Line by Line Absorption Coefficient Solver Report

YUXIN WANG, EPFL, Switzerland
SUPERVISOR: DR. MANDY XIA, EPFL, Switzerland

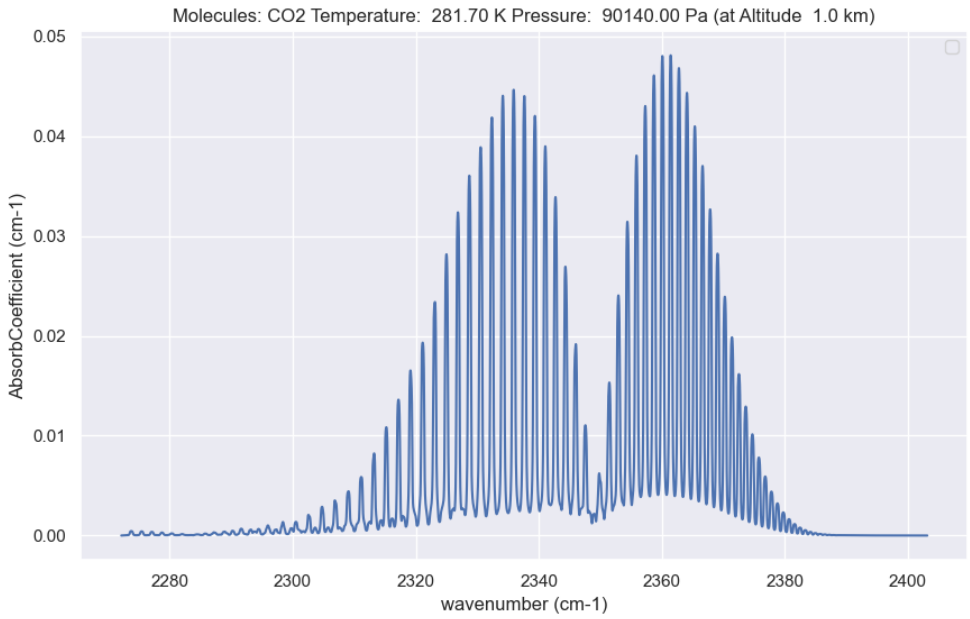

Fig. 1. The absorption of CO2 at altitude 1.0 km in the real atmosphere

Abstract This project tackles the issue of computing the line-by-line absorption coefficient, which is one of the complex challenges when simulating the Earth's atmosphere. The solution is implemented based on the Atmospheric Radiative Transfer Simulator book [2] and the Radis library [8] in Python. The absorption coefficient is computed using data from the Hitran line database. As a result, this might help to enable direct access to the measured data if we want to simulate some gas spectra. Simple optimizations is added to the implementation to improve computation speed while ensuring accuracy.

Additional Key Words and Phrases: line absorption coefficient, atmosphere gas absorption, spectroscopy, Radis, Atmospheric Radiative Transfer Simulator (ARTS)

Authors' addresses: Yuxin Wang, yuxin.wang@epfl.ch, EPFL, Lausanne, Vaud, Switzerland; Supervisor: Dr. Mandy Xia, mengqi.xia@epfl.ch, EPFL, Lausanne, Vaud, Switzerland.

Manuscript submitted to ACM

ACM Reference Format:
Yuxin Wang and Supervisor: Dr. Mandy Xia. 2023. Line by Line Absorption Coefficient Solver Report. 1, 1 (June 2023), 14 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Simulating the Earth's atmosphere throws out a difficult challenge because of the usually intricate and versatile processes occurring within the atmosphere. One crucial aspect of the atmospheric simulation is how to accurately account for the absorption of sunlight by atmospheric gases. This process is not easy primarily because there exists a high-oscillatory shape of gases in the spectral domain. To simulate this absorption properly, it is necessary to obtain a correct line-by-line absorption coefficient, which quantifies the gases' interaction within a specific range of light wavelengths. This procedure requires specialized solvers, like Radis[8], which is able to handle the conversion of molecular spectra data to absorption coefficients. These coefficients are necessary inputs for atmospheric simulation models, which allow scientists to study and understand the behavior of the Earth's atmosphere even when the conditions change a lot. The simulation models could also be applied in other areas like rendering.

Gas absorption involves both spectral lines (line absorption) and continua (continuum absorption) [2]. Continua can be either physically meaningful or empirical corrections for line-by-line calculations. Their magnitude depends on the specific setup of the line-by-line calculation. Hence, combining continua and line absorptions effectively requires expertise. In this project, I mainly focus on the part of line-by-line absorption at local thermodynamic equilibrium (LTE). The initial step starts by studying the basic concepts and then following computational procedures outlined in the Atmospheric Radiative Transfer Simulator (referred to as ARTS later) book and understanding the source code from Radis [7].

To verify the accuracy of the absorption coefficient calculated by my implemented approach, I utilized the line-by-line gas data from the Hitran database [3] and also the data from the real atmosphere at different altitudes. By the comparison between the absorption coefficient computed from Radis and that obtained from my implementation, it is easier to check if my implementation has any errors. Additionally, having access to the source code of Radis helps me to refine and complete my own implementation.

Furthermore, I also did some analysis the absorption coefficient calculation implementation to see if there could be any optimizations to achieve better performance in efficiency and memory usage. The goal is to reduce the computational time while not affecting the accuracy of the computed absorption coefficient. For now, the optimization is mainly based on removing nested loops to realize parallelization and make use of the third party which has better performance on certain calculations such as Fast Fourier Transform.

