Journal of Quantitative Spectroscopy & Radiative Transfer 261 (2021) 107476

Contents lists available at ScienceDirect

Journal of Quantitative Spectroscopy & Radiative Transfer

journal homepage: www.elsevier.com/locate/jqsrt

A discrete integral transform for rapid spectral synthesis

D.C.M. van den Bekerom[a,1,*], E. Pannier[b]

[a] Department of Mechanical and Aerospace Engineering, The Ohio State University, Columbus, OH 43210, USA
[b] Laboratoire EM2C, CNRS UPR288, CentraleSupélec, Université Paris Saclay, 3 rue Joliot Curie, 91190 Gif sur Yvette

ARTICLE INFO

Article history:
Received 27 October 2020
Revised 8 December 2020
Accepted 8 December 2020
Available online 13 December 2020

Keywords:
Spectroscopy
Synthetic code
Data-analysis
Optimization

ABSTRACT

Accurate synthetic spectra that rely on large Line-By-Line (LBL)-databases are used in a wide range of applications such as high temperature combustion, atmospheric re-entry, planetary surveillance and laboratory plasmas. Conventionally synthetic spectra are calculated by computing a lineshape for every spectral line in the database and adding those together, which may take multiple hours for large databases. In this paper we propose a new approach for spectral synthesis based on an integral transform: the synthetic spectrum is calculated as the integral over the product of a Voigt profile and a newly proposed three-dimensional "*lineshape distribution function*", which is a function of spectral position and Gaussian- & Lorentzian width coordinates. A fast discrete version of this transform based on the Fast Fourier Transform (FFT) is proposed, which improves performance compared to the conventional approach by several orders of magnitude while maintaining accuracy. Strategies that minimize the discretization error are discussed. A Python implementation of the method is compared against state-of-the-art spectral code RADIS, and is since adopted as RADIS's default synthesis method. The synthesis of a benchmark CO_2 spectrum consisting of 1.8 M spectral lines and 200k spectral points took only 3.1 s using the proposed method (10^{11} lines $\times$ spectral points/s), a factor ~300 improvement over the state-of-the-art, with the relative improvement generally increasing for higher number of lines and/or number of spectral points. An experimental GPU-implementation of the method was also benchmarked, which demonstrated another 2~3 orders performance increase, achieving up to $5 \cdot 10^{14}$ lines $\times$ spectral points/s.

1. Introduction

1.1. The challenge of large spectral databases

The field of spectroscopy concerns itself with measuring and interpreting spectra to extract physical quantities of interest. A crucial step in this process is the computation of a synthetic spectrum that can be compared with the experimental spectrum to extract meaningful data. As available computational power has steadily grown in the past decades, so have the number and size of accurate Line-By-Line (LBL) spectral databases [1-7]. The database, together with operational parameters such as temperature and pressure, is supplied to a numerical procedure that computes the output synthetic spectrum. Applications for these synthetic spectra are numerous and diverse, including (high temperature) combustion [8], atmospheric re-entry [9], planetary surveillance (including Earth [10], Mars [11], and exo-planets [12,13]), and plasma discharges [14-16]. A compilation of many large databases can be found e.g. at the Virtual Atomic & Molecular Data centre (VAMDC) web-portal, which gathers various types of atomic and molecular line lists and data sources [7].

The current work was initially motivated by the need for fast synthetic spectral code for fitting FTIR-emission [14] and -absorption spectra [15,16] in non-equilibrium plasma discharges in CO_2. The recorded FTIR-spectra were broadband ($>400 cm^{-1}$), high resolution ($<0.1 cm^{-1}$), and time-resolved (StepScan) measurements, so fitting the experimental spectra required synthesizing numerous spectra with a large number of spectral points at every timestep. The high (vibrational) temperatures required the use of the HITEMP-database in favour of the much smaller HITRAN-database, significantly adding to the required computation time.

The size of a database is dictated by the number of lines that have a significant contribution at the conditions the database is intended for. As the temperature of a spectrum increases, the number of lines that must be included in a database to accurately reproduce experimental spectra increases dramatically. As an example let us consider commonly used databases for the CO_2-molecule: The HITRAN database [1] for CO_2, which is valid up to 700 K, consists of ~560k lines. The HITEMP database [2],

* Corresponding author.
E-mail address: dcmvdbekerom@gmail.com (D.C.M. van den Bekerom).
[1] Current address: Combustion Research Facility, Sandia National Laboratories, Livermore, CA 94550, USA.

https://doi.org/10.1016/j.jqsrt.2020.107476
0022-4073/

Glossary

$I(\nu)$	Spectrum to be synthesized (continuous)
i	Index of a single line/transition in the database
S_0^i	Line-strength of line i
ν_0^i, w_G^i, w_L^i	Spectral position, Gaussian width, and Lorentzian width, respectively, of line i
$S(\nu_0,w_G,w_L)$	Lineshape distribution function (continuous)
ν_0, w_G, w_L	Spectral position, Guassian width, and Lorentzian width coordinate (continuous)
$\hat{\nu}$	Spectral position coordinate in Fourier space
S_{klm}	Lineshape distribution matrix (discrete)
k, l, m	Index for gridpoint on the spectral position grid, Gaussian width-grid, or Lorentzian width-grid
k^i, l^i, m^i	Gridpoint immediately preceding the line parameter (resp. position, G-width, L-width)
δk, δl, δm	Boolean variable for selecting gridpoint preceeding- or succeeding line parameter
k', l', m'	Net gridpoint index ($k' = k^i + \delta k$)
ν_k, $\nu[k]$	Spectral coordinate (discrete)
$w_{G,l}$, $w_G[l]$	Guassian linewidth coordinate (discrete)
$w_{L,m}$, $w_G[m]$	Lorentzian linewidth coordinate (discrete)
α	Ratio of Lorentzian to Gaussian width w_L/w_G
d	Degree to which how Lorentzian/Gaussian a Voigt profile is, $d = \frac{w_L - w_G}{w_L + w_G} = \frac{\alpha-1}{\alpha+1}$; $d = -1$ for pure Gaussian and $d = 1$ for pure Lorentzian.
$a^i_{\delta k,\delta l,\delta m}$	Contribution to lineshape distribution function by line i (discrete)
a_ν^i, a_G^i, a_L^i	Weight for spectral position, Gaussian linewidth, or Lorentzian linewidth, for line i
t_ν^i, t_G^i, t_L^i	Grid alignment for spectral position, Gaussian linewidth, or Lorentzian linewidth, for line i
x_ν	Dimensionless spectral coordinate ($x_\nu \equiv \nu/w_V$)
ξ	Dimensionless spectral coordinate in Fourier space
x_G, x_L	Logarithmic width-grids
$\Delta\nu$	Spectral grid spacing
Δx_ν	Dimensionless spectral grid spacing ($\Delta x_\nu = \Delta\nu/w_V^i$), which is different for each line i
$\Delta x_{\nu,G}$	Dimensionless spectral grid spacing normalized by Gaussian width ($\Delta x_\nu = \Delta\nu/w_G^i$)
Δx_G, Δx_L	Grid spacing of the logarithmic Gaussian and Lorentzian width grids
n	Index for wavenumber axis of output spectrum /Index for Taylor series /Dummy variable
$\delta(x)$	Dirac delta; zero when $x \neq 0$, $\int \delta(x)dx = 1$
δ_{ij}	Kronecker delta; unity when $i = j$, zero otherwise
$g(\nu;\, w_G,w_L)$	Exact (Voigt) lineshape, centred at $\nu = 0$ (continuous)
$\hat{g}(\hat{\nu};\, w_G, w_L)$	FT of $g(\nu;\, w_G,w_L)$
$\tilde{g}(\nu;\, w_G, w_L)$	Approximation of $g(\nu;\, \mathrm{w_G},\mathrm{w_L})$
$G(x)$	Zero-centred, unity width, dimensionless lineshape
$\breve{G}(x)$	Approximation of $G(x)$ with approximated widths, but exact line position
$\tilde{G}(x)$	Approximation of $G(x)$ with both approximated widths and line position
$\hat{G}(\xi)$	FT of $G(x)$

valid up to 2000 K, already consists of ~11 M lines, while the CDSD-4000 database [3], valid up to 4000 K, consists of ~630 M lines. The problem of large line-by-line databases is even worse for larger molecules; uncompressed ab-initio databases such as ExoMol [4] or TheoReTS [5] contain up to billions of lines. Line databases for complex molecules can be particularly large, with the uncompressed database for Methane at high temperatures (3000 K) containing over 150 billion lines [6].

1.2. Optimization and reduction strategies

A number of optimization and line reduction strategies have been developed over the years to minimize the database size and maximize performance of synthetic spectral codes. In these calculations the usual performance bottleneck is the application of the physical lineshape broadening. Typical strategies to reduce the calculation times include reduction the number of lines, improving the calculation of the broadening itself, or using approximate models.

Because calculation of an exact Voigt lineshape is computationally expensive, the lineshape itself is often approximated by an empirical formula [17-19]. Another widely used and simple strategy is to truncate the lineshape at the wings where the intensity is "small". As we will see, this truncation may introduce significant approximation errors, which can be particularly large in high temperature spectra where the sum of many weak lines can add up to significant contributions. Computing the lineshape over different overlaying spectral grids, with a high resolution near the line position and a lower resolution on the band wings, is another strategy that seeks to minimize the calculation time of the lineshape. ExoCross [20] uses the Humlicek [21] approximation to calculate the Voigt broadening close to the line centre, and a reference Voigt profile that can be vectorized in the band wings, with a truncation at 25 cm^{-1}. This vectorized Voigt approach improves calculations by one order of magnitude. NASA's SMART-algorithm employs several nested grids to completely resolve narrow lines in the spectrum in their LBLABC model [22]. Some authors calculate the lineshapes in Fourier space [23,24], which allows separate calculations of the Lorentzian and Gaussian lineshapes. Tan [23] suggested an approach employing this separation based on binning and pre-computing Lorentzian lineshapes for a low resolution grid of widths, but the result showed significant discrepancies with their benchmarks.

An even simpler way to reduce calculation times is simply to reduce the number of spectral lines involved in the calculation. Rey et al. [5] proposed to separate the database into two separate groups of "strong" and "weak" lines in their TheoReTS database. While the strong lines would be processed normally, weak lines would be added together in a quasi-continuum, which was then converted into so-called "super-lines" with a constant spacing at a predetermined spectral resolution. Although this solution helped to significantly reduce the database size by a few orders of magnitude, the quasi-continuum created by the super-lines had to be re-evaluated for every temperature, making it somewhat cumbersome to implement in spectral codes.

Hargreaves et al. [25] addressed this issue by converting the "super-lines" into so-called "effective lines" by assigning an effective lower state energy, allowing these lines to be converted to any temperature. As a result these effective lines appear in the database as regular lines which can therefore be used as input for any spectral synthetic code without modification of the code. Even with this compression technique, the HITEMP CH_4 database still contains ~32 M lines. While the effective lines solve the performance issue to a certain extent, information on the constituent lines is still lost. Moreover, determining which lines of the original database may be compressed into effective lines still requires advance knowledge of the database and may involve arbitrary decisions on what lines are considered "weak" or "strong".

Alternatively, instead of treating weak lines individually they can be added to a single large (quasi-) continuum, that is not processed in the step where the lineshape is applied. The concept of the quasi-continuum has been suggested in both experimental and theoretical high temperature studies [26,27]. The quasi-continuum is also used by the NEQAIR software, which uses continuum precomputed from the CDSD-4000 [3] database to calculate CO_2 [9]. RADIS [28] also has the option to use a continuum to speedup computation, which identifies weak lines at runtime and adds them to a continuum that is not processed in the line-broadening step.

Another approach in radiative transfer makes use of LBL models combined with a Monte Carlo method and an appropriate wavenumber selection scheme to choose statistically meaningful wavenumbers [29]. A significant performance improvement is achieved, but the method is still limited by the performances of the LBL algorithm itself. Nonetheless, such schemes were used in the modelling of high-temperature oxy–natural gas combustion [8].

Finally, approximate models can be used to derive the emission and absorption coefficients many lines together. For instance, statistical narrow-band (SNB) and correlated-K (C-K) models [30] were developed from high-resolution line-by-line spectra. These models lead to many orders of magnitude of improvement and can be used in CFD simulations. However, they must be tabulated beforehand from high-resolution LBL spectra, for each of the thermodynamic conditions required. They also result in a loss of spectral information.

1.3. This work

While the line reduction strategies discussed here go a long way in improving performance, in many cases trade-offs between accuracy or convenience must be made, and the resulting database can still contain many millions of lines. It is therefore clear that there is a need for a high-performance spectral synthesis procedure that is compatible with these extensive databases.

In this paper we propose a novel approach that drastically reduces spectral computation times while at the same time producing high quality spectra, that is not predicated on the removal or compression of lines in the database. The approach is based on two observations, namely that 1.) A spectrum may be rapidly synthesized by convolution of a stick spectrum and lineshape function, as long as the lineshape is constant for all lines, and that 2.) Any lineshape may be accurately represented as the weighted sum of two or more other lineshapes, allowing lines to be distributed over several spectra that themselves each have a constant lineshape. More formally, the output spectrum is expressed as an integral transform that takes the spectral database as input parameters, and its implementation relies on convolutions to significantly reduce the number of lineshapes that need to be generated. Additionally, by expressing these convolutions as products in Fourier space, the Fast Fourier Transform can be leveraged to further accelerate spectral computation. The new method was implemented in the RADIS spectral modelling package [28] and is available since version 0.9.20.

