# SSH-aerosol

SSH-aerosol user manual and test cases

## About

**Purpose:** introduction to SSH-aerosol and launching simulations for test cases. Very basic post-processing is also presented.

**Authors:**
Karine Sartelet, `karine.sartelet@enpc.fr`;
Florian Couvidat, `florian.couvidat@ineris.fr`;
Zhizhao Wang, `zhizhaow@ucr.edu`;
Youngseob Kim, `youngseob.kim@enpc.fr`;
Cédric Flageul, `cedric.flageul@univ-poitiers.fr`;
Victor Lannuque `victor.lannuque@ineris.fr`

**SSH-aerosol version:** 2.0



## Contents

# 1 Introduction

The SSH-aerosol model represents the physicochemical transformation undergone by aerosols in the troposphere. The term aerosol designs here particles with the surrounding gas. SSH-aerosol is designed to be modular and the user can choose the physical and chemical complexity required. The model is based on the merge of three state-of-the-art models:

- SCRAM: The Size-Composition Resolved Aerosol Model (Zhu et al., 2015) that simulates the dynamics and the mixing state of atmospheric particles. It classifies particles by both composition and size, based on a comprehensive combination of all chemical species and their mass-fraction sections. All three main processes involved in aerosol dynamics (coagulation, condensation/evaporation and nucleation) are included.
- SOAP: The Secondary Organic Aerosol Processor (Couvidat and Sartelet, 2015) is a thermodynamic model that computes the partitioning of organic compounds. It takes into account several processes involved in the formation of organic aerosol (hygroscopicity, absorption into the aqueous phase of particles, non-ideality, and phase separation) and computes the formation of organic aerosol either with a classic equilibrium representation (the partitioning of organic compounds is instantaneous) or with a dynamic representation (where the model solves the dynamic of the condensation/evaporation limited by the viscosity of the particle). The dynamic representation was successfully used (Kim et al., 2019) for the first study with a 3D air quality model on the impact of particle viscosity on SOA formation.
- $H^2O$: The Hydrophilic/Hydrophobic Organics (Couvidat et al., 2012) mechanism uses a molecular surrogate approach to represent the myriad of formation of semi-volatile organic compounds formed from the oxidation in the atmosphere of volatile organic compounds. The mechanism was shown to give satisfactory results for SOA formation (for example in Kim et al. (2019)).



# 2 Hardware and software requirement

The random access memory (RAM) requirement for a 0D simulation is very small.

SSH-aerosol is written in the programming languages FORTRAN and C++. It can run on PC or a cluster with the GNU compilers (gfortran, gcc and g++) under a Linux system. Intel compiler should also work. If not, please report to `ssh-aerosol-help@liste.enpc.fr`.

The following external libraries are required:

- C++ library Blitz++ (`https://github.com/blitzpp/blitz`). See section 2.1 for the instructions.
- NetCDF library may be required if you have precomputed coagulation partition coefficients, you can download it from the following site: `http://www.unidata.ucar.edu/downloads/netcdf/index.jsp`. See section 2.2 for the instructions.
- SWIG (`http://www.swig.org/index.php`). See section 2.3 for the instructions.
- Header file for Python (Python.h). It is included in `libpython3-dev`. When you install `python3-dev`, `libpython3-dev` is installed too.

All installations of the external libraries can be done on your local directory without admin rights. If your local directory is `$HOME/usr/local`, you may make `YourLocal` for your convenience, for example,

```
export YourLocal=$HOME/usr/local
```

For convenience, a shell script `tools/install_ssh-aerosol.sh` is provided to automate the installation of all requirements for running SSH-aerosol, including the external libraries discussed. The script can also install `SCons` using `apt-get` and set up the necessary environment paths, as described in Section 3.1. Note that using apt-get may require administrative permissions.

Once the script is executed and the installation is complete, users can then generate the SSH-aerosol executable in the *ssh-aerosol* folder by running the commands detailed in Section 3.1:

```
./clean; ./compile
```

In addition, SSH-aerosol can be installed using `Docker`. This allows users across platforms (e.g., macOS, Windows, and Linux/Unix) to install and run SSH-aerosol inside a Docker container that has been set up with a Linux environment and all the requirements for running SSH-aerosol. Please refer to Section 2.4 for detailed installation instructions.

## 2.1 Install Blitz++

If you install Blitz++ using a package manager like apt or yum, you need to install `libblitz0-dev` and `libblitz0v5`.

If you can not install using a package manager, you can install it from source files. To do that, `cmake` is required to build Blitz++ from source files.

If `cmake` is not installed, the following guide provides instructions on installing. You can skip if `cmake` is already installed.

```
wget https://github.com/Kitware/CMake/releases/download/v3.23.2/cmake-3.23.2.tar.gz
tar xvf cmake-3.23.2.tar.gz
./configure --prefix=$YourLocal
make
make install
```

- to build `Blitz++` from the source

```
wget https://github.com/blitzpp/blitz/archive/refs/heads/master.zip
unzip master.zip

cd blitz-master
mkdir build
cd build
cmake ..
make lib
```

And edit `CMakeCache.txt` to indicate where blitz is installed. Please indicate the full path to YourLocal

```
CMAKE_INSTALL_PREFIX:PATH=/home/xxx/usr/local
```

```
make install
```

Please set the following paths to the installed `Blitz++` in your environment before compiling.

```
export CPLUS_INCLUDE_PATH=/home/xxx/usr/local/include:$CPLUS_INCLUDE_PATH
export LD_LIBRARY_PATH=/home/xxx/usr/local/lib:$LD_LIBRARY_PATH
```

## 2.2 Install netCDF

You have to install `netCDF-C` and then `netCDF-fortran`.

### 2.2.1 `netCDF-C`

If you install `netCDF-C` using a package manager like apt or yum, you need to install `libnetcdf-dev`.

For the installation from the source, Version 4.7.3 is tested.

```
wget https://downloads.unidata.ucar.edu/netcdf-c/4.7.3/netcdf-c-4.7.3.tar.gz
tar xvf netcdf-c-4.7.3.tar.gz
cd netcdf-c-4.7.3
./configure LDFLAGS="-L$HOME/usr/local/lib" --disable-hdf5 --prefix=$HOME/usr/local
make
make install
```

### 2.2.2 `netCDF-fortran`

If you install `netCDF-fortran` using a package manager like apt or yum, you need to install `libnetcdff-dev`.

For the installation from the source, Version 4.5.2 is tested.

```
wget https://downloads.unidata.ucar.edu/netcdf-fortran/4.5.2/netcdf-fortran-4.5.2.tar.gz
tar xvf netcdf-fortran-4.5.2.tar.gz
cd netcdf-fortran-4.5.2
./configure CPPFLAGS="-I$HOME/usr/local/include" LDFLAGS="-L$HOME/usr/local/lib"
--prefix=$HOME/usr/local
make
make install
```

## 2.3 `SWIG`

If you install `SWIG` using a package manager like apt or yum, you need to install `swig`.

If you can not install using a package manager, you can install it from source files.

```
wget http://prdownloads.sourceforge.net/swig/swig-4.0.2.tar.gz
tar xvf swig-4.0.2.tar.gz
cd swig-4.0.2
./configure --without-pcre --prefix=$HOME/usr/local
make
make install
```

## 2.4 Installation with Docker

The following steps guide you through installing SSH-aerosol using Docker, which allows for a consistent setup across different platforms (macOS, Windows, and Linux/Unix):

1. **Install Docker**: Make sure Docker is installed and running on your system. Refer to the Docker installation guide if needed: `https://docs.docker.com/get-docker/`.

2. **Open a command-line application**: A command-line application is required to launch Docker. `terminal` on Mac and `cmd.exe` on Windows can be used. Open your command-line application and go to your working directory `YourDirectory`

3. **Prepare SSH-aerosol for Docker**: Before building the Docker image, copy the SSH-aerosol folder in your working directory `YourDirectory` and make sure it is named `ssh-aerosol`.

   `ssh-aerosol` folder will be copied into the Docker container during the build process. If this folder is not present, you will need to manually copy the `ssh-aerosol` folder to the Docker container later.

4. **Navigate to the Docker Setup Folder**: Copy the files required to build the Docker environment from `ssh-aerosol/tools/install_docker` directory into `YourDirectory`.

   ```
   cp ssh-aerosol/tools/install_docker/Dockerfile .
   cp ssh-aerosol/tools/install_docker/setup_ssh-aerosol.sh .
   ```

   You will find two files there: `Dockerfile` and `setup_ssh-aerosol.sh` in `YourDirectory`.

5. **Build the Docker Image**: Run the following command to build a Docker image named `ssh-aerosol`:

   ```
   docker build -t ssh-aerosol .
   ```

   This process may take a few minutes, depending on your machine's performance.

6. **Run the Docker Container**: Once the image is built, start a container by running:

   ```
   docker run -it ssh-aerosol
   ```

   This will open the Docker container, and you will be in the working directory: `/root`. If the container starts successfully, you should see the `ssh-aerosol` folder and the `setup_ssh-aerosol.sh` script in this directory.

7. **Complete the Compilation**: To finish the setup, run:

   ```
   cd ssh-aerosol
   ./clean
   ./compile
   ```

   This step will clean up and then build the SSH-aerosol executable using `SCons` (See Section 3.1). If the process completes successfully, you should see the message `scons: done building targets.` on the screen, along with a symbolic link named `ssh-aerosol` pointing to the executable located in `src/ssh-aerosol`. Please refer to Section 3 for other compilation methods.

# 3 Compilation

## 3.1 Compilation with SCons

Before the compilation, make sure the construction tool: SCons has already been installed. If not you can obtain it through the instructions of the site: `http://www.scons.org/wiki/SconsTutorial1`

After all the required software and library are ready, you may need to modify `compile` to indicate the right path.

```
# Path to blitz and netcdf libraries
export LD_LIBRARY_PATH=$Home/usr/local/lib:$LD_LIBRARY_PATH
# Path to blitz and netcdf files
export CPLUS_INCLUDE_PATH=$HOME/usr/local/include:$CPLUS_INCLUDE_PATH
# path to netcdf.mod
export F90FLAGS=-I$HOME/usr/local/include:$F90FLAGS
# path to swig
export SWIG=$HOME/usr/local/bin/swig
```

Finally, the compilation can be done by typing a simple command: `compile`.

```
./compile
```

All files created during the compilation can be removed by typing: `clean`.

```
./clean
```

## 3.2 Compilation with Makefile and fortran interface

An alternative method is to compile SSH-aerosol with Makefiles to use an interface with fortran programs. Compiling with Makefiles create library module that can be used to link SSH-aerosol to a fortran program.
To compile with Makefiles, fill mymodules.sh to provide link to the different librairies necessary to launch SSH (like Blitz and SSH). Then launch the command ./make.sh that will create the SSH librairies.
In the repertory example_fortran, an example to call SSH-aerosol is provided in the program launch.f90. A Makefile is provided to illustrate the compilation of launch.f90 with a Makefile. In this example, the launch_ssh_aerosolonly subroutine is called (only calculates the fortran of aerosol without the gas phase chemistry). Different exemple are provided depending the values of parameter itest in launch.f90.

## 3.3 Coupling to other models using the SSH-aerosol library

Regarding the coupling of SSH-aerosol and 3D codes, a dedicated Application Programming Interface (API) is available in SSH-aerosol. It allows the 3D code to easily initialise, read or modify the internal variables and perform time-advancement if transport and chemistry processes are coupled to the 3D code by a first order scheme. The usual work-flow allowing to couple a 3D code with SSH-aerosol using the API is now briefly described.

Firstly, SSH-aerosol has to be compiled to make a library from the sources.

```
compile -s=yes
```

A shared library `libssh-aerosol.so` is made in `src` folder.

Then the 3D code needs to compile to load the library. For example, it can be done as follows for a script written in C.

```
gcc -g -o c_example example.c -ldl ../src/libssh-aerosol.so -rdynamic
```

Once the 3D code has loaded the SSH-aerosol shared library, dedicated API functions allow the external code to set the pressure, temperature, humidity and gas and aerosol concentrations and run SSH-aerosol. The repository of the code `example_using_so` provides a small C program, Fortran and Python showing how such a shared library can be used.

# 4 Code structure

The SSH-aerosol package contains different repertories for source code, configuration files, input files, output files, and visualisation of outputs.

## 4.1 Source code

The folder *src* is where all source code files are stored. The main program file is *ssh-aerosol.f90*. Besides, there are 2 sub-folders under the ***src*** directory:

- ***scons*** folder contains the files necessary for compiling the program
- ***include*** folder contains the source code, which is itself separated into different folders:
  - ***Module*** contains SCRAM subroutines to model aerosol dynamics
  - ***SOAP*** contains SOAP subroutines to model organic aerosol thermodynamics and inorganic aerosol thermodynamics if the inorganic module ISORROPIA is not used.
  - ***CHEMISTRY*** contains many files for gas-phase chemical mechanisms, including $H^2O$.
  - ***AtmoData*** is a tool for data processing in atmospheric sciences.
  - ***RDB*** contains subroutines for size redistribution used in SCRAM.
  - ***isorropia_aec*** contains ISORROPIA subroutines, a well-known module for the computation of inorganic thermodynamic equilibrium between gas and aerosol.
  - ***INC*** contains include files which define some system parameter variables.

## 4.2 Input files

Simulations require a configuration file that details input and output conditions, as well as various simulation setups. The folder *INIT* is where the example configuration files (with the suffix `.ssh`) for SSH-aerosol are sorted. The main options used in the input file are discussed in Section 6. The following example configuration file, "**namelist.ssh**" will be used to illustrate further discussions.

## 4.3 Output files

Simulation results are stored in the folder specified in the variable *namelist.ssh* under the variable *output_directory*. The type of output files can also be selected in the *namelist.ssh* file: *output_type = 1* for outputs in text format, *output_type = 2* for outputs in binary format and *output_type = 3* for outputs in netcdf format. If *output_type* is set to 0, no output is written

(potentially useful for coupling situations). The *graph* folder contains a few Python routines for post-processing and displaying results (with text outputs only). At the end of the simulation, a report file recording the main settings of the simulation and five types of outputs are automatically generated/updated in the output folder.

### 4.3.1 Text and binary outputs

In the case of binary or text output files (i.e. *output_type = 1 or 2*), a file is generated for each time series. In each file, a total of N + 1 rows of data (one value per row) are listed. N represents the number of iterations and the first line in the file records the initial concentration. The results are organized into 5 sub-folders:

- Subfolder ***gas***: this subfolder contains files that record the time variation of gas phase mass concentration ($\mu$g m$^{-3}$) of each species. The species name (which is defined by the species file entered in the *namelist.ssh*) is adopted as the file name (ex.: *HNO3.txt*).

