Supplement of

How non-equilibrium aerosol chemistry impacts particle acidity: the GMXe AERosol CHEMistry (GMXe–AERCHEM, v1.0) sub-submodel of MESSy

Simon Rosanka et al.

Correspondence to: Simon Rosanka (srosanka@uci.edu, s.rosanka@fz-juelich.de)

The copyright of individual parts of the supplement might differ from the article licence.
Figure S1. Box-plots comparing observations of each observation network (EPA, EMEP, and EANET) with modeled values by ISORROPIA-II and AERCHEM for (a) sulfate ($\text{SO}_4^{2-}$), (b) nitrate ($\text{NO}_3^-$), (c) ammonium ($\text{NH}_4^+$), and (d) chloride ($\text{Cl}^-$). For each box-plot, the box extends from the first to the third quartile. The whiskers encompass data points lying within a range of 1.5 times the inter-quartile range. The black line and the black diamond represent the median and mean, respectively. To allow for an easy comparison between all datasets, outliers are not represented.
Figure S2. Mean yearly aerosol pH for the soluble accumulation mode (as) simulated by (a) ISORROPIA-II and by (b) AERCHEM. Subfigure (c) represents the absolute difference of the yearly means. Please note that for the figure showing the absolute pH differences, an increase in acidity (decrease in pH) is indicated by red shading, whereas an increase in pH is indicated in blue. For comparison, observed fine particle acidity, based on the dataset published by Pye (2020), is indicated by circles in subfigures (a) and (b).
Figure S3. Mean yearly aerosol pH for the soluble coarse mode (cs) simulated by (a) ISORROPIA-II and by (b) AERCHEM. Subfigure (c) represents the absolute difference of the yearly means. Please note that for the figure showing the absolute pH differences, an increase in acidity (decrease in pH) is indicated by red shading, whereas an increase in pH is indicated in blue. For comparison, observed fine particle acidity, based on the dataset published by Pye (2020), is indicated by circles in subfigures (a) and (b).
The representation of the formation and evolution of secondary organic aerosols (SOA) is computationally demanding. A detailed representation is thus outside the scope of this study. In order to be able to calculate the water uptake due to organic aerosols, we implement a simple computationally efficient sub-submodel in GMXe (GMXe-SOA) to calculate the formation of SOA from isoprene, α-pinene, β-pinene, toluene, and xylenes. This model follows the description presented by Tsigaridis and Kanakidou (2003), Zhang et al. (2007), and O’Donnell et al. (2011).

In short, we implement the two-product model for SOA formation based on the work of Odum et al. (1996). They demonstrated that for a reaction yielding multiple semi-volatile products, the aerosol yield can be represented by assuming that only two condensable species are produced (SP$_1$ and SP$_2$, respectively). For known mass-based stoichiometric coefficients for each product ($\alpha_1$ and $\alpha_2$, respectively), the gas-phase reaction of a given precursor (PRE, e.g., isoprene) with a given oxidant (OX, e.g., OH, O$_3$, NO$_3$) is given by:

\[ \text{PRE} + \text{OX} \rightarrow \alpha_1 \times \text{SP}_1 + \alpha_2 \times \text{SP}_2 \]  

(1)

Tables S1 and S2 provide a summary of all rate constants and SOA parameters, respectively.

For each condensable species (i) produced from VOC oxidation, the equilibrium between the gas and aerosol phase is governed by:

\[ A_i = K_i M_0 G_i \]  

(2)

where $K_i$ is the partitioning coefficient of the condensable species, $A_i$ and $G_i$ are the aerosol-phase and gas-phase concentrations of the condensable species, respectively. $M_0$ is the total particulate phase concentration, which is the sum of the primary particles (PP) that can serve as absorptive medium and the sum of the aerosol phase concentrations ($A_i$) of all condensable species:

\[ M_0 = PP + \sum_i A_i \]  

(3)

Here, we represent the bulk partitioning coefficient ($K_i(T)$) at a specific temperature (T) as:

\[ K_i(T) = K_{i,\text{ref}} \times \frac{T}{T_{\text{ref}}} \times \exp \left( \frac{\Delta H_i}{R} \left( \frac{1}{T} - \frac{1}{T_{\text{ref}}} \right) \right) \]  

(4)

where $K_{i,\text{ref}}$ is the reference partitioning coefficient at 298 K, $T_{\text{ref}}$ the reference temperature of 298 K, R the universal gas constant, and $\Delta H_i$ the enthalpy of vaporization of the specific compound. In this study, we assume $\Delta H_i$ to be 156 kJ mol$^{-1}$, the upper estimate discussed in Zhang et al. (2007). Table S2 lists all reference partitioning coefficients at 298 K ($K_{i,\text{ref}}$).

By assuming that the total amount of semi-volatile products repartitions in each time step and all SOA has the potential to fully evaporate from the particles at each time step and defining $S_i$ as the total mass of semi-volatile SOA of the given species ($S_i = A_i + G_i$), we can rearrange Eq. (2) to solve for the gas and aerosol mass to:

\[ G_i = \frac{S_i}{1 + K_i(T)M_0}, \text{and} \]  

\[ A_i = \frac{K_i(T)M_0S_i}{1 + K_i(T)M_0} \]  

(5)

(6)

Substituting equation (6) into equation (3), we can solve $M_0$ iteratively, since PP and $S_i$ are known.