2 RELATED WORK

2.1 HITRAN

HITRAN (high-resolution transmission), as a molecular absorption database[3], is a comprehensive and up-to-date source of molecular spectroscopy for a wide range of atmospheric gases. HITRAN contains detailed information about molecular spectroscopic parameters, such as molecule isotopes, line strengths, state energies, pressure shifts and pressure-broadening coefficients, which are all essential for computing the absorption coefficient (absorbcoeff).

2.2 The Atmospheric Radiative Transfer Simulator (ARTS)

ARTS (Atmospheric Radiative Transfer Simulator) is a versatile radiative transfer model specifically designed for spectra[2]. It provides C++ code that could be used in many radiative transfer-related calculations in the microwave region including the gas absorption in our case. More importantly, the ARTS team offers a theory book to explain in detail the crucial steps of the absorbcoeff calculation, which is a great theoretical supplementary for me to better comprehend what is and how to obtain the absorbcoeff of gases.

2.3 RADIS

RADIS (Radiative Solver)[8] is a Python library that aims to do radiative transfer calculations and spectral data analysis, similar to ARTS. It develops a comprehensive set of functionalities that contain the precise line-by-line gas absorption calculation. The qualified performance on spectroscopic simulations of RADIS is a great guideline for my own implementation using Python. As RADIS has a user-friendly interface and the capability of accessing data also from HITRAN, I also used the result computed by it to be a good ground truth for the absorbcoeff computed by me.

3 BACKGROUND AND THEORY

According to ARTS[2], the line absorption of a gas molecule is summarized in one equation:

$$\alpha(\nu) = nS(T)F(\nu) \tag{1}$$

where $\boldsymbol{\alpha}$ represents the absorbcoeff. $\mathbf{S(T)}$ is the line strength. $\boldsymbol{F(\nu)}$ is the line shape. $\mathbf{T}$ is the desired temperature at which the absorption is computed at. $\boldsymbol{\nu}$ is the wavenumber. Note that ARTS uses frequency (unit as Hz) as the argument of this function but HITRAN provides the gas data in terms of wavenumber (unit as cm^{-1}), so in my project, the calculation of lines is based the argument as wavenumber. (the wavenumber could be converted from frequency by $light\ velocity \times frequency$)

Then, because of the additive property of absorption, the total absorbcoeff is sum of absorptions of all the lines. In the context of gas data from HITRAN, the Equation. 1 is applied to every line and finally add the results up to get the absorbcoeff of this gas. Accordingly, if there is more than one gas molecule to be computed, the total absorbcoeff is obtained by adding the $\boldsymbol{\alpha}$ of all molecules together.

In addition, only the state of local thermodynamic equilibrium(LTE) is considered in this project. In other words, conditions like temperature, energy states, and density of molecules are at equilibrium in the system where the effect of collisions between molecules outsteps the effect of radiation. So generally in LTE, simpler thermodynamic relationships take place in the simulation process of gas absorption.

3.1 Line Strength (S)

Line strength means the intensity or magnitude of the absorption line of the gas molecule. It indicates how much radiation a molecule absorbs at this line. Referred to [9], the line strength S could be computed as:

$$S(T) = S(T_{ref})\frac{Q(T_{ref})}{Q(T)}\frac{exp(-c_2E''/T)}{exp(-c_2E''/T_{ref})}\frac{[1-exp(-c_2\nu/T]}{[1-exp(-c_2\nu/T_{ref}]} \tag{2}$$

- $\boldsymbol{T_{ref}}$ = 296 K is a standardized room temperature as a constant.

- **Q** represents the partition function. It describes the distribution of gas molecular energy states[2], which depends on the temperature variable **T**. The value of **Q(T)** of any molecules and isotopologues could be directly read from the HITRAN database.
- c_2 = 1.4387769 (cm · K) is the second radiation constant, which is calculated as

$$\frac{h \times c}{k \times 100} \tag{3}$$

where **h** = 6.62607015 x 10^{-27} (erg[1] · s) is the Planck constant, **c** = 2.99792458 x 10^8 (m/s) is the light velocity, and **k** = 1.380649 × 10^{-16} (erg / K) is the Boltzmann constant.

- E'' is the lower state energy before the transition (like absorption or emission of radiation) of the molecule to a higher energy level, which is also available from HITRAN data.

3.2 Line shape (Broadening) (F)

Line shape refers to the spectral line profile which defines how the line intensity changes along the wavelength range. Factors like the finite lifetime of an excited state of an isolated molecule, the thermal motion of molecules, and collisions between molecules will influence the line shape. These influences end in natural broadening, **Doppler broadening**, and **pressure broadenin**g respectively. Doppler and pressure broadening have much more significance than natural broadening so only these two are applied in this project. [2]

In the end, the effects of Doppler and pressure broadening should be combined together which is called the **Voigt line shape**, to be multiplied by the line intensity **S**.

3.2.1 ***Pressure broadening*** *(Lorentz line shape).*

The Lorentz line shape could be computed by

$$F_L(\nu) = \frac{\gamma_L}{\pi} \frac{1}{(\nu - \nu_0)^2 + {\gamma_L}^2} \tag{4}$$

where γ_L is the half-width at half of the maximum (HWHM) of the Lorentz line. HWHM is a very important factor when computing the line profile as it measures the line width and resolution of spectral lines. The smaller HWHM gives a narrower and more resolved line; the larger HWHM gives a broader and less resolved line. So the HWHM of Lorentz at pressure **p** and temperature **T** is defined as (referred to [9]):

$$\gamma_L(p,T) = (\frac{T_{ref}}{T})^{n_{air}} (\gamma_{air}(p_{ref}, T_{ref})(p - p_{self}) + \gamma_{self}(p_{ref}, T_{ref})(p_{self}) \tag{5}$$

where the n_{air} is the coefficient of the temperature dependence of the air-broadened half-width. p_{self} is the partial pressure of the gas molecule.

