The absorption models of ARTS 2.6

Richard Larsson

June 5, 2024

Overview

Propagation matrix formalism

All the methods

Absorption

- This talk is about all the different absorption models of ARTS 2.6.
- I will quickly go through almost all of them, but will only show examples of a few.

Point-to-point radiative transfer

Derivative form

The core expression of ARTS to propagate the radiation through the atmosphere

$$\frac{d\vec{I}}{dr} = -\mathbf{K}\left(\vec{I} - \vec{J}\right)$$

relies on the propagation matrix $\mathbf{K}$ and the source vector $\vec{J}$ to change the radiation $\vec{I}$ as it propagates through the atmosphere.

Point-to-point form

If $\mathbf{K}$ and $\vec{J}$ are constant between 2 points

$$\vec{I}_{i+1} = \exp\left(-\mathbf{K}r\right)\left(\vec{I}_i - \vec{J}\right) + \vec{J}$$

The Stokes Vector

Unit vectors

$$\begin{array}{llcl}
\text{Vertical:} & \hat{e}_v & = & \begin{bmatrix} 0 & 1 & 0 \end{bmatrix} \parallel [\ \uparrow\ ], \\
\text{Horizontal:} & \hat{e}_h & = & \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \parallel [\ \rightarrow\ ], \\
\text{Plus } 45^\circ\text{:} & \hat{e}_{+45^\circ} & = & \begin{bmatrix} \cos\left(\frac{\pi}{4}\right) & \sin\left(\frac{\pi}{4}\right) & 0 \end{bmatrix} \parallel [\ \nearrow\ ], \\
\text{Minus } 45^\circ\text{:} & \hat{e}_{-45^\circ} & = & \begin{bmatrix} \cos\left(\frac{7\pi}{4}\right) & \sin\left(\frac{7\pi}{4}\right) & 0 \end{bmatrix} \parallel [\ \searrow\ ], \\
\text{Left Circular:} & \hat{e}_{lc} & = & \begin{bmatrix} 0 & 0 & -1 \end{bmatrix} \parallel [\ \circlearrowleft\ ], \\
\text{Right Circular:} & \hat{e}_{rc} & = & \begin{bmatrix} 0 & 0 & 1 \end{bmatrix} \parallel [\ \circlearrowright\ ].
\end{array}$$

Polarized Stokes Vector

$$\vec{I} = [I_v + I_h \vee \cdots,\ I_v - I_h,\ I_{+45^\circ} - I_{-45^\circ},\ I_{lc} - I_{rc}]^\top$$

The Propagation Matrix

The Symmetry of the Propagation Matrix

$$\mathbf{K} = \begin{bmatrix} A & B & C & D \\ B & A & U & V \\ C & -U & A & W \\ D & -V & -W & A \end{bmatrix},$$

Examples

- If the propagation does not polarize, only A is used.
- Zeeman effect makes use of all 7 components.
- Other physics use a subset, like Faraday rotation is purely in U

Algorithm for propagation matrix exponential

To compute the matrix exponential, it is possible to split this into the diagonal and non-diagonal parts:

$$\exp\left(-Ar\right)\exp\left(\mathbf{K}^{'}\right),$$

where the latter part is the non-diagonal part of the matrix. Using Cayley-Hamilton theorem, we can write this as

$$\exp\left(\mathbf{K}^{'}\right) = c_0\mathbf{I} + c_1\mathbf{K}^{'} + c_2{\mathbf{K}^{'}}^2 + c_3{\mathbf{K}^{'}}^3,$$

where the eigenvalues can be found from the characteristic polynomial and a bit of math.

Algorithm for propagation matrix exponential (cont 1)

Eigenvalues (note that capitalized variables are pre-scaled with r):

$$\begin{aligned}
0 &= \lambda^4 + b\lambda^2 + c \\
b &= U^2 + V^2 + W^2 - B^2 - C^2 - D^2 \\
c &= -(DU - CV + BW)^2 \\
s &= \sqrt{b^2 - 4c} \\
x^2 &= \sqrt{\frac{s-b}{2}} \\
y^2 &= \sqrt{\frac{s+b}{2}}
\end{aligned} .$$