O’Donnell et al. (2011) demonstrate that an equilibrium SOA partitioning between different size classes is consistent, when assuming unity for the activity coefficient for all SOA species, with the fraction of non-volatile absorbing aerosol mass in each mode:

\[ \frac{A_{i,\text{NV}}}{A_i} = \frac{M_{\text{NV},i}}{M_{\text{NV}}} \]  

(7)
where $M_{NV}$ is the total non-volatile absorbing mass, $A_i$ the total aerosol phase mass of SOA species $i$, $M_{NV,j}$ the non-volatile absorbing mass in mode $j$, and $A_{i,j}$ the aerosol phase mass of SOA species $i$ in mode $j$. This allows us to calculate the partitioning for each SOA species for each aerosol mode in which SOA partitioning is permitted.

In a last step, the water uptake due to each SOA species for each mode is calculated. Here, we assume a variable hygroscopicity parameter for each SOA species. Table S2 provides an overview on the hygroscopicity parameters used for each SOA species.

Table S1. Temperature dependent rate constants for particular precursors with OH, O$_3$, and NO$_3$. Unless noted, reaction rates are presented by the Arrhenius equation, given as: $K = A \times \exp\left(-\frac{E_a}{RT}\right)$, where $T$ is temperature in Kelvin (K), $A$ is given in s$^{-1}$, and $-E_a/R$ is given in K.

<table>
<thead>
<tr>
<th>Precursor</th>
<th>OH</th>
<th>O$_3$</th>
<th>NO$_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$A$</td>
<td>$-E_a/R$</td>
<td>$A$</td>
</tr>
<tr>
<td>Isoprene</td>
<td>$2.7 \times 10^{-11}$</td>
<td>$390$</td>
<td>$1.03 \times 10^{-14}$</td>
</tr>
<tr>
<td>$\alpha$-pinene</td>
<td>$1.21 \times 10^{-11}$</td>
<td>$444$</td>
<td>$1.01 \times 10^{-15}$</td>
</tr>
<tr>
<td>$\beta$-pinene</td>
<td>$2.38 \times 10^{-11}$</td>
<td>$357$</td>
<td>$1.5 \times 10^{-17}$</td>
</tr>
<tr>
<td>Toluene</td>
<td>$5.96 \times 10^{-11}$</td>
<td>$2.34 \times 10^{-12}$</td>
<td>$-6694$</td>
</tr>
<tr>
<td>Xylene</td>
<td>$1.72 \times 10^{-11}$</td>
<td>$a$</td>
<td>$a$</td>
</tr>
</tbody>
</table>

$^a$ Average of temperature dependence for ortho-, meta- and para-isomers: $(2.4 \times 10^{-13} \times \exp(-\frac{6586}{T}) + 5.37 \times 10^{-13} \times \exp(-\frac{6029}{T}) + 1.91 \times 10^{-13} \times \exp(-\frac{6586}{T}))/3$

Table S2. SOA parameters used for the implemented two-product model. $\alpha_1$ and $\alpha_2$ represent the stoichiometric coefficients and $K_{1,\text{ref}}$ and $K_{2,\text{ref}}$ represent the reference equilibrium partitioning coefficient, respectively. Values for $\alpha_1$, $\alpha_2$, $K_{1,\text{ref}}$, and $K_{2,\text{ref}}$ are taken from Zhang et al. (2007) and sources therein. $\kappa_1$ and $\kappa_2$ represent the hygroscopicity parameter for each SOA product. Hygroscopicity parameter taken from Pye et al. (2017) and sources therein.

<table>
<thead>
<tr>
<th>Precursor</th>
<th>$\alpha_1$</th>
<th>$\alpha_2$</th>
<th>$K_{1,\text{ref}}$ [m$^3$µg$^{-1}$]</th>
<th>$K_{2,\text{ref}}$ [m$^3$µg$^{-1}$]</th>
<th>$\kappa_1$</th>
<th>$\kappa_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Isoprene</td>
<td>0.232</td>
<td>0.0288</td>
<td>0.00459</td>
<td>0.8628</td>
<td>0.14</td>
<td>0.15</td>
</tr>
<tr>
<td>$\alpha$-pinene</td>
<td>0.0864</td>
<td>0.3857</td>
<td>1.1561</td>
<td>0.0847</td>
<td>0.10</td>
<td>0.10</td>
</tr>
<tr>
<td>$\beta$-pinene</td>
<td>0.0864</td>
<td>0.3857</td>
<td>1.1561</td>
<td>0.0847</td>
<td>0.10</td>
<td>0.10</td>
</tr>
<tr>
<td>Toluene</td>
<td>0.071</td>
<td>0.138</td>
<td>0.5829</td>
<td>0.0209</td>
<td>0.15</td>
<td>0.10</td>
</tr>
<tr>
<td>Xylene</td>
<td>0.038</td>
<td>0.167</td>
<td>0.4619</td>
<td>0.0154</td>
<td>0.14$^a$</td>
<td>0.14$^a$</td>
</tr>
</tbody>
</table>

$^a$ Average for all SOA products presented by Pye et al. (2017).
2 GMXe–AERCHEM and GMXe–SOA manual