Because the approach does not use approximate models for the broadening and retains all spectral lines, it achieves a high accuracy while potentially improving performance by several orders of magnitude. The approach is generic in the sense that it doesn't rely on any prior knowledge of the database, making it not only compatible with any existing spectral code, but also straightforward and easy to implement. This paper describes the details of the proposed procedure and compares the performance of a Python implementation to the optimized spectral modelling package RADIS [28]. The full details of the proposed method are presented in Section 2. A specific implementation is suggested in Section 3. Comparisons between RADIS's original and newly proposed synthesis methodsare presented in Section 4, including a comparison to an experimental GPU-implementation. Finally, in Section 5 we present conclusions and a short summary of the proposed method.

2. Proposed approach

2.1. Continuous transform

The calculation of synthetic spectra from a spectral database requires information of the line-position ν_0^i, the line strength S_0^i, and linewidths – which typically consists of a Gaussian width w_G^i and a Lorentzian width w_L^i, where the superscript i indicates a quantity belonging to a particular line i. The conventional procedure for calculating synthetic spectra is to first define a spectral axis ν and then for each line calculate the normalized lineshape $g(\nu - \nu_0^i; w_G^i, w_L^i)$, multiplied by the line strength S_0^i, which are then added to the output spectrum $I(\nu)$. By doing this for every line in the database, the full synthetic spectrum is calculated:

$$I(\nu) = \sum_i S_0^i g\left(\nu - \nu_0^i; w_G^i, w_L^i\right) \tag{2.1}$$

Under certain conditions, the lineshape $g(\nu)$ will be identical for every line in the database. This describes e.g. an emission spectrum where broadening is dominated by the instrumental function of the spectrometer in good approximation. In such a case, the spectrum may be synthesized by taking the convolution of a lineshape function with constant Gaussian width w_G^0 and Lorentzian width w_L^0, and a stick spectrum $S(\nu_0)$:

$$I(\nu) = \int S(\nu_0) g\left(\nu - \nu_0; w_G^0, w_L^0\right) d\nu_0, \tag{2.2}$$

where the spectral coordinate ν_0 is now a continuous axis, and the stick spectrum $S(\nu_0)$ can be populated by summing delta peaks at the line position ν_0^i and line strength S_0^i corresponding to each line i:

$$S(\nu_0) = \sum_i S_0^i \delta\left(\nu_0 - \nu_0^i\right). \tag{2.3}$$

Here $\delta(x)$ is the Dirac delta function which is zero when $x \neq 0$, and $\int \delta(x) dx = 1$. In general however, the lineshape may be different for every line i. Eq. (2.2) can easily be extended to reflect this, by adding two more continuous axes for the Gaussian and Lorentzian widths; w_G and w_L. We therefore propose to express the synthesis of spectrum $I(\nu)$ more generally as the following integral transform:

$$I(\nu) = \iiint S(\nu_0, w_G, w_L) g(\nu - \nu_0; w_G, w_L) d\nu_0 dw_G dw_L \tag{2.4}$$

where $g(\nu;\ w_G,w_L)$ is a Voigt profile centred at $\nu = 0$ and the transform consists of a convolution over the line position ν_0 and integrals over the Gaussian and Lorentzian widths w_G and w_L. The spectral information is contained in the 3-dimensional quantity $S(\nu_0,w_G,w_L)$, which we will designate the *lineshape distribution function*. It may be thought of as a generalization of the stick spectrum in Eq. (2.3), and is given by:

$$S(\nu_0, w_G, w_L) \equiv \sum_i S_0^i \delta\left(\nu_0 - \nu_0^i\right) \delta\left(w_G - w_G^i\right) \delta\left(w_L - w_L^i\right), \tag{2.5}$$

By comparing Eqs. (2.4) and (2.5) to Eq. (2.1), it can be seen that the produced outputs $I(\nu)$ are identical to each other. Here $S(\nu_0,w_G,w_L)$ may represent any spectrum like e.g. absorbence, emissivity, scattering, etc., by substituting S_0^i with the corresponding quantity.

While we will continue to focus on spectra with Voigt-profiles in the rest of the paper, the procedure may in principle be extended to include as many lineshape parameters as are needed to

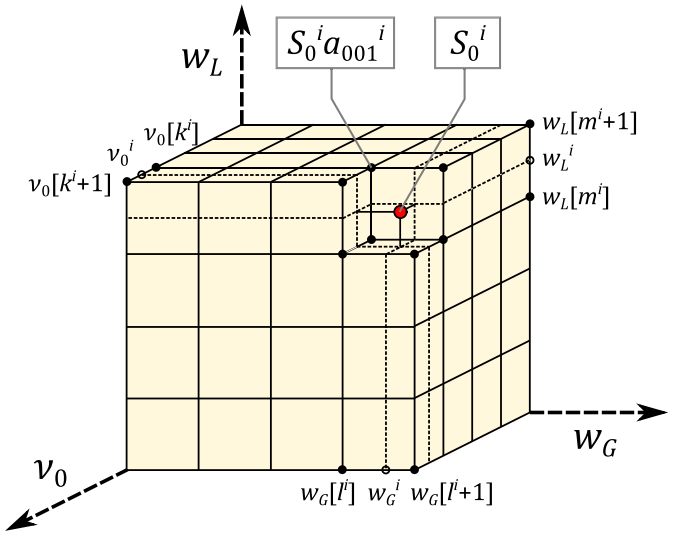

Fig. 1. the spectral information $\boldsymbol{S^i(\nu_0^i, w_G^i, w_L^i)}$ that is added to the 3D spectral matrix for each line $\boldsymbol{i}$ generally does not line up with the spectral position- & linewidth-grids, and must therefore be distributed over the $2 \times 2 \times 2 = 8$ nearest neighbour gridpoints.

accurately represent the physical lineshape, e.g. by using Speed-dependant Voigt-profiles [31] (4 parameters), or Hartmann-Tran profiles [32] (7 parameters) instead of the Voigt-profile. It should be noted however that since the dimensionality of S corresponds directly to the number of parameters, adding more parameters will significantly increase both the size of S and the computational complexity, which may defeat the purpose of the method beyond a certain number of parameters.

Traditional numerical implementations of Eq. (2.1) only use a numerical grid for discretization of the wavenumber parameter, but not for other parameters. By implementing the spectral synthesis according to Eqs. (2.4)-(2.5), the Gaussian and Lorentzian width can also be discretized, allowing for an arbitrary accuracy of the synthetic spectrum by controlling the respective grid-sizes. The computational complexity of the suggested procedure differs from the traditional implementation in that it decouples the processing of the lineshape synthesis (Eq. (2.4)) and processing of the database lines (Eq. (2.5)), which can clearly be beneficial for a large number of spectral points or lines in the database.

2.2. Discrete transform

In order for the integral transform to be applicable to real-world synthetic code, a discrete version of the transform must be developed. Eq. (2.4) may be discretized by replacing the integrals with summations:

$$I_n = \sum_k \sum_l \sum_m S_{klm} g\left(\nu_n - \nu_k; w_{G,l}, w_{L,m}\right) \tag{2.6}$$

The remaining task for formulating the discrete transform is to find an accurate discretization of Eq. (2.5) which would yield the 3D *lineshape distribution matrix* S_{klm}. In the continuous case, each line i adds a single point of magnitude S_0^i to the lineshape distribution function $S(\nu_0,w_G,w_L)$. In the discrete version, such point will generally not line up with the discrete 3D-grid and will have to be distributed to the $2 \times 2 \times 2 = 8$ nearest neighbour gridpoints for each spectral line, as shown in Fig. 1. We label the coordinates of these gridpoints as follows:

$$k' = k^i + \delta k \tag{2.7a}$$

$$l' = l^i + \delta l \tag{2.7b}$$

$$m' = m^i + \delta m \tag{2.7c}$$

$$\delta k, \delta l, \delta m = 0, 1 \tag{2.7d}$$

where k^i, l^i, and m^i are the indices of the gridpoints of the ν_0,w_G, and w_L-grids, respectively, that are immediately below the line parameters ν_0^i, w_G^i, and w_L^i. The variables δk, δl, and δm are either 0 or 1, to indicate whether the gridpoint k', l', or m' is respectively directly below or directly above the line parameter.

For every line i the linestrength S_0^i must be distributed to each of the 8 gridpoints. This distribution is implemented by multiplying S_0^i by matrix $a^i_{\delta k,\delta l,\delta m}$ which describes the relative contribution of the gridpoint at (k', l', m'), before adding it to S_{klm}. The $2 \times 2 \times 2$ matrix $a^i_{\delta k,\delta l,\delta m}$ is given by:

$$a^i_{\delta k,\delta l,\delta m} \equiv f_{\delta k}\left(a_\nu^i\right) f_{\delta l}\left(a_G^i\right) f_{\delta m}\left(a_L^i\right), \quad f_n(a) \equiv \begin{cases} 1-a & \text{if } n = 0 \\ a & \text{if } n = 1 \end{cases} \tag{2.8}$$

where a_ν^i, a_G^i, and a_L^i are weights that are related to their relative position in the 3D grid with a presumed value between 0 and 1, and may be found by interpolation of respectively the spectral positions, Gaussian widths, and Lorentzian widths. These weights are in principle a function of the exact coordinates ν_0^i, w_G^i, and w_L^i, and should be assigned values that minimize the discretization error. Proposed closed form expressions for these weights are discussed in Section 3. Regardless of the particular choice of weights however, by its definition the lineshapes always remain normalized, i.e.:

$$\sum_{\boldsymbol{\delta k},\boldsymbol{\delta l},\boldsymbol{\delta m}=0,1} \boldsymbol{a^i_{\delta k,\delta l,\delta m}} = 1. \tag{2.9}$$

Finally, the lineshape distribution matrix S_{klm} can now be calculated using:

$$S_{klm} = \sum_i \sum_{\delta k,\delta l,\delta m=0,1} S_0^i a^i_{\delta k,\delta l,\delta m} \delta_{k,k'} \delta_{l,l'} \delta_{m,m'} \tag{2.10}$$

where $\delta_{i,j}$ is the Kronecker delta which is unity when $i = j$, and zero otherwise, and which should not be confused with the indices of summation δk, δl, and δm. The spectrum can then be calculated by plugging S_{klm} into Eq. (2.6).

2.3. Line integrated spectra

The integral transform approach can yield additional performance increase when applied to the synthesis of line-integrated spectra, which may generally be represented as:

$$I(\nu) = \int I_z(\nu) dz, \tag{2.11}$$

which is valid for both absorption and optically thin emission. In terms of the proposed integral transform, a line integrated spectrum can be expressed as follows:

$$I(\nu) = \int \left[\iiint S_z(\nu_0, w_G, w_L, z) g(\nu - \nu_0; w_G, w_L) d\nu_0 dw_G dw_L \right] dz \tag{2.12}$$

But since the lineshape function $g(\nu - \nu_0; w_G,w_L)$ itself is not a function of z, the order of integration can be swapped, yielding:

$$I(\nu) = \iiint \left[\int S_z(\nu_0, w_G, w_L, z) dz \right] g(\nu - \nu_0; w_G, w_L) d\nu_0 dw_G dw_L \tag{2.13}$$

In other words, only the lineshape distribution function $S_z(\nu_0,w_G,w_L,z)$ needs to be integrated, meaning no additional lineshapes have to be synthesized regardless of the number of axial points, which can yield a significant additional performance increase when synthesizing line integrated spectra.

3. Suggested implementation

In the previous section the general aspects of the integral transform were treated. In the current section we will discuss implementation details for maximum performance and accuracy. Concretely, we discuss how the convolution can be replaced by the Fast Fourier Transform (FFT) and suggest closed form expressions for the weights encountered in Eq. (2.8). These implementation suggestions will generally lead to high performance and accuracy, but may not be generally applicable or result in the best performance for each and every case. As an example, when the number of spectral points is small, replacing the convolution with the FFT will in fact reduce performance.

3.1. FFT for convolution

The integral transform relies on a convolution over the spectral coordinate, which may be expressed as a product in Fourier space. By doing so, the Fast Fourier Transform (FFT) [33] can be leveraged for calculating the spectrum. After replacing the convolution, the integral transform may be expressed as:

$$I_n = \sum_l \sum_m \mathfrak{F}^{-1}\left\{\mathfrak{F}[S_{klm}] \cdot \mathfrak{F}\left[g\left(\nu_k; w_{G,l}, w_{L,m}\right)\right]\right\} \tag{3.1}$$

where the FT is over the spectral axis with index k. Because the lineshape function is known in advance, its FT may be calculated directly. In the case of a Voigt profile this becomes:

$$\begin{aligned}\hat{g}\left(\hat{\nu}; w_G, w_L\right) &\equiv \mathfrak{F}[g(\nu; w_G, w_L)] \\ &= \frac{1}{\Delta\nu} \exp\left\{-\left[\frac{\left(\pi\,\hat{\nu} w_G\right)^2}{4\ln 2} + \pi\,\hat{\nu} w_L\right]\right\}\end{aligned} \tag{3.2}$$

where by choosing this definition, w_G and w_L are now defined as FWHM-widths. Moreover, due to the linearity of the FT, the inverse transform may be placed outside of the summations, further reducing the number of FT's needed:

$$I_n = \mathfrak{F}^{-1}\left\{\sum_l \sum_m \mathfrak{F}[S_{klm}] \cdot \hat{g}\left(\hat{\nu}_k; w_{G,l}, w_{L,m}\right)\right\} \tag{3.3}$$

Eq. (3.3) can now be combined with (2.8) and (2.10) to yield a fast numerical implementation of the discrete transform.