Solve:

$$\begin{array}{lcccccccc}
e^{x} & = & c_0 & + & c_1 x & + & c_2 x^2 & + & c_3 x^3 \\
e^{-x} & = & c_0 & - & c_1 x & + & c_2 x^2 & - & c_3 x^3 \\
e^{iy} & = & c_0 & + & i c_1 y & - & c_2 y^2 & - & i c_3 y^3 \\
e^{-iy} & = & c_0 & - & i c_1 y & - & c_2 y^2 & + & i c_3 y^3
\end{array} .$$

Algorithm for propagation matrix exponential (cont 2)

Yield:

$$\begin{aligned}
c_0 &= \frac{x^2 \cos y + y^2 \cosh x}{x^2+y^2} \\
c_1 &= \frac{x^2 \frac{\sin y}{y} + y^2 \frac{\sinh x}{x}}{x^2+y^2} \\
c_2 &= \frac{\cosh x - \cos y}{x^2+y^2} \\
c_3 &= \frac{\frac{\sinh x}{x} - \frac{\sin y}{y}}{x^2+y^2}.
\end{aligned}$$

Limits:

$$\begin{aligned}
c_0 :&\quad \lim_{x\to 0} c_0 = && \lim_{y\to 0} c_0 && = 1 \\
c_1 :&\quad \lim_{x\to 0} c_1 = && \lim_{y\to 0} c_1 && = 1 \\
c_2 :&\quad && \lim_{x\to 0 \wedge y \to 0} c_2 && = \frac{1}{2} \\
c_3 :&\quad && \lim_{x\to 0 \wedge y \to 0} c_3 && = \frac{1}{6} \\
&\quad \vee && \lim_{x\to 0} c_3 && = \frac{1}{y^2} - \frac{\sin y}{y^3} \\
&\quad \vee && \lim_{y\to 0} c_3 && = \frac{\sinh x}{x^3} - \frac{1}{x^2}.
\end{aligned}$$

Summing up

This is how the propagation matrix is computed in ARTS:

$$\mathbf{K} = \sum_i \mathbf{K}_i$$

- It is simply summed up.
- Each contribution to the propagation matrix is assumed to be independent.
- Each contribution is computed separately.
- So what can we compute in ARTS?

All the methods

Absorption model type	ARTS method name	○ × ●
Particulate absorption	propmat_clearskyAddParticles	○
Faraday rotation	propmat_clearskyAddFaraday	○
Lookup tables	propmat_clearskyAddFromLookup	○
Predefined absorption models	propmat_clearskyAddPredefined	●
Collision-induced absorption	propmat_clearskyAddCIA	●
Cross-section fitting	propmat_clearskyAddXsecFit	●
Line-by-line absorption (LBL)	propmat_clearskyAddLines	○
LBL - Line mixing (HITRAN)	propmat_clearskyAddHitranLineMixingLines	×
LBL - Line mixing (on-the-fly)	propmat_clearskyAddOnTheFlyLineMixing	×
LBL - Zeeman effect	propmat_clearskyAddZeeman	●

Before that

- There is a new method in ARTS 2.6 we did not have before: propmat_clearsky_agendaAuto. It analyses your species selection and line-by-line data and sets the agenda for you.
- Another new method for ARTS 2.6 is our pure python method pyarts.cat.download.retrieve. It downloads the corresponding absorption catalog data for you.
- Finally, if you want to just explore the absorption models, we have a new simple GUI in ARTS 2.6: propmat_clearsky_agendaGUI.

Particulate absorption in ARTS

- This is not my expertise. This module is used when the assumption that particulate scattering is just away from the path.
- You choose between adding the unpolarized absorption vector to A or to add the full extinction matrix to the full propagation matrix.

Faraday effect

The Faraday effect in ARTS is treated as a pure rotation of the circular polarization state. This rotation is given by

$$U = \left| \frac{n_e e^3}{4\pi^2 c \epsilon_0 m_e^2 \nu^2} \right| \vec{r} \cdot \vec{B},$$

where n_e is the number of electrons, e is the electron charge, c is the speed of light, ϵ_0 is the vacuum permittivity, m_e is the electron mass, ν is the frequency of the radiation, $\vec{r}$ is the line-of-sight vector of the radiation, and $\vec{B}$ is the magnetic field vector.