- Subfolder ***aero***: this subfolder contains files that record the time variation of particulate mass concentration ($\mu$g m$^{-3}$) of each species for each size section and external mixing case. The file is named after the species name followed by the 'sizemix' index (index = 1, 2, ..., N where N is the number of size bin x number of external mixing index). The concept of 'sizemix' is explained in Section 4.3.3.

  For example, file *PNO3_1.txt* notes the time variation of nitrate particulate mass concentration in the first size section (and first external mixing state if considered).
  Time variation of total organic particles, inorganic particles, black carbon, dust, $PM_1$, $PM_{2.5}$, $PM_{10}$ and particulate water mass concentration ($\mu$g m$^{-3}$) are also located in the subfolder *aero*, under the name *Organic.txt*, *Inorganic.txt*, *Black_Carbon.txt*, *Dust.txt*, *PM1.txt*, *PM2.5.txt*, *PM10.txt* and *Water.txt*, respectively.
  Time variations of pH in each size section and external mixing case are also recorded in pH files (ex.: *pH_1.txt*).

- Subfolder ***TM***: this subfolder contains files that record the time variation of total mass concentration ($\mu g/m^3$) and total aerosol mass concentration summed on all size sections ($\mu$g m$^{-3}$). The files are named after the aerosol species name, (followed by the name of its gas phase precursor for recording total mass) and followed by '*_TM*'.
  For example, the file *PSO4_TM.txt* notes the time variation of sulfate particulate mass concentration, while the file *PSO4_SULF_TM.txt* notes the time variation of the total sulfate concentration in both the gas phase and the particulate phase.

- Subfolder ***number***: this subfolder contains files that record the time variation of particulate number concentrations (particles m$^3$) for each size section and external mixing case. The file is named '*NUMBER_*' followed by the 'sizemix' index (ex.: *NUMBER_1.txt*).
  The file *TNUM.txt* that records the time variation of total number concentration (particles m$^{-3}$) is located in this subfolder as well.

- Subfolder ***diameter***: this subfolder contains files that record the time variation of the average diameter ($\mu$m) for each size section and external mixing case. The file is named '*DIAMETER_*' followed by the 'sizemix' index (ex.: *DIAMETER_1.txt*).

### 4.3.2 NETCDF outputs

In the case of netcdf files (i.e. output type = 3), five files are generated corresponding to the 5 folders of binary or text outputs. Writing of NetCDF outputs is faster than text or binary ones and is recommended for very detailed chemical mechanisms to limit the quantity of small output files. The five generated files are:

- ***gas.nc***: this file includes the evolution of the mass concentrations of each gas species (as variables) in ($\mu$g m$^{-3}$) versus time in seconds.
- ***aero.nc***: this file includes the evolution of the pH and mass concentrations of each aerosol species as variables in $\mu$g m$^{-3}$, for each size section and external mixing case (regrouped in a 'sizemix' dimension), versus time in seconds. Information about each 'sizemix' index is given by the *low_boundary(sizemix)*, *high_boundary(sizemix)*, *Sizebin(sizemix)* and *ext_mix(sizemix)* variables included in the netcdf file. Special aerosol species (organics, inorganics, black carbon, dust, $PM_1$, $PM_{2.5}$, $PM_{10}$, and particulate water) are also recorded in the file versus time.
- ***TM.nc***: this file includes the evolution of the mass concentrations of each condensable species as variables in $\mu$g m$^{-3}$, versus time in seconds, in gas, in aerosol or in both phases depending on the 'Phase' dimension (*1=gas*, *2=aero* and *3=total*). Variable names correspond to the aerosol species' names.
- ***number.nc***: this file contains the variation of particle number concentrations (particle m$^3$) for each size section and external mixing case (regrouped in a 'sizemix' dimension), and the total concentration, versus time in seconds. Information about each 'sizemix' index is given by the *low_boundary(sizemix)*, *high_boundary(sizemix)*, *Sizebin(sizemix)* and *ext_mix(sizemix)* variables included in the netcdf file.
- ***diameter.nc***: this file includes the variation of the average diameter ($\mu$m) for each size section and external mixing case (regrouped in a "sizemix" dimension) versus time in seconds. Information about each 'sizemix' index is given by the *low_boundary(sizemix)*, *high_boundary(sizemix)*, *Sizebin(sizemix)* and *ext_mix(sizemix)* variables included in the netcdf file.

### 4.3.3 About the 'sizemix' index

The sizemix index is introduced to cover all cases of size bin and mixing-state sections in a single vector. If the mixing-state of particles is not considered, the sizemix index corresponds to the sizebin index. Conversely, in the case where the mixing state is considered but with a single-size bin, the sizemix index corresponds to the index of the mixing-state index. If both the mixing state and size distribution are considered, the correspondences are as presented in Table 1.

| | mix_ind = 1 | mix_ind = 2 | mix_ind = 3 |
|---|---|---|---|
| size_ind = 1 | 1 | 2 | 3 |
| size_ind = 2 | 4 | 5 | 6 |
| size_ind = 3 | 7 | 8 | 9 |

Table 1: Exemple of sizemix index correspondences to external mixing (mix_ind) and size bin (size_ind).

## 4.4 Tools

The folder *tools* contains a few Python routines that may be useful for the numerical representation of aerosols, as detailed below.

### 4.4.1 Partitioning mass and number into sections

The functions of the routine *upsample.py* may be used to partition, i.e. split, a section into several sections, while conserving both mass and number concentrations. An example is provided in the file *test_upsample.py*, which splits a section into 2, 3, and 12 logspaced sections, conserving mass and number during the splitting. This algorithm may be useful to split emissions, which are not enough size-resolved, onto a finer size discretization.

The following scheme is employed: We assume the mass concentration M and the number concentration N of a section, ranging from diameters $d_{min}$ to $d_{max}$, are related such that $M = \frac{\pi}{6}\rho\bar{d}^3 N$ where $\rho$ is the density of the aerosols in that section and $\bar{d} = \sqrt{d_{min}d_{max}}$ is the geometric mean diameter of the section boundaries.

Provided that $\{d_i\}$ is a partition of the interval $[d_{min}, d_{max}]$, such that $d_{i-1} < d_i$ for all $i = 1, ..., n$ and $d_0 = d_{min}$, $d_n = d_{max}$, we can derive a mass and number concentrations that satisfy the condition $M_i = \frac{\pi}{6}\rho\bar{d_i}^3 N_i$, where $\bar{d_i} = \sqrt{d_{i-1}d_i}$ is the geometric mean diameter of the $i$-th section, in a way that conserves both total mass and number.

By setting

$$M_i = M\frac{d_i^{3/2} - d_{i-1}^{3/2}}{d_n^{3/2} - d_0^{3/2}}$$

one can easily check that $\sum_{i=1}^{n} M_i = M$ and $\sum_{i=1}^{n} N_i = N$.

### 4.4.2 Discretization of lognormal modes into sections

If the aerosol distribution is known as the sum of lognormal modes, such as in Seigneur et al. (1986), it can be discretized into sections. This is done in the python file *lognormal_50.py*, which discretizes into 50 sections the "Hazy" and "Urban" distribution of Seigneur et al. (1986).

## 4.5 Array structure of variables inside the code

The variables and arrays corresponding to the number of compounds in the different phases, as well as the concentrations are defined in the routine *src/Module/ModuleInitialisation.F90*. Arrays are allocated at the beginning of the program and deallocated at the end. The array *concentration_number* contains the number concentrations for each size and composition section. *concentration_mass* contains the mass concentrations of each chemical compound for each size and composition section. The order of the chemical compounds is given in the file *species-list/species-list-aer.dat*.

Each organic compound can be only hydrophilic, only hydrophobic, or both hydrophilic and hydrophobic. This property affects the partitioning of compound between the organic phase (if the compound is hydrophobic) and the aqueous phase (if the compound is hydrophilic). This property is specified in the column *partitioning* of the file *species-list/species-list-aer.dat*. If some compounds are both hydrophilic and hydrophobic, then the variable *coupled_phases* of the paragraph *&physic_condensation* of the namelist file should be set to 1.

In *soap.cpp*, the organic and aqueous concentrations are stored in different arrays. The organic-phase concentrations in the different layers are noted *Ap_layer*, while the aqueous-phase concentrations are noted *Aaq*. If thermodynamic equilibrium is assumed (*ISOAPDYN=0* in

the namelist), then the sum of the organic and the aqueous phases is stored and output in the concentration array. If dynamic exchanges between the gas and particle phases are modelled (*ISOAPDYN = 1* in the namelist), in the code, the variable *i_hydrophilic* is set to 1 and both the aqueous and organic phases of compounds are followed. In that case, for each organic compound (independently of the intrinsic property of the compound), the organic phase is stored before the aqueous phase of the compound.

Note that in the case of a viscous aerosol, if several layers are considered, for each organic compound, the mass concentration is stored for each layer. In that case, for each size and composition section, the array *concentration_mass* contains the mass concentrations of each chemical compound and in each layer if the compound is organic and hydrophobic. An example of the storage of mass concentrations for a size and composition section $i$ is displayed in Fig. 1. In this example, 3 compounds are considered, and three layers are used to represent the condensation/evaporation of the compound 2, which is the only compound hydrophobic and organic out of the 3.

| *Section i* | Species 1 | Species 2 Organic phase Layer 1 | Species 2 Organic phase Layer 2 | Species 2 Organic phase Layer 3 | Species 2 Aqueous phase | Species 3 |
|---|---|---|---|---|---|---|

Figure 1: Storage of the concentrations of chemical compounds/species in the array /itconcentration_mass for a section $i$. Case of 3 compounds/species with 3 layers for species 2, which is a hydrophobic organic compound.

# 5 Run a simulation

To run a simulation, please run the compiled program `ssh-aerosol` followed by the main configuration `namelist.ssh`.

Please check first the name of the path to the simulation results. This path is indicated as `output_directory` in `namelist.ssh`.

```
./ssh-aerosol namelist.ssh
```

`namelist.ssh` contains only mandatory options. The full list of options used is found in `namelist.out`.

Many configuration files for the test cases are available in the directory `INIT`.

To run a test case, you need to replace the main configuration file with the configuration file of the test case (See section 8).

```
./ssh-aerosol INIT/namelist_cond.ssh
```

# 6 Main options

The following table lists all the options in the namelist. Some parameters are not compulsory in the namelist. If the parameter is not provided, a default value is used. Some parameters are indicated as "Optional", they are not used unless a specific option is activated. In that case, the parameter must be provided. For example, parameter *N_groups* is only used when

*tag_external=1*. The order of the sections should not be modified in the namelist, however the order of the options within a section may be changed.

| Options | Default value | Description |
|---|---|---|
| Section: /setup_meteo/ Described in section 6.1 | | |
| latitude | Compulsory | Latitude (in degree): use to compute photolysis rate as a function of time and location |
| longitude | Compulsory | Longitude (in degree): use to compute photolysis rate as a function of time and location |
| Temperature | Compulsory | Temperature in K |
| Pressure | Compulsory | Pressure in Pa |
| Humidity | Compulsory | Specific humidity in kg/kg. Either specific or relative humidity must be provided |
| Relative_Humidity | Compulsory | Relative humidity. Either specific or relative humidity must be provided |
| meteo_file | None | File to provide meteorological data as a function of time |
| Section: /setup_time/ Described in section **??** | | |
| initial_time | Compulsory | Time (in s) at beginning of simulation |
| final_time | Compulsory | Time (in s) at end of simulation |
| delta_t | Compulsory | General timestep (in s) of SSH-aerosol |
| Section: /initial_condition/ Described in section 6.3 | | |
| with_init_num | 0 | Are initial number concentrations provided? |
| tag_init | 0 | Internally (:0) or externally (:1) mixed aerosols |
| tag_dbd | | Size bound generated (:0) or read (:1) |
| N_sizebin | Compulsory | Number of particle sizebin |
| wet_diam_estimation | 1 | Method used to estimate wet diameter as a function of dry diameter, humidity, and composition. 0: based on Isorropia. 1: based on water concentration. |
| init_gas_conc_file | Compulsory | File on initial concentrations of gases |
| init_aero_conc_mass_file | Compulsory | File on initial mass concentrations of particles |
| init_aero_conc_num_file | Compulsory | File on initial number concentrations of particles |
| Section: /initial_diam_distribution/ described in section 6.3 | | |
| diam_input | Compulsory | Initial diameter of the size distribution |
| Section: /emissions/ described in section 6.8 | | |
| tag_emis | 0 | 0: no emissions. 1: internally mixed emissions. 2: externally mixed emissions. |
| with_emis_num | 0 | Put 1 to provide particle number emissions |
| emis_gas_file | Optional | Compulsory if *tag_emis* is not set to 0. File with emissions of gases |
| emis_aero_mass_file | Optional | Compulsory if *tag_emis* is not set to 0. File with mass emissions of particles |
| emis_aero_num_file | Optional | Compulsory if *emis_num* is not set to 0. File with number emissions of particles |
| Section: /mixing_state/ described in section 6.4 | | |
| tag_external | 0 | Internally (:0) or externally (:1) mixed aerosols |
| N_groups | Optional | Compulsory if *tag_external=1*. Number of groups of aerosol compounds for which the composition is discretised. |