When using the FT for convolution calculations without modification, the operation will result in a circular convolution. In order to implement a linear convolution instead, the S_{klm} matrix must be padded with N_ν zeros [34,35] along the k-axis, for a total size of $2N_\nu$. By employing the FFT, the total computational complexity may now be expressed as $c_1 N_i + c_2(1 + N_G N_L) N_\nu \log N_\nu$, where N_G and N_L are the number of grid-points for the Gaussian and Lorentzian width axes and c_1 and c_2 some numerical constants. Provided N_G and N_L can be kept small, it is thus possible to achieve a performance that is several orders of magnitude smaller than the more conventional $N_i \times N_\nu$ scaling. Typical realistic values for grid-sizes are $N_G = 2 \sim 4$ and $N_L = 8 \sim 12$ for high accuracy spectra.

Another benefit of using the FT is that the lineshapes are always normalized, even when the linewidths are similar to- or smaller than the grid spacing. Although the FT returns truthful values whenever $\Delta\nu \ll w_V^i$ and $\Delta\nu \gg w_V^i$, discretization errors in the form of ringing are introduced whenever $\Delta\nu \approx w_V^i$. Apart from the obvious solution of decreasing the spectral grid-spacing $\Delta\nu$, this ringing can be resolved by convolving the spectrum with an instrument function that has a width much larger than the linewidth $w_{instr} \gg w_V^i$.

Finally, expressing the convolution as an FT has a major additional benefit: The Voigt profile no longer has to be approximated but is directly given as the convolution of Gaussian and Lorentzian lineshapes, as also recognized by Mendenhall [24]. The implication is that by choosing smaller grid-spacings of the width-grids, the user is able to approximate the Voigt profile with arbitrary accuracy. This is in contrast with most contemporary Voigt-approximations [17-19], where the approximation accuracy is fixed.

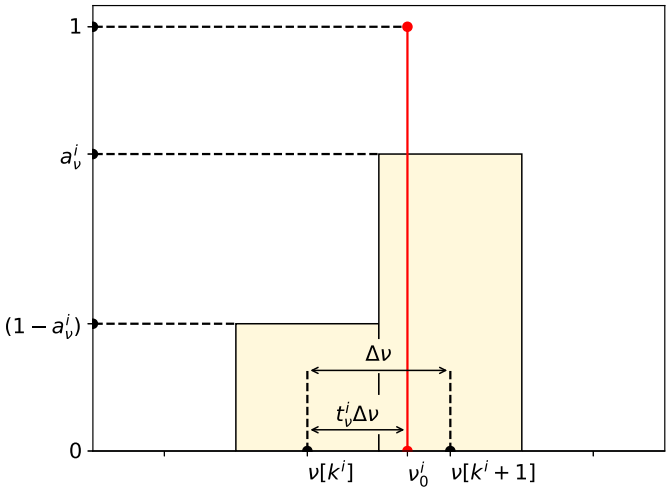

Fig. 2. When the spectral data is discretized, it is distributed over the two nearest neighbours. The weight a_ν^i directly determined by the grid-alignment t_ν^i of the spectral line at position ν_0^i.

3.2. Weights and error estimations

3.2.1. Grid-alignment and simple weights

In most cases, the exact line position ν_0^i will fall somewhere between the two spectral gridpoints with indices k^i and $k^i + 1$. To describe the relative position in the grid we introduce the grid-alignment t_ν^i that is zero when $\nu_0^i = \nu[k^i]$, unity when $\nu_0^i = \nu[k^i + 1]$, and for any other value of ν_0^i is given by:

$$t_\nu^i \equiv \frac{\nu_0^i - \nu\left[k^i\right]}{\nu\left[k^i + 1\right] - \nu\left[k^i\right]} \tag{3.4}$$

This relation is shown schematically in Fig. 2. While the grid may in principle have a grid-spacing that varies with index k, it is advantageous to keep the grid spacing constant, i.e. $\nu[k+1] - \nu[k] = \Delta\nu$. When a constant grid spacing $\Delta\nu$ is used, the spectral coordinate at gridpoint k may be calculated directly by:

$$\nu[k] = \nu[0] + k\Delta\nu, \tag{3.5}$$

where $\nu[0]$ is the first (smallest) spectral position in the grid. Moreover, with constant spacing the grid-index k^i corresponding to a line i having exact position ν_0^i can be calculated directly:

$$k^i = \left\lfloor \frac{\nu_0^i - \nu[0]}{\Delta\nu} \right\rfloor, \tag{3.6}$$

where $\lfloor \bullet \rfloor$ is the floor operator that returns the integer part of the argument.

The grid-alignment for the Gaussian and Lorentzian width-grids are defined in a similar fashion, but with the difference that we first take the logarithm of the widths, i.e.:

$$t_G^i \equiv \frac{\ln\left(w_G^i\right) - \ln\left(w_G\left[l^i\right]\right)}{\ln\left(w_G\left[l^i + 1\right]\right) - \ln\left(w_G\left[l^i\right]\right)}, \tag{3.7a}$$

$$t_L^i \equiv \frac{\ln\left(w_L^i\right) - \ln\left(w_L\left[m^i\right]\right)}{\ln\left(w_L\left[m^i + 1\right]\right) - \ln\left(w_L\left[m^i\right]\right)} \tag{3.7b}$$

Taking the logarithm of the widths can be rationalized in two ways: The first is that since the range of widths encountered in a spectrum can span multiple orders of magnitude, using a width-axes with a logarithmic scaling is an efficient way to cover all the widths in the spectrum. The second is that it makes it easier to estimate the discretization error, which is explored in more detail in the following sections. When a logarithmic width-grid is used with a constant grid-spacing, the width of consecutive grid-points increases by a constant factor. In such a case, the width-axes are given by:

$$x_G[l] \equiv \ln\left(\frac{w_G[l]}{w_G[0]}\right) = l\Delta x_G \;\leftrightarrow\; w_G[l] = w_G[0]\exp\left(l\Delta x_G\right), \tag{3.8}$$

$$x_L[m] \equiv \ln\left(\frac{w_L[m]}{w_L[0]}\right) = m\Delta x_L \;\leftrightarrow\; w_L[m] = w_L[0]\exp\left(m\Delta x_L\right), \tag{3.9}$$

where $x_G[l]$ and $x_L[m]$ define the logarithmic width-axes, and Δx_G and Δx_L are the constant grid spacings of the logarithmic width-grids. In such a case, the grid-indices l^i and m^i may again be calculated directly by:

$$l^i = \left\lfloor \frac{\ln\left(w_G^i\right) - \ln\left(w_G[0]\right)}{\Delta x_G} \right\rfloor, \tag{3.10a}$$

$$m^i = \left\lfloor \frac{\ln\left(w_L^i\right) - \ln\left(w_L[0]\right)}{\Delta x_L} \right\rfloor \tag{3.10b}$$

Equipped with expressions for the grid-alignment for each of the three grids t_ν^i, t_G^i, and t_L^i, we now return to the weights encountered in chapter 2, Eq. (2.6). Taking the spectral axis as example, the weight a_ν^i was defined such that a fraction $(1-a_\nu^i)$ of the line is added at index k^i, and a fraction a_ν^i of the line is added at index k^i+1. Intuitively, a_ν^i should thus be zero at k^i, unity at k^i+1, and increase with the distance from $\nu[k^i]$ – which is exactly the behaviour of the alignment t_ν^i. As a first approximation we may therefore use the grid-alignments directly as weights:

$$a_\nu^i = t_\nu^i \tag{3.11a}$$

$$a_G^i = t_G^i \tag{3.11b}$$

$$a_L^i = t_L^i \tag{3.11c}$$

We will refer to this choice of weights as "simple weights" for the remainder of the paper. In order to determine how well these weights discretize the lineshape distribution function, an error measure must be defined which can then be expressed it in terms of these weights.

3.2.2. Estimating the discretization error

In this section we will estimate the contribution to the overall discretization error by a single isolated line for a given set of weights. To facilitate the error analysis, it is convenient to work in dimensionless quantities. The Voigt-profile $g_V(\nu;\, w_G,w_L)$ takes three quantities with unit [cm^{-1}] as input and returns the line profile in units of [cm]. For the error estimation we will be considering the dimensionless Voigt-profile with unity width $G_V(x_\nu;\, \alpha)$ which can be obtained by multiplying the ordinary profile $g_V(\nu;\, w_G,w_L)$ with the FWHM of the Voigt function w_V and dividing the spectral axis by w_V:

$$G_V(x_\nu;\alpha) \equiv w_V g_V(\nu; w_G, w_L), \tag{3.12a}$$

$$x_\nu \equiv \frac{\nu}{w_V}, \tag{3.12b}$$

$$\alpha \equiv \frac{w_L}{w_G} \tag{3.12c}$$

The dimensionless Voigt profile described above may be used to calculate the contribution to the discretization error by a single isolated line. Before we can determine the discretization error however, we must define what is precisely meant by "the" error. While a definition of the error may in principle be chosen arbitrarily, we chose to use the RMS-error, which we define as:

$$E_{RMS}{}^2 \equiv \int \left|\tilde{G}_V(x_\nu) - G_V(x_\nu)\right|^2 dx_\nu, \tag{3.13}$$

where $\tilde{G}_V(x_\nu)$ is the approximated line profile and $G_V(x_\nu)$ the exact line profile. Deriving the error directly from the Voigt profiles is difficult, as they involve calculating convolutions of approximating lineshape functions. By choosing a squared error measure, Plancherel's theorem [34] allows the error-analysis to be done entirely in Fourier space where convolutions translate simply to products, thus greatly simplifying the error-analysis.

With the definition of the error in Eq. (3.13), an analytical expression can be derived that estimates the contribution to the discretization error by a single isolated line. Full details of the derivation are given in the appendix, here we will just present the result:

$$\begin{aligned} &E_{RMS}(a_\nu, a_G, a_L, t_\nu, t_G, t_L, \Delta x_\nu, \Delta x_G, \Delta x_L, \alpha)^2 \\ &\quad\approx c_{L1}^2 S_{L1L1}^\circ + 2c_{L1}c_{L2}S_{L1L2}^\circ + c_{L2}^2 S_{L2L2}^\circ + 2c_{G1}c_{L1}S_{G1L1}^\circ \\ &\quad + 2c_{G1}c_{L2}S_{G1L2}^\circ + 2c_{G2}c_{L1}S_{G2L1}^\circ \\ &2c_{G2}c_{L2}S_{G2L2}^\circ + c_{G1}^2 S_{G1G1}^\circ + 2c_{G1}c_{G2}S_{G1G2}^\circ + c_{G2}^2 S_{G2G2}^\circ \\ &\quad + 2c_{G1}c_{\nu 2}S_{G1\nu 2}^\circ + 2c_{G2}c_{\nu 2}S_{G2\nu 2}^\circ \\ &\quad + 2c_{L1}c_{\nu 2}S_{L1\nu 2}^\circ + 2c_{L2}c_{\nu 2}S_{L2\nu 2}^\circ + c_{\nu 1}^2 S_{\nu 1\nu 1}^\circ + c_{\nu 2}^2 S_{\nu 2\nu 2}^\circ \end{aligned} \tag{3.14}$$

where the "constants" c_{Xn} are constant with respect to α and depend only on a_X, t_X, and Δx_X for $X = \nu, G, L$, and are given by:

$$c_{X1}(a_X, t_X, \Delta x_X) = (a_X - t_X)\Delta x_X, \tag{3.15}$$

$$c_{X2}(a_X, t_X, \Delta x_X) = \frac{1}{2}\left({t_X}^2 - 2a_X t_X + a_X\right){\Delta x_X}^2. \tag{3.16}$$

The error integrals $S_{XiYj}^\circ(\alpha)$ depend only on α and are given by Eqs. (A.31) and (A.32) in the appendix. Having an analytical expression for the discretization error allows us to analytically derive optimized weights, which will be discussed in the next section.

3.2.3. Optimized weights

The optimized weights can now be found by setting the derivatives of E_{RMS} to the weights to zero and solving for a_ν, a_G and a_L:

$$\frac{\partial E_{RMS}}{\partial a_\nu} = 0, \quad \frac{\partial E_{RMS}}{\partial a_G} = 0, \quad \frac{\partial E_{RMS}}{\partial a_L} = 0 \tag{3.17}$$

This set of equations was solved by the symbolic mathematics package *Sympy*. The resulting expressions for a_G and a_L were expanded into individual terms, yielding hundreds of terms per weight. After eliminating every Δx_X-term of order 2 or higher, all but a few terms disappear and we achieve our final result:

$$a_\nu^i = t_\nu^i, \tag{3.18a}$$

$$a_G^i = t_G^i + \frac{R_{G\nu}t_\nu^i\left(t_\nu^i-1\right)\Delta {x_{\nu,G}^i}^2 + R_{GG}\left(\alpha^i\right)t_G^i\left(t_G^i-1\right)\Delta x_G^2 + R_{GL}\left(\alpha^i\right)t_L^i\left(t_L^i-1\right)\Delta x_L^2}{2\Delta x_G}, \tag{3.18b}$$

$$a_L^i = t_L^i + \frac{R_{LG}\left(\alpha^i\right)t_G^i\left(t_G^i-1\right)\Delta x_G^2 + R_{LL}t_L^i\left(t_L^i-1\right)\Delta x_L^2}{2\Delta x_L}, \tag{3.18c}$$

Table 1
Parameters for the approximation of the correction constants $R_{GG}(\alpha)$ and $R_{LG}(\alpha)$ described in Eq. (3.20).