Lookup table calculations in ARTS

The lookup table calculations are used to interpolate data from a precomputed cross-section table. The core expression is an interpolation routine over:

$$\alpha_u = n_s LUP(\nu, P, T, VMR, H_2O)$$

As always, the absorption coefficient is

$$A = \sum_u \alpha_c$$

Predefined absorption models in ARTS

- A collection of models that does not fit in anywhere else, and that are useful for absorption calculations.
- These are vastly different but can be put into some categories:
  - Full line-by-line models.
  - A liquid water absorption model.
  - Continuum absorption models.
- Note that they are all based on our own porting of the original code or model and NOT the original code.
- Also note that it is not possible to combine them freely with other calculations of ARTS.
- Finally, all of these models are to date unpolarized.

Predefined line-by-line absorption models in ARTS

These are the models that are available in ARTS.

- Oxygen
  - Rosenkranz's models for oxygen. (v 1998, v 2021, v 2022).
  - Liebe's models (v 1989, and v 2020)
  - Tretyakov's model (v 2005)
- Water
  - Rosenkranz models (v 1998, v 2021, v 2022)
  - Libe's model (v 1989)

Predefined continuum absorption models in ARTS

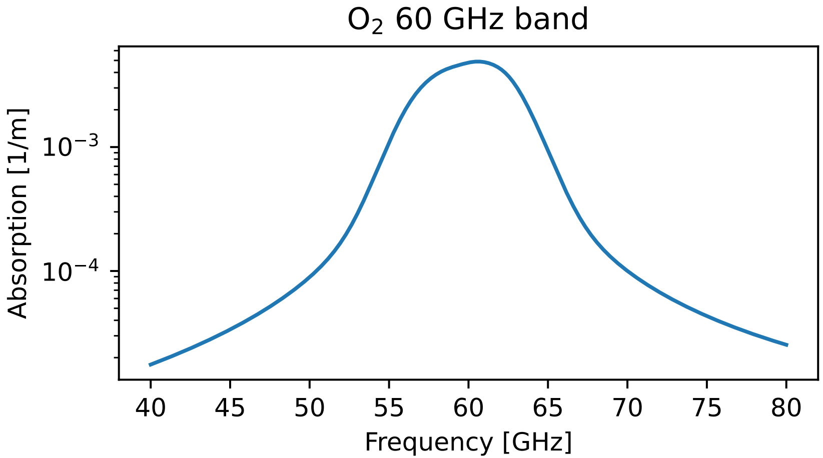
These are the available models in ARTS.

- Carbon dioxide
  - MTCKD (v 2.52)
- Water
  - MTCKD (v 2.52, v 3.2, v 3.5, v 4)
  - Rosenkranz (v 1998)
- Oxygen
  - MTCKD (v 1, v 2.52; CIA)
  - Rosenkranz (v 1998)
- Nitrogen
  - MTCKD (v 2.52; CIA)
  - Rosenkranz (v 1998)

Example of a Predefined model in ARTS

```
ws = pyarts.Workspace()
ws.jacobianOff()
ws.stokes_dim = 1
ws.rtp_pressure = 1e5
ws.rtp_temperature = 250.
ws.rtp_vmr = [0.21]
ws.rtp_mag = [50e-6, 0, 0]
ws.rtp_los = [0, 0]

ws.abs_speciesSet(species=["O2-PWR2022"])
ws.propmat_clearsky_agenda_checked = 1

ws.f_grid = np.linspace(40e9, 80e9, 1001)
ws.propmat_clearskyInit()
ws.propmat_clearskyAddPredefined()
```

Example of a Predefined model in ARTS



Collision-induced absorption in ARTS

- Two molecules that "collide" can generate a dipole that absorbs and emits radiation. This is called collision-induced absorption (CIA).

- For a pair of species s_1 and s_2 the absorption coefficient is given by:

$$\alpha_c = n_{s_1} n_{s_2} CIA(T, \nu),$$

where n_{s_1} and n_{s_2} are the number densities of the species and $CIA(T, \nu)$ is an interpolation routine in temperature and frequency based on input data. As always, the absorption coefficient is

$$A = \sum_c \alpha_c$$

CIA models in ARTS 2.6

The catalog database contains the following models:

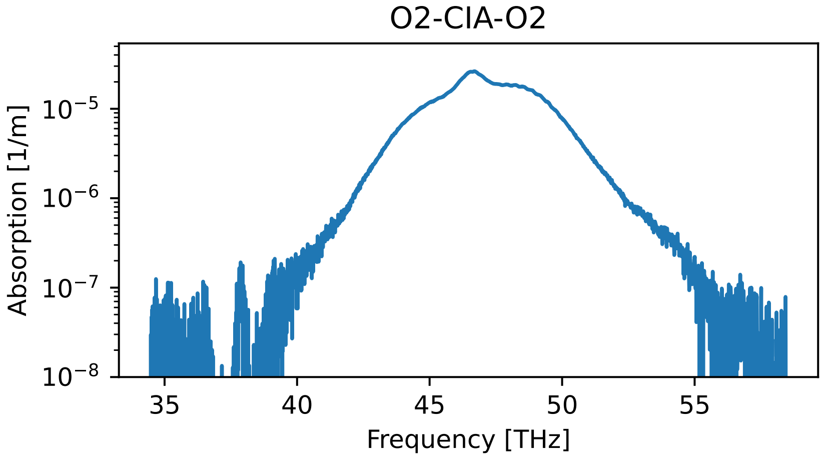
CH4-CIA-He	CO2-CIA-Ar	CO2-CIA-CH4
CO2-CIA-CO2	CO2-CIA-H2	CO2-CIA-He
H2-CIA-CH4	H2-CIA-H	H2-CIA-H2
H2-CIA-He	He-CIA-H	N2-CIA-H2
N2-CIA-H2O	N2-CIA-He	N2-CIA-N2
O2-CIA-CO2	O2-CIA-N2	O2-CIA-O2

All based on HITRAN data (adapted to ARTS).

Example of CIA in ARTS

```
ws = pyarts.Workspace()
ws.jacobianOff()
ws.stokes_dim = 1
ws.rtp_pressure = 1e5
ws.rtp_temperature = 250.
ws.rtp_vmr = [0.21]

ws.abs_speciesSet(species=["O2-CIA-O2"])
ws.propmat_clearsky_agenda_checked = 1

ws.abs_cia_data = [pyarts.arts.CIARecord.fromxml("cia/O2-CIA-O2.xml")]
ws.f_grid = ws.abs_cia_data.value[0].data[0].grids[0]
ws.propmat_clearskyInit()
ws.propmat_clearskyAddCIA()
```

Example of CIA in ARTS



Cross-section fitting in ARTS

The cross-section fitting in ARTS is based on the HITRAN database where a lot of effort has gone into massaging the data to a useful form. The core expression is:

$$\alpha_x = n_s \left(p_{00} + x p_{10} + y p_{01} + x^2 p_{20} \right) ,$$

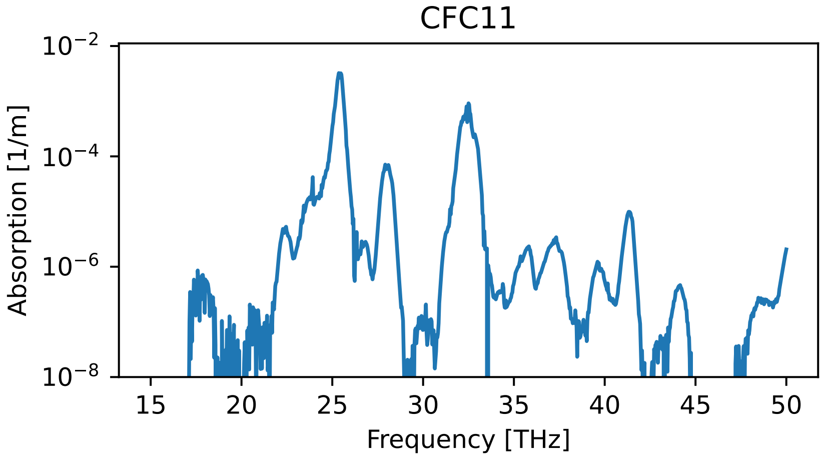
with $x = T / T_0$ and $y = \frac{P}{P_0}$, and p_{ij} are the best-fit coefficients. The absorption coefficient is then

$$A = \sum_x \alpha_x$$

Example of cross-section fitting in ARTS

```python
ws = pyarts.Workspace()
ws.jacobianOff()
ws.stokes_dim = 1
ws.rtp_pressure = 1e5
ws.rtp_temperature = 250.
ws.rtp_vmr = [220e-9]

ws.abs_speciesSet(species=["CFC11-XFIT"])
ws.propmat_clearsky_agenda_checked = 1

ws.ReadXsecData(basename="xsec/")
ws.f_grid = np.linspace(15e12, 50e12, 1001)
ws.propmat_clearskyInit()
ws.propmat_clearskyAddXsecFit()
```

Example of cross-section fitting in ARTS



LBL in ARTS

The core expression

$$\alpha_l = S_l(T, p, \cdots) N_l(\nu, \cdots) F_l(\nu, \cdots),$$

where S is some line strength operator, N is a line shape normalization operator, and F is the line shape operator.