3.2.2 ***Doppler broadening*** *(Gaussian line shape).*

The Gaussian line shape could be computed by

$$F_D(\nu) = \frac{\sqrt{\ln 2}}{\gamma_D \sqrt{\pi}} exp(-(\ln 2)(\frac{\nu - \nu_0}{\gamma_D})^2) \tag{6}$$

Similarly to the Lorentz profile, γ_D is the half-width at half of the maximum (HWHM) of the Gaussian line, which is defined as (referred to [2]):

[1] unit of energy, could be "J"

$$\gamma_D = \frac{\nu}{c}\sqrt{\frac{2kT\ln 2}{m}} \tag{7}$$

where **m** is the mass of the gas molecule (g). $\boldsymbol{m = M/Na}$, **M** is the molar mass (g/mol), and **Na** is the Avogadro constant.

3.2.3 ***Voigt line shape*** *(combination of Lorentz and Gaussian profile).*

The Voigt line profile is defined as:

$$F_{Voigt}(\nu, \nu_0) = \int F_L(\nu, \nu')F_D(\nu, \nu')d\nu' \tag{8}$$

This integral is not applicable to be computed in the real world which means it is a problem with no definite solutions [2]. Many different models and approaches ([4], [6], [5], [11]) have been carried out to approximate this Voigt line shape. However, my implementation design chose to follow the discrete integral transform algorithm from the RADIS team [10] because it is fast and reliable to produce accurate results. Please see more details in the *LineShape Broadening* subsection in the Implementation Details section.

3.3 Number density (n)

n in Equation. 1 stands for number density, i.e., the number of absorbing particles (**N**) per unit volume (**V** in cm^3). By applying the Ideal Gas Law [12], the equation of number density can be derived as:

$$\begin{aligned} pV &= n'RT \\ PV = n' \cdot NA \cdot k \cdot T &= N \cdot k \cdot T \\ \frac{P}{kT} &= \frac{N}{V} = n \end{aligned} \tag{9}$$

given that **R** is the gas constant = $\boldsymbol{k \cdot NA}$ (Avogadro constant); $\boldsymbol{n'}$ is the amount of substance and $N = n' \cdot NA$

4 IMPLEMENTATION DETAILS

Based on the theories above and the RADIS source code of calculating the absorption coefficient (absorbcoeff), the pipeline in my implementation including the major steps is as the pseudocode (1).

The details of the major steps in the implementation are explained in the following sections.

4.1 Loading Data

The molecular data (like the molecular mass, and the state energies) utilized for this project is mainly read from the online database HITRAN [3]; Real atmosphere data (e.g., the real temperature and pressure at the altitude of 50km) is from [1], a NetCDF data file.

4.1.1 ***HITRAN Data***.

As loading data from HITRAN is not the main issue I want to focus on in this project, I just utilized the *HITRANDatabaseManager* class from *radis.api.hitranapi* to load the dataframe given the inputs:

- molecule name: e.g., "CO2" or "H2O", which will be translated to numbers as a unique id for each molecule.
- isotope number: the desired isotope of this molecule. There could be as many isotopes as the user want. (e.g., "1" or "2, 3" or all available isotopes if "all")

Algorithm 1 AbsorbCoeff of gas molecules

```
altitude ← 50
pressure, temperature, molecules, molecule_fractions ← get_atm_gas_data(altitude)
abscoeff_res ← 0.0
for mol in molecules do
    df ← load_hitran(mol, isotope, wmin, wmax)
    wavenumber ← generate_wavenumber_range(wmin, wmax, wstep)
    if len(df) == 0 then
        continue
    end if
    calc_linestrength(df, Tgas, molecule)
    calc_lineshift(df, pressure)                    ▷ apply line center shift due to the pressure shift coefficient

    molar_mass ← get_molar_mass(df)
    calc_lorentz_hwhm(df, Tgas, diluent, mole_fraction, pressure)
    calc_gauss_hwhm(df, Tgas, molar_mass)

    line_profile_LDM ← calc_lineshape_LDM(df, wavenumber)        ▷ calculate the broadening (line shapes)
    sumoflines ← apply_lineshape_LDM(df['S'], wavenumber, line_profile_LDM)    ▷ compute S * F convolution

    number_density ← mole_fraction × ((pressure)/(k × Tgas))
    abscoeff ← number_density × sumoflines                                     ▷ compute n * S * F

    abscoeff_res ← abscoeff_res + abscoeff                                     ▷ absorption is additive
end for
```

Table 1. Example of the HITRAN data

id	iso	wav (ν)	int(S_{ref})	El (E'')	Pshft	airbrd(γ_{air})
0	1	0.00067	0.0021	345	-0.0039	0.0687
0	2	0.00077	0.0303	264	-0.002	0.0651
0	4	0.00087	0.0001	278	-0.0446	0.0554

- wavenum_min & wavenum_max: the minimum and maximum wavenumbers to define the range of line data in which the calculation takes place.

The loaded data is stored in a Pandas dataframe (mentioned as **df** later). Here is an example of what the data looks like in Table [1].

4.1.2 ***Real Atmosphere Data***.