| | | |
|---|---|---|
| N_frac | Optional | Compulsory if *tag_external=1*. Number of mass fraction sections used in the discretisation of composition. |
| kind_composition | 0 | 0: automatic composition discretization. 1: provided by the user. |
| Section: /fraction_distribution/ described in section 6.4 | | |
| frac_input | Optional | Compulsory is *kind_composition=1*. Bounds of mass fraction sections. |
| Section: /gas_phase_species/ described in section 6.5 | | |
| species_list_file | Compulsory | List of gaseous species |
| Section: /aerosol_species/ described in section 6.4 | | |
| aerosol_species_list_file | Compulsory | List of aerosol species. |
| aerosol_structure_file | Optional | File to provide UNIFAC group decomposition of organic aerosol species. |
| Section: /physic_gas_chemistry/ described in section 6.9.1 | | |
| tag_chem | 0 | Put 1 to activate chemical reactions. |
| attenuation | 1. | value below 1 in case of cloud attenuation of photolysis, and it is equal to 1 if no cloud attenuation (clear sky). |
| option_photolysis | 1 | 1: photolysis rates estimated in the program, 2: read from binary files |
| time_update_photolysis | 100000 | Photolysis rate from binary files read again at this time value in seconds |
| with_heterogeneous | 0 | 1: with heterogeneous reactions, 0: without |
| with_adaptive | 0 | 1: activate automatic timestep to solve chemistry |
| adaptive_time_step_tolerance | 0.001 | Tolerance on relative error accepted to solve chemistry |
| min_adaptive_time_step | 0.001 | Minimal time step (in s) to solve chemistry |
| RO2_list_file | Optional | List of $RO_2$ radical species accounted for in the $RO_2$ pool. |
| tag_RO2 | 0 | 0: no reaction with $RO_2$ pool, 1: only generated $RO_2$, 2 only background $RO_2$, 3: both background and generated $RO_2$ |
| photolysis_dir | "./photolysis/" | Directory where binary files on photolysis data are. |
| photolysis_file | "./photolysis/photolysis-cb05.dat" | List of Photolysis reactions |
| n_time_angle | 9 | Number of time angles in photolysis binary files |
| nsza | 11 | Number of solar angles used in reactions file when option_photolysis = 1 |
| time_angle_min | 0 | Minimal time angle in binary files (in hour) |
| delta_time_angle | 1 | Resolution of time angle in binary files (in hour) |
| n_latitude | 10 | Number of latitude levels in photolysis binary files |
| latitude_min | 0 | Minimal latitude in binary files |
| delta_latitude | 10 | Resolution of photolysis binary files (in degree) |
| n_altitude | 9 | Number of altitude levels in photolysis binary files |
| altitude_photolysis_input | 0.0, 1000.0, 2000.0, 3000.0, 4000.0, 5000.0, 10000.0, 15000.0, 20000.0 | Altitude levels in photolysis binary files (in meter) |

| | | |
|---|---|---|
| keep_gp | Advanced: 0 | If 1, the gas/particle partitioning of SVOC compounds is assumed to be constant. To be used only in the equilibrium mode with high time step. Expert parameter. Should be used with caution. |
| kwall_gas | 0. | Wall loss rate of gaseous SVOC (in $s^{-1}$). Fixed value. |
| kwall_particle | 0 | Wall loss rate of particles (in $s^{-1}$). Fixed value. |
| Cwall | 0 | wall equivalent concentration (in $\mu g/m^3$). Has to be provided to account for the wall losses of SVOC. |
| eddy_turbulence | | eddy diffusion coefficient (in $s^{-1}$). Used to compute *kwall_gas* and *kwall_particle* according to chamber characteristics (and also diameters for kwall_particle). |
| surface_volume_ratio | 0 | Surface on volume ratio of the chamber (in $m^{-1}$). Used to compute *kwall_gas* and *kwall_particle* according to chamber characteristics |
| kwp0 | 0 | minimal wall loss rate of particles due to electrostatic forces. Used to compute *kwall_particle* as a function of diameters. |
| radius_chamber | 0 | Radius of the chamber (in m). Used to compute *kwall_particle* according to chamber characteristics. |
| | | |
| Section: /physic_particle_numerical_issues/ described in section 6.9.2 | | |
| DTAEROMIN | 1.e-5 | Time step used to solve aerosol formation and evolution (in s) |
| redistribution_method | 0 | 0: no redistribution, 3: euler_mass, 4: euler_number, 5: hemen, 6: euler_coupled, 10: Moving Diameter, 11: SIREAM, 12: euler_couple_siream |
| with_fixed_density | 1 | 1: Constant density. 0: density computed as a function of composition |
| fixed_density | $1.4 \times 10^3$ | Default value for aerosol density (in $kg/m^3$) |
| splitting | 1 | 0: coagulation and (condensation/evaporation+nucleation) are splitted (solved separately). 1: the processes are solved together. |
| | | |
| Section: /physic_coagulation/ described in 6.9.3 | | |
| with_coag | 0 | With coagulation (1) or not (0) |
| i_compute_repart | 1 | 0: repartition coefficient are read. 1: ,repartition coefficient are computed |
| i_write_repart | 0 | Put 1 to write repartition coefficient after calculation. |
| Coefficient_file | Optional | Path to (written or read) repartition coefficient |
| Nmc | $10^6$ | Number of Monte Carlo points used to compute repartition coefficients |
| | | |
| Section: /physic_condensation/ described in section 6.9.4 | | |
| with_cond | 0 | With condensation (1) or not (0) |
| Cut_dim | Compulsory | Diameter (in $\mu$m) above which the dynamics condensation/evaporation of inorganic (and also organics for *soap_inorg=1*) is solved |
| ISOAPDYN | 0 | 0: partitioning of SVOC at equilibrium, 1: solved dynamics of SVOC condensation/evaporation |
| IMETHOD | 1 | numerical solver used to solved the dynamics of condensation/evaporation of SVOC. 0: explicit method. 1: implicit backward Euler. |

| | | |
|---|---|---|
| soap_inorg | 0 | Thermodynamic module for inorganics. 0: ISORROPIA, 1: SOAP (coupled thermodynamics between organics and inorganics) |
| nlayer | 1 | Number of layers in the organic phase of aerosols. |
| with_kelvin_effect | 1 | 1: with kelvin_effect, 0: without |
| tequilibrium | 0.1 | characteristic time above which condensation/evaporation of SVOC is solved dynamically (for ISOAPDYN=1 and IMETHOD=0) |
| dorg | $10^{-12}$ | Diffusion coefficient in the organic phase in $m^2/s$. Used for *nlayer¿1*. If set to 0, the diffusion coefficient will be computed as a function of composition |
| coupled_phases | 0 | Parameter used to track (1) or not (0) concentrations in the aqueous phase separateley from the organic phase. Have to be set to 1 when some compounds are both hydrophilic and hydrophobic and when *ISOAPDYN=1*. |
| activity_model | 3 | Option to compute activity coefficients of organic compounds. 1: ideal, 2: UNIFAC (short range interactions), 3: AIOMFAC (short, medium and long range interactions) |
| epser | 0.01 | Tolerance for the relative error for time step adjustment on the condensation/evaporation of inorganic compounds |
| epser_soap | 0.01 | Tolerance for the relative error for time step adjustment on the condensation/evaporation of organic compounds |
| niter_eqconc | Advanced: 1 | Recompute local equilibrium with the thermodynamic module every niter_eqconc iterations. Can be used in coupling with 3D models to decrease CPU time. Expert parameter. Should be used with caution. |
| niter_water | Advanced: 1 | Recompute water concentrations with the thermodynamic module every niter_eqconc iterations. Can be used in coupling with 3D models to decrease CPU time. Expert parameter. Should be used with caution. |
| co2_conc_ppm | 410 | $CO_2$ concentrations (in ppm). Used only when carbonates are simulated with *soap_inorg=1*. |
| NACL_IN_THERMODYNAMICS | 0 | 0: Consider $Na^+$ and $Cl^-$ in the thermodynamic modules. |
| SOAPlog | 2 | Parameter to extract Henry's law parameter (calculated from the structure and the saturation vapor pressure) and the smiles decomposition in UNIFAC functional groups calculated by the model. 0=no output, 1=on screen, 2=in written files (*smile2UNIFAC.decomp* and *henryfromSOAP.dat*). |
| reaction_soap_file | No reactions | Path to a file listing all intra-particle reactions. *Example given by species-list/REACTIONS.particle.soap* |
| Section: /physic_nucleation/ described in section 6.9.5 | | |
| with_nucl | 0 | With nucleation (1) or not (0) |
| nucl_model_binary | 0 | type of binary nucleation model considering water and sulfuric acid. 0=none, 1=Vehkamaki et al. (2002), 2=Kuang et al. (2008) |
| nucl_model_ternary | 0 | type of ternary nucleation model considering water, sulfuric acid, and ammonia. 0=none, 1=Napari et al. (2002), 2=Merikanto et al. (2007, 2009) |
| scal_ternary | 1 | Scaling factor for ternary nucleation rate. Ternary nucleation rate with be multiplied by the scaling factor. |

| | | |
|---|---|---|
| nucl_model_hetero | 0 | With (1) or without (0) heteromolecular nucleation. |
| scal_hetero | 0.1 | Scaling factor for heteromolecular nucleation rate. Heteromolecular nucleation rate with be multiplied by the scaling factor. |
| nesp_org_h2so4_nucl | 0 | Number of organics involved in heteromolecular nucleation |
| name_org_h2so4_nucl_species | Optional | List of organics species involved in heteromolecular nucleation. Must have the same dimension than *nesp_org_h2so4_nucl*. |
| nucl_model_org | 0 | With (1) or without (0) organic nucleation. |
| scal_org | 0.1 | Scaling factor for organic nucleation. Nucleation rate with be multiplied by the scaling factor. |
| nexp_org | 0.1 | Exponent of the power law in the computation of the organic nucleation rate. |
| nesp_org_nucl | 0 | Number of organics involved in organic nucleation |
| name_org_nucl_species | Optional | List of organic species involved in nucleation. Must have the same dimension than *nesp_org_nucl*. |
| | | |
| Section: /output/ described in section 6.11 | | |
| output_directory | Compulsory | Path to the output dir. |
| output_type | 1 | 0: no output, 1: text files, 2: binary files, 3: NETCDF files. |
| particles_composition_file | Compulsory | Discretisation of the particle composition determined by the model in case mixing-state is followed. The file is made into output_directory. |
| output_aero_list | All species | List of aerosol species in output. Used when *output_type=0* |
| output_gas_list | All species | List of gaseous species in output. Used when *output_type=0* |

The configuration options discussed below can be used in the configuration file *namelist.ssh*. These options are organized into different sections for clarity. If an option is not specified (either commented out with '!' or omitted from the configuration file), a default value is applied,as defined in the Table.

## 6.1 Meteorology

Input data concerning latitude (in degrees), longitude (in degrees), Temperature (in Kelvin), Pressure (in Pascal), and Humidity (Relative humidity in fraction or specific humidity in grams of water vapor per kilogram of air) are listed in the group setup_meteo.

```
&setup_meteo
latitude = 48.2,                ! Latitude
longitude = 2.22,               ! Longitude
Temperature = 273.16,     ! Temperature
Pressure = 1.01325e05,          ! Pressure
Humidity = 5.E-3,      ! Specific humidity
Relative_Humidity = 0.6 ! if 0 or not given, compute RH from specific humidity.
                        ! If not, specify RH value.
/
```

To take into account variable meteorological input data, an additional input file can be used. An example file is given in `inputs/meteo.dat`. This file name should be added in the group *setup_meteo* as follows:

```
Relative_Humidity = 0.6,
meteo_file = "inputs/meteo.dat"
/
```

`meteo.dat` file should include time, temperature, pressure, relative humidity, and specific humidity. If you don't have relative humidity but specific humidity, you put zero to relative humidity. And if you put non-zero relative humidity, specific humidity is not used. Please make sure that meteorological data cover the whole simulation period. To do it, the first Time data should be less or equal to `initial_time` and the last Time data should be greater or equal to `final_time` in the group *setup_time* (see **??**). This meteorological file is used in the mcm test case described in Sect.8.6.

```
Time,Temperature,Pressure,RelativeHumidity,SpecificHumidity
19040400.0,298.00,1.01325e05,0.6,5.33039774746E-3
19058400.0,298.00,1.01325e05,0.6,5.33039774746E-3
```

## 6.2 Time

The group *setup_time* lists the initial time of the simulation (in seconds from $1^{st}$ January), the final time (in seconds from $1^{st}$ January), and the time step output of the simulation (in seconds). This time step corresponds to the time step when concentrations are written in the output files, but also to the time step used for splitting the resolution of gaseous chemistry, aerosol processes, and emissions. Note that gaseous chemistry and aerosol processes are then solved with smaller time steps.

```
&setup_time
initial_time = 0.0,          ! in seconds from January 1st
final_time = 43200.0,
delta_t = 43200.0,
time_emis = 0,
/
```

## 6.3 Initial conditions

The group *initial_condition* lists the initial conditions of the simulation. The number of size sections is defined in the variable *N_sizebin*. The variable *tag_dbd* defines whether particle size bounds are either generated in the program by assuming they are equally spaced logarithmically (*tag_dbd = 0*) or whether they are read (*tag_dbd = 1*). If they are read, they need to be specified in the group *initial_diam_distribution*. The variable *tag_init* defines whether the particles are internally mixed (*tag_init = 0*) or not (*tag_init = 1*) initially. It needs to be set to 0 in the current model version. The variable *with_init_num* defines whether number concentrations are estimated from mass concentrations and diameters of each size section (*with_init_num = 0*) or whether number concentrations are read (*with_init_num = 1*). The variable *wet_diam_estimation* is equal to 0 if isorropia is called initially to estimate the liquid water content of particles and the wet diameter. If *wet_diam_estimation* is equal to 1, the initial wet diameter is estimated from the input water concentrations (and so it is equal to the dry diameter if water concentration is zero initially). Finally, the names of the files containing initial gas and mass concentrations and number concentrations (if *with_init_num = 1*) are specified by the variables *init_gas_conc_file*, *init_aero_conc_mass_file* and *init_num_conc_num_file* respectively. The unit for gas and aerosol mass concentrations is $\mu$g m$^{-3}$, and the unit for number concentration is #particles m$^{-3}$.

```
&initial_condition
with_init_num = 1,         ! 0 estimated from mass and diameter;
  ! 1 number conc. for each bin is read
tag_init = 0,              ! initial method for aerosol species
```

```
                          ! (0 internally mixed,
                          ! 1 mixing_state resolved (notavailable)
wet_diam_estimation = 1   ! Initial estimation of wet diameter
                          ! (0 = isorropia, 1=none)
tag_dbd = 1,          ! Method for defining particle size bounds
          ! (0 if they are auto generated, 1 if bounds are read)
N_sizebin = 50,           ! Number of size bin
init_gas_conc_file = "inputs/init_gas.dat",        ! Initial data for gas
init_aero_conc_mass_file = "inputs/init_aero.dat", ! Initial data for aero mass
init_aero_conc_num_file = "inputs/init_num.dat",   ! Initial data for aero number
/

&initial_diam_distribution
diam_input = 1.0000000000000000E-03 1.2022644346174130E-03
1.4454397707459280E-03 1.7378008287493760E-03 2.0892961308540399E-03
2.5118864315095799E-03 3.0199517204020170E-03 3.6307805477010140E-03
4.3651583224016601E-03 5.2480746024977272E-03 6.3095734448019337E-03
7.5857757502918377E-03 9.1201083935590985E-03 1.0964781961431851E-02
1.3182567385564069E-02 1.5848931924611141E-02 1.9054607179632480E-02
2.2908676527677741E-02 2.7542287033381681E-02 3.3113112148259113E-02
3.9810717055349727E-02 4.7863009232263859E-02 5.7543993733715687E-02
6.9183097091893658E-02 8.3176377110267125E-02 1.0000000000000001E-01
1.2022644346174140E-01 1.4454397707459279E-01 1.7378008287493751E-01
2.0892961308540400E-01 2.5118864315095812E-01 3.0199517204020171E-01
3.6307805477010152E-01 4.3651583224016621E-01 5.2480746024977298E-01
6.3095734448019380E-01 7.5857757502918444E-01 9.1201083935590987E-01
1.0964781961431860E+00 1.3182567385564070E+00 1.5848931924611140E+00
1.9054607179632490E+00 2.2908676527677749E+00 2.7542287033381689E+00
3.3113112148259121E+00 3.9810717055349771E+00 4.7863009232263849E+00
5.7543993733715766E+00 6.9183097091893693E+00 8.3176377110267090E+00
1.0000000000000011E+01
/
```

Furthermore, there is an option to use fixed concentration profiles per time step for certain species defined by the user. A simple example of this can be seen in the mcm test case, where two files containing the temporal concentration profiles of inorganic compounds in gas and aerosol phases are specified in the namelist:

```
&initial_condition
...
! Constant concentration file for gas-phase species
cst_gas_file ="./inputs/inputs-mcm/gas.cst",
! Constant concentration file for aerosol-phase species
cst_aero_file ="./inputs/inputs-mcm/aero.cst",
...
```

In the file for gas-phase species, each line should contain the species name and the temporal concentrations ($\mu$g m$^{-3}$) during the simulation time. In the case of aerosol species, each line should include the aerosol species name, the size bin number, and the aerosol concentration ($\mu$g m$^{-3}$) per time step in the targeted size bin. If the time of the simulation does not match

the time used in the constant profils, there will probably be no error message output. Thus, please verify that the output concentrations are consistent with the input files.