XY =	A	B	C_1	C_2	p_0	p_1	p_2
GG	− 1	2	$\frac{15\pi-32}{6\pi-16}$	$\frac{15}{2\ln 2}$	1.50	0	2
LG	1	0	$\frac{3}{6\pi-16}\sqrt{\frac{2\pi}{\ln 2}}$	$\frac{15}{(2\ln 2)^2}$	2.25	1	4

with $\Delta x^i_{\nu,G} \equiv \frac{\Delta\nu}{w^i_G}$. The correction terms $R_{G\nu}$, $R_{GG}(\alpha^i)$, $R_{GL}(\alpha^i)$, R_{LL}, and $R_{LG}(\alpha^i)$ are ratios given of the error-integrals given by Eqs. (A.35)–(A.40) in the appendix, some of which are functions of $\alpha^i \equiv \frac{w^i_L}{w^i_G}$. The calculations of these terms are given in the appendix and can be simplified to:

$$R_{G\nu} = 8\ln 2 \tag{3.19a}$$

$$(R_{L\nu} = 0) \tag{3.19b}$$

$$R_{GG} \approx f_{GG}(\alpha) \tag{3.19c}$$

$$R_{LG} \approx f_{LG}(\alpha) \tag{3.19d}$$

$$R_{GL} = -(2\ln 2)\alpha^2 \tag{3.19e}$$

$$R_{LL} = 1 \tag{3.19f}$$

The functions $R_{GG}(\alpha)$ and $R_{LG}(\alpha)$ cannot be reduced to simple expressions and may be calculated either by numerically evaluating the integrals or by using the analytical expressions in the appendix. It is convenient however to have a closed form expression available to calculate the numerical values in real-time. We therefore provide a relatively simple but adequate parametrization of the following form:

$$f_{XY}(\alpha) = \frac{A}{\left[\left(\frac{\alpha^{p_1}}{C_1}\right)^{1/p_0} + \left(\frac{\alpha^{p_2}}{C_2}\right)^{1/p_0}\right]^{p_0}} + B \tag{3.20}$$

For a suitable choice of constants, this parametrization approaches both $R_{GG}(\alpha)$ and $R_{LG}(\alpha)$ asymptotically at $\alpha = 0$ and $\alpha = \infty$, fixing the constants A, B, C_1, C_2, p_1, and p_2. The only free parameter left, p_0, was then chosen such that $f_{XY}(\alpha) = R_{XY}(\alpha)$ at $\alpha = 1$. The resulting discretization errors were not noticeably different when $f_{XY}(\alpha)$ was used for the correction constants instead of the exact constants $R_{XY}(\alpha)$. The constants for $f_{XY}(\alpha)$ are given in Table 1.

The expression for the optimized weights (Eq. (3.18)) could be interpreted as the sum of the simple weights and some additional correction terms. It should be emphasized however that the correction terms are not necessarily small: When α goes to zero (close to full Gaussian), R_{LG} goes to infinity, and when α goes to infinity (close to full Lorentzian), R_{GL} goes to negative infinity. This means that the optimized weights can become significantly larger than unity or even negative, which may seem to be incompatible with the initial interpretation of the weights, where they are used to continuously fade between the two lineshapes on either side of the grid. Because the line-strengths are normalized regardless of their weights however (Eq. (2.9)), there is nothing fundamentally against weights larger than one or smaller than zero, as long as it indeed reduces the discretization error. Whether this is indeed the case will be explored in the next section.

3.2.4. Error comparison

Having defined two sets of weights (i.e. “simple” – Eq. (3.11) and “optimized” – Eq. (3.18)), we will now compare the RMS-discretization error for each of the two sets of weights. To depict how well the optimized weights minimize the discretization error, contour-plots of the error are shown in Fig. 3 for a typical set of grid-spacings: $\Delta x_\nu = 0.05$, $\Delta x_G = 0.14$, $\Delta x_L = 0.2$. The figure consists of four collections of figures for varying values of the ratio of Lorentzian to Gaussian width $\alpha = \frac{1}{25}, \frac{1}{2}, 2, 25$. For each value of α, nine figures are plotted with varying combinations of grid alignments with $t_G = 0.1,\ 0.5,\ 0.9$ and $t_L = 0.1,\ 0.5,\ 0.9$. Because the weight for the spectral position is $a_\nu = t_\nu$ for both simple and optimized weights, it is not varied and kept fixed at $t_\nu = 0.5$, where the error is largest. The contour plot for any given combination of α, t_G, and t_L shows isoerror lines as a function of choice of Gaussian weight a_G (horizontal) and Lorentzian weight a_L (vertical). The solid coloured lines were calculated by direct numerical evaluation of the RMS-error, while the black dashed lines were calculated by the analytical approximation derived in the previous section (Eq. (3.14)). In most cases, the analytical approximation shows good agreement with the direct numerical calculation.

The position of the simple and optimized weights in this graph are indicated by the square & cross, and the circle & point, respectively. From the figure it can be seen that in most cases the optimized error indeed minimizes the error adequately, especially compared to the simple weights. When the Lorentzian component is large ($\alpha = 25$), the optimized weights are not as close to the point of minimal error, but this is only apparent at points far away from $t_X = 0.5$, where the absolute error is small anyway. If the grid-spacings Δx_ν, Δx_G, and Δx_L are reduced, not only does the absolute error decrease, but because the analytical approximation of the error becomes more accurate, the optimized weights also better minimize the error. We note that while the grid sizes Δx_G and Δx_L are direct user-input values, the dimensionless spectral grid-size $\Delta x_\nu \equiv \Delta\nu/w_V$ is different for every line and may be adjusted by choosing different values for the (regular) spectral grid spacing $\Delta\nu$.

While the error-contours give a good indication of how close the chosen weights are to the weights with the absolute minimal error, they don’t give any insight in the magnitude of the error. The number of parameters were first further reduced before looking at the magnitude of the error. Firstly, the variables a_ν,a_G, and a_L can be eliminated by plugging in Eq. (3.11) for simple weights or Eq. (3.18) for optimized weights, respectively. The number of variables is further reduced by considering that the grid-alignments t_ν,t_G, and t_L are generally uniformly distributed, and thus the error may be averaged by integrating over t_ν, t_G, and t_L from 0 to 1, yielding an estimate of the average RMS-error that is only a function of Δx_ν, Δx_G, Δx_L, and α. For a particular choice of Δx_ν, Δx_G, Δx_L, the resulting function depends only on α and can thus be readily compared to other approximations for Voigt functions.

In Fig. 4 the numerically calculated mean RMS-error for simple (solid red) and optimized (solid purple) weights for grid-sizes of $\Delta x_\nu = 0.05$, $\Delta x_G = 0.14$, $\Delta x_L = 0.2$ are plotted as a function of $d \equiv \frac{w_L - w_G}{w_L + w_G} = \frac{\alpha-1}{\alpha+1}$, which is -1 for a pure Gaussian and 1 for a pure Lorentzian. The dashed black curves indicate the analytical approximations of the mean RMS discretization error, which show good agreement for $d < 0$, but underestimate the error for the optimized weights when $d \to 1$ (large Lorentzian component). This underestimation may be explained by the fact that the optimized weights almost completely eliminate the 2nd order components to the error, and the error is thus dominated by higher order components that are not included in the analytical approximation.

The RMS-errors for the simple and optimized weights are compared to the RMS-error for some popular approximations of the Voigt-function: the two approximations proposed by Whiting [17] and the approximation proposed by Liu et al. [19]. The figure clearly shows that for this choice of grid-sizes, the RMS-error

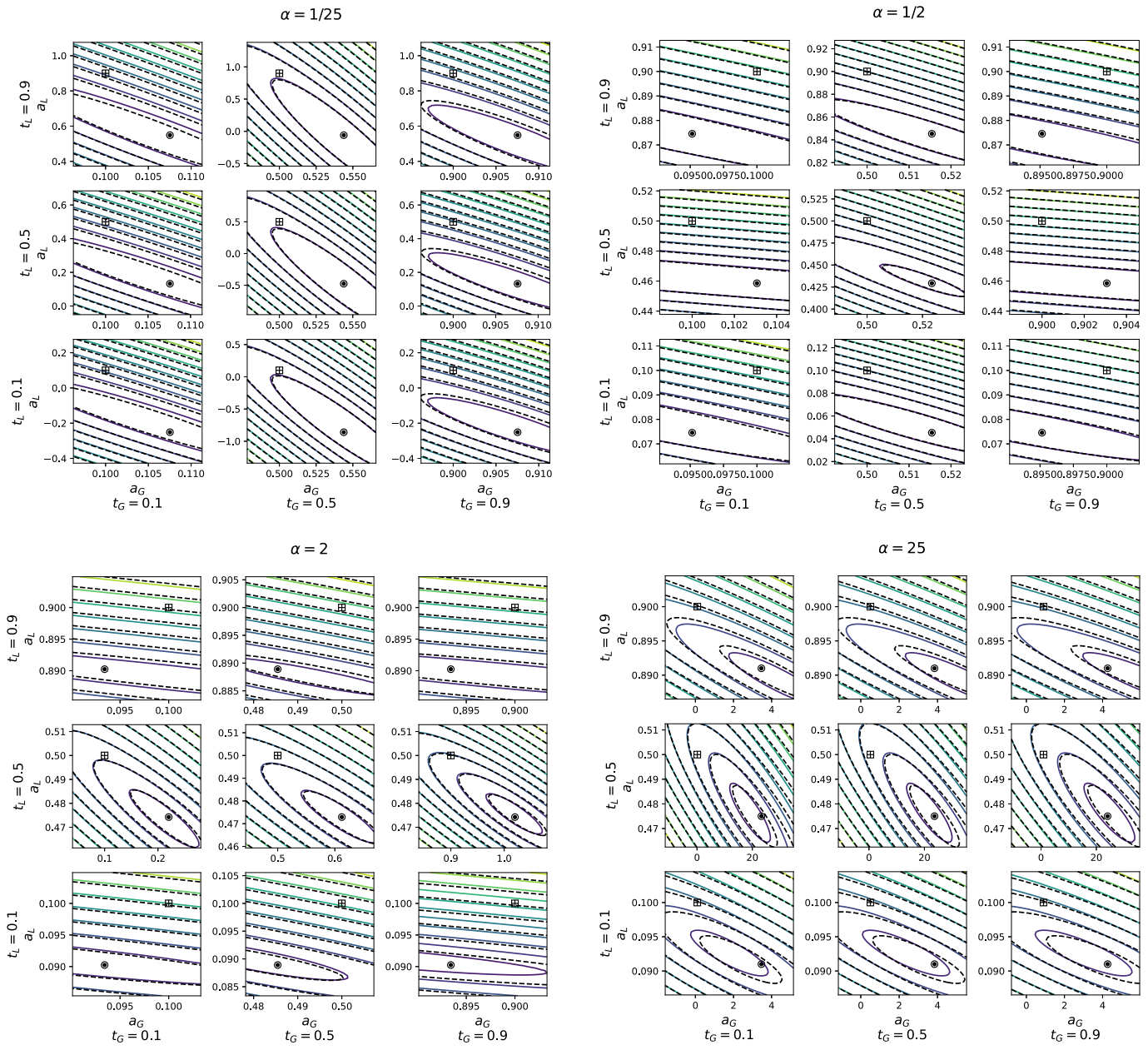

Fig. 3. Contour-plots of the discretization error plotted versus weights $\boldsymbol{a_G}$ and $\boldsymbol{a_L}$, for a selection of $\boldsymbol{t_G}$ & $\boldsymbol{t_L}$ and a range of ratios $\boldsymbol{\alpha} = \boldsymbol{w_L}/\boldsymbol{w_G}$, using $\boldsymbol{\Delta x_\nu} = 0.05$, $\boldsymbol{\Delta x_G} = 0.14$, $\boldsymbol{\Delta x_L} = 0.2$, and $\boldsymbol{a_\nu} = \boldsymbol{t_\nu} = 0.5$. The solid coloured contours represent the numerically calculated RMS discretization error while the dashed black contours represent the analytical approximation, which was used for determining the optimized weights. For each combination of $\boldsymbol{t_G}$ & $\boldsymbol{t_L}$ the simple weights, given by $\boldsymbol{a_X} = \boldsymbol{t_X}$, are indicated by the square and cross. The optimized weights, given by eq 3.22 are indicated by the circle and dot. Overall, the optimized weights can be seen to be close to the point of minimum discretization error. For smaller values of $\boldsymbol{\Delta x_\nu}$, $\boldsymbol{\Delta x_G}$, and $\boldsymbol{\Delta x_L}$, the optimized weights will be closer to the absolute minimum error.

of the proposed method are significantly smaller than for those of widely used contemporary Voigt approximations.

4. Results

The proposed method was implemented in the RADIS spectral code version 0.9.20 [28,36]. A minimal Python implementation of the procedure is also available in Supplementary materials. By default, RADIS already implements many optimizations, such as line-shape truncation, line intensity cut-off, and semi-continuum calculation [28]. All computational heavy tasks are run in C-compiled code, and calculation times in RADIS are comparable with other state-of-the-art solvers in the literature. Therefore, it is a good candidate for a comparison of the proposed method. RADIS version 0.9.25 was used for the benchmark presented here, on a personal laptop (i7–6700 HQ @ 2.6 GHz, 16 GB RAM).