Unpolarized

$$A = \sum_l \mathrm{Re}\left(\alpha_l\right)$$

$B = C = D = U = V = W = 0$

LBL - The operators

- The line strength S_l operator is the line strength operator. It can be LTE or non-LTE (by temperature or ratio). It can contain line mixing or not (first order or second order Rosenkranz approximations).
- The line-normalization operator N_l comes from a choice of options. You should probably always use SFS:

$$N_l = \frac{\nu}{\nu_0}\left(\frac{1 - \exp(-h\nu/kT)}{1 - \exp(-h\nu_0/kT)}\right)$$

- The line shape operator F_l is the line shape operator. It can be one of Doppler, Lorentz, Voigt, speed-dependent Voigt, or Hartmann-Tran profile. All data we provide is Voigt profile because even though it is well known that line shapes are not symmetric, the Voigt profile data is most of what is available.

LBL not in ARTS

Refraction

The expression for refraction is

$$R = \sum_l \mathrm{Im}\left(\alpha_l\right)$$

such that:

$$n = 1 + \frac{R}{2}.$$

(I am not sure about the factor 1/2, it comes up in literature sometimes.) However, we do not use this in ARTS as it is not clear 1) how to sum up R for non-line-by-line models and 2) if the expression itself is accurate enoguh for all purposes. It would be interesting to find out.

LBL - Zeeman effect

The change in line strength

$$S_l(T, p, \cdots) = C \begin{pmatrix} J'' & 1 & J' \\ M_J'' & \Delta M & -M_J' \end{pmatrix}^2 S_l'(T, p, \cdots),$$

where $C = 1.5$ if $\Delta M = 0$ and $C = 0.75$ otherwise. J and M are quantum numbers of the absorption line. Most databases provide J and the Zeeman effect needs to be computed for the pairs in $\{M_J'' : -J'', \cdots, J''\}$ and $\{M_J' : -J', \cdots, J'\}$ with $|\Delta M| \leq 1$. Note: $\Delta M = M_J' - M_J''$.

The change in line shape

$$F_l(\nu, \cdots) = F_l'(\nu - \frac{\mu_B}{h}\left(g_{J''} M_J'' - g_{J'} M_J'\right) ||\vec{B}||, \cdots)$$

LBL - Zeeman effect

Polarization state

$$\begin{aligned}
A &= \sum_l^{\text{if } |\Delta M|=1} \mathrm{Re}\left(\alpha_l\right)\left(1+\cos^2\theta\right) &+& \sum_l^{\text{if } |\Delta M|=0} \mathrm{Re}\left(\alpha_l\right)\sin^2\theta \\
B &= \sum_l^{\text{if } |\Delta M|=1} \mathrm{Re}\left(\alpha_l\right)\left(\cos 2\eta\sin^2\theta\right) &-& \sum_l^{\text{if } |\Delta M|=0} \mathrm{Re}\left(\alpha_l\right)\left(\cos 2\eta\sin^2\theta\right) \\
C &= \sum_l^{\text{if } |\Delta M|=1} \mathrm{Re}\left(\alpha_l\right)\left(\sin 2\eta\sin^2\theta\right) &-& \sum_l^{\text{if } |\Delta M|=0} \mathrm{Re}\left(\alpha_l\right)\left(\sin 2\eta\sin^2\theta\right) \\
D &= \sum_l^{\text{if } |\Delta M|=1} 2\Delta M \mathrm{Re}\left(\alpha_l\right)\left(\cos\theta\right) && \\
U &= \sum_l^{\text{if } |\Delta M|=1} 4\Delta M \mathrm{Im}\left(\alpha_l\right)\left(\cos\theta\right) && \\
V &= \sum_l^{\text{if } |\Delta M|=1} 2\mathrm{Im}\left(\alpha_l\right)\left(\sin 2\eta\sin^2\theta\right) &-& \sum_l^{\text{if } |\Delta M|=0} 2\mathrm{Im}\left(\alpha_l\right)\left(\sin 2\eta\sin^2\theta\right) \\
W &= \sum_l^{\text{if } |\Delta M|=1} 2\mathrm{Im}\left(\alpha_l\right)\left(\cos 2\eta\sin^2\theta\right) &-& \sum_l^{\text{if } |\Delta M|=0} 2\mathrm{Im}\left(\alpha_l\right)\left(\cos 2\eta\sin^2\theta\right)
\end{aligned}$$