MESSy includes a very comprehensive set of chemical reactions in both, the gas-phase and the aqueous-phase. To simplify the usage and enhance the consistency between all mechanisms used for the different phases (gas – MECCA, aqueous phase – SCAV, aerosol phase – GMXe–AERCHEM) the full mechanism is hosted within the MECCA submodel. Before compiling the MESSy code, the user is able to choose the required mechanisms. The selection of the aqueous-phase mechanism used in clouds and deliquescent aerosols needs to be performed separately. In order to make use of both GMXe–AERCHEM and GMXe–SOA the user needs to perform the following steps in the given order before compiling the MESSy code:

1. Select the in-cloud aqueous-phase mechanism and adjust the creation of cloud and aerosol tracer in SCAV
2. Select the aqueous-phase mechanism for deliquescent aerosols in GMXe–AERCHEM
3. Modify and select the gas-phase mechanism in MECCA
4. Adjust the GMXe setup for the usage of GMXe–AERCHEM and GMXe–SOA

2.1 Selecting the in-cloud aqueous-phase mechanism and related adjustments in SCAV

The choice of the in-cloud aqueous-phase mechanism employed is determined by the xscav script:

```bash
cd messy/mbm/scav/mechanism
./xscav
```

The script will prompt various queries with the intention of creating the mechanism. A general description of the creation script is provided in messy/mbm/caaba/manual/caaba_mecca_manual.pdf. It is strongly recommended to utilize the precompiled batch files which are outlined within the batch directory. For the simulation performed in this manuscript, the jamoc.bat was used in order to employ the JAMOC mechanism.

In order to utilize GMXe–AERCHEM, the SCAV submodel needs to make use of cloud tracer and evaporate all cloud species to aerosol tracers upon cloud evaporation. The creation of aerosol tracer in SCAV is enabled in the submodel interface layer. Before compiling the MESSy code, the user needs to ensure that the following logical switches are set to .TRUE. in the messy/smil/messy_scav_si.f90 file:

```fortran
! l_trac_aer: make aerosol tracers for kpp species; using this switch, 
! but not l_trac_aer_all will create tracers only for charged 
! compounds
l_trac_aer   = .TRUE.
```

In order to enable the creation of cloud tracer in SCAV, the user needs to enable the appropriate switch in the SCAV namelist (scav.nml):

```fortran
&CPL
[...]
! Create cloud tracer for all KPP species
l_trac_cloud   = T,
[...]
/
```

2.2 Selecting the aqueous-phase mechanism for deliquescent aerosols in GMXe–AERCHEM

The choice of the aqueous-phase mechanism employed for deliquescent aerosols is determined by the xaerchem script:
The script will prompt various queries with the intention of creating the mechanism. For an explanation of all options, the user may consult messy/mbm/caaba/manual/caaba_mecca_manual.pdf. It is recommended to consult the same general description manual. It is strongly recommended to utilize the precompiled batch files which are outlined within the batch directory. For the simulation performed in this manuscript, the jamoc.bat was used in order to employ the JAMOC mechanism.

2.3 Necessary changes to the MECCA submodel and selecting the gas-phase mechanism

In its standard setup, MECCA represents the reactive aerosol uptake of N$_2$O$_5$ across all aerosol modes. At the same time, the newly developed GMXe–AERCHEM sub-submodel also represents this uptake for each individual mode for which GMXe–AERCHEM is used. In order to avoid a double accounting of this process, the user needs to disable the uptake of N$_2$O$_5$ across all aerosol modes before selecting the gas-phase mechanism by excluding in messy/mbm/caaba/mecca/gas.eqn:

\[
\begin{align*}
<HET301> N2O5 &= 2 \text{ NO3m}_c + 2 \text{ Hp}_c : \{\text{TrHet} \} \text{khet}_\text{Tr}(iht\_N2O5)\{\$\}; \{\&\text{SGN}\} \\
<HET301a> N2O5 &= 2 \text{ NO3m}_n + 2 \text{ Hp}_n : \{\text{TrHet} \} \text{khet}_\text{Tr}(iht\_N2O5_n)\{\$\}; \{\&\text{SGN}\} \\
<HET301b> N2O5 &= 2 \text{ NO3m}_k + 2 \text{ Hp}_k : \{\text{TrHet} \} \text{khet}_\text{Tr}(iht\_N2O5_k)\{\$\}; \{\&\text{SGN}\} \\
<HET301c> N2O5 &= 2 \text{ NO3m}_a + 2 \text{ Hp}_a : \{\text{TrHet} \} \text{khet}_\text{Tr}(iht\_N2O5_a)\{\$\}; \{\&\text{SGN}\} \\
<HET301d> N2O5 &= 2 \text{ NO3m}_c + 2 \text{ Hp}_c : \{\text{TrHet} \} \text{khet}_\text{Tr}(iht\_N2O5_c)\{\$\}; \{\&\text{SGN}\}
\end{align*}
\]

and enable the mode specific uptake by including the following in the same file:

\[
\begin{align*}
<HET301a> N2O5 &= 2 \text{ NO3m}_n + 2 \text{ Hp}_n : \{\text{TrHet} \} \text{khet}_\text{Tr}(iht\_N2O5_n)\{\$\}; \{\&\text{SGN}\} \\
<HET301b> N2O5 &= 2 \text{ NO3m}_k + 2 \text{ Hp}_k : \{\text{TrHet} \} \text{khet}_\text{Tr}(iht\_N2O5_k)\{\$\}; \{\&\text{SGN}\} \\
<HET301c> N2O5 &= 2 \text{ NO3m}_a + 2 \text{ Hp}_a : \{\text{TrHet} \} \text{khet}_\text{Tr}(iht\_N2O5_a)\{\$\}; \{\&\text{SGN}\} \\
<HET301d> N2O5 &= 2 \text{ NO3m}_c + 2 \text{ Hp}_c : \{\text{TrHet} \} \text{khet}_\text{Tr}(iht\_N2O5_c)\{\$\}; \{\&\text{SGN}\}
\end{align*}
\]

In addition, the user needs to disable the calculation of the reactive aerosol uptake for each mode considered in GMXe–AERCHEM by adjusting mecca.nml. By default, GMXe–AERCHEM represents non-equilibrium aqueous-phase chemistry for the soluble accumulation (as, mode 3 in GMXe) and coarse (cs, mode 4 in GMXe) mode. Accordingly, mecca.nml needs to be adjusted to:

\[
\begin{align*}
&\text{CPL_KHET} \\
&\text{aerosol submodel and modes:} \\
&\text{asm(1) = 'gmxe', '1,2,5,6,7'} \\
&\
\end{align*}
\]

If other heterogeneous uptake processes are considered in MECCA, the user is advised to keep mecca.nml unchanged and exclude the uptake of N$_2$O$_5$ for each GMXe-AERCHEM mode in messy/mbm/caaba/mecca/gas.eqn. For a standard GMXe-AERCHEM setup the uptake of N$_2$O$_5$ in MECCA would only include:

\[
\begin{align*}
<HET301a> N2O5 &= 2 \text{ NO3m}_n + 2 \text{ Hp}_n : \{\text{TrHet} \} \text{khet}_\text{Tr}(iht\_N2O5_n)\{\$\}; \{\&\text{SGN}\} \\
<HET301b> N2O5 &= 2 \text{ NO3m}_k + 2 \text{ Hp}_k : \{\text{TrHet} \} \text{khet}_\text{Tr}(iht\_N2O5_k)\{\$\}; \{\&\text{SGN}\}
\end{align*}
\]

The choice of the gas-phase mechanism employed is determined by the xmecca script:

\[
\begin{align*}
&\text{cd messy/mbm/caaba/mecca} \\
&./xmecca
\end{align*}
\]

The script will prompt various queries with the intention of creating the mechanism. For an explanation of all options, the user may consult messy/mbm/caaba/manual/caaba_mecca_manual.pdf. It is strongly recommended to utilize the precompiled batch files which are outlined within the ini directory. For the simulation performed in this manuscript, the mecca_mom3d.ini was used in order to employ the MOM mechanism.
2.4 GMXe setup for GMXe–AERCHEM and GMXe–SOA

Adhering to the general methodology employed by MESSy for managing submodels (Jöckel et al., 2006, 2010) all necessary controls to utilize both newly implemented sub-submodels (i.e., GMXe–AERCHEM and GMXe–SOA) are established within the GMXe namelist (gmxe.nml). Namelist S1 provides an overview of the recommended GMXe configuration when employing GMXe–AERCHEM and GMXe–SOA. In the following, each setting is described in detail. The supplemental material of Pringle et al. (2010) provides a detailed description of all GMXe options not discussed in this section.

2.4.1 GMXe–AERCHEM sub-submodel options

The GMXe–AERCHEM sub-submodel is enabled by using the appropriate logical switch in the control setup:

```FORTRAN
&CTRL
[...]
1_aerchem = T, ! calculate aerosol chemistry using a kpp based reaction mechanism
[...]
/
```

In its default setup, all GMXe calculations are performed in the radiation routine of MESSy. When using GMXe–AERCHEM, all GMXe calculations need to be performed in the `physc` routine of MESSy to represent the intended operator splitting described in the main manuscript (see Figure 1).

```FORTRAN
&CPL
[...]
driver_call = 'physc' ! Swich indicated where GMXe_driver is called from
                       ! (radiation or physc)
[...]
/
```

When using GMXe–AERCHEM, all aerosols are assumed to be in a metastable state. This is ensured by using the following setting:

```FORTRAN
&CTRL_GMXE_TD
[...]
LHYSTER = F, ! Inclusion of hysteresis effect (includes solids/metastable)
[...]
/
```

All additional GMXe–AERCHEM settings are defined in its control instant (`CTRL_GMXE_AERCHEM`):

- lmode: Defines the lower mode for which the KPP routine will be executed. The default values is 3 representing the soluble accumulation mode (as).
- umode: Defines the upper mode for which the KPP routine will be executed. The default values is 4 representing the soluble coarse mode (cs).

2.4.2 GMXe–SOA sub-submodel options

The GMXe–SOA sub-submodel is enabled by using the appropriate logical switch in the control setup:

```FORTRAN
&CTRL
[...]
l_soa = T, ! calculate SOA with an explicit SOA model
[...]
/
```
All additional GMXe–SOA settings are defined in its control instant (CTRL_GMXE_SOA):

- `lmode_soa`: Defines the lower mode for which the SOA calculations will be executed. The default value is 2 representing the soluble aitken mode (ks).

