


SpectralModel: a high-resolution framework for petitRADTRANS 3

Doriann Blain¹, Paul Mollière¹, and Evert Nasedkin¹

¹ Max Planck Institut für Astronomie, DE  Corresponding author

DOI: [10.xxxxxx/draft](https://doi.org/10.xxxxxx/draft)

Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Open Journals](#) 

Reviewers:

- [@openjournals](#)

Submitted: 01 January 1970

Published: unpublished

License

Authors of papers retain copyright,
and release the work under a
Creative Commons Attribution 4.0
International License ([CC BY 4.0](#)).

Summary

Atmospheric characterisation from spectroscopic data is a key to understand planetary formation. Two types of observations can be performed for this kind of analysis. Space-based observations (e.g., using the James Webb Space Telescope, JWST), are not impeded by the Earth's atmosphere, but are currently limited to low resolving powers (< 3000), which can lead to ambiguities in some species detections. Ground-based observations (e.g., using the Very Large Telescope, VLT), on the other hand, can benefit from large resolving powers ($\approx 10^5$), allowing for unambiguous species detection, but are impacted by telluric spectral lines. `petitRADTRANS` (pRT) is a radiative transfer package used for computing emission or transmission spectra of planetary atmospheres (Mollière et al., 2019). The package has a non-negligible user base, the original article being cited in 264 refereed works at the time of writing. pRT is already relatively easy to use on space-based, low-resolution observations. However, while the package technically has the capacity to analyse high-resolution spectra, thanks to its ability to incorporate high-resolution ($\mathcal{R} = 10^6$) line lists, ground-based observations analysis is a complex and challenging task. The new `SpectralModel` object provides a powerful and flexible framework that streamlines the setup necessary to model and retrieve high-resolution spectra.

Statement of need

Calculating a spectrum using pRT's core object `Radtrans` is a two-step process in which the user first instantiates the object, giving parameters that control the loading of opacities. The second step is for the user to call one of the `Radtrans` function, giving “spectral” parameters such as the temperatures or the mass fractions of the atmosphere, that will be used in combination with the loaded opacities to generate the spectrum.

However, these two steps are by themselves often insufficient to build a spectrum in a real-life scenario. The spectral parameters may individually rely on arbitrarily complex models requiring their own parameters, and may depend on each other. For example, getting mass fractions from equilibrium chemistry requires knowing the temperature profile, and the mean molar mass requires knowing the mass fractions (see e.g. the built-in pRT functions). Common operations such as convolving the spectrum, scaling it to stellar flux, or more specifically for high-resolution spectra, Doppler-shifting the spectrum and including the transit effect, must be done by post-processing the `Radtrans`-generated spectrum. Finally, using a retrieval requires to code a “retrieval model” including all the steps described above. This induces, especially for first-time users, a significant setup cost. The alternative is to use one of pRT's built-in models, but this lacks flexibility.

The `SpectralModel` object extends the base capabilities of the `petitRADTRANS` package by providing a standardized but flexible framework for spectral calculations. It has been especially designed to effectively erase the setup cost of modelling the spectral Doppler-shift, the transit effect, and of implementing the preparation step necessary for ground-based high-

42 resolution observations analysis. SpectralModel is also interfaced with pRT's retrieval
43 module (Nasedkin et al., 2024), and as such is an easy-to-use tool to perform both high- and
44 low-resolution atmospheric retrievals.

45 The combination of ease-of-use and flexibility offered by SpectralModel makes it a powerful tool
46 for high-resolution (but also low-resolution) atmospheric characterisation. With the upcoming
47 first light of a new generation of ground based telescopes, such as the Extremely Large
48 Telescope, SpectralModel makes petitRADTRANS ready for the new scientific discoveries
49 that will be unveiled in the next era of high-resolution observations.