How to compute g_J for O$_2$

$$\begin{aligned}
H^{J-1,J-1}_{\eta\Lambda SJ} =& \langle \eta, [(N=J-1,\Lambda)N=J-1,S]JM_J|\hat{H}_{\text{eff}}|\eta,[(N=J-1,\Lambda)N=J-1,S]JM_J\rangle \\
=& BJ(J-1) - DJ^2(J-1)^2 + HJ^3(J-1)^3 \\
&+ \left[\gamma + \gamma_D J(J-1) + \gamma_H J^2(J-1)^2\right](J-1) \\
&+ \left[\lambda + \lambda_D J(J-1) + \lambda_H J^2(J-1)^2\right]\left(\frac{2}{3} - \frac{2J}{2J+1}\right) \\
H^{J+1,J+1}_{\eta\Lambda SJ} =& \langle \eta, [(N=J+1,\Lambda)N=J+1,S]JM_J|\hat{H}_{\text{eff}}|\eta,[(N=J+1,\Lambda)N=J+1,S]JM_J\rangle \\
=& B(J+2)(J+1) - D(J+2)^2(J+1)^2 + H(J+2)^3(J+1)^3 \\
&- \left[\gamma + \gamma_D (J+2)(J+1) + \gamma_H (J+2)^2(J+1)^2\right](J+2) \\
&+ \left[\lambda + \lambda_D (J+2)(J+1) + \lambda_H (J+2)^2(J+1)^2\right]\left(\frac{2}{3} - \frac{2(J+1)}{2J+1}\right) \\
H^{J-1,J+1}_{\eta\Lambda SJ} =& \langle \eta, [(N=J-1,\Lambda)N=J-1,S]JM_J|\hat{H}_{\text{eff}}|\eta,[(N=J+1,\Lambda)N=J+1,S]JM_J\rangle \\
=& H^{J+1,J-1}_{\eta\Lambda SJ} = \left[\lambda + \lambda_D(J^2+J+1) + \lambda_H(J^2+J+1)^2\right]\frac{2\sqrt{J(J+1)}}{2J+1},
\end{aligned} \tag{8}$$

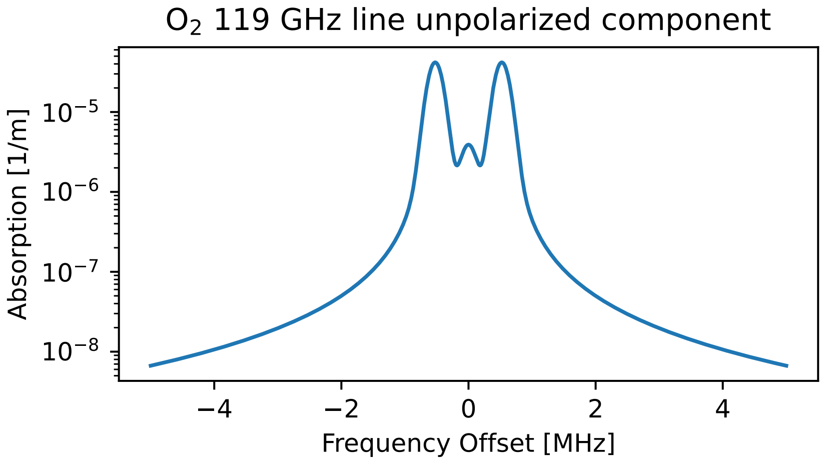
$$\tan(2\phi_{\eta\Lambda SJ}) = \frac{2H^{J+1,J-1}_{\eta\Lambda SJ}}{H^{J-1,J-1}_{\eta\Lambda SJ} - H^{J+1,J+1}_{\eta\Lambda SJ}}.$$