4.1. Number of lines

Fig. 5 shows a comparison of the calculation times using the conventional approach (Eq. (2.1)) with RADIS's default settings and the discrete integral transform approach described in this paper, respectively, as a function of the number of lines in the spectrum. Absorption spectra of CO_2 at 3000 K were synthesized randomly sampling an increasing number of lines in the 2000 – 2400 cm^{-1} range of the HITEMP-2010 database (total number of lines: 1.8 M). With a resolution of $\Delta\nu = 0.002$ cm^{-1}, the total number of spectral points was 200,000. For the conventional method, the calcula-

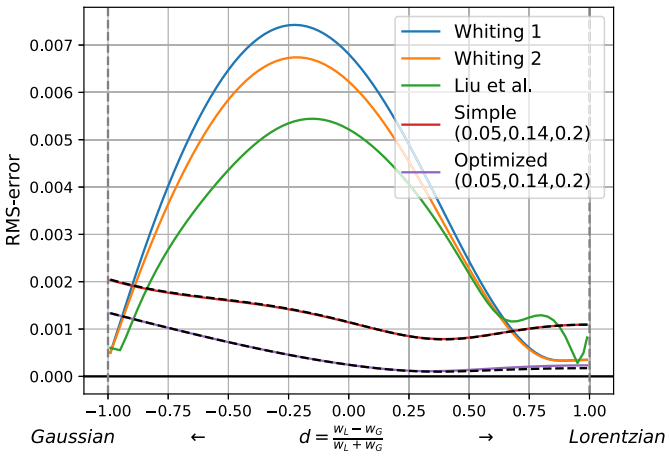

Fig. 4. RMS discretization errors for different approximations to the Voigt function as a function of $\boldsymbol{d} \equiv \frac{w_L - w_G}{w_L + w_G} = \frac{\alpha - 1}{\alpha + 1}$, which is -1 for a pure Gaussian and 1 for a pure Lorentzian. The solid curves indicate numerically calculated errors of the approximations by Whiting et al. [17] (blue & orange), by Liu et al. [19] (green), and mean RMS-errors for the proposed method using simple weights (red) and optimized weights (purple). Black dashed lines depict the analytically approximated RMS-error for the proposed method.

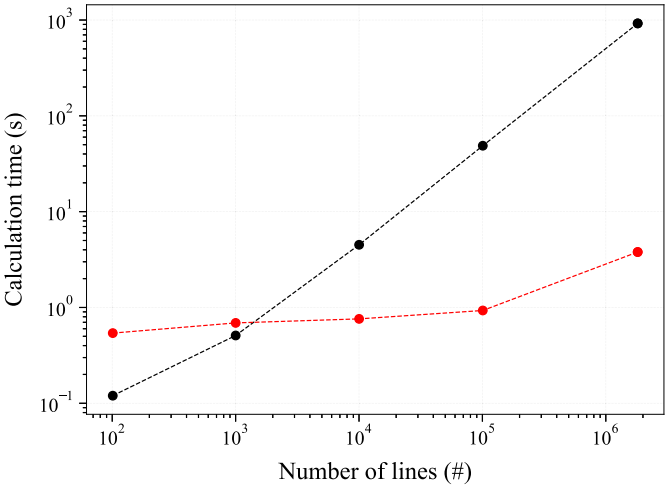

Fig. 5. Calculation times with the conventional (black) and proposed (red) approach, using optimized weights.

tion time increases about linearly with the number of lines from 0.1 s for 100 lines to 923 s for 1.8 M lines). When employing the proposed method with $\Delta x_G = 0.14$ and $\Delta x_L = 0.2$, the calculation time is initially higher (0.5 s for 100 lines) because of the overhead for setting up the lineshape grid. However, it remains below a few seconds for all cases even as the number of lines increases, which makes a significant difference for line numbers above 10k. For 1.8 M lines, the proposed method requires about 3.8 s to calculate the spectra (about 10^{11} lines $\times$ spectral points/s), which is about 300 times faster than when the usual method (923 s).

Fig. 6 shows the spectra calculated for the two rightmost points in Fig. 5 (1.8 M lines). The two methods yield the same spectra within 0.25%.

4.2. Truncation

As discussed in the introduction, it is possible to increase performance with the conventional approach by truncating lineshapes. While it is possible to achieve calculation times similar to that of the proposed approach, this is at a cost of greatly reduced accuracy. Fig. 7 shows a CO_2 synthetic spectrum calculated over 200k points between 2000 and 2400 cm^{-1}. For this example, only 10,000 randomly chosen lines were calculated. Calculation with a large truncation of 50 cm^{-1} is used as a reference. In all the calculations, the area of each lineshape remains normalized to 1. Using a truncation of 0.1 cm^{-1}, synthetic spectra can be produced spectra in about 1 s, however clear truncation effects are visible as shown in the inset of Fig. 7, showing a mismatch of peak intensity up to 15% with the reference.

Fig. 8 shows a comparison between the same reference spectrum and the new approach. In this case the integral transform method reproduces the reference spectrum adequately but is about two orders of magnitude faster. The calculation time of about 1 s is similar to the highly truncated spectrum of Fig. 7, without the adverse effects of truncation.

In Fig. 9 we compare the residual (top) and computation time (bottom) of a number of different spectra. The residue is calculated by referencing the spectra to a "ground truth" spectrum that is obtained by computing an exact Voigt function as the convolution of a Gaussian and Lorentzian lineshape with a truncation width of 50 cm^{-1}. The figure compares three different synthesis methods: the conventional method using the Whiting approximation at varying truncation widths (black solid curve), the conventional method using an exact Voigt lineshape (i.e. convolution of Gaussian and

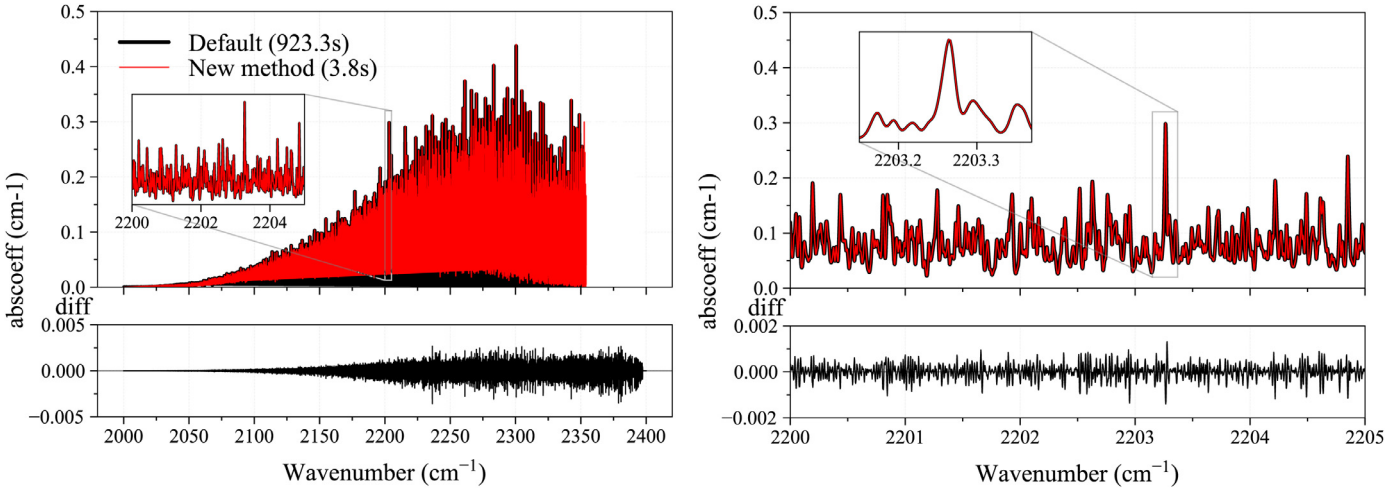

Fig. 6. Comparison of two calculated spectra: 2000 – 2400 cm^{-1} (resolution 0.002 cm^{-1}: 200k spectral points), HITEMP-2010 CO_2 (1.8 M lines), lineshape truncation in default mode: 5 cm^{-1}. The figure on the right is a zoom of the region shown in the inset on the left.

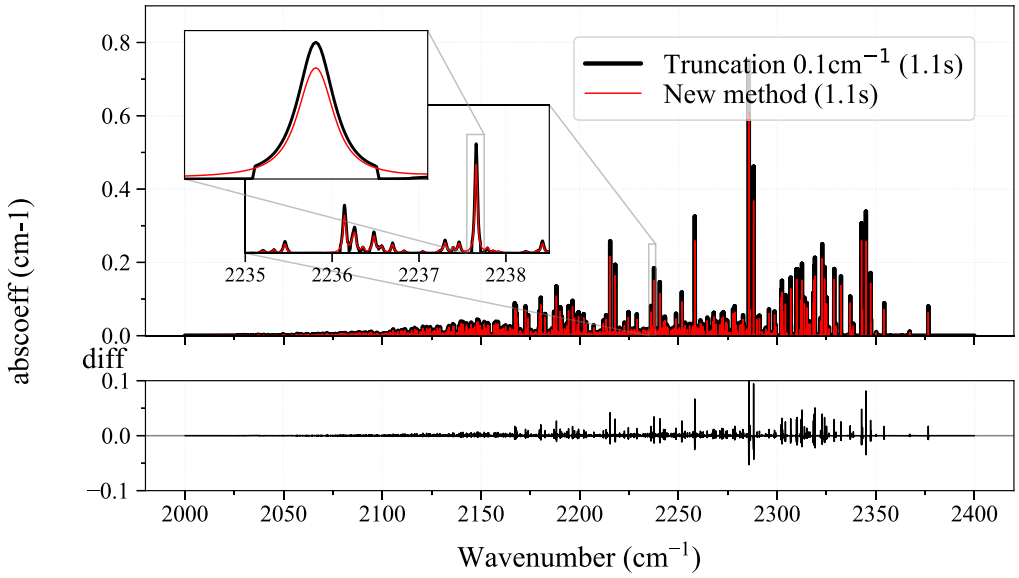

Fig. 7. CO_2 synthetic spectra (10k randomly chosen lines, 200k points) calculated by truncating the lineshape to 0.1 cm^{-1} (black) and with the proposed method (red). Insets show lines at higher resolution.

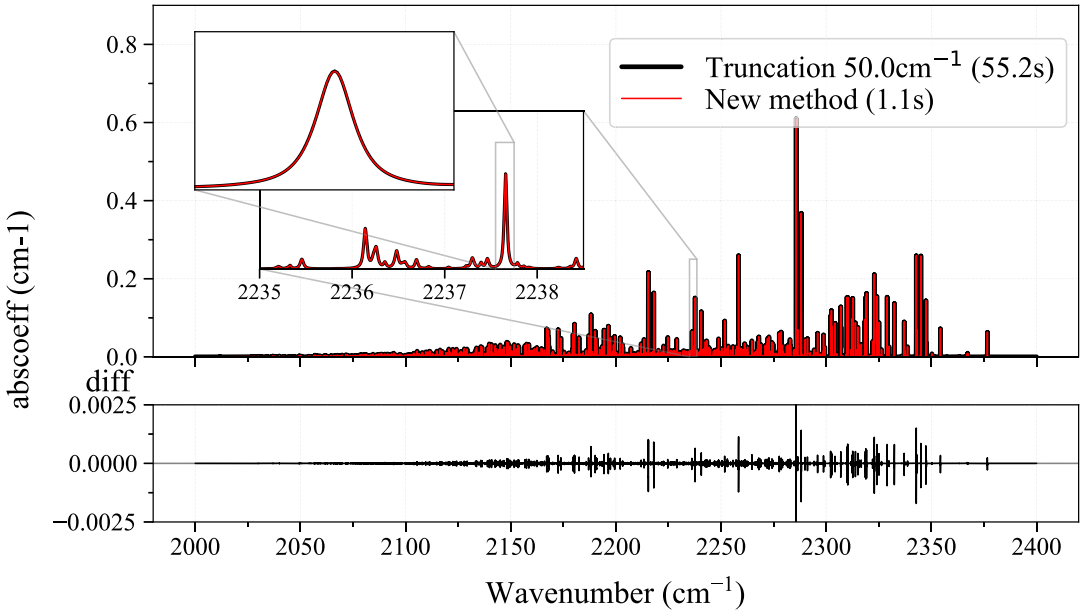

Fig. 8. CO_2 synthetic spectra (10k randomly chosen lines, 200k points) calculated by truncating the lineshape to 0.1 cm^{-1} (black) and with the proposed method, optimised weights (red). Insets show lines at higher resolution.

Lorentzian – black dashed curve), and the newly proposed method that inherently has no lineshape truncation (red dot). The points at 0.05 and 50 cm^{-1} correspond to Fig. 7 and Fig. 8, respectively. We note that the residue for the proposed method is larger than for some of the spectra synthesized with the conventional method. This is likely due to the discretization errors in the spectrum calculated by the integral transform and could be improved by choosing smaller grid-spacing, but could also be due to the fact that the 50 cm^{-1} truncation for the "ground truth" reference spectrum still introduces truncation errors. The corresponding calculation times are also shown (bottom). For the conventional method, a lineshape truncation of at least 5 to 10 cm^{-1} appears necessary to obtain a correct accuracy, which requires calculation times of 5 – 10 s. By comparison, the proposed integral transform method does not require such an optimization.

4.3. Pseudo-continuum

Finally, we compare the proposed method with another optimization method, where only strong lines and resolved and the others added in a pseudo-continuum. This pseudo-continuum approach is often found in literature codes, such as SPARTAN [37], or NEQAIR [9] for CO_2. Here we use the RADIS [28,36] implementation.

