters.
- inv_uni_ecluncs.txt: Text file containing the eclipse depth uncertainties corresponding to the 47 filters.
- inv_uni_emission.brt: BART configuration file to perform the retrieval. Inverted emission case.
- inv_uni_filters.txt: Text file containing the paths to the filter files corresponding to the depths.
- inv_uni_tradepths.txt: Text file containing the transit depths. Order corresponds to the filter filters in 00inputs/filters/.
- inv_uni_trauncs.txt: Text file containing the transit depth uncertainties.
- inv_uni_transmission.brt: BART configuration file to perform the retrieval. Inverted transmission case.
- retrievals.plc: Configuration file for Pylineread. Creatse line list input file for Transit.

4.6.19 s04hjclearisoBarstowEtal

This test performs a synthetic retrieval on Model 0 of Barstow et al. (2020).

Files in s04hjclearisoBarstowEtal/:

- BarstowEtal_model0_chimera.brt: BART configuration file, for CHIMERA data.
- BarstowEtal_model0_nemesis.brt: BART configuration file, for NEMESIS data.
- BarstowEtal_model0_taurex.brt: BART configuration file, for Tau-REx data.

4.6.20 s05hjcloudisoBarstowEtal

This test performs a synthetic retrieval on Model 1 of Barstow et al. (2020).

Files in s05hjcloudisoBarstowEtal/:

- BarstowEtal_model1_chimera.brt: BART configuration file, for CHIMERA data.
- BarstowEtal_model1_nemesis.brt: BART configuration file, for NEMESIS data.
- BarstowEtal_model1_taurex.brt: BART configuration file, for Tau-REx data.

4.6.21 r01hd189733b

HD 189733b is one of the most observed exoplanets, with observations across the NIR and MIR. It is therefore a good testcase for retrieval algorithms as the plethora of data points allows for the constraining of some parameters.

This test uses space-based photometric and spectroscopic observations of the dayside emission of HD 189733b to perform a retrieval of atmospheric parameters and abundances. Data presented in the following publications are used:

- Charbonneau et al. 2008, ”The Broadband Infrared Emission Spectrum of the Exoplanet HD 189733b”
- Grillmair et al. 2008, ”Strong water absorption in the dayside emission spectrum of the planet HD189733b”
- Swain et al. 2009, ”Molecular Signatures in the Near Infrared Dayside Spectrum of HD 189733b”
- Agol et al. 2010, ”The Climate of HD 189733b from Fourteen Transit and Eclipses Measured by Spitzer”

Following the setup of Line et al. (2014, verified by M. Line priv. comm.), we use the following data for Spitzer IRAC channels 1 and 2:

- 0.001533 ± 0.000029
- 0.001886 ± 0.000071

Files in r01hd189733b/:

- HD189733b.brt: Configuration file for BART.
- HD189733b.plc: Configuration file for Pylineread. Creates line list input file for Transit. Note: requires line lists. See Section 3.2 for information on downloading line lists if the user wishes to download them. Be aware that this will take some time and will use ∼15 GB of storage.

For further testing, we include additional BART configuration files for this test case. For their associated output, see the Reproducible Research Compendium for BART1.

5 Example

This section will detail methods of running BARTTest. Since BARTTest is configured for Linux, all commands given are for that operating sytem.

First, the user must clone BARTTest from Github to their machine. Navigate in a terminal to the location where BARTTest will be cloned. Then type

```
git clone https://github.com/exosports/BARTTest BARTTest/
```

to clone it into a directory called BARTTest/ within your current directory. Then, navigate into the BARTTest directory.

```
cd BARTTest/
```

BARTTest comes with BART's code output and results already there (except the large input files, like line lists and opacity files). For this example, we will start fresh as if that is not the case. Type

```
make clean
```

into the terminal to delete all of the existing output from BART. Note that reproducing all of that output will take many, many CPU hours, so it will be easier to re-clone BARTTest than to run the code if the user wishes to undo the deletion of that output.