Currently, this option is available in the two-step chemical solver for gas-phase chemistry (*tag_twostep*=1 in the namelist). The fixed concentrations are assigned only once at the beginning of each simulation time for aerosol dynamics. It is, therefore, possible that the concentrations of constant species in the output may differ from those in the input file. Meanwhile, the total concentrations (gas + aerosol) should be consistent with those in the input file, as the aerosol dynamics should not affect the total mass.

### 6.4 Mixing state

The group *mixing_state* defines the mixing state of particles. The variable *tag_external* is set to 0 for internally-mixed particles and to 1 for mixing-state resolved particles. The variable *N_groups* defines the number of group of aerosol compounds for which the composition is discretised. It is set to 1 for internal mixing, because the composition of compounds is then not discretised. In the case of mixing-state resolved particles, the belonging of each compound to a group is specified in the input file detailing the aerosol compounds and their properties. The variable *N_frac* determines the number of mass fraction sections used in the discretisation of composition. Finally, the variable *kind_composition* determines whether the fraction are discretized by the program (they are then evenly discretised, *kind_composition = 1*), or whether they are read. If they are read, they need to be specified in the group *fraction_distribution*.

```
&mixing_state
tag_external = 0,     ! Mixing state(0 for internally mixed,
              ! 1 for mixing-state resolved)
N_groups = 1,         ! Nb of species groups
N_frac = 1,        ! Nb of mass fraction sections
kind_composition = 1, ! Fraction discretization methods
      ! (1 for auto discretization and
                      ! 0 for manual discretization)
/


&fraction_distribution
frac_input= 0.0 1.0,  ! Set fraction bounds manually
/
```

### 6.5 Gas and aerosol species

The gas phase species are detailed in the group *gas_phase_species*, where the variable *species_list_file* contains the name of the file with the list of gas-phase species. In this file, gas-phase species are listed together with their molar weight in g/mol.

```
    &gas_phase_species
    species_list_file = "./species-list/species-list-cb05.dat",
    reaction_list_file =    "./src/include/CHEMISTRY/cb05/CB05.reactions",
/
```

`species_list_file` describes the gas-phase species and their molar mass.

`reaction_list_file` describes the list of the chemical reactions. If it is not given, `CB05.reactions` is used by default.

Aerosol species are detailed in the group *aerosol_species*, where the variable *aerosol_species_list_file* contains the name of the file with the list of aerosol species. In this file, each aerosol species is listed on a line, together with specific properties: the species type (1 for dust, 2 for elemental carbon, 3 for inorganics and 4 for organics), the group to which the species belong in case of mixing-state resolved particles, their molar weight (g/mol) and gaseous precursors, the collision factor, molecular diameter (Angstrom), surface tension (N/m), accommodation coefficient (between 0 and 1), density in $kg/m^3$ and whether or not the species is non-volatile ("1" for condensable but low-volatility species and "0" for others). Note that the condensation of non-volatile species is solved dynamically. The following column "partitioning" corresponds to whether the species condense on only an organic phase (keyword: HPHO), only an aqueuous phase (keyword: HPHI) or both (keyword: BOTH). In the next column, the smiles structure of the species can be species, or its decomposition into the 60 functional groups (see Table 4), then the saturation vapor pressure (in Torr), the enthalpy of vaporization, the Henry constant, and the reference temperature at which it is specified. Note that the Henry constant is not needed (in that case enter "–") if both the saturation vapor pressure and the smiles structure (or decomposition into functional groups) are provided.

```
&aerosol_species
aerosol_species_list_file = "./species-list/species-list-aer.dat",
/
```

Particles can be made of compounds of different categories: the category 1 corresponds to dust and other unspeciated compounds (PMD), the category 2 corresponds to black carbon (PBC), the category 3 corresponds to inorganics (sulfate (PSO4), nitrate (PNO3), ammonium (PNH4), chloride (PCL), sodium (PNA)) and the category 4 corresponds to organics. Characteristics of the surrogates for aerosol organic species are detailed in Table 2 and Table 3.

### 6.6 Use of generic organic aerosol species

To add an organic aerosol species in SSH-aerosol, two different methods can be used. The first method consists to add manually the species in the file src/include/SOAP/species.cxx file and to provide the different properties (saturation vapor pressure, henry's law constant, decomposition in functional group). However, this method is quite tedious, especially for a large number of species.
An alternative method was therefore developed using "Generic species". Generic species are used by SSH-aerosol when a species is not found in SSH-aerosol, but are present in the aerosol species file. If the different parameters are provided (smiles code, saturation vapor pressure, enthalpy of vaporization, molar mass, etc...); the species is created and automatically decomposed in UNIFAC functional groups with an evaluation of the necessary properties (for example the Henry's law constant is evaluated from the saturation vapor pressure and the activity coefficient at infinite dilution computed with UNIFAC).

Here is an overview of the UNIFAC structural groups used in SSH-aerosol to specify the aerosol structure. There are three ways in which users can provide the chemical structures of generic aerosol species:

- provide the SMILEs code of the generic species in the aerosol species list
- provide the list of UNIFAC functional groups (see Table 4) in aerosol species-list

Table 2: Characteristic of the surrogates for aerosol organic species: abbreviation of the surrogate, type (A: hydrophilic, type B: hydrophobic, type C: hydrophobic non-volatile and not used to compute activity coefficients), Molecular structure, Molecular weight MW ($g.mol^{-1}$), OM/OC ratio r, Henry's law constant H ($\times 10^8$ M/atm), partitioning constant $K_p$ ($m^3\ \mu g^{-1}$) for an ideal organic phase of molar mass 200 g/mol at 298 K, Enthalpy of vaporization ($kJ.mol^{-1}$), and precursors. The abbreviation n.v. means non volatile.

| Surrogate | Type | Molecular structure | MW | r | H | $K_p$ | $\Delta H_{vap}$ | Precursor |
|---|---|---|---|---|---|---|---|---|
| BiMT | A | methyl tetrol | 136 | 2.3 | 330 | 0.064 | 38 | |
| BiPER | A | methyl dihydroxy dihydroperoxide | 168 | 2.8 | 81 | 0.036 | 38 | |
| BiDER | A | methyl tetrol | 136 | 2.3 | 891 | 0.227 | 38 | isoprene |
| BiMGA | A | methyl glyceric acid (MGA) | 120 | 2.5 | 5.3 | 0.007 | 43 | |
| BiNGA | B | nitrate derivative of MGA | 165 | 3.5 | 0.4 | 0.007 | 43 | |
| BiNIT3 | B | methyl hydroxy trinitrate butane | 272 | 4.6 | 0.4 | 0.064 | 38 | |
| BiA0D | A | pinonaldehyde | 168 | 1.4 | 0.02 | 0.0003 | 50 | |
| BiA1D | A | norpinic acid | 170 | 1.5 | 1.12 | 0.428 | 50 | mono- |
| BiA2D | A | pinic acid | 186 | 1.7 | 2.7 | 0.650 | 109 | terpenes |
| BiA3D | B | 3-methyl-1,2,3-butane tricarboxylic acid | 204 | 2.1 | - | n.v. | - | |
| BiNIT | B | Nitrooxy-limonene-ol | 215 | 1.8 | 0.02 | 0.037 | 50 | |
| Monomer | C | $C_{10}H_{14}O_9$ | 278 | 2.3 | - | n. v. | - | |
| Dimer | C | $C_{19}H_{28}O_{11}$ | 432 | 1.9 | - | n. v. | - | |
| BiBlP | B | C15 hydroxy nitrate aldehyde | 298 | 1.7 | 2.34 | 154.9 | 175 | sesqui- |
| BiBmP | B | C15 oxo aldehyde | 236 | 1.3 | - | 0.309 | 175 | terpenes |
| AnBlP | B | methyl nitro benzoic acid | 167 | 2 | 2.48 | 1.37 | 50 | xylenes |
| AnBmP | B | methyl hydroxy benzoic acid | 152 | 1.6 | 12.4 | 0.011 | 50 | and |
| AnClP | C | No structure | 167 | 2 | - | n. v. | - | toluene |

Table 3: Characteristic of the surrogates for aerosol organic species: abbreviation of the surrogate, type (A: hydrophilic, type B: hydrophobic, type C: hydrophobic non-volatile and not used to compute activity coefficients), Molecular structure, Molecular weight MW ($g.mol^{-1}$), OM/OC ratio r, Henry's law constant H ($\times 10^8$ M/atm), partitioning constant $K_p$ ($m^3\ \mu g^{-1}$) for an ideal organic phase of molar mass 200 g/mol at 298 K, Enthalpy of vaporization ($kJ.mol^{-1}$), and precursors. The abbreviation n.v. means non volatile.

| Surrogate | Type | Molecular structure | MW | r | H | $K_p$ | $\Delta H_{vap}$ | Precursor |
|---|---|---|---|---|---|---|---|---|
| ACIDMAL | A | maleylacetic acid | 158 | 2.2 | 8685 | 2.02 | 50 | phenol catechol benzene |
| DHMB | A | dihydroxymethyl benzoquinone | 154 | 1.8 | 36.2 | 0.026 | 50 | methyl-catechol cresol |
| PSYR | A | $C_8H_{10}O_5$ | 186 | 1.9 | 14.5 | 0.0123 | 50 | syringol |
| GHDPerox | A | SOA (hydroperoxide) | 174 | 2.1 | 63.3 | 0.171 | 50 | guaiacol |
| PAHlN | A | dihydroxytere phtalic acid | 198 | 2.1 | n. v. | - | - | naphtalene methyl-naphtalene |
| PAHhN | A | phthalic acid | 166 | 1.7 | 14.9 | 0.093 | 50 | |
| POAlP | B | primary OA of low volatility | 280 | 1.3 | - | 1.1 | 106 | - |
| POAmP | B | primary OA of medium volatility | 280 | 1.3 | - | 0.0116 | 91 | - |
| POAhP | B | primary OA of high volatility | 280 | 1.3 | - | 0.00031 | 79 | - |
| SOAlP | C | secondary OA of low volatility | 392 | 1.8 | - | 110 | 106 | POAlP |
| SOAmP | B | secondary OA of medium volatility | 392 | 1.8 | - | 1.16 | 91 | POAmP |
| SOAhP | B | secondary OA of high volatility | 392 | 1.8 | - | 0.031 | 79 | POAhP |

- provide the list of UNIFAC functional groups in a separate file that is specified in the namelist by the name *aerosol_structure_file*.

The list of the UNIFAC functional groups used in SSH-aerosol contains 60 types of functional groups, as shown in Table 4.

| Index | UNIFAC main Group | UNIFAC subgroup (and corresponding number) |
|---|---|---|
| 1-4 | alkyl group (C) | 1: CH3, 2: CH2, 3: CH1, 4: C |
| 5-8 | alkyl directly linked to an alcohol (C[OH]) | 5: CH3, 6: CH2, 7: CH, 8: C |
| 9-12 | alkyl in a chain between two alcohols (Calcohol) | 9: CH3, 8: CH2, 7: CH, 8: C |
| 13-17 | alkyl in tails of alcohol chain (Ctail) | 13: CH3, 14: CH2, 15: CH, 16: C |
| 17-21 | double bound C=C group | 17: CH2=CH, 18: CH=CH, 19: CH2-C, 20: CH=C, 21: C=C |
| 22,23 | aromatic carbon (AC) | 22: AC-H, 23: AC |
| 24-26 | aromatic carbon linked to alkyl (AC-C) | 24: AC-CH3, 25: AC-CH2, 26: AC-CH |
| 27 | alcohol (OH) | 27: OH |
| 28 | water (H2O) | 28: H2O |
| 29 | aromatic carbon linked to acohol (ACOH) | 29: AC-OH |
| 30,31 | ketone (RCO) | 30: CH3CO, 31: CH2CO |
| 32 | aldehyde (HCO) | 32: CHO |
| 33,34 | ester (RC(=O)O) | 33: CH3COO, 34: CH2COO |
| 35-37 | ether (COC) | 35: CH3O, 36: CH3O, 37: CHO |
| 38 | acid (COOH) | 38: COOH |
| 39 | nitroaromatic (ACNO2) | 39: AC-NO2 |
| 40-42 | nitrate (CNO3) | 40: CH2ONO2, 41: CHONO2, 42: CONO2 |
| 43-45 | hydroxyperoxide (CO-OH) | 43: CH2OOH, 44: CHOOH, 45: COOH |
| 46-54 | peroxide (CO-OC) | 46: CH3OOCH2, 47: CH3OOCH, 48: CH3OOC, 49: CH2OOCH2, 50: CH2OOCH, 51: CH2OOC, 52: CHOOCH, 53: CHOOC, 54: COOC |
| 55 | peroxyacyl nitrates (PAN) | 55: PAN |
| 56 | Peroxyacetyl acid (C(=O)OOH) | 56: COOOH |
| 57 | Anhydre (O=COC=O) | 57: OCOCO |
| 58-60 | Nitroalkyl (CNO2) | 58: CH3NO2, 59: CH2NO2, 60: CHNO2 |

Table 4: List of UNIFAC Structural Groups used in SSH-aerosol to specify the aerosol structure.