The pseudo-continuum was used in the benchmark cases [28] of the RADIS code to calculate water absorption spectrum for the main 3 isotopes in the 1000 – 2000 cm^{-1} range, for a total of 100k spectral points and 2.9 M lines. In Fig. 10 we reproduce this benchmark (top) and compare it to the new integral transform method (bottom). The performance improvement due to the pseudo-continuum is about 2 orders of magnitude. The inte-

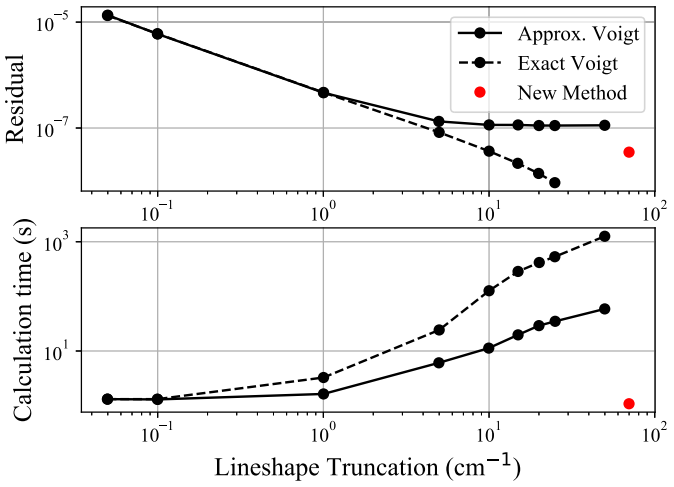

Fig. 9. Comparison of spectra calculated with different truncation (black dots), different truncation and the Whiting approximation for the Voigt lineshape (black solid line) and the proposed method (red), in terms of accuracy (up, where all are compared to the 50 cm-1 truncation without Whiting approximation) and in terms of computational performance (down). Proposed method has no truncation and therefore is shown on the side.

gral transform method achieves similar improvement and is even 3 times faster, but also yields a much better accuracy.

4.4. GPU

In the conventional method, adding lines to the output spectrum requires information on the entire spectrum before the line is added and may thus be difficult to parallelize. In the proposed method, the lines that are added to the lineshape distribution matrix are mostly independent of other lines, which is therefore much more straightforward to parallelize. To compare how much performance gain can be expected from parallel execution, a GPU-implementation of the proposed method was implemented in an experimental version of RADIS, version 0.9.26. To maximize performance, the entire line database is first transferred to the GPU's global memory, after which spectra for any given condition can be calculated at high speed. Due to this setup, there is a lot of overhead for calculating the first spectrum of a series, which typically takes about an order of magnitude longer than any subsequent spectrum. Once the database is transferred to the GPU, calculations of subsequent spectra (that use the same database) can all be done on the GPU resulting in smaller computation times compared to the initial spectrum. In many cases, such as in fitting routines, a large number of spectra need to be synthesized in which case the extra overhead for the first line becomes insignificant.

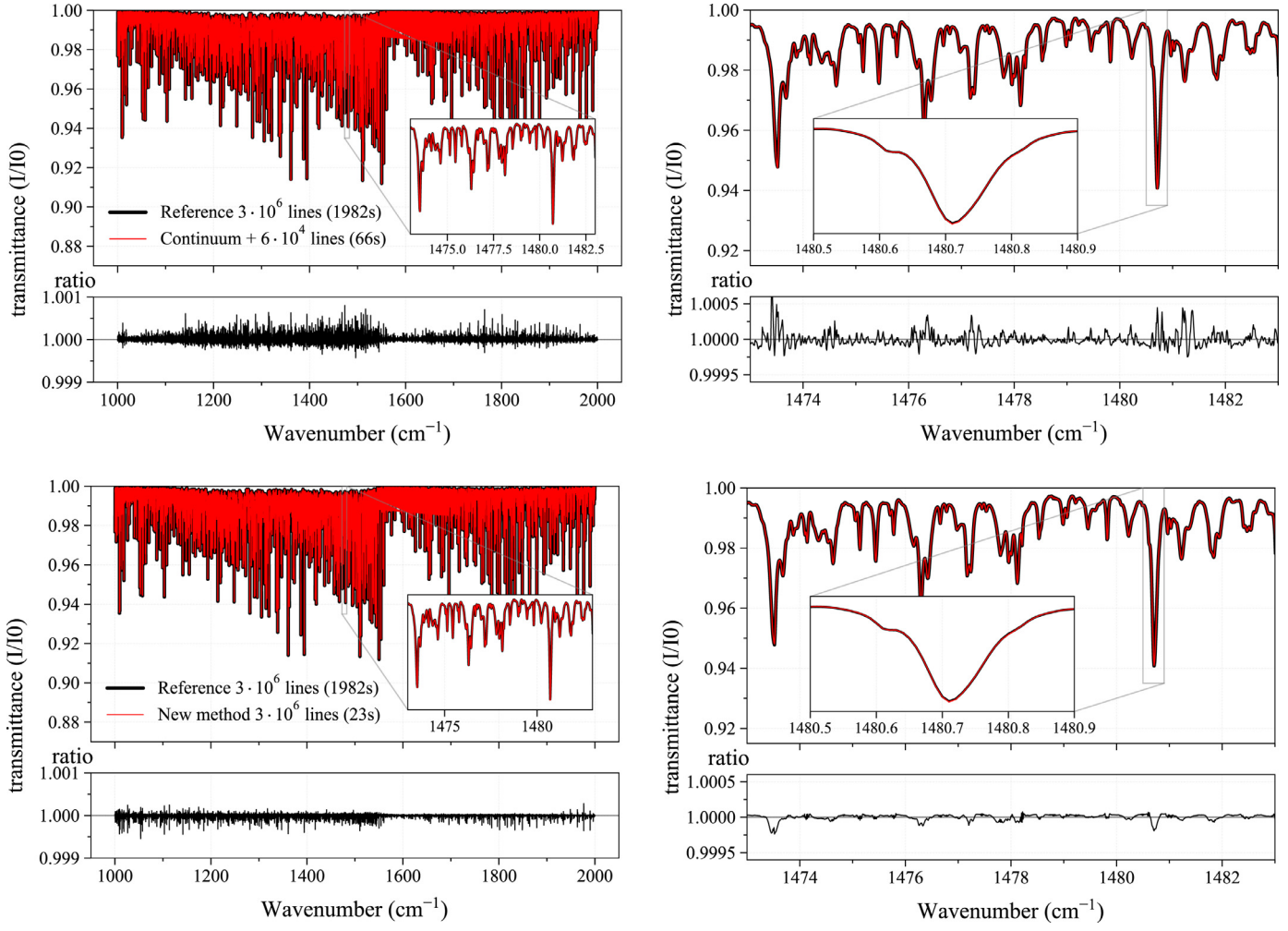

Fig. 10. Water transmittance spectrum at 2200 K (HITEMP-2010, 3 M lines, 100k spectral points) calculated with the standard method labelled "reference" (black), is compared with the pseudo-continuum plus 60k lines (top, red), and the integral transform method with all lines (bottom, red). Figures on the right are zooms of the region shown in the inset on the left.

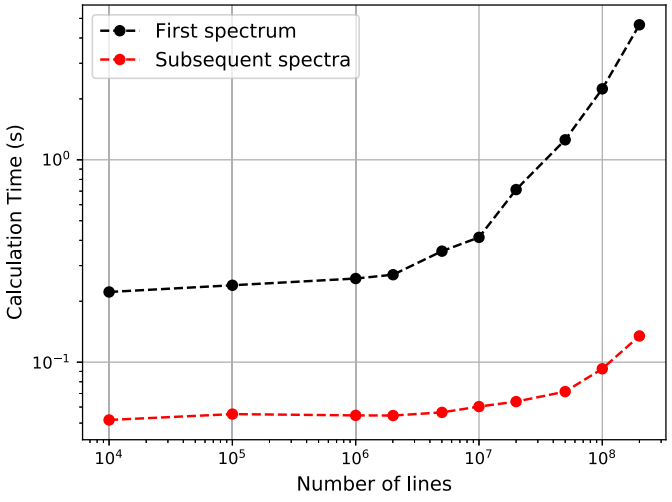

Fig. 11. Benchmark of a GPU-implementation of the proposed method for CO_2 absorption spectra using the CDSD4000 spectral database over a spectral range of 1800–2400cm^{-1} and $\Delta\nu = 0.002$ cm^{-1}, with $\boldsymbol{N_G} = 4$ and $\boldsymbol{N_L} = 8$. The curves show the computation time for a varying number of lines included in the spectrum. The first spectrum in a series takes significantly longer to compute due to CPU-to-GPU overhead.

To benchmark the GPU code, absorption spectra of CO_2 were synthesized using the CDSD4000 database in the range 1800–2400cm^{-1} with a spectral resolution of $\Delta\nu = 0.002$ cm^{-1} ($3 \cdot 10^5$ spectral points). In this case not the grid spacing, but the grid sizes of the width-axes were kept constant at $N_G = 4$ and $N_L = 8$, while the grid-spacings were adjusted to accommodate the entire range of Gaussian and Lorentzian widths respectively. The benchmark was performed on an Intel i7 9750H CPU and Nvidia RTX 2070 max-Q GPU. The number of lines included in the database was varied from $1 \cdot 10^4$ to a maximum of $2 \cdot 10^8$ lines, which was dictated by the GPUś memory size of 8 GB. The benchmark results are shown in Fig. 11. For a given number of lines in the database, 27 spectra were produced with a temperature from 300 to 3000 K. As motivated earlier, the computation time for the first spectrum (black) is much longer than that for subsequent spectra (red), mainly due to the cost of transferring the database to the GPU-memory. Computation times for subsequent spectra for a given database size were typically very similar and independent of the temperature. For up to 1 M lines the computation time does not increase with the number of lines, suggesting the computation time is dominated by overhead, including calculation of the FT's. For larger number of lines, the computation time increases with the number of lines. At 200 M lines, the computation time is 5 s for the first line, but still only 0.15 s for subsequent lines. In terms of the product of number of lines and spectral points per second, the performance of the GPU-implementation is $1.2 \cdot 10^{13} s^{-1}$ and $5 \cdot 10^{14} s^{-1}$ for the first and subsequent spectra respectively, which to our knowledge is the fastest reported to date.

4.5. Memory requirements

In addition to performance and accuracy, an important performance indicator for a spectral synthesis approach is the memory requirement. Both the CPU and GPU implementation are implemented with single-precision floating point numbers and load the entire database before starting the procedure. The calculation of the line-list (S_0^i, ν_0^i, w_G^i, and w_L^i) relied on 7 parameters per line: line position, line strength, pressure broadening constant, pressure broadening temperature coefficient, isotopoglogue mass, pressure shift, and lower state energy. This implies $M_{DB} = 28 \times N_i$ bytes are needed to load the database in memory, but a more efficient use of memory is certainly possible by using smaller types than a 4-byte float wherever the number of significant figures permits. In addition to the database, the lineshape distribution matrix S_{klm} needs to be stored. This takes $M_{LDM} = 8 \times N_\nu \times (1 + N_G \times N_L)$ bytes, assuming single point precision and zero padding are used. For large databases, the memory requirement for S_{klm} is insignificant compared to the database memory requirement, and the total number of required bytes simplifies to M_{DB}. In the case of the uncompressed 3000 K methane list of 150 billion lines [6], this would come down to 4.2 TB.

It should be noted however that the database doesn't have to be loaded in memory in its entirety, and instead parts of the database may be loaded as they are processed. In this case, the total memory requirement reduces to only M_{LDM} bytes, which is in the order of a couple dozen megabytes for a typical realistic matrix (e.g. $8 \times 300\,000 \times (1 + 4 \times 8) = 79$ MB for the GPU example of Section 4.4) and does not scale with the database size. Because the filling of S_{klm} (Eq. (2.10)) doesn't involve many calculations, the drawback of loading the database during processing is that the overall performance will likely be limited by the memory transfer rate. This limitation will be particularly noticeable for a GPU implementation, as suggested by the difference between the first and consequent spectra (black and red curves, Fig. 11).

5. Conclusion

We have proposed a new method for generating synthetic spectra that has the potential to dramatically improve performance while maintaining or even improving accuracy. By replacing the typical summation of prototype lineshapes with an integral transform, only a couple dozen lineshapes have to be generated instead of thousands to millions of lineshapes. A discrete version of the integral transform was formulated which can be implemented using the Fast Fourier Transform, further improving performance. A set of weights was proposed that interpolate the line parameters and minimize the discretization error. Depending on the number of lines in the spectrum, we showed that a performance increase of a couple orders of magnitude compared to fully optimized, state-of-the art spectral synthesis code can be obtained. A GPU-implementation of the proposed method further increases the performance by 2~3 orders.

In summary, the proposed method consists of calculating the synthetic spectrum using Eq. (3.3), with the Voigt profile given by Eq. (3.2) and the lineshape distribution matrix given by Eq. (2.8) & (2.10). A set of weights must be chosen for which we provided either Eqs. (3.11) or (3.18) for consideration, and whose inputs can be calculated using Eqs. (3.4)-(3.10).

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.jqsrt.2020.107476.