In addition to the temperature and pressure at different altitudes (0km - 120km), this data also provides the real fraction of the seven major gas molecules in the atmosphere **[H2O, O3, N2O, CO, CH4, CO2, O2]** for each altitude. As the molecule fraction is also a necessary factor in the absorption coefficient computation especially when there is more than one molecule, this information with temperature and pressure are passed to the computation as inputs. And when computing the absorption of these real atmosphere gases, **all** isotopes of these gas molecules are taken into account.

4.1.3 ***Remarks*** *on gas data.*

Note that though the available spectral range of gases listed on HITRAN is very wide where the wavenumber could

be from less than 0.1 cm^{-1} to above 10000 cm^{-1}, there is no guarantee that HITRAN does provide data at every wavenumber. For example, from my test, HITRAN doesn't have any line data for "O2" in the range of wavenumber 1900-2400 cm^{-1} although this stays inside the range of O2'S line data on HITRAN.

As a consequence, this missing data of molecules will cause Radis to return *"Empty Database"* error immediately. However, when I computed the absorption of the real atmosphere data, it is hard to always find a range that ensures the non-empty line data provided for all 7 gases. Hence, I modified the source code of Radis to only return a warning if no line data for the queried wave range but continue on the calculation and sum up the absorption coefficients of the gases whose data is not empty. It is the same of how I deal with empty data in my implementation.

4.2 Line Strength Calculation

Referred to the equation of line strength (2) and the data fetched from HITRAN (Table. 1), $\boldsymbol{S(T)}$ is computed using the $\boldsymbol{Q_{ref}/Q_{gas}}$ ratio and other parameters from the dataframe, such as **df.int**, **df.El**, **df.wav**.

For the $\boldsymbol{Q(T_{ref})/Q(T_{gas})}$ ratio, HITRAN provides an api $\boldsymbol{partitionSum(molecule\_id, isotope, temperature)}$, so by passing T_{ref} and T_{gas} to the **partitionsum** function respectively and walk through the isotopes in **df**, this $Qref/Qgas$ ratio of all isotopes could be calculated.

4.3 Lorentz HWHM Calculation

From the Equation. 5, $\boldsymbol{\gamma_L(P,T)}$ is the sum of the air broadening part and the self-broadening part. The code is also implemented as adding up the two parts together. The partial pressure of the gas molecule is computed from its fraction:

$$p_{self} = molecule\_fraction \times pressure \tag{10}$$

p_{air} is calculated similarly with its faction equals $[1 - gas\_fraction]$. Hence, p_{air} and p_{self} are specific to each molecule. Other parameters could be extracted directly from the dataframe.

4.4 Gauss HWHM Calculation

The parameters to compute γ_D are either known constants or from the dataframe. Though the molar mass **M** is not inside the dataframe, HITRAN also provides a txt file including the molar mass of all the available molecules in its online database, so **M** of the gas molecule is easy to read from the txt file.

4.5 Add LineShape Broadening

As mentioned above, the Lorentz line shape and Gaussian line shape are combined together to be the Voigt line shape and multiplied by the line intensity. The custom way to achieve this is by approximating the Voigt lineshape for each line which is then combined with the line intensity and add up this calculation for all lines like Equation.11 (in the paper([10]))

$$\sum_i S_0^i Voigt(\nu - \nu_0^i; w_G^i, w_L^i) \tag{11}$$

where **i** means one line, $\boldsymbol{\nu_0^i}$ means the line center, $\boldsymbol{w_G}$ and $\boldsymbol{w_L}$ are the **gaussian HWHM** and **Lorentz HWHM**.

If there are a huge amount of lines in the data, this step becomes extremely computationally consuming. [10] comes up with an optimization where the convolution of lineshapes is converted to a discrete integral transform and is further transferred to be computed in the Fourier domain. [10] verified that this method is about 50 times faster than the original implementation of Radis while not decreasing the accuracy at all.

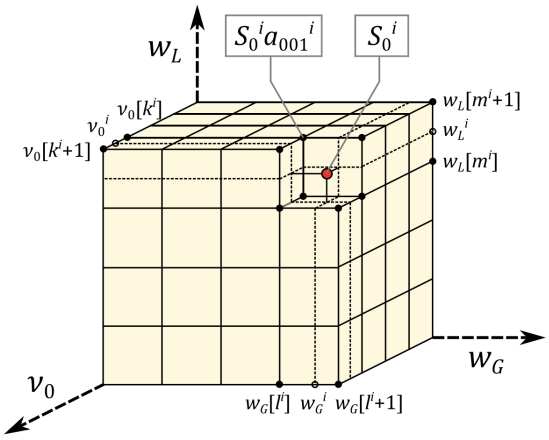

Fig. 2. Visualization of the LDM of $S^i * (v_0^i, w_G^i, w_L^i)$. $\boldsymbol{a_{001}^i}$ is the weight of $\boldsymbol{S_0^i}$ on the pointed gridpoint. **k, l, m** are the gridpoint indices for the line position grids in Fourier space (ν), Gaussian width grids, and Lorentz width grids respectively.

Knowing that the Voigt profile which depends on variables $(\boldsymbol{\nu}, \boldsymbol{w_G}, \boldsymbol{w_L})$ is different for different lines, Equation.11 could be written as a more general form in integral transform:

$$\iiint S(v_0, w_G, w_L) Voigt(\nu - v_0; w_G, w_L) dv_0 dw_G dw_L \tag{12}$$

And Equation.12 is further converted to a discrete transform to be computable:

$$\sum_k \sum_l \sum_m S_{klm} Voigt(v_n - v_k; w_{G,l}, w_{L,m}) \tag{13}$$

Based on this derivation, a discrete 3d grid is constructed with three axes of ν, w_G, and w_L. The Voigt profile is computed on each grid point and S_{klm} is the **Lineshape Distribution Matrix (LDM)** where the strength S^i for each line is distributed to $2 \times 2 \times 2 = 8$ nearest neighbor gridpoints and weighted by the interpolation of ν, w_G, and w_L.