50 The SpectralModel object

51 Main features

52 Spectral parameter calculation framework

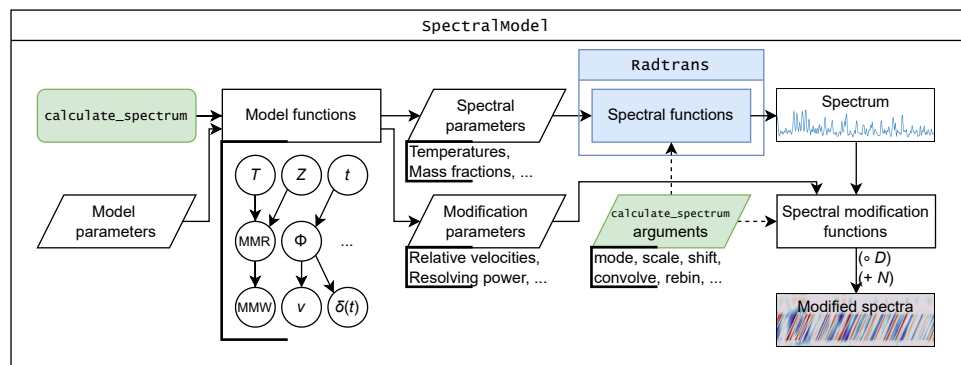


Figure 1: Flowchart of SpectralModel.calculate_spectrum function. The annotation below the model functions represents an example of execution order of these function after topological sorting, involving the temperature (T), the metallicity (Z), the time (t), the mass fractions (MMR), the mean molar masses (MMW), the orbital phases (ϕ), the relative velocities (v), and the transit effect (δ).

53 SpectralModel provides a framework to automatise the calculation of the spectral parameters.
54 Each spectral parameter is linked to a function, called here "model function", which calculates
55 its value. This feature can be extended to the parameters required for these functions, and
56 so on. Before calculating spectra, the function's execution order is automatically determined
57 through a topological sorting algorithm¹ (Kahn, 1962). SpectralModel comes with built-in
58 functions (Blain et al., 2024) for all the spectral parameters, so that the object can be used
59 "out-of-the box". Parameters that ultimately do not depend on any function are called "model
60 parameters", and must be given during instantiation.

61 In addition, SpectralModel provides built-in functions (Blain et al., 2024) to scale, convolve,
62 Doppler-shift, rebin, include planet transit effect, and prepare a spectrum after it has been
63 calculated. Similarly to model functions, these "spectral modification functions" must be given,
64 if used, their own model parameters during instantiation.

65 The spectral calculation is done within the calculate_spectrum function (see Figure 1). The
66 spectral mode (emission or transmission), as well as which of the spectral modification to
67 activate (i.e. only scaling, or both convolving and rebinning, etc.), are controlled through the
68 function's arguments ("spectral modification parameters").

¹Cyclic dependencies are not supported.

69 Automatic optimal wavelength range calculation

70 High-resolution spectra require high-resolution opacities, which, when loaded, can take a lot of
 71 Random-Access Memory (RAM). For example, loading pRT's 1H2-160__P0KAZATEL line-by-line
 72 line list ($\mathcal{R} = 10^6$) between 1 and 2 μm takes 804 MB² of RAM. Moreover, it is not unusual
 73 for a model to incorporate multiple species opacities. Fast retrievals in pRT are also performed
 74 in parallel, using multiple processes on distributed memory. The total RAM usage of the
 75 opacities is thus the amount of bytes taken by one species on the required wavelength range,
 76 times the number of species, time the number of processes. Using too many processes can
 77 therefore overload hardware RAM. To avoid out-of-memory errors, the number of processes
 78 must be limited, decreasing retrieval speed.

79 A way to slightly reduce this memory usage is to load exactly the wavelength range required
 80 for an analysis, instead of relying on manual inputs. This task is complicated in high-resolution
 81 retrievals due to parameters influencing the Doppler-shift (that is, the radial velocity semi-
 82 amplitude K_p , the rest frame velocity shift V_{rest} , and the mid transit time offset T_0) being
 83 retrieved. SpectralModel comes with a class method with_velocity_range, which takes into
 84 account the (uniform) prior range of these parameters to automatically calculate the optimal
 85 wavelength range to load.