Appendix

The dimensionless profile can be expressed as a convolution of the dimensionless, unity width Gaussian and Lorentzian profiles $G_G(x)$ and $G_L(x)$:

$$G_V(x_\nu; \alpha) = G_G\left(\frac{x_\nu}{\alpha_G}\right) * G_L\left(\frac{x_\nu}{\alpha_L}\right), \tag{A.1}$$

which is a function of the dimensionless spectral coordinate $x_\nu \equiv \nu/w_V$, and the dimensionless profiles $G_G(x)$ and $G_L(x)$ are given by:

$$G_G(x) = 2\sqrt{\frac{\ln 2}{\pi}}\exp\left(-4\ln 2x^2\right),\ G_L(x) = \frac{2}{\pi\left(1+4x^2\right)}, \tag{A.2}$$

and the ratios α, α_G, and α_L dimensionless widths that are calculated as ratios of the FWHM's:

$$\alpha \equiv \frac{w_L}{w_G},\ \alpha_G \equiv \frac{w_G}{w_V},\ \alpha_L \equiv \frac{w_L}{w_V}. \tag{A.3}$$

While α_G and α_L are in principle a function of only α, no exact relation exists and therefore an approximation is often used [18]:

$$\alpha_G \approx \frac{2}{C_1\alpha+\sqrt{C_2\alpha^2+4}},\ \alpha_L \approx \frac{2\alpha}{C_1\alpha+\sqrt{C_2\alpha^2+4}}, \tag{A.4}$$
$$C_1 = 1.0692,\ C_2 = 0.86639$$

Deriving the error directly from the lineshape functions is difficult, as they involve calculating convolutions of approximating lineshape functions. This analysis is greatly simplified by looking at the Fourier space, where convolutions are manifested as multiplications. The big advantage of choosing Eq. (3.13) as the error measure is that through Plancherel's theorem [28], the error in real space is equal to the RMS error in Fourier space, i.e.:

$$\int\left|\tilde{G}_V(x_\nu;\alpha) - G_V(x_\nu;\alpha)\right|^2 dx_\nu = \int\left|\tilde{\hat{G}}_V(\xi;\alpha) - \hat{G}_V(\xi;\alpha)\right|^2 d\xi,$$
$$\xi \equiv \hat{\nu} w_V \tag{A.5}$$

In Fourier space The Voigt lineshape may be written as the product of Gaussian and Lorentzian lineshapes:

$$\hat{G}_V(\xi;\alpha) = \hat{G}_G(\alpha_G\xi)\hat{G}_L(\alpha_L\xi), \tag{A.6}$$

and $\hat{G}_G(\xi)$ and $\hat{G}_L(\xi)$ the unity-width lineshapes for Gaussian and Lorentzian:

$$\hat{G}_G(\xi) = \exp-\frac{(\pi\xi)^2}{4\ln 2},\ \hat{G}_L(\xi) = \exp-\pi|\xi|. \tag{A.7}$$

Explicit expressions for the approximated Gaussian and Lorentzian lineshape functions $\breve{\hat{G}}(\xi)$ are then given by:

$$\breve{\hat{G}}_X\left(\hat{\nu}w_X\right) = (1-a_X)\hat{G}_X\left(\hat{\nu}w_X[l]\right) + a_X\hat{G}_X\left(\hat{\nu}w_X[l+1]\right) \tag{A.8}$$

$$\breve{\hat{G}}_X(\xi) = (1-a_X)\hat{G}\left(\xi\frac{w_X[l]}{w_X}\right) + a_X\hat{G}_X\left(\xi\frac{w_X[l+1]}{w_X}\right) \tag{A.9}$$

$$\breve{\hat{G}}_X(\xi) = (1-a_X)\hat{G}_X(\exp[\ln\xi - t_X\Delta x_X]) + a_X\hat{G}_X(\exp[\ln\xi - (t_X-1)\Delta x_X]) \tag{A.10}$$

$$\breve{\hat{G}}_X(\xi) = (1-a_X)\sum_{n=0}^{\infty}\frac{1}{n!}\frac{\partial^n\hat{G}_X}{\partial(\ln\xi)^n}(-t_X\Delta x_X)^n + a_X\sum_{n=0}^{\infty}\frac{1}{n!}\frac{\partial^n\hat{G}_X}{\partial(\ln\xi)^n}(-(t_X-1)\Delta x_X)^n \tag{A.11}$$

$$\breve{\hat{G}}_X(\xi) = \sum_{n=0}^{\infty}\frac{(-1)^n}{n!}\left[(1-a_X)t_X^n + a_X(t_X-1)^n\right]\frac{\partial^n\hat{G}_X}{\partial(\ln\xi)^n}\Delta x_X^n \tag{A.12}$$

We will evaluate this for $X = G, L$ and up to second order, i.e. $n = 0,\ 1,\ 2$. The n^{th} derivative of $\hat{G}_X$ with respect to $\ln\xi$ is given by:

$$\frac{\partial^n\hat{G}_X}{\partial(\ln\xi)^n} = \hat{h}_{Xn}(\xi)\hat{G}_X(\xi) \tag{A.13}$$

and $\hat{h}_{Xn}(\xi)$ is:

$$\left(\hat{h}_{X0}(\xi) = 1\right) \tag{A.14}$$

$$\hat{h}_{G1}(\xi) = -\frac{\pi^2\xi^2}{2\ln 2},\ \hat{h}_{L1}(\xi) = -\pi|\xi| \tag{A.15}$$

$$\hat{h}_{G2}(\xi) = \frac{\pi^2\xi^2}{2\ln 2}\left(\frac{\pi^2\xi^2}{2\ln 2} - 2\right),\ \hat{h}_{L2}(\xi) = \pi|\xi|(\pi|\xi| - 1) \tag{A.16}$$

With these expressions, the discretization error may be expressed as the sum of the exact lineshape and some error terms $\hat{e}_{Xn}(\xi)$, that vanish for increasingly large n:

$$\breve{\hat{G}}_X(\xi) = \left(1+\hat{e}_X\right)\hat{G}_X(\xi) \tag{A.17}$$

with:

$$\hat{e}_X(\xi) \equiv \sum_{n=1}^{\infty}\hat{e}_{Xn}(\xi),\ \hat{e}_{Xn}(\xi) \equiv c_{Xn}\hat{h}_{Xn}(\xi) \tag{A.18}$$

where:

$$c_{X1}(a_X, t_X, \Delta x_X) = (a_X - t_X)\Delta x_X \tag{A.19}$$

$$c_{X2}(a_X, t_X, \Delta x_X) = \frac{1}{2}\left(t_X^2 - 2a_Xt_X + a_X\right)\Delta x_X^2 \tag{A.20}$$

Now the approximated Voigt lineshape can be expressed as:

$$\breve{\hat{G}}_V(\xi;\alpha) = \left[1+\hat{e}_G(\alpha_G\xi)\right]\left[1+\hat{e}_L(\alpha_L\xi)\right]\hat{G}_V(\xi;\alpha) \tag{A.21}$$

While $\breve{\hat{G}}_V$ gives the lineshape approximation by discretizing the Gaussian and Lorentzian width, the line position is not discretized yet. We do this as follows:

$$\tilde{G}_V(x_\nu;\alpha) = (1-a_\nu)\breve{G}_V(x_\nu + t_\nu\Delta x_\nu;\alpha) + a_\nu\breve{G}_V(x_\nu + (t_\nu-1)\Delta x_\nu;\alpha) \tag{A.22}$$

In Fourier space this looks like:

$$\tilde{\hat{G}}_V(\xi;\alpha) = (1-a_\nu)\breve{\hat{G}}_V(\xi;\alpha)\exp[-2\pi i t_\nu\Delta x_\nu\xi] + a_\nu\breve{\hat{G}}_V(\xi;\alpha)\exp[-2\pi i(t_\nu-1)\Delta x_\nu\xi] \tag{A.23}$$

Expand the exponential:

$$\tilde{\hat{G}}_V(\xi;\alpha) = \left[1+\hat{e}_\nu(\xi)\right]\breve{\hat{G}}_V(\xi;\alpha) \tag{A.24}$$

$$\hat{e}_\nu(\xi) \equiv (1-a_\nu)\sum_{n=1}^{\infty}\frac{1}{n!}(-2\pi i t_\nu\Delta x_\nu\xi)^n + a_\nu\sum_{n=1}^{\infty}\frac{1}{n!}(-2\pi i(t_\nu-1)\Delta x_\nu\xi)^n \tag{A.25}$$

$$\hat{e}_\nu(\xi) = \sum_{n=1}^{\infty}\frac{(-1)^n}{n!}\left[(1-a_\nu)t_\nu{}^n + a_\nu(t_\nu-1)^n\right](2\pi i\xi)^n\Delta x_\nu{}^n \tag{A.26}$$

That is, perhaps somewhat surprisingly, $\hat{e}_\nu(\xi)$ can be expressed in exactly the same way as $\hat{e}_G(\xi)$ and $\hat{e}_L(\xi)$, with the functions $\hat{h}_{\nu n}(\xi)$ given by:

$$\hat{h}_{\nu n}(\xi) = (2\pi i\xi)^n \tag{A.27}$$

The approximated lineshape including line position discretization $\tilde{\hat{G}}_V(\xi;\alpha)$, can now be expressed in terms of the exact lineshape function $\hat{G}_V(\xi;\alpha)$ as follows:

$$\tilde{\hat{G}}_V(\xi;\alpha) = \left[1+\hat{e}_\nu(\xi)\right]\left[1+\hat{e}_G(\alpha_G\xi)\right]\left[1+\hat{e}_L(\alpha_L\xi)\right]\hat{G}_V(\xi;\alpha), \tag{A.28}$$

where α_G and α_L are both functions of α. The discretization error can thus be expressed as some scaled version of the original, undiscretized lineshape $\hat{G}_V(\xi;\alpha)$, with the error terms $\hat{e}_X$ a function of the parameters a_ν, a_G, a_L, t_ν, t_G, t_L, $\Delta\nu$, Δx_G, Δx_L, α and ξ.

By taking only error terms up to second order into account, the squared error evaluates to an expression with 16 distinct error terms:

$$\begin{aligned}\hat{E}_V^2(\xi;\alpha) &\equiv \left|\tilde{\hat{G}}_V(\xi;\alpha) - \hat{G}_V(\xi;\alpha)\right|^2 \\ &\approx \big(\hat{e}_{G1}^2 + 2\hat{e}_{G1}\hat{e}_{G2} + \hat{e}_{G2}^2 + 2\hat{e}_{G1}\hat{e}_{L1} + 2\hat{e}_{G1}\hat{e}_{L2} + 2\hat{e}_{G2}\hat{e}_{L1} \\ &\quad + 2\hat{e}_{G2}\hat{e}_{L2} + \hat{e}_{L1}^2 + 2\hat{e}_{L1}\hat{e}_{L2} + \hat{e}_{L2}^2 + 2\hat{e}_{G1}\hat{e}_{\nu2} + 2\hat{e}_{G2}\hat{e}_{\nu2} \\ &\quad + 2\hat{e}_{L1}\hat{e}_{\nu2} + 2\hat{e}_{L2}\hat{e}_{\nu2} + \hat{e}_{\nu1}^2 + \hat{e}_{\nu2}^2\big)\hat{G}_V^2(\xi;\alpha)\end{aligned} \tag{A.29}$$

By taking the integral over $\hat{E}_V^2(\xi;\alpha)$ with respect to $d\xi$, we get an expression for the error that's independent of the spectral coordinate ξ:

$$\begin{aligned}E_{RMS}(a_\nu, a_G, a_L, t_\nu, t_G, t_L, \Delta x_\nu, \Delta x_G, \Delta x_L, \alpha)^2 &\approx c_{L1}^2 S^\circ_{L1L1} \\ &+ 2c_{L1}c_{L2}S^\circ_{L1L2} + c_{L2}^2 S^\circ_{L2L2} + 2c_{G1}c_{L1}S^\circ_{G1L1} + 2c_{G1}c_{L2}S^\circ_{G1L2} \\ &+ 2c_{G2}c_{L1}S^\circ_{G2L1} + 2c_{G2}c_{L2}S^\circ_{G2L2} + c_{G1}^2 S^\circ_{G1G1} + 2c_{G1}c_{G2}S^\circ_{G1G2} \\ &+ c_{G2}^2 S^\circ_{G2G2} + 2c_{G1}c_{\nu2}S^\circ_{G1\nu2} + 2c_{G2}c_{\nu2}S^\circ_{G2\nu2} + 2c_{L1}c_{\nu2}S^\circ_{L1\nu2} \\ &+ 2c_{L2}c_{\nu2}S^\circ_{L2\nu2} + c_{\nu1}^2 S^\circ_{\nu1\nu1} + c_{\nu2}^2 S^\circ_{\nu2\nu2}\end{aligned} \tag{A.30}$$

Where $S^\circ_{XiYj}(\alpha)$ is defined by [38]:

$$\begin{aligned}S^\circ_{XiYj}(\alpha) &= S^\circ_{YjXi}(\alpha) \equiv \int_{-\infty}^{\infty} h_{Xi}(\alpha_X\xi)h_{Yj}(\alpha_Y\xi)\hat{G}_V^2(\xi;\alpha)d\xi \\ &= \frac{\sqrt{2\ln 2}}{\pi\alpha_G}\sum_{n=0}^{N} c_n f_n\left(\alpha\sqrt{2\ln 2}\right)\left(\alpha\sqrt{2\ln 2}\right)^n \\ f_n(x) &\equiv \begin{cases}\sqrt{\pi}\ \mathrm{erfcx}\, x & \text{if } n \text{ is } even \\ 1 & \text{if } n \text{ is } odd\end{cases}, \quad \mathrm{erfcx}\, x \\ &\equiv (1-\mathrm{erf}\, x)\exp x^2, \ \ \mathrm{erf}\, x \equiv \tfrac{2}{\sqrt{\pi}}\int_0^x \exp -t^2 dt\end{aligned} \tag{A.31}$$

and c_n can be found in the table below:

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
S°_{L1L1}	0	0	1/2	−1	1	0	0	0	0
S°_{L1L2}	0	0	1/2	−2	5/2	−1	1	0	0
S°_{L2L2}	0	0	1/2	−3	19/4	−9/2	5	−1	1
S°_{G1L1}	0	1	−3/2	1	−1	0	0	0	0
S°_{G1L2}	0	1	−9/4	7/2	−4	1	−1	0	0
S°_{G2L1}	0	0	3/4	−5/2	3	−1	1	0	0
S°_{G2L2}	0	0	9/8	−23/4	33/4	−6	13/2	−1	1
S°_{G1G1}	3/4	−5/2	3	−1	1	0	0	0	0
S°_{G1G2}	−3/8	13/4	−21/4	5	−11/2	1	−1	0	0
S°_{G2G2}	33/16	−95/8	39/2	−89/4	53/2	−19/2	10	−1	1