4.5.1 *LineShape LDM Calculation*.

The goal is to construct the 3d matrix (see Fig. [2] from the paper [10]) with the wavenumbers ν as one axis and initialize the Gaussian and Lorentz width axes in the Fourier domain where the step of w_G and w_L axes are fixed for all molecules and the step of wavenumber on the grid could be user-defined. Then only the Voigt shape of the points on the w_G and w_L axes are computed. Because of the Fourier transform, the Voigt profile is the product of the Gaussian Fourier profile and Lorentz Fourier profile. As illustrated in the pseudocode [2], **line_profile_LDM** corresponds to $Voigt(\nu - v_0; w_G, w_L)$ in Equation.13.

4.5.2 *Apply LineShape LDM*.

In this step, the line strength should be (1) distributed to LDM; (2) also converted to the Fourier space and multiplied by the Voigt profile constructed in the previous section on the grids; (3) and the combination result is converted back to the real space. As illustrated by the pseudocode [3], **sumoflines** corresponds to the result of Equation. 13.

Algorithm 2 Voigt lineshapes in Fourier space along the wavenumber, gaussian & Lorentz width axes

$wL_{\text{dat}} \leftarrow$ df.hwhm_lorentz.values $\times 2$ ▷ full half width at half max (FWHM)
$wG_{\text{dat}} \leftarrow$ df.hwhm_gauss.values $\times 2$ ▷ FWHM
$wL \leftarrow$ _init_w_axis(wL_{dat}, step_L)
$wG \leftarrow$ _init_w_axis(wG_{dat}, step_G)

$line\_profile\_LDM \leftarrow \{\}$
$w\_lineshape\_ft \leftarrow$ numpy.fft.rfftfreq($2 \times$ len($wavenumber$), step_w)
for l in range(len(wG)) **do**
  $line\_profile\_LDM[l] \leftarrow \{\}$
  for m in range(len(wL)) **do** ▷ compute the effect of hwhm to the line shape in Fourier space directly
    $line\_profile\_LDM[l][m] \leftarrow$ calc_voigt_ft($w\_lineshape\_ft, wG[l]/2, wL[m]/2$)

    $line\_profile\_LDM[l][m] \leftarrow line\_profile\_LDM[l][m]/line\_profile\_LDM[l][m][0]$ ▷ normalization based on the Fourier frequency of 0
  end for
end for

Algorithm 3 Apply lineshapes to the line strength in LDM

$(ki0, ki1, avi) \leftarrow$ get_indices($wavenum$, wavenumber_range)
$(li0, li1, aGi) \leftarrow$ get_indices(log(wG_dat), log(wG))
$(mi0, mi1, aLi) \leftarrow$ get_indices(log(wL_dat), log(wL))
$awV00 \leftarrow (1 - aGi) \cdot (1 - aLi)$
$awV01 \leftarrow (1 - aGi) \cdot aLi$
$awV10 \leftarrow aGi \cdot (1 - aLi)$
$awV11 \leftarrow aGi \cdot aLi$
$Iv0 \leftarrow S \cdot (1 - avi)$
$Iv1 \leftarrow S \cdot avi$ ▷ interpolation of the line strength
$LDM \leftarrow$ numpy.zeros($2 \times$ len($wavenumber\_range$), len(wG), len(wL))
_add_at($LDM, ki0, li0, mi0, Iv0 \cdot awV00$) ▷ Distribute all line intensities on the 2x2x2 bins.
_add_at($LDM, ki0, li0, mi1, Iv0 \cdot awV01$)
... ▷ Repeat for all bins
_add_at($LDM, ki1, li1, mi1, Iv1 \cdot awV11$)

$Ildm\_FT \leftarrow 1j \cdot$ numpy.zeros(len(line_profile_LDM[0][0]))
for l in range(len(wG)) **do**
  for m in range(len(wL)) **do**
    $lineshape\_FT \leftarrow$ line_profile_LDM[l][m]
    $Ildm\_FT \leftarrow Ildm\_FT +$ numpy.fft.rfft($LDM[:, l, m]$) $\cdot lineshape\_FT$
  end for
end for
$sumoflines \leftarrow$ numpy.fft.irfft($Ildm\_FT$)

4.6 Testing

The tests here aim to validate the computation accuracy by comparing the differences in the absorption coefficient returned by using the RADIS library and that computed by my implementation both in

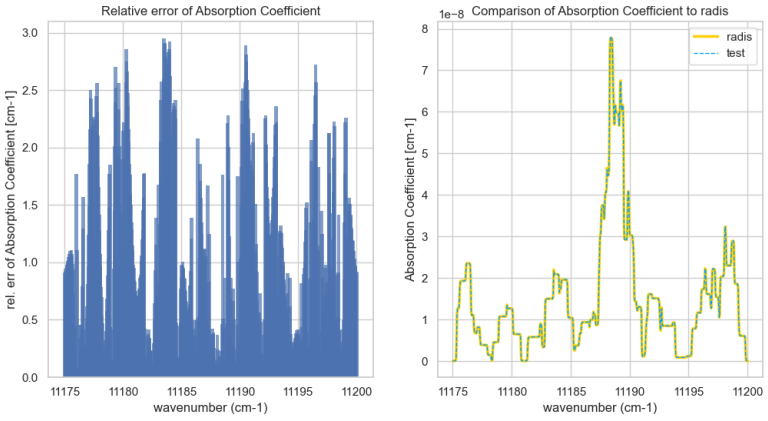

Fig. 3. Test on CO2_CH4_T500_P5560

(1) the quantitative values of the absorption coefficients. The difference should be less than a threshold (set to 5×10^{-3} for now).
(2) the shape of the plot of absorption coefficient versus the wavenumber. The shapes of results from RADIS and mine should look similar and have a lot of overlaps.