**UNIFAC functional groups from SMILES** To use SMILES, simply put the SMILES code in the smiles column of the used aerosol species list. While this method should work properly for the vast majority of molecules (it has been been for the species formed from the oxidation of toluene, monoterpene and sesquiterpene MCM mechanism), we encourage you to check to the decomposition given by the code. If there are molecules not handled properly, the code may stop

or give you some warning (in some rare cases, the decomposition may not have work properly without showing messages). If *SOAPlog = 1* in the namelist, the decomposition into functional groups is shown on screen. If *SOAPlog = 2* in the namelist, the decomposition into functional groups is written in file *smile2UNIFAC.decomp*.

It should be noted that in numerous cases, the exact decomposition into functional groups is not possible. An approximate one is then used. It is especially the case when a carbon atom is share by two different groups. For example, the hydroperoxide group CHnOOH consists of OOH and CHn. If CHn is already included in another group (like another CHnOOH, a ketone, ether, nitrate, aromatic ring, etc.), the decomposition is not possible. In that case, another carbon is added (or a carbon is moved a from another place) so that the decomposition becomes possible. In that case, a warning is displayed: "WARNING: exact structure not possible, use of an approximate one".

**UNIFAC functional groups in aerosol species-list** This method can be used to do a manual decomposition of functional groups (for example, in case the decomposition of smiles does not work for some molecules).

Each functional group should be written in the exact format "&0.00E+0" without any space between elements. An example is shown in *species-list-aer-mcm-bcary-fgl.dat*.

For species PALCOMOXOOH, the complete manual decomposition should be written as follows in one line:

```
&0.00E+00&0.00E+00&0.00E+00&0.00E+00&0.00E+00&0.00E+00
&0.00E+00&0.00E+00&0.00E+00&0.00E+00&0.00E+00&0.00E+00
&0.00E+00&0.00E+00&0.00E+00&0.00E+00&0.00E+00&0.00E+00
&0.00E+00&0.00E+00&0.00E+00&0.00E+00&0.00E+00&0.00E+00
&0.00E+00&0.00E+00&0.00E+00&0.00E+00&0.00E+00&0.00E+00
&1.00E+00&1.00E+00&0.00E+00&0.00E+00&0.00E+00&0.00E+00
&0.00E+00&0.00E+00&0.00E+00&0.00E+00&0.00E+00&1.00E+00
&0.00E+00&0.00E+00&0.00E+00&0.00E+00&0.00E+00&0.00E+00
&0.00E+00&0.00E+00&0.00E+00&0.00E+00&0.00E+00&0.00E+00
&0.00E+00&0.00E+00&0.00E+00&0.00E+00&0.00E+00
```

The values correspond to the number of the different UNIFAC subgroups that are separated by the & symbol. Be careful as the numbers have to be written exactly over 8 characters. For an example of the functional group decomposition see the configuration file *species-list-aer-mcm-bcary-fgl.dat*.

One easier alternative is to only prescribe the necessary groups. The code has then to begin with the $ symbol. You can provide the number of the UNIFAC subgroup and the corresponding number. Groups have to be separated with the "|" symbol.

Here is an example with PALCOMOXOOH:

```
$32=1|$33=1|$43=1
```

**UNIFAC functional groups in *aerosol_structure_file*** For this option, each line in the aerosol structure file should include the name of the aerosol species, as well as the list of functional groups, using a comma as the separator. The printout messages during the simulation can confirm the generation of generic species in SSH-aerosol. All three options for creating aerosol species are used in the MCM test case described in Sect.8.6.

## 6.7 Oligomer species

A monomer species (for example called "X") can be transformed into oligomer species with the bulk oligomerization parameterization of Couvidat et al. (2018a) by changing the file ./species-list/species-list-aer.dat and adding a new species called "OligoX" ("Oligo" + the name of the monomer). An example is provided in ./species-list/species-list-aer-oligo.dat
The molar mass of the oligomer has to be consistent with the molar mass of the monomer and the number of monomer block in the oligomer. The bulk oligomerization parameterization is presented in section 8.5.2 "Type 1: Bulk Oligomerization".

## 6.8 Emissions

The group *emissions* defines options linked to emissions. The variable *tag_emis* defines whether emissions are used (*tag_emis = 1*) or not (*tag_emis = 0*). Emissions are assumed to be internally mixed. Gas-phase emissions and/or particle-phase emissions can be specified. Number concentrations at emission may be determined from mass emissions and section diameters if the variable *with_emis_num* is set to 0. They are read from a file is the variable *with_emis_num* is set to 1. The name of the file containing the list of gas-phase emitted species and their emission rates should be specified using the variable *emis_gas_file*. Similarly, the name of the file containing the list of aerosol-phase emitted species and their emission rates should be specified using the variable *emis_aero_mass_file*. Note that the unit for emission rates is $\mu$g m$^{-3}$ s$^{-1}$. If number emissions are read, the name of the files containing emission rates should be specified using the variable *emis_aero_num_file*. The units of number emissions should be #particles m$^{-3}$ s$^{-1}$.

```
&emissions
tag_emis = 0,       ! 0 Without emissions,
                    ! 1 with internally-mixed emissions,
                    ! 2 with externally-mixed emissions
with_emis_num = 0,  ! 0 if number estimated from mass and diameter;
    ! 1 if read
emis_gas_file = "./inputs/emis_gas.dat",
emis_aero_mass_file = "./inputs/emis_aero.dat",
emis_aero_num_file = "./inputs/emis_aero_num.dat"
/
```

## 6.9 Numerical and physical options

### 6.9.1 Gas-phase chemistry

The group *physic_gas_chemistry* lists options related to gas-phase chemistry. The variable *tag_chem* defines whether gas-phase chemistry is used (*tag_chem = 1*) or not (*tag_chem =0*). Photolysis reactions may be taken into account (*with_photolysis = 1*) or ignored (*with_photolysis = 0*). In case photolysis reactions are taken into account, they may be attenuated by clouds. The cloud attenuation (variable attenuation) has a value below 1 in case of cloud attenuation of photolysis, and it is equal to 1 if no cloud attenuation (clear sky). Heterogeneous reactions at the surface of particles may be taken into account (variable *with_heterogeneous = 1*) or ignored (variable *with_heterogeneous = 0*). An adaptive time step may be used to solve gaseous chemistry (variable *with_adaptive*). It is advised to use the adaptative time step and to set the relative tolerance to decide if the time step is kept to 0.01 or 0.001. The minimum time step (in

seconds) that can be used in the solver is set with the variable *min_adaptive_time_step*. For simulation of concentrations inside atmospheric chambers, wall losses of gases and particles can be accounted. The user can use a specific wall loss rate for organic vapors (*kwall_gas*) and particles (*kwall_particle*). It should be noted that for the losses of organic vapors onto wall are reversible. The parameter *Cwall* representing an equivalent wall concentration has to be provided. If not provided or equal to zero, wall losses of vapors are accounted for. Instead of providing *kwall_gas* and *kwall_particle*, the user can provide the eddy diffusion parameter (*eddy_turbulence*), the ratio between the surface and volume of the chamber (*surface_volume_ratio*), the minimal particle loss rate due to particle charge (*kwp0*) and the radius of the chamber. In that case, wall loss rates are computed as a function of the properties of gases and the diameter of particles.

```
&physic_gas_chemistry
tag_chem = 0,     ! Tag of gas-phase chemistry
attenuation = 1.d0,              ! Cloud attenuation field (0 to 1)
 ! (1 = no attenuation)
option_photolysis = 1,   ! 1 if default photolysis rates,
                                 ! 2 if read from binary files.
time_update_photolysis = 100000. ! if photolysis are read,
 ! time in seconds between two reads
with_heterogeneous = 0,          ! Tag of heterogeneous reaction
with_adaptive = 1,               ! Tag of adaptive time step for chemistry
 ! 1 if adaptive time step.
adaptive_time_step_tolerance = 0.001, ! Relative tolerance to decide
   ! if the time step is kept
min_adaptive_time_step = 0.001,  ! Minimum time step in seconds
photolysis_dir = "./photolysis/", ! Directory where binary files are.
photolysis_file = "./photolysis/photolysis-cb05.dat",  ! Photolysis list
n_time_angle = 9,                ! Parameters specific to the tabulation
   ! of photolysis rates
time_angle_min = 0.d0,   ! if read from files
delta_time_angle = 1.d0,
n_latitude = 10,
latitude_min = 0.d0,
delta_latitude = 10.d0,
n_altitude = 9,
altitude_photolysis_input = 0.0, 1000.0, 2000.0, 3000.0, 4000.0, 5000.0,
10000.0, 15000.0, 20000.0,
kwall_particle = 1.e-4,     !Fixed wall loss rate of particles (s-1).
                            !Used if kwp0 and eddy_turbulence are not provided
kwall_gas=3.e-4,            !Fixed wall loss rate of gases (s-1).
                            !Used in Cwall and eddy_turbulence are not provided
Cwall=10000,                !Around 10800*surface_volume_ratio
                            !(Huang et al., 2018) in ug/m3
surface_volume_ratio=2.22d0,       !Ratio surface/volume of the chamber (m-1)
eddy_turbulence=0.1d0,             !In s-1
kwp0=8.e-5,              !Minimal wall loss of particles due to particle charge
radius_chamber=1.,       !Radius of the chamber (m).
                 !Used to calculate particle deposition from eddy turbulence.
/
```

### 6.9.2 Numerical issues related to aerosols

The group *physic_particle_numerical_issues* lists options related to numerical issues when solving aerosol dynamics. The variable *DTAEROMIN* specifies the The minimum time step (in seconds) that can be used in the solver. Different redistribution methods of mass and number concentrations onto the fixed diameter grids may be used (variable *redistribution_method*). If only the process of condensation/evaporation is considered for aerosol dynamics, then it is possible to not apply redistribution (*redistribution_method = 0*). If nucleation and/or coagulation is also considered, then a redistribution method should be chosen. It is advised to use the redistribution 10 (moving diameter) or 12 (Euler coupled). The different redistributions (*redistribution_method*) are Euler mass (3), Euler number (4), hemen (5), moving diameter (10), area-based as in SIREAM (11), Euler coupled (12). The density of particles may be computed during the simulation depending on the composition of particles if the variable *with_fixed_density* is set to 0. If it is set to 1, then the density is fixed through the simulation to the value set by the variable *fixed_density* (in kg $m^{-3}$). Note that the variable *fixed_density* needs to be set.

Numerically, the nucleation and condensation/evaporation of inorganics are always solved simultaneously, because nucleation and condensation/evaporation are competing processes. Coagulation may also be coupled to nucleation and condensation/evaporation if the variable splitting is set to 1. If splitting = 0, coagulation is splitted from nucleation and condensation/evaporation. If nucleation is taken into account, it is recommended to set splitting to 1.

```
&physic_particle_numerical_issues
DTAEROMIN = 1E-5,                ! Minimum time step
redistribution_method = 0,    ! Redistibution method: 0: no redistribution
! 10: 10 Moving Diameter, 12: euler_coupled
with_fixed_density = 1,         ! 1 if density is fixed - 0 else
fixed_density = 1.84D3,
splitting = 0,              ! 0 if coagulation and
! (condensation/evaporation+nucleation) are splitted
! 1 if they are not
/
```

### 6.9.3 Coagulation

The group *physic_coagulation* lists options related to coagulation. The variable *with_coag* defines whether coagulation is taken into account (*with_coag = 1*) or not (*with_coag =0*). Repartition coefficients may be computed in the simulation (*i_compute_repart = 1*) or read from a netcdf file (*i_compute_repart = 0*). If they are read from a file, its name should be specified (*Coefficient_file*). If they are computed, the number of Monte Carlo points used to compute them should be specified (*Nmc*). This number should be large enough and its value depends on the section discretisation used.

```
&physic_coagulation
with_coag = 1,                  ! Tag of coagulation
i_compute_repart = 1,           ! 0 if repartition coeff are not computed
                                ! but read from file, 1 if they are
i_write_repart = 0,             ! 1 to write repartition coeff file, 0 otherwise
Coefficient_file = "coef_s1_f1_b6.nc",  ! Repartition coefficient file
Nmc = 2000                      ! Number of Monte Carlo points to compute
                                ! repartition coefficients
/
```

### 6.9.4 Condensation/evaporation

The group *physic_condensation* lists options related to condensation/evaporation. The variable *with_cond* defines whether condensation/evaporation is taken into account (*with_cond = 1*) or not (*with_cond =0*). Kelvin effect may be taken into account (*with_kelvin_effect = 1*), as recommended if ultrafine particles are simulated, or ignored (*with_kelvin_effect = 0*). For the condensation/evaporation of inorganic compounds, *Cut_dim* corresponds to the diameter under which thermodynamic equilibrium is assumed. Set it to 0 to compute dynamically condensation/evaporation for all particles, and set it to a value larger than the largest diameter to assume thermodynamic equilibrium.

For the condensation/evaporation of organic compounds, the parameter ISOAPDYN determines whether thermodynamic equilibrium is assumed for all particles (*ISOAPDYN = 0*) or whether condensation/evaporation is computed dynamically (*ISOAPDYN = 1*). Even if it is computed dynamically, thermodynamic equilibrium may be used for the condensation/evaporation of small particles by setting a characteristic time under which equilibrium is assumed for organics (e.g. *tequilibrium* = 0.1 seconds). For numerical reasons, *tequilibrium* may not be set to 0, if condensation/evaporation of all particles is solved dynamically ( if*ISOAPDYN = 1*). Instead, *tequilibrium* should be set to a small value (e.g. 1.d-15).

The effect if viscosity can be accounted for with two method: a constant diffusion coefficient or one that evolves with the particle composition. If the value of the diffusion coefficient *dorg* in the organic phase is provided, the coefficient is assumed to be constant. Typical values would be 1.d-12 for non viscous particles and 1.d-24 for very viscous particles. If dorg is set to zero, then AIOMFAC-VISC algorithm is used to calculate activity coefficient (Gervasi et al., 2020).