The error-integrals with respect to ν can be derive from the ones to G and L:

$$S^\circ_{Xi\nu1}(\alpha) = S^\circ_{\nu1Xi}(\alpha) = 0 \tag{A.32a}$$

$$S^\circ_{Xi\nu2}(\alpha) = S^\circ_{\nu2Xi}(\alpha) = \frac{8\ln 2}{{\alpha_G}^2}S^\circ_{G1Xi}(\alpha) \tag{A.32b}$$

$$S^\circ_{\nu1\nu1}(\alpha) = -\frac{4}{{\alpha_L}^2}S^\circ_{L1L1}(\alpha) \tag{A.32c}$$

$$S^\circ_{\nu2\nu2}(\alpha) = \frac{64(\ln 2)^2}{{\alpha_G}^4}S^\circ_{G1G1}(\alpha) \tag{A.32d}$$

We can now find optimal weights by setting the derivatives to the weights to zero and solving for a_G and a_L:

$$\frac{\partial E_{RMS}}{\partial a_\nu} = 0, \ \ \frac{\partial E_{RMS}}{\partial a_G} = 0, \ \ \frac{\partial E_{RMS}}{\partial a_L} = 0 \tag{A.33}$$

After eliminating every $\log p$-term of order 2 and higher, all but a few terms disappear and we achieve our final result:

$$\begin{aligned}&a_\nu(t_\nu) = t_\nu \\ &a_G(t_\nu, t_G, t_L, \Delta x_{\nu,G}, \Delta x_G, \Delta x_L, \alpha) = t_G \\ &+\frac{R_{G\nu}t_\nu(t_\nu-1)\Delta x_{\nu,G}^2 + R_{GG}t_G(t_G-1)\Delta x_G^2 + R_{GL}t_L(t_L-1)\Delta x_L^2}{2\Delta x_G} \\ &a_L(t_G, t_L, \Delta x_G, \Delta x_L, \alpha) \\ &\int = t_L + \frac{R_{LG}t_G(t_G-1)\Delta x_G^2 + R_{LL}t_L(t_L-1)\Delta x_L^2}{2\Delta x_L}\end{aligned} \tag{A.34}$$

with:

$$R_{G\nu}(\alpha) = (8\ln 2)\frac{S_{L1L1}S_{G1G1} - {S_{G1L1}}^2}{S_{L1L1}S_{G1G1} - {S_{G1L1}}^2} = 8\ln 2 \tag{A.35}$$

$$R_{GG}(\alpha) = \frac{S_{L1L1}S_{G1G2} - S_{G1L1}S_{L1G2}}{S_{L1L1}S_{G1G1} - {S_{G1L1}}^2} \tag{A.36}$$

$$R_{GL}(\alpha) = \frac{S_{L1L1}S_{G1L2} - S_{G1L1}S_{L1L2}}{S_{L1L1}S_{G1G1} - {S_{G1L1}}^2} = -(2\ln 2)\alpha^2 \tag{A.37}$$

and:

$$R_{L\nu}(\alpha) = (8\ln 2)\frac{S_{G1G1}S_{G1L1} - S_{G1L1}S_{G1G1}}{S_{G1G1}S_{L1L1} - {S_{G1L1}}^2} = 0 \tag{A.38}$$

$$R_{LG}(\alpha) = \frac{S_{G1G1}S_{L1G2} - S_{G1L1}S_{G1G2}}{S_{G1G1}S_{L1L1} - {S_{G1L1}}^2} \tag{A.39}$$

$$R_{LL}(\alpha) = \frac{S_{G1G1}S_{L1L2} - S_{G1L1}S_{G1L2}}{S_{G1G1}S_{L1L1} - {S_{G1L1}}^2} = 1 \tag{A.40}$$

Where:

$$S_{XiYj}(\alpha) \equiv \frac{\pi\alpha_G}{\sqrt{2\ln 2}}S^\circ_{XiYj}(\alpha) = \sum_{n=0}^{N} c_n f_n\left(\alpha\sqrt{2\ln 2}\right)\left(\alpha\sqrt{2\ln 2}\right)^n \tag{A.41}$$

CRediT authorship contribution statement

D.C.M. van den Bekerom: Conceptualization, Methodology, Software, Validation, Formal analysis, Writing - original draft, Writing - review & editing, Visualization. **E. Pannier:** Software, Validation, Writing - review & editing, Visualization.

References

[1] Gordon IE, Rothman LS, Hill C, Kochanov RV, Tan Y, Bernath PF, Birk M, et al. The HITRAN2016 molecular spectroscopic database. J Quant Spectrosc Radiat Transfer December 2017;203:3–69. doi:10.1016/j.jqsrt.2017.06.038.

[2] Rothman LS, Gordon IE, Barber RJ, Dothe H, Gamache RR, Goldman A, Perevalov VI, Tashkun SA, Tennyson J. HITEMP, the high-temperature molecular spectroscopic database. J Quant Spectrosc Radiat Transfer October 2010;111(15):2139–50. doi:10.1016/j.jqsrt.2010.05.001.

[3] Tashkun SA, Perevalov VI. CDSD-4000: high-resolution, high-temperature carbon dioxide spectroscopic databank. J Quant Spectrosc Radiat Transfer June 1, 2011;112(9):1403–10. doi:10.1016/j.jqsrt.2011.03.005.

[4] Tennyson, Jonathan, and Sergei N Yurchenko. "ExoMol: molecular line lists for exoplanet and other atmospheres," 2012, 13.

[5] Rey Michaël, Nikitin Andrei V, Babikov Yurii L, Tyuterev Vladimir G. TheoReTS – an information system for theoretical spectra based on variational predictions from molecular potential energy and dipole moment surfaces. J Mol Spectrosc, New Visions of Spectroscopic Databases September 1, 2016;II(327):138–58. doi:10.1016/j.jms.2016.04.006.

[6] Rey Michael, Nikitin Andrei V, Tyuterev Vladimir G. Accurate theoretical methane line lists in the infrared up to 3000k and quasi-continuum absorption/emission modeling for astrophysical applications. Astrophys. J. September 27, 2017;847(2):105. doi:10.3847/1538-4357/aa8909.

[7] Albert Damien, et al. A decade with VAMDC: results and ambitions. Atoms 2020;8(4):76. doi:10.3390/atoms8040076.

[8] Zhao XY, Haworth DC, Ren T, Modest MF. A Transported probability density function/photon monte carlo method for high-temperature oxy–natural gas combustion with spectral gas and wall radiation.. Combust Theory Model April 1, 2013;17(2):354–81. doi:10.1080/13647830.2013.766365.

[9] Palmer G, Cruden B. Experimental Validation of CO2 Radiation Simulations. In: 43rd AIAA thermophysics conference, 2012-3188. Reston, Virigina: American Institute of Aeronautics and Astronautics; 2012. p. 1–15. doi:10.2514/6.2012-3188.
[10] Geller, and Murray. "NASA Technical Reports Server (NTRS)," 1992. https://ntrs.nasa.gov/citations/19960007788.
[11] Giuranna M, Fonte S, Longobardo A, Sindoni G, Wolkenberg P, Formisano V. PFS/MEX limb observations of 4.3-Mm CO2 non-LTE emission in the atmosphere of Mars. Icarus November 15, 2018;315:46–60. doi:10.1016/j.icarus.2018.06.018.
[12] Snellen Ignas AG, Brandl Bernhard R, Kok Remco Jde, Brogi Matteo, Birkby Jayne, Schwarz Henriette. Fast spin of the young extrasolar planet β Pictoris b. Nature May 2014;509(7498):63–5. doi:10.1038/nature13253.
[13] Watson CA, de Mooij EJW, Steeghs D, Marsh TR, Brogi M, Gibson NP, Matthews S. Doppler tomography as a tool for detecting exoplanet atmospheres. Mon Not R Astron Soc December 1, 2019;490(2):1991–2006. doi:10.1093/mnras/stz2679.
[14] Pannier Erwan, Laux Christophe O. Analysis of the jaxa nonequilibrium infrared emission spectra for Mars entry conditions. J Thermophys Heat Transfer October 2019;33(4):1127–31. doi:10.2514/1.T5646.
[15] Klarenaar BLM, Engeln R, van den Bekerom DCM, van de Sanden MCM, Morillo-Candas AS, Guaitella O. Time evolution of vibrational temperatures in a CO 2 glow discharge measured with infrared absorption spectroscopy. Plasma Sources Sci Technol October 26, 2017;26(11):115008. doi:10.1088/1361-6595/aa902e.
[16] Bekerom Dirk van den, Niek den Harder Teofil, Minea Nicola, Gatti Jose Palomares, Linares Waldo, Bongers Richard van de Sanden, van Rooij Gerard. Non-equilibrium microwave plasma for efficient high temperature chemistry. JoVE (J Vis Exper) August 1, 2017(126):e55066. doi:10.3791/55066.
[17] Whiting EE. An empirical approximation to the Voigt profile. J Quant Spectrosc Radiat Transfer June 1968;8(6):1379–84. doi:10.1016/0022-4073(68)90081-2.
[18] Olivero JJ, Longbothum RL. Empirical fits to the voigt line width: a brief review. J Quant Spectrosc Radiat Transfer February 1, 1977;17(2):233–6. doi:10.1016/0022-4073(77)90161-3.
[19] Liu Yuyan, Lin Jieli, Huang Guangming, Guo Yuanqing, Duan Chuanxi. Simple empirical analytical approximation to the Voigt profile. JOSA B May 1, 2001;18(5):666–72. doi:10.1364/JOSAB.18.000666.
[20] Yurchenko Sergei N, Al-Refaie Ahmed F, Tennyson Jonathan. EXOCROSS: a general program for generating spectra from molecular line lists. Astronom Astrophys June 2018;614:A131. doi:10.1051/0004-6361/201732531.
[21] Humlíček J. An efficient method for evaluation of the complex probability function: the Voigt function and its derivatives. J Quant Spectrosc Radiat Transfer April 1, 1979;21(4):309–13. doi:10.1016/0022-4073(79)90062-1.
[22] Meadows VS, Crisp D. Ground-based near-infrared observations of the Venus nightside: the thermal structure and water abundance near the surface. J Geophys Res 1996;101-E2:4595–622. doi:10.1029/95JE03567.
[23] Tan X. An ultrafast line-by-line algorithm for calculating spectral transmittance and radiance. J Quant Spectrosc Radiat Transfer 2013;129:101–8. doi:10.1016/j.jqsrt.2013.05.036.
[24] Mendenhall Marcus H. Fast computation of Voigt functions vis Fourier Transforms. J Quant Spectrosc Radiat Transfer 2007;105:519–24. doi:10.1016/j.jqsrt.2006.11.014.
[25] Hargreaves Robert J, Gordon Iouli E, Rey Michael, Nikitin Andrei V, Tyuterev Vladimir G, Kochanov Roman V, Rothman Laurence S. An accurate, extensive, and practical line list of methane for the HITEMP database. Astrophys J Suppl Ser April 1, 2020;247(2):55. doi:10.3847/1538-4365/ab7a1a.
[26] Hargreaves RJ, Bernath PF, Bailey J, Dulick M. Empirical line lists and absorption cross sections for methane at high temperatures. Astrophys J 2015(8) 813-1. doi:10.1088/0004-637X/813/1/12.
[27] Rey M, Nikitin AV, Tyuterev VlG. Theoretical hot methane line lists up to T=2000K for astrophysical applications. Astrophys J 2014;2 789-1. doi:10.1088/0004-637X/789/1/2.
[28] Pannier Erwan, Laux Christophe O. RADIS: a nonequilibrium line-by-line radiative code for CO2 and HITRAN-like database species. J Quant Spectrosc Radiat Transfer January 1, 2019;222–223:12–25. doi:10.1016/j.jqsrt.2018.09.027.
[29] Ren Tao, Modest Michael F. Line-by-Line random-number database for monte carlo simulations of radiation in combustion system. J Heat Transfer February 1, 2019;141(2):022701. doi:10.1115/1.4041803.
[30] Rivière Philippe. Updated band model parameters for H2O, CO2, CH4 and CO radiation at high temperature. Int J Heat Mass Transf 2012:10.
[31] Schreier Franz. Computational aspects of speed-dependent Voigt profiles. J Quant Spectrosc Radiat Transfer January 1, 2017;187:44–53. doi:10.1016/j.jqsrt.2016.08.009.
[32] Ngo NH, Lisak D, Tran H, Hartmann J-M. An isolated line-shape model to go beyond the Voigt profile in spectroscopic databases and radiative transfer codes. J Quant Spectrosc Radiat Transfer November 1, 2013;129:89–100. doi:10.1016/j.jqsrt.2013.05.034.
[33] Cooley James W, Tukey John W. An algorithm for the machine calculation of complex Fourier series.. Math Comput 1965;19(90):297–301. doi:10.2307/2003354.
[34] Oppenheim Alan V, Schafer Ronald W, Buck John R. Discrete-Time signal processing. 2nd ed. Upper Saddle River, N.J: Prentice Hall; 1999.
[35] Iftekharuddin Khan M, Awwal Abdul. Circular convolution and zero padding. Field Guide to Image Processing March 21, 2012:27–8. doi:10.1117/3.923354.ch27.
[36] Pannier, Erwan. Radis/Radis. Python. 2018. Reprint, RADIS, 2020. https://github.com/radis/radis.
[37] Silva, M.L.da. "The line-by-line radiative code SPARTAN," 2016. http://esther.ist.utl.pt/spartan/.
[38] Ng Edward W, Geller Murray. A table of integrals of the error functions. J Res Natl Bureau Standards Section B: Math Sci January 1969;73B(1):1. doi:10.6028/jres.073B001.