The code is tested using different data (including different molecules, multiple isotopes, various wavelengths range, various temperatures and pressures).

Fig. [3] shows an example of the plot from the test on **molecules = "CO2, CH4"**, **isotopes = {"CO2: "3, 4", CH4: "2"}**, **wavelength** from 892nm-896nm, **temperature = 500K**, **pressure = 5560 Pa**. The left bar chart is a visualization of the quantitative differences which refers to the relative error of the absorbcoeff (rel.err = absolute(absorbcoeff from RADIS - absorbcoeff by me)). On the right is the line chart of the absorbcoeff related to the wavenumber. Although there are small differences as shown by the relative error bar chart, the total shape of the absorbcoeff is almost the same as the absorbcoeff from RADIS as we can see from the well-overlapped blue and yellow lines.

5 OPTIMIZATIONS

By recording and observing the running time of the computation process, the step of convolving the Lorentz and Gaussian line shape (applying the Lineshape LDM) is the most time-consuming as it computes many fourier trnasform of a large array inside the nested loops (see Alg. [3]). The second most time-consuming step is when constructing the Lineshape LDM. Therefore, I mainly considered about how to optimize the code of these two steps.

5.1 Optimizing Applying LDM

The bottleneck is mainly from the nested for-loops to compute the discrete Fourier Transform for arrays on each grid point $[l, m]$. To unravel the loop, the $rfft$ could just be applied to the oringinal 3d matrix LDM along the correct axis. And then multiply LDM and the $line\_profile\_LDM$ matrix directly to do convolution. Hence, the for-loops become a one-line code as

$$Ildm\_FT += sum((rfft(np.array(LDM), axis = 0).transpose(1, 2, 0)) * line\_profile\_LDM, axis = (0, 1)) \quad (14)$$

5.2 Optimizing Calculating LDM

The main issue of this step is some repeated calculations inside the for-loops (see Alg. [2]). Inside the ***calc_voigt_ft*** function, the voigt profile is computed as the product of ***gauss_ft(wG[l])*** and ***lorentz_ft(wL[m])***. This implies the value ***gauss_ft(wG[l])*** of every l is recomputed the number of inner loops times which should stay the same for every inner loop. Similarly, the value ***lorentz_ft(wL[m])*** of every m is recomputed the number of outer loops times which shoulf stay the same for every outer loop. Therefore, I precomputed and saved all different ***gauss_ft(wG[l])*** and ***lorentz_ft(wL[m])***. Then in the loops, the program can just extract the values at the corresponding index of l and m, like Alg. [4].

Algorithm 4 Optimized Calculating Voigt Line Profile LDM

```
gauss_out_prod ← outer_product(wG/2, w_lineshape_ft)
gauss_profile ← guass_ft_func(gauss_out_prod)              ▷ shape: len(wG) x w
lorentz_out_prod ← outer_product(wL/2, w_lineshape_ft)
lorentz_profile ← lorentz_ft_func(lorentz_out_prod)        ▷ shape: len(wL) x w
line_profile_LDM: 3D matrix of shape (wg_len, wl_len, w_lineshape_ft_len)
for l in range(len(wG)) do
    for m in range(len(wL)) do
        line_profile_LDM[l][m] ← gauss_profile[l] × lorentz_profile[m]
    end for
end for
```

However, when avoiding the loops and repeated computations, new vairables as 2d or 3d matrices are created. Since there usually exist over thousands of lines for each molecule, the size of the matrices could be extremely large. As a result, the too large memory space taken to store the precomputed values might cause the program performance to be worse than using loops. The table. [2] shows the running time and memory usage of using loops and the optimized code on several tests:

(1) test1: molecule "CO, CO2", isotope = "CO2: 1, CO: 3", wavelength range: 4165 - 5265 nm
(2) test2: molecule "H2O, O3, N2O, CO, CH4, CO2, O2", isotope = "all", wavelength range: 4165 - 5000 nm
(3) test3: molecule "CO2", isotope = "all", wavelength: 750 - 1500 nm (wider range usually includes larger data size)

"Avg. Peak Memory" in the table represents the peak memory occupied in the computation process on the average of all molecules. The computation in *test1* and *test2* do not induce a very huge memory usage compared to *test3* which is over 3500 MB. So the speeding up by optimization performs well, but the optimization is affected a lot by the memory usage in *test3*, so the computation is not faster. This is a drawback caused by speeding up on top of creating a much larger memory allocation. So it would be valuable to find a way to attain a better tradeoff between the computation efficiency and memory so that the computation of larger data size could also run faster than the original implementation.

6 RESULTS AND VISUALIZATION

As a result, the user can input [any molecules with the corresponding isotopes and molecule fractions at any temperature (T) and pressure (P), and wavelength range the user is interested in]. Fig. [4] is a visualization of the computed result of **H2O** at T = 300 K and the standard atmospheric pressure 101.325 kPa: absorbcoeff of all isotopes is in the orange line while the green line represents the absorbcoeff when only considering the isotope = "3" of H2O assuming the fraction of H2O in the air is 0.1. It can be seen that isotope "1" is the major isotope in this absorption.