The variable *coupled_phases* should be set to 1 if the resolution of the aqueous and organic phases is coupled and to 0 if they are solved independently. The interactions between compounds in the particles may be assumed to be ideal (*activity_model = 1*), or activity coefficients may be computed with unifac (interactions between organics only, *activity_model = 2*) or with aiomfac (*activity_model = 3*).

Two different thermodynamic models may be used for inorganic compounds. They can be selected with the parameter *soap_inorg*: ISORROPIA (*soap_inorg = 0*) or SOAP (*soap_inorg = 1*).

The parameter *SOAPlog* is used to determine if the values of Henry's law constant and the SMILES decomposition of generic species calculated by SOAP should be printed or written in files. If *SOAPlog = 0*, no message is printed. If *SOAPlog = 1*, the messages are printed on screen. If *SOAPlog = 2*, the SMILES decomposition and the Henry's law constant are written in the file *smile2UNIFAC.decomp* and *henryfromSOAP.dat*, respectively. Intraparticle reactions (such as hydrolysis, oligomerization, hydratation) can be added via *reaction_soap_file*. This file and the reactions accounted for are described in Section 8.5.

```
&physic_condensation
with_cond = 0,              ! Tag of condensation/evaporation
Cut_dim = 0.0,              ! Diameter under which equilibrium
                            ! is assumed for inorganics
ISOAPDYN = 1,               ! 0 = equilibrium, 1 = dynamic
nlayer = 1,
with_kelvin_effect = 0,     ! 1 if kelvin effect is taken into account.
tequilibrium = 0.1,         ! time under which equilibrium
                            ! is assumed for organics.
dorg = 1.d-12,              ! diffusion coefficient
                            ! in the organic phase.
```

```
                          ! If zero, AIOMFAC-VISC is used
coupled_phases = 1,       ! 1 if aqueous and organic phases
                          ! are coupled
activity_model = 1,       ! 1: ideal, 2: unifac, 3: aiomfac
epser = 0.01,             ! relative error for time step adjustment
epser_soap = 0.01,        ! relative difference of ros2 in SOAP
soap_inorg = 0,           ! 0: use of isorropia, 1: inorganics computed by SOAP
SOAPlog = 1,                 !0=no text output (Henry + smiles decomposition) from SOAP, 1=
reaction_soap_file = "species-list/REACTIONS.particle.soap",    !Reaction file for intrapart
/
```

### 6.9.5 Nucleation

The group *physic_nucleation* lists options related to nucleation. The variable *with_nucl* defines whether nucleation is taken into account (*with_nucl = 1*) or not (*with_nucl =0*). If nucleation is taken into account, then the lower diameter bound should be about 1 nm. Several nucleation parameterizations are implemented. They can be used separately or together. An example of their use in 3D is illustrated in Sartelet et al. (2022).

- binary: water and sulfuric acid with the parameterisations of Vehkamaki et al. (2002) or Kuang et al. (2008)
- ternary: water, sulfuric acid and ammonia with the parameterisation of Napari et al. (2002) or Merikanto et al. (2007, 2009) To avoid artificially large nucleation rates in the parameterisation of Napari et al. (2002), a maximum nucleation rate of 1.d6 #particles $cm^{-3}$ is set. A scaling factor of the nucleation rate can be applied by setting the parameter scal_ternary to the scaling value.
- heteromolecular: sulfuric acid, and extremely-low volatile compounds from monoterpene autoxidation with the parameterisation of Riccobono et al. (2014). The names of the particle ELVOC compounds from monoterpenes autoxidation should be specified after the flag "name_org_h2so4_nucl_species"
- organics: extremely-low volatile compounds may nucleate by setting the parameter nucl_model_org to 1. The number of nucleating compounds should be specified (parameter name_org_nucl_species), and the species name provided (parameter name_org_nucl_species). The nucleation rate can be ajusted by specifying a exponent of the power law (parameter nexp_org) and scaling factor (parameter scal_org).

```
&physic_nucleation
with_nucl = 1,    ! Tag of nucleation
       ! Need to have the lowest diameter about 1 nm.
nucl_model_binary = 0,   !Tag of type of binary nucleation model
                         !(0=none, 1=Veahkamaki, 2=Kuang)
nucl_model_ternary = 1,  !Tag of type of ternary nucleation model
                         ! (0=none, 1=Napari, 2=Merikanto)
scal_ternary = 1.,       ! Scaling factor for ternary nucleation rate
nucl_model_hetero = 0,   ! Tag of heteromolecular nucleation (0=none, 1=yes)
scal_hetero = 1,         ! Scaling factor for heteromolecular nucleation rate
nesp_org_h2so4_nucl = 2, ! number of organics involved
```

```
                          ! in heteromolecular nucleation
name_org_h2so4_nucl_species = 'PMonomer' 'PDimer', ! Species names involved
                                                   ! in heteromolecular nucleation
nucl_model_org = 1,        ! Tag of organic nucleation (0=none, 1=yes)
scal_org = 1e-12,              ! Scaling factor for organic nucleation rate
nexp_org = 3,              ! Exponent of the power law
nesp_org_nucl = 2,         ! number of organics involved in organic nucleation
name_org_nucl_species = 'PMonomer' 'PDimer',  ! Species names involved in organic nucleatior
/
```

## 6.10 Using user-defined chemical scheme for SOA formation

Several chemical schemes are available for SOA formation. Although many test cases use the $H^2O$ scheme (Couvidat et al., 2012; Sartelet et al., 2020), others use near-explicit schemes, such as those presented in Lannuque et al. (2023); Lannuque and Sartelet (2024); Wang et al. (2023), or chemical schemes reduced from near-explicit chemical schemes using GENOA (Sartelet et al., 2024; Wang et al., 2023). You may customized the chemical scheme and apply a user-defined chemical scheme. This can be done by following the steps below:

**Step 1: Prepare a chemical scheme describing ozone and oxidant formation without SOA formation**
An example using the CB05 mechanism is given in `src/include/CHEMISTRY/cb05-ozone`:
`CB05-ozone.species` and
`CB05-ozone.reactions`.
Another example using RACM2 is given in `src/include/CHEMISTRY/racm2-2020rad-ozone`:
`RACM2-2020rad-ozone.species` and
`RACM2-2020rad-ozone.reactions`.

**Step 2: Prepare** `species_matching.dat` As species may have different names in different mechanisms, an instruction is required to match between generic species names and model species names in the file `species_matching.dat`. The file should be placed next to the species and reactions files. The instruction for CB05 mechanism is

```
hydroxyl OH
alpha_pinene API
limonene LIM
beta_pinene BPI
ozone O3
nitrate_radical NO3
hydroperoxy HO2
toluene TOL
methylperoxy MEO2
acetylperoxy C2O3
nitric_oxide NO
xylene XYL
isoprene ISOP
nitrogen_dioxide NO2
humulene HUM
sulfur_dioxide SO2
formaldehyde FORM
```

```
nitric_acid HNO3
peroxynitric_acid HNO4
carbon_monoxide CO
```

**Step 3: Prepare a used-defined scheme, for example, toluene** Prepare SOA chemical schemes for the different precursors for which you would like to simulate oxidation. Different pre-prepared schemes are already available in the repertory `src/include/CHEMISTRY/individual_schemes`. The $H^2O$ schemes, built using data from chamber experiments, are available for toluene, xylenes, benzene, cresol, humulene, methyl-naphthalene, naphthalene, syringol, beta-caryophyllene, guaiacol, isoprene. Near-explicit schemes are available for monoterpenes and sesquiterpene (beta-caryophyllene) (Wang et al., 2023), they are based on MCM and PRAM for monoterpenes and on MCM for beta-caryophyllene. Near-explicit schemes are available for toluene (Lannuque et al., 2023) and napthalene (Lannuque and Sartelet, 2024). Schemes reduced from the neat-explicit schemes using GENOA are available for monoterpenes and beta-caryophyllene (Wang et al., 2023) and for toluene (Sartelet et al., 2024).

**Step 4: Prepare** `user_defined_scheme.cfg` This file describes which mechanisms are taken into account. It should be located in `src/include/CHEMISTRY/cb05-ozone` if CB05 is used for ozone chemistry, or in `src/include/CHEMISTRY/racm2-2020rad-ozone` if RACM2 is used.

```
# This file is required by the script 'user_defined_scheme.py'.

[schemes]

toluene = yes
xylene = yes
alpha-pinene = yes
beta-pinene = yes
limonene = yes
isoprene = yes
humulene = yes
poa = yes
benzene = yes
cresol = yes
syringol = yes
guaiacol = yes
naphthalene = yes
methyl-naphthalene = yes


[toluene]

species = ../toluene/h2o/toluene.h2o.species
reactions = ../toluene/h2o/toluene.h2o.reactions
aerosol_species = ../toluene/h2o/species-list-aer-toluene.dat


[xylene]
```

```
species = ../xylene/h2o/xylene.h2o.species
reactions = ../xylene/h2o/xylene.h2o.reactions
aerosol_species = ../xylene/h2o/species-list-aer-xylene.dat

[alpha-pinene]

species = ../alpha-pinene/h2o/alpha-pinene.h2o.species
reactions = ../alpha-pinene/h2o/alpha-pinene.h2o.reactions
aerosol_species = ../alpha-pinene/h2o/species-list-aer-api.dat

[beta-pinene]

species = ../beta-pinene/h2o/beta-pinene.h2o.species
reactions = ../beta-pinene/h2o/beta-pinene.h2o.reactions
aerosol_species = ../beta-pinene/h2o/species-list-aer-bpi.dat

[limonene]

species = ../limonene/h2o/limonene.h2o.species
reactions = ../limonene/h2o/limonene.h2o.reactions
aerosol_species = ../limonene/h2o/species-list-aer-lim.dat

[isoprene]

species = ../isoprene/h2o/isoprene.h2o.species
reactions = ../isoprene/h2o/isoprene.h2o.reactions
aerosol_species = ../isoprene/h2o/species-list-aer-iso.dat

[humulene]

species = ../humulene/h2o/humulene.h2o.species
reactions = ../humulene/h2o/humulene.h2o.reactions
aerosol_species = ../humulene/h2o/species-list-aer-hum.dat

[poa]

species = ../poa/h2o/poa.h2o.species
reactions = ../poa/h2o/poa.h2o.reactions
aerosol_species = ../poa/h2o/species-list-aer-poa.dat

[benzene]

species = ../benzene/h2o/benzene.h2o.species
reactions = ../benzene/h2o/benzene.h2o.reactions
aerosol_species = ../benzene/h2o/species-list-aer-ben.dat

[cresol]

species = ../cresol/h2o/cresol.h2o.species
```

```
reactions = ../cresol/h2o/cresol.h2o.reactions
aerosol_species = ../cresol/h2o/species-list-aer-cre.dat

[syringol]

species = ../syringol/h2o/syringol.h2o.species
reactions = ../syringol/h2o/syringol.h2o.reactions
aerosol_species = ../syringol/h2o/species-list-aer-syr.dat

[guaiacol]

species = ../guaiacol/h2o/guaiacol.h2o.species
reactions = ../guaiacol/h2o/guaiacol.h2o.reactions
aerosol_species = ../guaiacol/h2o/species-list-aer-gua.dat

[naphthalene]

species = ../naphthalene/h2o/naphthalene.h2o.species
reactions = ../naphthalene/h2o/naphthalene.h2o.reactions
aerosol_species = ../naphthalene/h2o/species-list-aer-nap.dat

[methyl-naphthalene]

species = ../methyl-naphthalene/h2o/methyl-naphthalene.h2o.species
reactions = ../methyl-naphthalene/h2o/methyl-naphthalene.h2o.reactions
aerosol_species = ../methyl-naphthalene/h2o/species-list-aer-mna.dat
```

**Step 5: Compile**

```
> compile -b=user -c=cb05-ozone
-u=user_defined_scheme.cfg -m=species_matching.dat
```

or

```
> compile -b=user -c=racm2-2020rad-ozone
-u=user_defined_scheme.cfg -m=species_matching.dat
```

Default files are `user_defined_scheme.cfg` and `species_matching.dat` if `-u` or `-m` are not explicitly indicated.

**Remove files made for the user-defined scheme** If you want a deep cleaning including the files which are generated from the scripts for the user-defined scheme, please add the flag `-a` next to `clean`.

```
> clean -a
```

### 6.11 Output

The output directory may be specified, as well as the format of the files (No output if *output_type = 0*, text if *output_type = 1*, binary if *output_type = 2*, and NetCDF if *output_type = 3*). The

structure of output files and folders is described in section 4.3. When no output (*output_type = 0*), other variables can be used to have outputs only for specified gas-phase and/or aerosol-phase species (See section. 7.2).

```
&output
output_directory = "results/coag/",
output_type = 1               ! 1: text, 2: binary
/
```

# 7 Coupling

## 7.1 Coupling with external tools

SSH-aerosol can be coupled with external (3D) tools using the shared library (`libssh-aerosol.so`) available in the `src` folder after compilation. The `.so` file is produced with the command `./compile --sharedlib=yes`. A prototype of the typical workflow is described hereafter.