$$\begin{aligned}
g_{J_{N=J-1}} &= (g_S+g_r)\left[\frac{\cos^2(\phi_{\eta\Lambda SJ})}{J} - \frac{\sin^2(\phi_{\eta\Lambda SJ})}{J+1}\right] + \frac{2g^e_l\cos(2\phi_{\eta\Lambda SJ})}{2J+1} - g_r, \\
g_{J_{N=J}} &= \frac{g_S+g_r}{J(J+1)} - g_r, \\
g_{J_{N=J+1}} &= (g_S+g_r)\left[\frac{\sin^2(\phi_{\eta\Lambda SJ})}{J} - \frac{\cos^2(\phi_{\eta\Lambda SJ})}{J+1}\right] - \frac{2g^e_l\cos(2\phi_{\eta\Lambda SJ})}{2J+1} - g_r.
\end{aligned} \tag{12}$$

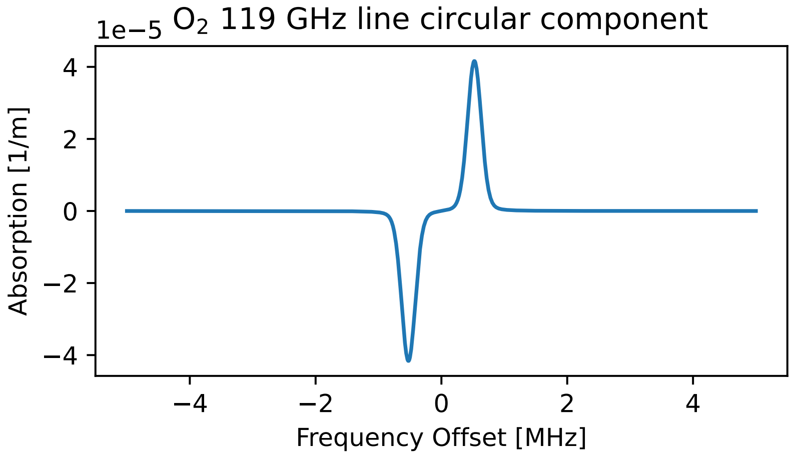
Example of LBL - Zeeman effect in ARTS

```
ws.rtp_pressure = 1e0
ws.rtp_temperature = 250.
ws.rtp_vmr = [0.21]
ws.rtp_mag = [10e-6, 20e-6, 30e-6]
ws.rtp_los = [30, 0]

ws.abs_speciesSet(species=["O2-Z-66-110e9-120e9"])
ws.propmat_clearsky_agenda_checked = 1

ws.abs_lines_per_speciesReadSpeciesSplitCatalog(basename="lines/")
ws.lbl_checked = 1
ws.Wigner6Init()
ws.f_grid = np.linspace(-5e6, 5e6, 1001) + line_f0
ws.propmat_clearskyInit()
ws.propmat_clearskyAddZeeman()
```

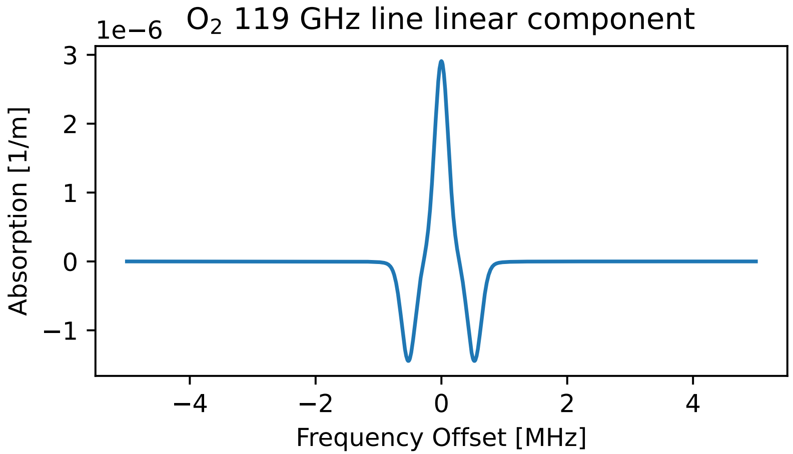
Example of LBL - Zeeman effect in ARTS



Example of LBL - Zeeman effect in ARTS



Example of LBL - Zeeman effect in ARTS



Questions?