- `umode_soa`: Defines the upper mode for which the SOA calculations will be executed. The default value is 5 representing the insoluble aitken mode (ki).

- `N_GASPREC(:, :)`: Defines the tracer name of each SOA precursor considered in the SOA calculations. GMXe–SOA will create the tracer, if it is not part of the gas-phase mechanism. If the tracer is not part of the gas-phase mechanism, ensure that `l_GASPREC(:, 2)` is set to true (T) for the precursor. In addition, appropriate emissions need to be implemented via the respective submodel.

- `l_GASPREC(:, 1)`: Logical switch defining if the SOA calculations will be performed for the specific precursor.

- `l_GASPREC(:, 2)`: Logical switch to decide whether total chemical degradation or only degradation due to SOA formation should be considered. If set to T, the total loss is considered.

- `l_OXI(:, :)`: Logical switch for oxidants (i.e., O₃, OH, and NO₃) to consider in the SOA calculation.

- `EXCL_STR_SOA(:, :)`: Tracer name of species to be excluded for bulk SOA uptake. By default, only mineral dust (DU) is excluded.
Namelist S1. Example GMXe setup when using GMXe-AERCHEM

1: ! -+ f90 -+ !
2: !
3: &CTRL
4: ! <=- GMXe general options
5: !
6: LGMXe = T , ! Aerosol composition scheme GMXe (interface)
7: LOUTPUT = T , ! Write GMXe channel to diagnose output
8: LMASS_DIAG = F , ! Mass balance check in gmXe_physc
9: LSTRAT = F , ! Exclude stratosphere (requires TROPOP channel)
10: LPSC = T , ! Exclude PSC region (requires PSC channel)
11: LNUCL = T , ! Calculate nucleation of aerosol particles
12: LCOND = T , ! Calculate amount of condensable species
13: ! due to kinetic limitations otherwise distribute equally across modes
14: LCOAG = T , ! Reshape the size distribution due to particle
15: ! coagulation (if LSIZE=.F.)
16: LCOAT = T , ! Calculate coating/ageing of primary particles
17: ! (requires THERMO interface (NEQM > −1))
18: LCLC = F , ! Update cloud cover
19: LADYN = F , ! Calculate aerosol dynamics (limits EQSAM3 gas/liquid/solid equilibration time by
20: ! transport time scales)
21: LCLWC = F , ! Update cloud liquid water concentration (sets LCDNC = T)
22: LCIWC = F , ! Update cloud ice water concentration (sets LICNC = T)
23: LCDNC = F , ! Update cloud droplet number concentration (CLWC calc. by ECHAM5)
24: LICNC = F , ! Update ice crystal number concentration (CIWC calc. by ECHAM5)
25: LGAS = T , ! Update gas phase chemistry - Gases : H3PO4, H2SO4, HNO3, HCl, HBr, HI, H2CO3
26: C2H2O4, C6H8O7, Hg, NH3
27: LAEROSOL = T , ! Update aerosol phase chemistry - Cations : H+, NH4+, Na+, K+, Mg++, Ca++, Fe++/Fe+++!
28: C2H2O4, C6H8O7, Hg, NH3
29: ! Anions : PO4−−−, SO4−−, HSO4−, NO3, Cl−, Br−, I−
30: −, CO3−−, HCO3−, OH−, CH2O−−,
31: ! Solute : NH3, C3H6O, C3H6O, C6H12O6,
32: C6H13O5, C12H22O11,
33: ! Gases : H3PO4, H2SO4, HNO3, HCl, HBr, HI, H2CO3
34: C2H2O4, C6H8O7, Hg, NH3
35: ! Note - Within EQSAM3 all major cation/anion combinations as well as gases
36: ! and (neutral) solutes are considered for
37: ! the aerosol water uptake;
All these compounds, but especially the salts are important for the aerosol water mass and hygroscopic growth due their different RH regimes, where they are able to absorb or desorb water vapor from the atmosphere (deliquescence/efflorescence relative humidity, respectively). Resolving the hygroscopic growth is important, as this determines the aerosols and trace gases, as well as the interaction of the aerosols with sunlight (direct forcing) and clouds (indirect forcing).

LNUMBER = F, ! T = Calculate aerosol numbers from aerosol mass; the GMXe channel contains a channel element 'number' ! F = Transport aerosol numbers as tracers (initial aerosol numbers are assigned from emissions, etc)

LWETRAD = T, ! T = Calculate wet aerosol radius from EQSAM3 thermodynamics; F from bulk thermodynamics

LDRYRAD = T, ! T = Calculate dry aerosol radius from EQSAM3 thermodynamics; F from bulk thermodynamics

LDRYDENS = T, ! T = Calculate dry aerosol density from EQSAM3 thermodynamics; F from bulk thermodynamics

LEVAP = T, ! Calculate H2SO4 evaporation (default T)

LCALCPM = T, ! Calculate Particulate Matter PM1/2.5/10 (default F, requires LCALCPM = T)

LDRYDENS = T, ! T = Calculate dry aerosol density from EQSAM3 thermodynamics; F from bulk thermodynamics

LEVAP = T, ! Calculate H2SO4 evaporation (default T)