86 Interface with pRT's retrieval module

87 The Retrieval object has been extended to support spectra with up to 3 dimensions, intended
 88 to be spectral order, exposure (time), and spectral pixel (wavelength). It now also has a class
 89 method that instantiates a Retrieval object from Data objects³. The Data object has also
 90 been extended in two ways: it now allows taking directly data as arrays, instead of requiring a
 91 file, and allows taking directly Radtrans (or by extension SpectralModel) objects, instead of
 92 generating a new one during a Retrieval instantiation.

93 In addition, SpectralModel's model parameters and spectral modification functions can be
 94 advantageously used to simplify the retrieval setup compared to Radtrans'. This removes the
 95 need for several steps:

- 96 ■ building the RetrievalConfig object, as this has been automated,
- 97 ■ declaring the fixed parameters, as all model parameters that are not retrieved parameters
 98 are *de facto* fixed parameters,
- 99 ■ writing the retrieval model function, as it is given by the SpectralModel itself.

100 Ground-based high-resolution spectra contain telluric and stellar lines that must be removed.
 101 This is usually done with a “preparing” pipeline (also called “detrrending” or “pre-processing”
 102 pipeline). To this end, a new retrieval.preparing sub-module has been implemented,
 103 containing the “Polyfit” pipeline (Blain et al., 2024) and the “SysRem” pipeline (Tamuz et al.,
 104 2005). To perform a retrieval when the data are prepared with “Polyfit”, the forward model
 105 must be prepared in the same way (Blain et al., 2024). This forward model preparation step
 106 can be activated when calculating a spectrum with SpectralModel.

107 Ground-based data simulation

108 Data (F) taken from ground telescopes can be expressed as $F = M_{\Theta} \circ D + N$ (Blain et
 109 al., 2024), where M_{Θ} is an exact model with true parameters Θ , D (“deformation matrix”)
 110 represents the combination of telluric lines, stellar lines, and instrumental deformations (pseudo-
 111 continuum, blaze function, ...), and N is the noise. The operator “ \circ ” represents the element-wise
 112 product. Telluric lines, noise, and other deformations can be included in a SpectralModel
 113 object. A time-varying airmass can be added as model parameter to better model the telluric
 114 lines. Finally, a command-line interface (CLI) with ESO's SKYCALC sky model calculator has
 115 been implemented, adapting the CLI provided on the ESO's website.

²According to numpy.ndarray.nbytes.

³pRT's retrieval module allows for the retrieval of a combination of datasets.

116 **Workflows**

117 Examples for these workflows are available in the pRT's documentation.

118 **Spectra calculation**

119 Calculating spectra with `SpectralModel` is done in two steps:

- 120 1. Instantiation: similarly to `Radtrans`, this step is done to load the opacities, and thus
121 requires the same parameter as a `Radtrans` instantiation. In addition, the user can
122 provide model parameters, that will give the spectral parameters and the modification
123 parameters. Finally, a custom dict can be given if the user desires to use different
124 functions than the built-in ones.
- 125 2. Calculation: spectral calculation is done with a unique function. The spectrum type
126 (emission or transmission), as well as modification flags (for scaling, Doppler-shifting,
127 etc.) are given as arguments.

128 **Retrievals**

129 Retrieving spectra with `SpectralModel` is done in seven steps:

- 130 1. Loading the data,
- 131 2. For high-resolution ground-based data: preparing the data,
- 132 3. Setting the retrieved parameters, this is done by filling a dict,
- 133 4. Setting the forward model, by instantiating a `SpectralModel` object,
- 134 5. Instantiating a `Data` object with the `SpectralModel` dedicated function,
- 135 6. Instantiating a `Retrieval` object from the previously built `Data` object(s),
- 136 7. Running the retrieval.