### 7.1.1 Prerequisite

The following piece of **C** code is taken from the open-source CFD code Code_Saturne. The subroutine `_get_dl_function_pointer` is used to interact with the members of the shared library object.

```
/*----------------------------------------------------------------------------
 * Get a shared library function pointer
 *
 * parameters:
 *   handle           <-- pointer to shared library (result of dlopen)
 *   name             <-- name of function symbol in library
 *   errors_are_fatal <-- abort if true, silently ignore if false
 *
 * returns:
 *   pointer to function in shared library
 *----------------------------------------------------------------------------*/

static void *
_get_dl_function_pointer(void        *handle,
                         const char  *lib_path,
                         const char  *name,
                         bool         errors_are_fatal)
{
  void  *retval = NULL;
  char  *error = NULL;

  dlerror();    /* Clear any existing error */

  retval = dlsym(handle, name);
  error = dlerror();

  if (error != NULL) { /* Try different symbol names */
```

```
    char *name_ = NULL;
    dlerror();    /* Clear any existing error */
    int _size_ = strlen(name) + strlen("_");
    BFT_MALLOC(name_, _size_ + 1, char);
    strcpy(name_, name);
    strcat(name_, "_");
    retval = dlsym(handle, name_);
    error = dlerror();
    BFT_FREE(name_);
  }

  if (error != NULL && errors_are_fatal)
    bft_error(__FILE__, __LINE__, 0,
              _("Error while trying to find symbol %s in lib %s: %s\n"),
              name,
              lib_path,
              dlerror());

  return retval;
}
```

### 7.1.2 Initialisation

First, the external code should load the shared library using `dlopen`.

```
  /* Load the shared object */
  _aerosol_so = dlopen(lib_path, RTLD_LAZY);
```

Then, it should decide whether SSH-aerosol outputs to the terminal or to a file.

```
  /* Declare SSH-aerosol as not running standalone */
  {
    typedef void (*cs_set_sshaerosol_t)(bool*);
    cs_set_sshaerosol_t fct =
      (cs_set_sshaerosol_t) _get_dl_function_pointer(_aerosol_so,
                                     lib_path,
                                     "api_set_sshaerosol_standalone",
                                     true);
    bool flag = false;
    fct(&flag);
  }
  /* Force SSH-aerosol to write output to a file */
  if (cs_glob_rank_id <= 0) {
    typedef void (*cs_set_sshaerosol_t)(bool*);
    cs_set_sshaerosol_t fct =
      (cs_set_sshaerosol_t) _get_dl_function_pointer(_aerosol_so,
                                                     lib_path,
                                                    "api_set_sshaerosol_logger",
                                                     true);
    bool flag = true;
    fct(&flag);
```

```
}
```

Then, SSH-aerosol can be initialized.

```
/* Initialize SSH-aerosol */
{
  const char namelist_ssh[40] = "namelist_coag.ssh";
  typedef void (*cs_set_sshaerosol_t)(char*);
  cs_set_sshaerosol_t fct =
    (cs_set_sshaerosol_t) _get_dl_function_pointer(_aerosol_so,
                                                   lib_path,
                                                  "api_sshaerosol_initialize",
                                                   true);
  fct(&namelist_ssh);
}
```

### 7.1.3 Time advancement

The time advancement in the external 3D code could look as follows: for each time step, a loop on the cells is performed and the code

- Sets (initialises) the time step in SSH-aerosol using `api_set_sshaerosol_dt`
- Sets the Pressure, Temperature, pH, ... in SSH-aerosol using `api_set_sshaerosol_temperature` and similar functions
- Sets the gaseous concentrations in SSH-aerosol using `api_set_sshaerosol_gas_concentration`
- Sets the aerosol concentrations in SSH-aerosol using `api_set_sshaerosol_aero_concentration`
- Sets the aerosol numbers in SSH-aerosol using `api_set_sshaerosol_aero_number`
- Advance in time (computes one time step) for the gaseous chemistry and for the aerosol chemistry in the given cell using `api_call_sshaerosol_gaschemistry` and `api_call_sshaerosol_aerochemistry`
- Reads the new concentrations in the given cell from SSH-aerosol using the `api_get_*` functions and use them in the external code

### 7.1.4 Finalisation

At the end of the simulation, the external tool should call the subroutine `api_sshaerosol_finalize`. Then, the shared library can be released using `dlclose`.

```
dlclose(_aerosol_so);
```

### 7.1.5 Parallelism

If the external tool is running/parallelized with MPI, each MPI process of the external tool should load the shared library and perform all the aforementioned operations. However, please note that only one MPI process can use the logger (`api_set_sshaerosol_logger`).

## 7.2 Adaptation for the GENOA algorithm

SSH-aerosol is coupled with the mechanism reduction algorithm known as the GENerator of reduced Organic Aerosol mechanisms (GENOA) (Wang et al., 2022, 2023), allowing users to generate condensed SOA mechanisms from near-explicit VOC mechanisms tailored to their specific scale and accuracy requirements. In GENOA, SSH-aerosol is mainly utilized for aerosol structure decomposition and mechanism reduction evaluation. This section summarizes all major functions and options used in GENOA (current version v2.0), which can also be customized by the users according to their own needs.

### 7.2.1 Compilation option

To use GENOA-specific options, SSH-aerosol needs to be compiled by adding the GENOA mode: code-genoa (or `-g`) compilation option with values of *fast* or *complete*.

Here are the commands you can use to compile SSH-aerosol under the GENOA mode:

```
## Use one of the following commands to compile SSH-aerosol under:
# ... the fast GENOA mode
>>> ./compile -g=fast
>>> ./compile -genoa=fast
# ... the complete GENOA model
>>> ./compile -g=complete
>>> ./compile -genoa=complete
```

In GENOA mode, simulation outputs may be modified:

- Under both the fast and complete modes: the output is a concentration file named ***concs.txt*** under the user-defined result folder *output_directory*.
- Under the complete mode: same outputs as those generated with the normal mode except for adding concs.txt.
- Under the fast mode: if no specified species lists are given in the "namelist.ssh", no other results are generated (same as output_type = 0 discussed in Sect. 6.11).

By default, there is no printout message under the fast mode. To allow printout messages on the screen, the user can activate the ssh_standalone option under the GENOA fast mode in the *src/include/Module/ModuleInitialization.f90* file:

```
....
#ifdef UNDER_GENOA_FAST_MODE
    tag_genoa = 1
    ssh_standalone = .false. ! Change here to .true. !
    ssh_logger = .false.
    iout_round = .false.
#endif
...
```

The printout message will show up during simulation runs after recompilation.

### 7.2.2 Clean option

Besides the compilation option, SSH-aerosol also provides a GENOA clean option (-genoaclean or -g) that reduces the size of the SSH-aerosol folder and only preserves the essential codes to run SSH-aerosol simulations:

```
## Use one of the following commands to clean SSH-aerosol - irreversible !!!
# Require user confirmation
>>> ./clean -genoa=fast -genoaclean=yes
>>> ./clean -g=fast -gcl=yes
```

The genoa mode can be set to complete. To prevent deleting accidentally, the -genoaclean option won't work automatically except for running with the genoa compilation mode. A user confirmation will be also requested as illustrated below:

```
>>> ./clean -g=fast -gcl=yes

...
GENOA version: fast
Attempting to clean up unused folders/files for GENOA runs.
 CAUTION: This action cannot be reversed!

Are you sure you want to remove ALL items that are not used with
 the GENOA v3 algorithm? (y/N): y

...
[start cleaning process]
```

To disable the confirmation, set -genoaclean to *cleanall*.

### 7.2.3 Configuration option

Several configuration variables/options are primarily developed based on the use of GENOA, but can also be used under normal mode. Before looking into one after one, here is a summary of those that have not been introduced in the previous sections:

```
! Additional variables for initial_condition section:
& initial_condition
...
! A file contains different initial conditions for specific species
init_species_file = "inputs"

! Additional variables for the output section:
&output
...
! Output concentrations for targeted species in subfolder gas/
output_gas_list = "species1,species2,...,species_N"

! Output concentrations for targeted species in subfolder aero/
output_aero_list = "species1,species2,...,species_N"
```

```
! Output fractional errors computed to those in the given files
ref_conc_files_in = "results1/ref/concs.txt,...,results_N/concs.txt"

! Add concentrations of those species as columns in file concs.txt
err_species_list = "O3,OH"

! Add concentrations of nout_soa SOA groups as columns in file concs.txt
! For internal mixing only, SOA groups are given by Index_groups(s)
nout_soa =
```

- init_species_file that can set different initial concentrations for specified species. It needs to be used with the specified simulation command
- output_gas_list and output_aero_list options allow you to output simulation results only for specified gas-phase and aerosol-phase species. These options are provided to save simulation runtime, particularly when opening, closing, and writing files is time-consuming. Note that these options only work when GENOA is in fast compile mode or when out_type is set to 0. Multiple species can be specified, using commas (",") as separators. Please ensure there are N-1 commas for N species in the inputs.

The example configuration file *namelist.ssh* provides an example of how to use these options. Note that they will not be used if they are not specified or if "—" is provided in the configuration file. The options `output_gas_list` and `output_aero_list` only work in the Genoa fast mode. The names of the species provided must be predefined in the species lists.

## 7.3 Use of external chemical mechanisms

Our model also provides the option to simulate SOA formation using VOC degradation schemes from the Master Chemical Mechanism (MCM) developed by Jenkin et al. (1997), as well as from the Generator for Explicit Chemistry and Kinetics of Organics in the Atmosphere (GECKO-A) (Camredon et al., 2007). Since the conversion process for both MCM and GECKO-A mechanisms is similar, this section will focus on explaining how to convert an MCM scheme for use in SSH-aerosol as an example.

### 7.3.1 Download external mechanism files

Users first need to download the reaction list in FACSIMILE format and the species list in tab-separated format from the MCM website.

Detailed instructions for this step are as follows:

1. Visit the MCM website at `https://mcm.york.ac.uk/MCM/`. Users can use the "browser" option or the search function to select the compounds whose mechanisms they wish to export. Add the selected compounds to the "marklist". Users can then can view their marklist by clicking the basket icon in the top right corner of the website.
2. Once the marklist is ready, click "export" or go to the export section to access the export page. Users will need to export both the reaction list and the species list. To export the reaction list for the selected compounds, choose the output format "FACSIMILE" and click the "Download" button. To export the species list, select "Species List (Tab-Separated Values)" as the output format, and download it. These files will be required for the format conversion in the next step.

3. On the "export page", there are two options: whether to include inorganic reactions, and whether to include generic rate coefficients. The latter does not affect the FACSIMILE output format, so either choice is acceptable. It is recommended to exclude inorganic reactions, as they are already provided within SSH-aerosol.

### 7.3.2 Use the converter

User then need to use the format converter to convert the MCM mechanism from the downloaded files into format compatible with SSH-aerosol. The converter is available at `https://github.com/tool-genoa/Converter`, and a detailed README file explains its structure, usage, and provides conversion examples.

To use the converter, users need to:

1. Download the converter from the GitHub page. The converter is written in Python and requires Python3 along with the libraries OpenBabel (which can be installed using `apt-get` or `pip`) and UManSysProp (which can be downloaded from the GitHub page).

2. Modify the user-defined parameters in the `parameters.py` file by following the instructions provided in the comments. Examples of how to make these modifications can be found in the ***examples/scripts*** folder, with example inputs located in ***examples/inputs***. The major user-defined parameters that should be changed include defining the output directory, specifying the input paths for the reaction and species lists, and selecting the input mechanism types. Since MCM does not provide aerosol properties, users will also need to define variables such as the method to compute saturation vapor pressures for condensable species. The converter will use UManSysProp to compute aerosol-related properties accordingly.

3. After configuring the parameters, you can run the converter with the command: `python3 converter.py`. If the process runs successfully, the converted mechanism will be available in the output directory defined in `parameters.py`. Example output mechanisms can be found in the ***examples/outputs*** folder.

Users will obtain a set of mechanism files as the output of running the converter, where `[chem_id]` is the user-defined mechanism identifier:

1. `[chem_id].species`: A species list file that can be directly used in SSH-aerosol simulations.

2. `[chem_id].reactions`: A reaction list file that can be directly used in SSH-aerosol simulations.

3. `[chem_id].aer.1st`: An aerosol species list file containing SMILES structures, where aerosol species are disconnected from their precursors (precursors for organic aerosols are not specified). This file is used to generate the SOAP output information required for further conversion with functional group decomposition as explained below.

4. `[chem_id].aer`: An aerosol species list file containing SMILES structures, where aerosol species are connected to their precursors. This file can be used to obtain SOAP output information along with the generated species file, and to run SSH-aerosol simulations with SMILES structures.

5. `[chem_id].aer.vec`: An aerosol species list file that includes functional group decomposition information. This file can be used to run SSH-aerosol simulations.

6. `[chem_id].mol`: A molecular properties file containing the molecular properties of all species in the mechanism. Together with the reaction file, this file can be used in the converter and GENOA to import a mechanism into the SSH-aerosol format.

These files can be used in SSH-aerosol simulations following instructions similar to Steps 1 and 2 in Section 6.10.

To include functional group decomposition information (e.g., `smile2UNIFAC.soap`, which is required as the variable `soapfile` in `parameters.py`), users need to run the converter twice:

1. Run the converter for the first time with `soapfile = None` in the `parameters.py` script.

2. Run an SSH-aerosol simulation with the generated aerosol species list that has the suffix ***.aer.1st***. The simulation settings do not matter, as the goal is to activate the decomposition tool in SSH-aerosol to get functional group decomposition for aerosol species. Since this aerosol list is dissociated from the reaction and species lists, it can be run with other mechanisms, meaning users do not need to change much in the namelist except for the path to the aerosol species list. The run will print information on the screen or to a file for aerosols using SMILES formatting as input in the given aerosol species list.

3. Save this information to a file if needed and provide the converter with the path to this file as the variable `soapfile`.

4. Run the converter a second time. This time, the content of the aerosol species list file (with the suffix ***.aer.vec***) should be updated with the correct decompositions.

For more details or if you encounter any issues when converting chemical schemes to the SSH-aerosol format, please contact the author.

# 8 Test cases

This section presents a few test-cases to demonstrate how the model works, as well as the different model capabilities.

During the practical session, we will work in the directory *ssh-aerosol*. To compile the program, please type *compile*. Before compiling, you may clean previous compilation by typing *clean*.

The main options of the simulations are detailed in the namelist files (for example *INIT/namefile_coag.ssh*), which are in the folder *INIT*, and initial conditions (meteorological, gas and aerosol concentrations) are required. The simulation can be run by typing *ssh-aerosol INIT/namefile_coag.ssh*.

The list of species and parameters are detailed in the files *species-list/species-list-aer.dat* and *species-list/species-list-cb05.dat*. The name and location of the files can be modified in the configuration file *namelist*.ssh*.

Different processes can be considered (set the flag to 1) or ignored (set the flag to 0): emissions (flag *tag_emis*), gaseous chemistry (flag *tag_chem*), coagulation (flag *with_coag*), condensation (flag *with_cond*), nucleation (flag *with_nucl*). Internal mixing (flag *tag_external* set to 0) or mixing-state resolved particles (flag *tag_external* set to 1) can be considered.

## 8.1 Dynamic of coagulation, condensation/evaporation and nucleation

In the literature, to test the overall behaviour of PM models, the following basic tests of condensation and coagulation of sulfate are often considered (Binkowski and Roselle, 2003; Seigneur

et al., 1986; Zhang et al., 1999). A tri-modal PM distribution is considered initially, with particles being made exclusively of sulfate. The parameters of the initial distribution considered here are those of hazy conditions for the condensation test with a sulphuric acid production rate of 9.9 $\mu$g m$^{-3}$, and those of urban conditions for the coagulation test (Seigneur et al., 1986; Zhang et al., 1999), because these two tests represent the two most stringent conditions for coagulation and condensation. Temperature is taken as 283.15 K. Simulations are conducted for 12 hours. The reference solutions for the coagulation and condensation tests are those of Zhang et al. (1999) obtained with other models.