LCALCPM = T, ! Calculate Particulate Matter PM1/2.5/10 (default F, requires LCALCPM = T)

LDRYDENS = T, ! T = Calculate dry aerosol density from EQSAM3 thermodynamics; F from bulk thermodynamics

LEVAP = T, ! Calculate H2SO4 evaporation (default T)

LCALCPM = T, ! Calculate Particulate Matter PM1/2.5/10 (default F, requires LCALCPM = T)

LDRYDENS = T, ! T = Calculate dry aerosol density from EQSAM3 thermodynamics; F from bulk thermodynamics

LEVAP = T, ! Calculate H2SO4 evaporation (default T)

LCALCPM = T, ! Calculate Particulate Matter PM1/2.5/10 (default F, requires LCALCPM = T)

LDRYDENS = T, ! T = Calculate dry aerosol density from EQSAM3 thermodynamics; F from bulk thermodynamics

LEVAP = T, ! Calculate H2SO4 evaporation (default T)

LCALCPM = T, ! Calculate Particulate Matter PM1/2.5/10 (default F, requires LCALCPM = T)

LDRYDENS = T, ! T = Calculate dry aerosol density from EQSAM3 thermodynamics; F from bulk thermodynamics

LEVAP = T, ! Calculate H2SO4 evaporation (default T)

LCALCPM = T, ! Calculate Particulate Matter PM1/2.5/10 (default F, requires LCALCPM = T)

LDRYDENS = T, ! T = Calculate dry aerosol density from EQSAM3 thermodynamics; F from bulk thermodynamics

LEVAP = T, ! Calculate H2SO4 evaporation (default T)

LCALCPM = T, ! Calculate Particulate Matter PM1/2.5/10 (default F, requires LCALCPM = T)

LDRYDENS = T, ! T = Calculate dry aerosol density from EQSAM3 thermodynamics; F from bulk thermodynamics

LEVAP = T, ! Calculate H2SO4 evaporation (default T)

LCALCPM = T, ! Calculate Particulate Matter PM1/2.5/10 (default F, requires LCALCPM = T)

LDRYDENS = T, ! T = Calculate dry aerosol density from EQSAM3 thermodynamics; F from bulk thermodynamics

LEVAP = T, ! Calculate H2SO4 evaporation (default T)

LCALCPM = T, ! Calculate Particulate Matter PM1/2.5/10 (default F, requires LCALCPM = T)

LDRYDENS = T, ! T = Calculate dry aerosol density from EQSAM3 thermodynamics; F from bulk thermodynamics

LEVAP = T, ! Calculate H2SO4 evaporation (default T)

LCALCPM = T, ! Calculate Particulate Matter PM1/2.5/10 (default F, requires LCALCPM = T)

LDRYDENS = T, ! T = Calculate dry aerosol density from EQSAM3 thermodynamics; F from bulk thermodynamics

LEVAP = T, ! Calculate H2SO4 evaporation (default T)
! string for sigma must contain the width of each mode, separated by ";"
! If the number of entries does not match nmod, the simulation will be terminated
sigma_nml = '1.59;1.59;1.59;2.0;1.59;1.59;2.0'

! string for crdiv must contain the lower boundary of each mode in [cm],
! separated by ";"
! If the number of entries does not match nmod, the simulation will be terminated
crdiv_nml = '0.0005e-4;0.006e-4;0.06e-4;0.7e-4;0.06e-4;0.06e-4;0.7e-4'

! string for cmodes must contain the names for each mode, separated by ";
! If the number of entries does not match nmod, the simulation will be terminated
Note, that many submodels coupling to aerosol species use small letters...
cmodes_nml = 'ns;ks;as;cs;ki;ai;ci'

! enter all gas phase species which should be considered in the thermodynamics
! the first string is for the name - CASE SENSITIVE
! the second string is for the modes, in which this species should occur
! the second string should contain the numbers of the modes, separated by ";
CASK_GASES ( 1 , : ) = "HNO3" , "0;1;2;3;4;5;6;7"
CASK_GASES ( 2 , : ) = "H2SO4" , "0;1;2;3;4;5;6;7"
CASK_GASES ( 3 , : ) = "HCl" , "0;1;2;3;4;5;6;7"
CASK_GASES ( 4 , : ) = "NH3" , "0;1;2;3;4;5;6;7"

! enter all cation species which should be considered in the thermodynamics
! the first string is for the name - CASE SENSITIVE
! the second string is for the modes, in which this species should occur
! the second string should contain the numbers of the modes, separated by ";
CASK_CATIONS ( 1 , : ) = "Hp" , "1;2;3;4;5;6;7"
CASK_CATIONS ( 2 , : ) = "NH4p" , "1;2;3;4;5;6;7"
CASK_CATIONS ( 3 , : ) = "Nap" , "1;2;3;4;5;6;7"
CASK_CATIONS ( 4 , : ) = "Kp" , "1;2;3;4;5;6;7"
CASK_CATIONS ( 5 , : ) = "Capp" , "1;2;3;4;5;6;7"
CASK_CATIONS ( 6 , : ) = "Mgpp" , "1;2;3;4;5;6;7"