137 In addition, a new corner plot function, based on the corner package ([Foreman-Mackey, 2016](#)),
138 has been implemented to ease the representation of the retrieval results with this framework.

139 **The petitRADTRANS 3 update**

Test	pRT 2.7.7 time (s)	pRT 3.1.0 time (s)	pRT 2.7.7 RAM (MB)	pRT 3.1.0 RAM (MB)
Opacity loading, 'c-k'	3.2	0.9	–	–
Opacity loading, 'lbl'	6.3	0.4	–	–
Emission, 'c-k'	6.4	5.2	2428	1472
Emission, 'lbl'	7.8	4.4	3929	2643
Transmission, 'c-k'	1.2	0.6	992	757
Transmission, 'lbl'	6.6	3.1	3929	2230

- Times are measured using the `cProfile` standard library, from the average of 7 runs.
- "RAM": peak RAM usage as reported by the `tracemalloc` standard library.
- 'c-k': using correlated-k opacities (CH_4 and H_2O), from 0.3 to 28 μm .
- 'lbl': using line-by-line opacities (CO and H_2O), from 0.9 to 1.2 μm .
- All spectra calculations are done using 100 pressure levels. Emission scattering is activated in 'c-k' mode.
- Results obtained on Debian 12.5 (WSL2), CPU: AMD Ryzen 9 3950X @ 3.50 GHz.

140 Along with `SpectralModel`, major changes have been made to pRT. The changes focus on
141 optimisations (both for speed and RAM usage) for high-resolution spectra computing, but this
142 also impacts the correlated-k (low-resolution) part of the code (see [Table 1](#)). To speed-up
143 "input data" (opacities, pre-calculated equilibrium chemistry table, star spectra table) loading

times, pRT's loading system has been overhauled and the loaded files have been converted from a mix of ASCII, Fortran unformatted and [HDF5](#) files to HDF5-only. Opacities now also follow an extended [ExoMol database](#) naming and structure convention. The package's code has also been rationalised, clarified, and refactored. Finally, several quality-of-life features (e.g., missing requested opacities can be automatically downloaded from the project's [Keeper library](#), or the Planet object) have been implemented.

Acknowledgements

We thank the pRT users, who greatly helped improving the package by sharing their suggestions and reporting their issues.

References

- Blain, D., Sánchez-López, A., & Mollière, P. (2024). A formally motivated retrieval framework applied to the high-resolution transmission spectrum of HD 189733 b. *The Astronomical Journal*, 167(4), 179. <https://doi.org/10.3847/1538-3881/ad2c8b>
- Foreman-Mackey, D. (2016). Corner.py: Scatterplot matrices in python. *Journal of Open Source Software*, 1(2), 24. <https://doi.org/10.21105/joss.00024>
- Kahn, A. B. (1962). Topological sorting of large networks. *Commun. ACM*, 5(11), 558–562. <https://doi.org/10.1145/368996.369025>
- Mollière, P., Wardenier, J. P., Boekel, R. van, Henning, Th., Molaverdikhani, K., & Snellen, I. A. G. (2019). petitRADTRANS: A Python radiative transfer package for exoplanet characterization and retrieval. *Astronomy & Astrophysics*, 627, A67. <https://doi.org/10.1051/0004-6361/201935470>
- Nasedkin, E., Mollière, P., & Blain, D. (2024). Atmospheric retrievals with petitRADTRANS. *Journal of Open Source Software*, 9(96), 5875. <https://doi.org/10.21105/joss.05875>
- Tamuz, O., Mazeh, T., & Zucker, S. (2005). Correcting systematic effects in a large set of photometric light curves. *Monthly Notices of the Royal Astronomical Society*, 356(4), 1466–1470. <https://doi.org/10.1111/j.1365-2966.2004.08585.x>