Table 2. Performance on several tests

test	calc_LDM_time (s)		apply_LDM_time(s)		total_time(s)		Avg. Peak Memory(MB)	
	loop	optimize	loop	optimize	loop	optimize	loop	optimize
test1	0.011	0.007	0.286	0.132	0.367	0.195	3.13	29
test2	0.451	0.218	12.642	4.609	13.958	5.111	36	328
test3	0.884	1.682	45.188	114.551	46.197	117.623	331	3579

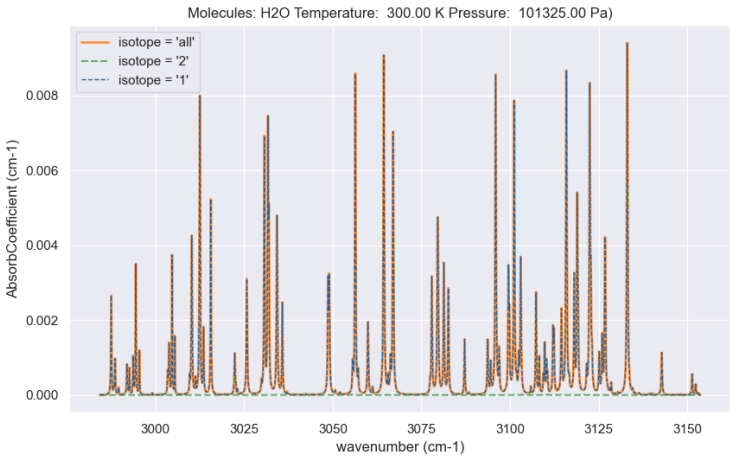

Fig. 4. Absorbcoeff of H2O_T300_P101325 in differnt isotopes

To view the absorption in the real atmosphere, data from the real atmospheric dataset[1] is input. Fig. [5] shows what the absorbcoeff looks like for the 7 gases **[H2O, O3, N2O, CO, CH4, CO2, O2]** with all isotopes considered in the atmosphere condition at an altitude of 50 km (the red line) or 60 km (the blue line). The wavelength range taken is 785 - 900 nm (wavenumber as 11100 - 12700 cm^{-1}) because we are interested in the absorption of infrared rays.

Despite the absorbcoeff, the parameters during the computation process like the line strength **S** could also be plotted as Fig. [6], which is the line strength of H2O at T = 300K temperature and P = 101.325kPa.

It is also possible to compare the absorption of different molecules, as pictured in Fig. [7]. The two lines belong to the absorbcoeff of **N2O** and **O3** when under the same physical conditions (T & P) and with the same molecule fraction = 0.2. The graph clearly shows that the absorption of N2O is much larger than O3 in this range.

7 DISCUSSION AND CONCLUSION

In summary, this project addresses the intricate problem of computing the line-by-line absorption coefficient for realistic atmospheric simulations. By leveraging established computational methodologies, accessing atmosphere gas data, and optimizing the implementation, it aims to enhance the accuracy and efficiency of atmospheric simulations, opening doors to new insights and further investigations into possible new ways of dealing with gas absorption in simulation.

The current implementation could calculate the absorption coefficient of mixed gas molecules at a certain temperature, pressure, and in a given wavelength range. However, there are still limitations to the wave range selected. On one hand,