From here, the execution method will be dependent on what test(s) the user wishes to execute. The common step to (almost) all of the options is

```
make bart
```

as this command will clone and compile BART and its submodules from Github.

To run the forward modeling cases that do not require line lists (f01 – f07), type

```
make quicktests
```

into the terminal. It will execute f01 through f07 sequentially and produce plots of the output of those specific tests. Note that if the user deleted the existing BART output, there will be some error messages related to the files missing for other tests, like f08 and f09.

If the user wishes to run all of the forward modeling tests (f01 – f09), full line lists are required. Type

```
make linelists
```

to download most of the necessary line lists. Then, type

```
make forwardtests
```

into the terminal to run the tests. **Be aware of the hardware requirements for this** (see Section 3.1). This will take on the order of hours to execute.

If the user wishes to run only the comparison tests (c01 – c05), type

```
make comparisontests
```

into the terminal. This will take roughly as long as the previous option, as the comparison test is the most time-consuming part of the forward modeling tests. Note that full line lists are required.

To run all of the synthetic retrievals, type

```
make synthretrievals
```

into the terminal. **Be aware of the hardware requirements for this** (see Section 3.1). This will take on the order of a week to complete.

If the user wishes to run, e.g., the synthetic retrievals on a non-inverted atmosphere in only eclipse geometry, type

```
make noinv{\_}retrieval{\_}e
```

For additional options, see Table 1.

6 How to Use This Software to Test RT and Retrieval Codes

We would like you to contribute the results of your code to BARTTest!

BARTTest has been designed with one main goal in mind: to provide verification that RT and retrieval codes work as intended. To do so, there are numerous tests that are offered in the package, described above in Section 4.6.

To contribute, it is not absolutely necessary for all tests to be executed by some RT code. Rather, the comparison tests (see Sections 4.6.11 – 4.6.15) are the most important of the forward modeling tests due to the realistic setup; this best reflects the real cases RT codes are used for. Similarly, the HD 189733b case is the most important of the retrieval tests since it is based on real data. The more codes that yield the same (or similar) answer, the more confident we as a community can be that our codes are all implementing the same mathematics, and thus the more confident we can be in the results of our codes. If there are discrepancies between the results of different codes, the simpler forward modeling tests and synthetic retrieval tests may be able to diagnose the origin of those differences.

In order to contribute, the user must run their code for the exact same cases as BARTTest. The user may provide a file (in BARTTest/code-output/<yourcode>/) to re-format their code output to that of BART's output so that existing plotting functions may be used to produce plots of the results. Rather than attempting to explain this format, the user is encouraged to look at the output of BART in BARTTest/code-output/01BART/. If the user does not wish to reformat the output of their code to match this particular format, then they must include a file (in BARTTest/code-output/<yourcode>/) to produce plots of the results of their code (in BARTTest/results/<yourcode>/). This is to ensure that the plots provided can be easily reproduced.

Contributing results to BARTTest will follow a particular format. The code-output/ and results/ directories will have subdirectories for each RT code following a specific naming convention of ##code/ (e.g., 01BART/) as described in Section 4.2. The user may name the directory for their code output/results as they wish. The number preceeding the name of the code is determined by the order in which results are received via a pull request on Github; the user may pick a number but it may be changed depending on the order it is received (e.g., if two users submit results around the same time). If the user does not pick a brief name for their code, the authors will pick one as they see fit, whether based on the name of the code (e.g., BART) or based on the name of the author(s) if no code name exists. **Note that we do not ask the user to submit their RT/retrieval code itself! We only want the output of the code, as well as any other necessary files such as code used to convert the output to BART's format or code to produce plots of the output if it has not been re-formatted to BART's format.** Users that convert the output of their code to the same format as BART can use the existing plotting routines to produce plots of the results if they wish to save the maintainers of BARTTest some time. Note that if the re-formatted output is incorrect, the maintainers of BARTTest will not fix that and will leave the pull request open until the user fixes it.