


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Published: unpublished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 referred 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 line-by-line opacities ($\mathcal{R} = 10^6$), 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 to know the temperature profile, and the mean molar mass requires to know 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 on the `Radtrans`-generated spectrum in post-process. 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 preparing step necessary for ground-based high-

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

51 Main features

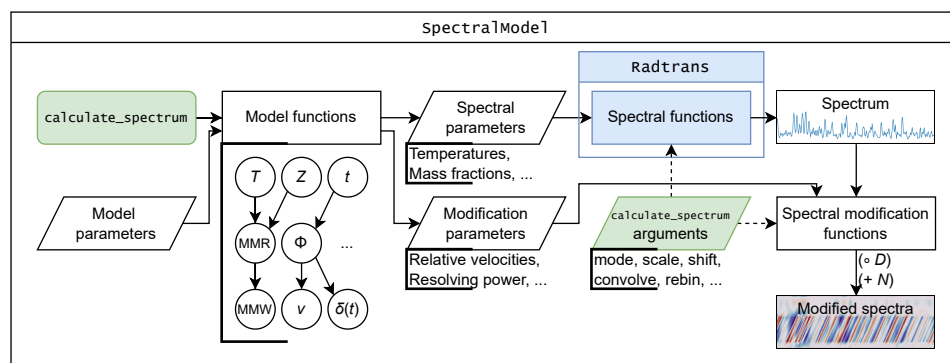
52 **Spectral parameter calculation framework**

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).

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

In addition, SpectralModel provides built-in functions (Blain et al., 2024) to scale, convolve, Doppler-shift, rebin, include planet transit effect, and prepare a spectrum after it has been calculated. Similarly to model functions, these “spectral modification functions” must be given, 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. Loaded opacities RAM usage is
 75 thus the amount of bytes taken by one species on the required wavelength range, times the
 76 number of species, time the number of processes. Using too many processes can thus overload
 77 hardware RAM, limiting retrieval speed.

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

85 Interface with pRT's retrieval module

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

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

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

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

106 Ground-based data simulation

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

²According to numpy.ndarray.nbytes.

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

115 **Workflows**

116 **Spectra calculation**

117 Calculating spectra with SpectralModel is done in two steps:

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

126 **Retrievals**

127 Retrieving spectra with SpectralModel is done in seven steps:

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

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

137 **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 (CH₄ and H₂O), from 0.3 to 28 μm.
- 'lbl': using line-by-line opacities (CO and H₂O), 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.

138 Along with SpectralModel, major changes have been made to pRT. The changes focus on
139 optimisations (both for speed and RAM usage) for high-resolution spectra computing, but this
140 also impacts the correlated-k (low-resolution) part of the code (see Table 1). To speed-up
141 “input data” (opacities, pre-calculated equilibrium chemistry table, star spectra table) loading
142 times, pRT’s loading system has been overhauled and the loaded files have been converted
143 from a mix of ASCII, Fortran unformatted and HDF5 files to HDF5-only. Opacities now also

144 follow an extended [ExoMol database](#) naming and structure convention. The package's code
145 has also been rationalised, clarified, and refactored. Finally, several quality-of-life features (e.g.,
146 missing requested opacities can be automatically downloaded from the project's [Keeper library](#),
147 or the Planet object) have been implemented.

148 Acknowledgements

149 Lorem.

150 References

- 151 Blain, D., Sánchez-López, A., & Mollière, P. (2024). A formally motivated retrieval framework
152 applied to the high-resolution transmission spectrum of HD 189733 b. *The Astronomical*
153 *Journal*, 167(4), 179. <https://doi.org/10.3847/1538-3881/ad2c8b>
- 154 Foreman-Mackey, D. (2016). Corner.py: Scatterplot matrices in python. *Journal of Open*
155 *Source Software*, 1(2), 24. <https://doi.org/10.21105/joss.00024>
- 156 Kahn, A. B. (1962). Topological sorting of large networks. *Commun. ACM*, 5(11), 558–562.
157 <https://doi.org/10.1145/368996.369025>
- 158 Mollière, P., Wardenier, J. P., Boekel, R. van, Henning, Th., Molaverdikhani, K., & Snellen,
159 I. A. G. (2019). petitRADTRANS: A Python radiative transfer package for exoplanet
160 characterization and retrieval. *Astronomy & Astrophysics*, 627, A67. [https://doi.org/10.](https://doi.org/10.1051/0004-6361/201935470)
161 [1051/0004-6361/201935470](https://doi.org/10.1051/0004-6361/201935470)
- 162 Nasedkin, E., Mollière, P., & Blain, D. (2024). Atmospheric retrievals with petitRADTRANS.
163 *Journal of Open Source Software*, 9(96), 5875. <https://doi.org/10.21105/joss.05875>
- 164 Tamuz, O., Mazeh, T., & Zucker, S. (2005). Correcting systematic effects in a large set of
165 photometric light curves. *Monthly Notices of the Royal Astronomical Society*, 356(4),
166 1466–1470. <https://doi.org/10.1111/j.1365-2966.2004.08585.x>