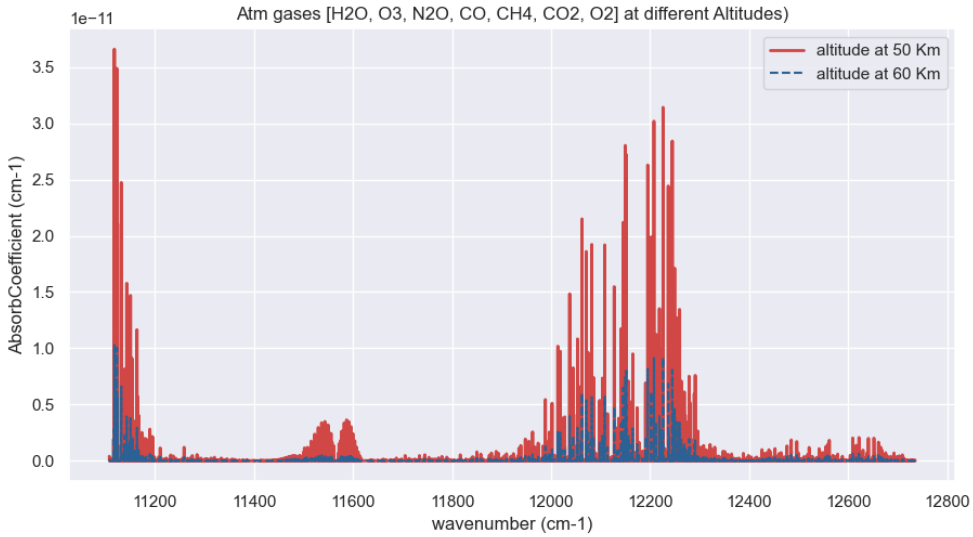

Fig. 5. Absorbcoeff of H2O_T300_P101325 in differnt isotopes

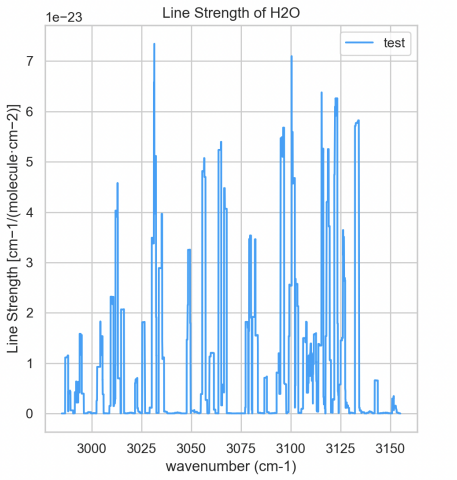

Fig. 6. Line strength vs wavenumber

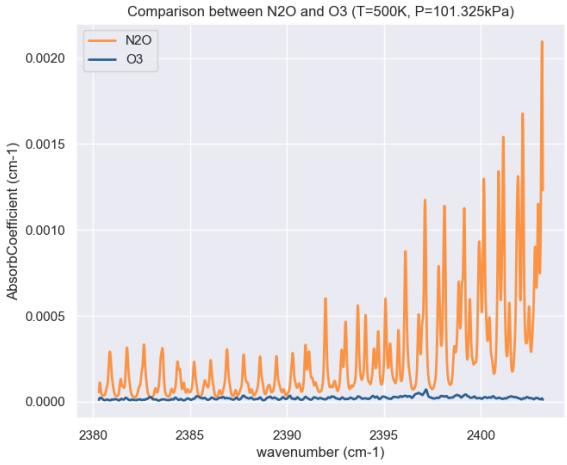

Fig. 7. Absorbcoeff of N2O and O3

HITRAN actually does not contain full data at every wavenumber, which is addressed in the *Loading data* section. On another hand, the optimized code is not efficient for all cases. For instance, it doesn't perform well when computing the absorption for a very wide wave range (like wavelength from 750 nm to 3500 nm) as the large line data size in such a range requires a huge memory during the optimized program. Due to the large occupation of used memory, the

performance of the program is dragged down. Therefore, it remains as future work to either achieve a better balance between memory usage and computation speed or develop other ways of optimization.

REFERENCES

[1] afgl_1986 us_standard.nc. 1986. *Real Atmosphere Gases Data of Altitudes from 0km to 120km.* https://github.com/Yuxin-99/line-absorbcoeff-solver/blob/main/Database/afgl_1986-us_standard.nc

[2] Stefan Buehler, Cory Davis, Claudia Emde, Patrick Eriksson, Nikolay Koulev, Thomas Kuhn, Oliver Lemke, Christian Melsheimer, and Manfred Brath. 2022. *ARTS Theory* (2.5.4 ed.). Chapter 2. https://atmtools.github.io/arts-docs-master/uguide/arts_theory.pdf

[3] HITRAN core team. 1973-2023. *HITRANonline, a high-resolution transmission molecular absorption database.* https://hitran.org/

[4] S.R. Drayson. 1976. Rapid computation of the Voigt profile. *Journal of Quantitative Spectroscopy and Radiative Transfer* 16, 7 (1976), 611–614. https://doi.org/10.1016/0022-4073(76)90029-7

[5] Martin Kuntz and Michael Höpfner. 1999. Efficient line-by-line calculation of absorption coefficients. *Journal of Quantitative Spectroscopy and Radiative Transfer* 63, 1 (1999), 97–114. https://doi.org/10.1016/S0022-4073(98)00140-X

[6] J.J. Olivero and R.L. Longbothum. 1977. Empirical fits to the Voigt line width: A brief review. *Journal of Quantitative Spectroscopy and Radiative Transfer* 17, 2 (1977), 233–236. https://doi.org/10.1016/0022-4073(77)90161-3

[7] Erwan Pannier and Christophe O. Laux. 2019. RADIS: A nonequilibrium line-by-line radiative code for CO2 and HITRAN-like database species. *Journal of Quantitative Spectroscopy and Radiative Transfer* 222-223 (2019), 12–25. https://doi.org/10.1016/j.jqsrt.2018.09.027

[8] RADIS Core Team. [n. d.]. *RADIS: a fast line-by-line code for high resolution infrared molecular spectra.* https://radis.readthedocs.io/en/latest/index.html

[9] L.S. ROTHMAN, C.P. RINSLAND, A. GOLDMAN, S.T. MASSIE, D.P. EDWARDS, J-M. FLAUD, A. PERRIN, C. CAMY-PEYRET, V. DANA, J.-Y. MANDIN, J. SCHROEDER, A. MCCANN, R.R. GAMACHE, R.B. WATTSON, K. YOSHINO, K.V. CHANCE, K.W. JUCKS, L.R. BROWN, V. NEMTCHINOV, and P. VARANASI. 1998. THE HITRAN MOLECULAR SPECTROSCOPIC DATABASE AND HAWKS (HITRAN ATMOSPHERIC WORKSTATION): 1996 EDITION. *Journal of Quantitative Spectroscopy and Radiative Transfer* 60, 5 (1998), 665–710. https://doi.org/10.1016/S0022-4073(98)00078-8

[10] D.C.M. van den Bekerom and E. Pannier. 2021. A discrete integral transform for rapid spectral synthesis. *Journal of Quantitative Spectroscopy and Radiative Transfer* 261 (2021), 107476. https://doi.org/10.1016/j.jqsrt.2020.107476

[11] E.E. Whiting. 1968. An empirical approximation to the Voigt profile. *Journal of Quantitative Spectroscopy and Radiative Transfer* 8, 6 (1968), 1379–1384. https://doi.org/10.1016/0022-4073(68)90081-2

[12] Benoît Paul Émile Clapeyron. 1834. *Ideal Gas Law (the general gas equation).* https://en.wikipedia.org/wiki/Ideal_gas_law