! enter all anion species which should be considered in the thermodynamics
! the first string is for the name - CASE SENSITIVE
! the second string is for the modes, in which this species should occur
! the second string should contain the numbers of the modes, separated by ";
CASK_ANIONS ( 1 , : ) = "SO4mm" , "1;2;3;4;5;6;7"
CASK_ANIONS ( 2 , : ) = "HSO4m" , "1;2;3;4;5;6;7"
CASK_ANIONS ( 3 , : ) = "OHm" , "1;2;3;4;5;6;7"
CASK_ANIONS ( 4 , : ) = "Clm" , "1;2;3;4;5;6;7"
CASK_ANIONS ( 5 , : ) = "NO3m" , "1;2;3;4;5;6;7"

! enter all solute species which should be considered in the thermodynamics
! the first string is for the name - CASE SENSITIVE
! the second string is for the modes, in which this species should occur
! the second string should contain the numbers of the modes, separated by ";
CASK_SOLUTES ( 1 , : ) = "H2O" , "0;1;2;3;4;5;6;7"
CASK_SOLUTES ( 2 , : ) = "DU" , "3;4;6;7"
CASK_SOLUTES ( 3 , : ) = "OC" , "2;3;4;5"
CASK_SOLUTES ( 4 , : ) = "BC" , "2;3;4;5"

! enter all bulk species which should be considered in the thermodynamics
! the first string is for the name - CASE SENSITIVE
! the second string is for the modes, in which this species should occur
! the second string should contain the numbers of the modes, separated by ";
CASK_BULK ( 1 , : ) = "SS" , "2;3;4"
CASK_BULK ( 2 , : ) = "DU" , "3;4;6;7"
CASK_BULK ( 3 , : ) = "OC" , "2;3;4;5"
CASK_BULK ( 4 , : ) = "BC" , "2;3;4;5"

! ---- Thermodynamic specific options
LDRY = F, ! Force aerosol particles to be dry (no aerosol water)
LHYSTER = F, ! Inclusion of hysteresis effect (includes solids/metastable)
&CTRL_GMXE_AERCHEM

! namelist for aerchem subsubmodel
lmode = 3  ! lower mode boundary for aerchem
umode = 4  ! upper mode boundary for aerchem
/

&CTRL_GMXE_SOA

! namelist for SOA subsubmodel
lmode_soa = 2  ! lower mode boundary for SOA
umode_soa = 5  ! upper mode boundary for SOA

! tracer name for each SOA precursor
Isoprene, Toluene, Xylene, Cresol, Alkenes, total terpenes, a−Pinene, b−Pinene,
N_GASPREC(:, :) = 'Isop', 'Tol', 'Xyl', 'Cre', 'Alk', 'Terp', 'aPin', 'bPin'

! logical switch for species to consider
Isoprene, Toluene, Xylene, Cresol, Alkenes, total terpenes, a−Pinene, b−Pinene
l_GASPREC(:, 1) = T, T, T, T, T, T, T

! logical switch to decide whether total chemical degradation or only degradation due to SOA formation should be considered
T means that total loss is considered
Isoprene, Toluene, Xylene, Cresol, Alkenes, total terpenes, a−Pinene, b−Pinene,
l_GASPREC(:, 2) = T, T, T, T, T, T, T

! logical switch for oxidants to consider
O3, OH, NO3
l_OXI(:, :) = T, T, T

! string with potential names of species excluded for bulk SOA uptake
! maximum number of species at the moment is 10
EXCL_STR_SOA(:, :) = 'DU', ',', ',', ',', ',', ',', ',', ',', ',', '
/

&CTRL_GMXE_PASSIVE

! namelist for passive aerosol tracers
! NOTES: If passive aerosols are turned on here, then consider the following:
! − SCAV: if it should be possible to scavenge them, then the full name has to be included in the out_string_aer in scav.nml;
! − AEROPT: if it is turned on and the passive tracers should NOT contribute to the AOT, then exclude them in aeropt.nml;
! − emissions: so far only offlem emissions, see offlem46.nml
! and emis_PA46.nml
! − tracers are initialized in messy_gmx_e5.nml
num_pa = 1  ! number of passive aerosol tracers
pamode1 = 2  ! lower mode boundary for passive aerosol
pamode2 = 4  ! upper mode boundary for passive aerosol
/

&CPL

l_calc_emis = T  ! calculate direct emission of compounds within gmxe
l_tendency = T  ! note that 3D or X−NLEV2D are always added in the tendency
1. Choice of tropopause channel

Tropopchannel = 'tropop' ! channel containing tropopause index
TropopIndex = 'tp_i' ! tropopause index channel element

2. Choice of PSC channel

Psccchannel = 'msbm' ! channel containing PSC mask
Psccreg = 'PSC_region' ! mask for psc region
phase = 'phase' ! phase of psc (to be used as a mask)
driver_call = 'physc' ! Switch indicated where GMXe_driver is called from

emission setup (new way)
for each emission flux one EMIS_CASK should be filled
all possible fluxes are listed in the messy_gmxem.e5.f90
in the subroutine gmxem_e5_init
certain characteristics of the fluxes are defined there as well,
e.g. if only a fraction of the total flux should be used for
another flux (e.g. SO2 emissions, 2.5% are used for particulate sulfate emissions)
emis_casks = array:

1st entry: name (used for identification in the
list of the e5 file section emis_init)
2nd entry: total scaling factor for incoming flux
3rd entry: channel name of the emission flux
4th entry: name of the mass emission flux object
5th entry: name of the corresponding number
6th entry: emission flux object (If it does not exist a
number is calculated from the mass)
7th entry: separated list of tracers which should
receive emissions from this flux
7th entry: separated list of fractions of this emission
Theoretically, one can define an emission flux several times for several tracers,
BUT, BE AWARE, that is means that the number flux is also added several times, which
is most likely not desired
WARNING: In case the 1st entry from a cask is not matching any of the fluxes in the list
it is ignored

SEASALT
AEROCOM
SULPHATE

EMIS_CASK(1,:) = "seasalt_mass_cs", "1.46", "onemis","mss_cs_lsce" ,","Nap_cs; Mgpp_cs; Capp_cs; Kp_cs; Clm_cs; SS_cs; SO4mm_cs", "0.306:0.037:0.012:0.011:0.550:0.007:0.077"
EMIS_CASK(2,:) = "seasalt_mass_as", "1.458905", "onemis","mss_as_lsce" ,","Nap_as; Mgpp_as; Capp_as; Kp_as; Clm_as; SS_as; SO4mm_as", "0.306:0.037:0.012:0.011:0.550:0.007:0.077"
EMIS_CASK(3,:) = "seasalt_mass_ks", "0.001095", "onemis","mss_as_lsce" ,","Nap_ks; Mgpp_ks; Capp_ks; Kp_ks; Clm_ks; SS_ks; SO4mm_ks", "0.306:0.037:0.012:0.011:0.550:0.007:0.077"

EMIS_CASK(4,:) = "so2_mlc_ks", "0.0125", "offemis", "EDGAR_ANTH_SO2_flux", ",", "SO4mm_ks; Hp_ks", "1.0:2.0"
EMIS_CASK(5,:) = "so2_mlc_as", "0.0125", "offemis", "EDGAR_ANTH_SO2_flux", ",", "SO4mm_ks; Hp_ks", "1.0:2.0"
278: EMIS_CASK(6,:) = "so2_mlc_ks", "0.0125", "offemis", "SO2_flux", "", "SO4mm_ks; Hp_ks","1.0;2.0"
279: EMIS_CASK(7,:) = "so2_mlc_as", "0.0125", "offemis", "SO2_flux", "", "SO4mm_ks; Hp_ks","1.0;2.0"
280: EMIS_CASK(8,:) = "so2_mlc_ks", "0.0125", "import_grid","VOLCANIC_SO2_SO2", "", "SO4mm_ks; Hp_ks","1.0;2.0"
281: EMIS_CASK(9,:) = "so2_mlc_as", "0.0125", "import_grid","VOLCANIC_SO2_SO2", "", "SO4mm_as; Hp_as","1.0;2.0"

282:
283: !--------------------------------------
284: ! ORGANIC CARBON
285: !--------------------------------------
286: ! Scaling factor contains water soluble fraction and conversion om2oc
287: ! ANTHROPOGENIC
288: EMIS_CASK(22,:) = "oc_mass_ff_ks", "0.91", "offemis", "EDGAR_ANTH_OC_flux", "", "OC_ks","1.0"
289: EMIS_CASK(23,:) = "oc_mass_ff_ki", "0.49", "offemis", "EDGAR_ANTH_OC_flux", "", "OC_ki","1.0"
290: EMIS_CASK(24,:) = "oc_mass_ff_ks", "1.0868", "import_grid", "EDGAR_AIR_OC_flux", "", "OC_ks","1.0"
291: EMIS_CASK(25,:) = "oc_mass_ff_ki", "0.2032", "import_grid", "EDGAR_AIR_OC_flux", "", "OC_ki","1.0"
292: ! BIOMASS BURNING
293: EMIS_CASK(26,:) = "oc_mass_bb_ks", "0.59728", "offemis", "OC_flux", "", "OC_ks","1.0"
294: EMIS_CASK(27,:) = "oc_mass_bb_ki", "1.00272", "offemis", "OC_flux", "", "OC_ki","1.0"

295:
296: !--------------------------------------
297: ! BLACK CARBON
298: !--------------------------------------
299: EMIS_CASK(28,:) = "bc_mlc_3d_ki", "1.0", "offemis", "EDGAR_ANTH_BC_flux", "", "BC_ki","1.0"
300: EMIS_CASK(29,:) = "bc_mlc_3d_ki", "1.0", "import_grid", "EDGAR_AIR_BC_flux", "", "BC_ki","1.0"
301: EMIS_CASK(30,:) = "bc_mass_3d_ki", "1.0", "offemis", "BC_flux","", "BC_ki","1.0"
302:
303: !--------------------------------------
304: ! DUST
305: !--------------------------------------
306: !KKDU-Asthita resolution dependent
307: ! T106 = 1.0
308: ! T63 = 0.8
309: ! T42 = 0.6
310: EMIS_CASK(40,:) = "dust_mass_ci", "1.00", "onemis", "kkdu_misc_emflux_ci", "", "DU_ci", "0.60"
311: EMIS_CASK(41,:) = "dust_mass_ai", "1.00", "onemis", "kkdu_misc_emflux_ai", "", "DU_ai", "0.60"
References