### 8.1.1 Coagulation

The configuration file for this test is *namelist_coag.ssh*. In the coagulation test case, only coagulation is considered by setting the variable of *with_coag* of the file *namelist_coag.ssh* to 1.

The partition coefficients may be precomputed in a C++ routine. Here, the coefficients are directly computed by SSH-aerosol (option *i_compute_repart*), using a Monte-Carlo method (Nmc represents the Monte Carlo number). The larger Nmc is, the more accurate the coefficients are, but the more CPU time it takes to compute them. Run the simulation by typing *ssh-aerosol INIT/namelist_coag.ssh*. You can compare the number and volume distribution of particles at the initial time and after 12 h by going to the repertory *graph* and by running the python script *dN_Vdlogd_coag.py*. SSH-aerosol does very well in representing the growth of particles by coagulation, as shown in Fig 2.

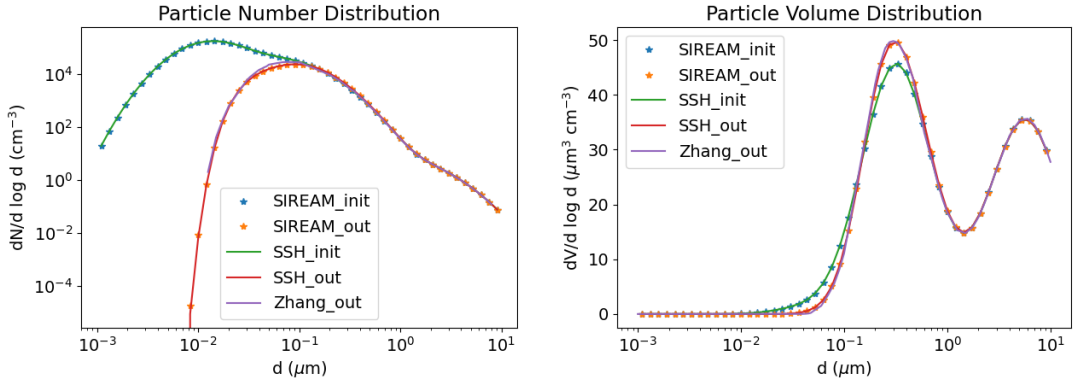


Figure 2: Coagulation test case. Number (left panel) and volume (right panel) concentrations.

### 8.1.2 Condensation of sulfate and redistribution algorithm

The condensation test with hazy conditions is very stringent, because the high condensation rate leads to a narrow Aitken mode. The configuration file for this test is *namelist_cond.ssh*. Only condensation/evaporation is considered by setting the variable *with_cond* of the file *namelist.ssh* to 1.

Run the simulation by typing *ssh-aerosol INIT/namelist_cond.ssh*. You can compare the number and volume distribution of particles at the initial time and after 12 h by going to the repertory *graph* and by running the python script *dN_Vdlogd_cond.py*. SSH-aerosol does very well in representing the Aitken mode as well as the growth of the accumulation mode if no redistribution is used (*redistribution_method* = 0 in *namelist_cond.ssh*), as shown in Fig 3.

Redistributing the mass and number concentrations amongst bins lead to numerical diffusion, as can be seen by using the redistribution methods 10 (moving diameter), 11 (area-based

(siream)) or 12 (Euler-coupled) (see Fig 4). You can change the redistribution by changing the flag *redistribution_method*.

In order to use a growth law, which is as close as possible to the original growth law used in Zhang et al. (1999), the accommodation coefficient is set to 1. It is however interesting to notice the sensitivity of results to the choice of the accommodation coefficient. The growth of the Aitken mode is strongly reduced by decreasing the accommodation coefficient from 1 to 0.1. You can change the accommodation coefficient in the file *species-list/species-list-aer.dat*.

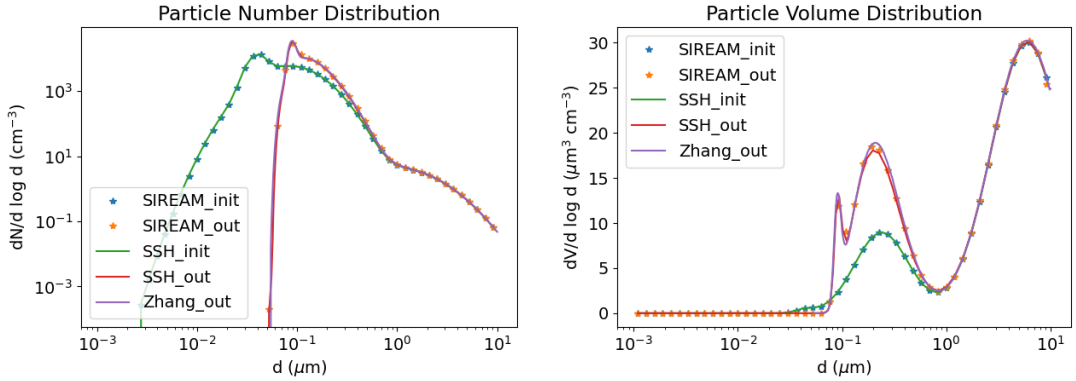


Figure 3: Condensation test case without redistribution. Number (left panel) and volume (right panel) concentrations.

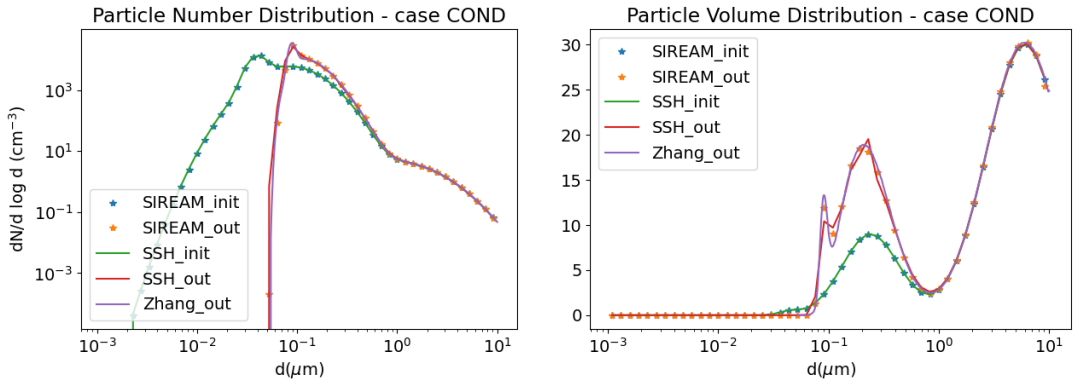


Figure 4: Condensation test case with Euler-coupled redistribution. Number (left panel) and volume (right panel) concentrations.

### 8.1.3 Condensation of low-volatility organics

Extremely-low volatility organic compounds (ELVOCs) are formed from the ozonolysis of monoterpenes (Chrit et al., 2017). Similarly to sulfate, these compounds have a very low saturation vapour pressure and they therefore should condense with a kinetic similar to sulfate if they had the same density. The test case on the sulfate condensation is redone with the ELVOC Monomer, by artificially setting its density to that of sulfate. The configuration file for this test is *namelist_cond_monomer.ssh*. The monomer density is modified in the file *species-list/species-list-aer-monomer.dat*. In the input files for initial conditions *init_gas.dat* and *init_aero.dat* of the

directory *inputs/inputs-cond-monomer/*, initial concentrations are assigned to monomer rather than sulfate. You can plot the number and volume distribution by using the python script *graph/dN_Vdlogd_cond_monomer.py*. The distributions are similar to those of the sulfate case.

### 8.1.4 Condensation/evaporation of inorganics

For inorganic compounds, differences between the particle compositions computed using equilibrium and dynamical sectional models have been stressed by numerous authors such as Sartelet et al. (2006). This test case simulates the case of the highly polluted day of 25 June 2001 of Sartelet et al. (2006). Measurements of PM and gaseous species made in Tokyo (Japan) are taken as initial conditions. Because the data were averaged continuously for 24 hours under varying meteorological conditions, this study can not assess the importance of the equilibrium approach compared to the dynamical approach. The model results obtained after thermodynamic equilibrium is reached are then compared.

The configuration file for this test is *namelist_cond-evap-inorg.ssh* for the case where condensation/evaporation is dynamic. Only condensation/evaporation is considered by setting the variable of *namelist_cond-evap-inorg.ssh with_cond* to 1. The variable *Cut_dim* corresponds to the diameter until which thermodynamic equilibrium is assumed. It is set to 0 to solve the condensation/evaporation with a dynamic approach.

To compare this simulation to a simulation where thermodynamic equilibrium is assumed for condensation/evaporation, please set *Cut_dim* to 40 (the maximum diameter of the particles considered here). This is done in the configuration file *namelist_cond-evap-inorg-eq.ssh*.

A hybrid method may also be used to compute condensation/evaporation: thermodynamic equilibrium is then computed only for size sections of mean diameter lower than a cutoff diameter, while condensation/evaporation is computed dynamically for size sections of diameter higher than the cutoff diameter. For example, a cutoff diameter *Cut_dim* of 0.1 $\mu$m is set in the configuration file *namelist_cond-evap-inorg-icut.ssh*.

Condensation/evaporation can also be simulated by using SOAP instead of Isorropia for inorganic thermodynamics by setting the variable *soap_inorg* to 1. For example, condensation/evaporation with this thermodynamic model can be simulated with *namelist_cond-evap-soapinorg-eq.ssh* at equilibrium and with *INIT/namelist_cond-evap-soapinorg_imethod.ssh* for a dynamic approach.

To assess the differences between these simulations, you can compare the time evolution of $NH_3$ and $HNO_3$, by running the python script *graph/gas_cond-evap.py*. As shown in Fig 5, the gas-phase concentrations quickly reach thermodynamic equilibrium. The concentrations computed with the hybrid approach are close to those computed by the dynamic approach.

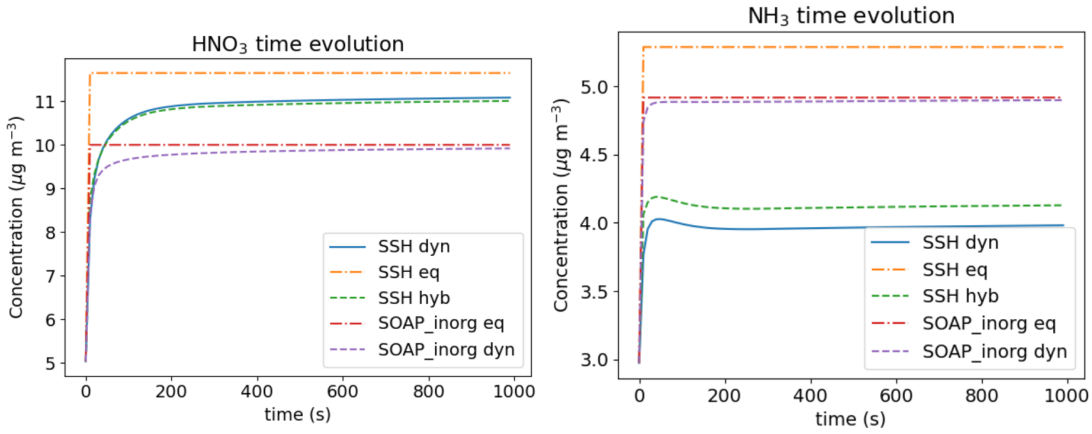


Figure 5: Condensation/évaporation of inorganic test case. Time evolution of $HNO_3$ concentrations (left panel) and $NH_3$ concentrations (right panel).

### 8.1.5 Kelvin effect

To illustrate the importance of the Kelvin effect for the growth of ultrafine particles, the test case of Devilliers et al. (2013) concerning the growth of ultrafine particles emitted from the exhaust of a diesel engine was simulated. As in Devilliers et al. (2013), a typical diesel engine emission initial distribution from Kittelson et al. (2006) is used here to study the gas/particle conversion of nonadecane (C19H40). It has a reference saturation vapour pressure of 6.1 $10^{-4}$ Pa at T = 298 K, which is very close to that of the model compound POAmP. Particles are assumed here to consist solely of POAmP. To show the importance of the Kelvin effect, two simulations are conducted: with and without the Kelvin effect.

The configuration files for this test are *namelist_kelvin.ssh* for the case where the Kelvin effect is modelled, and *namelist_kelvin_nokelv.ssh* for the case where it is not. The variable *ISOAPDYN* is set to 1 to indicate that the condensation/evaporation of organics is modelled dynamically. The variable *with_kelvin_effect* is also set to 1 to take into account the Kelvin effect in the configuration file *namelist_kelvin.ssh* and to 0 in the configuration file *namelist_kelvin_nokelv.ssh*.

To compare the number and volume size distribution simulated with and without Kelvin effect, you can run the python script *graph/dN_Vdlogd_kelvin.py*.

The results in Fig 6 show clearly that the Kelvin effect must be taken into account when the evolution of small particles is simulated: particles are much less affected by condensation/evaporation when it is not included in the model. The ultrafine particles of the initial distribution have been transferred to the gas phase while the coarse ones have grown to a greater size range.

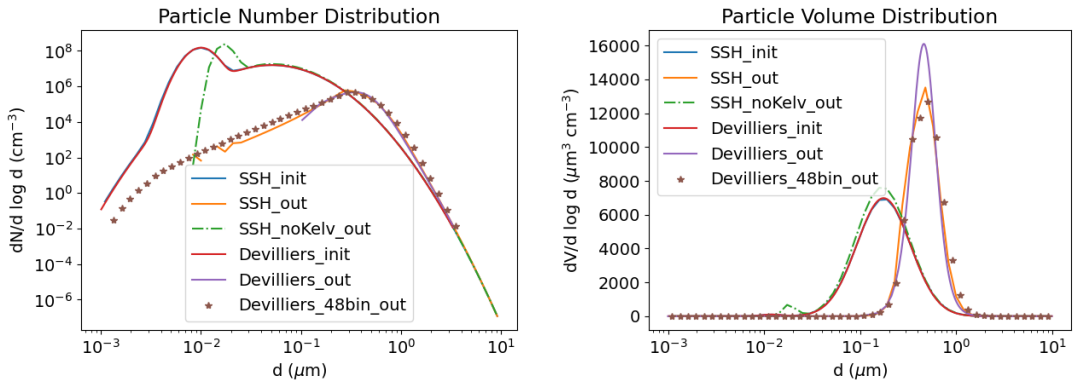


Figure 6: Kelvin test case. Number (left panel) and volume (right panel) concentrations.

### 8.1.6 Condensation and thermodynamic assumption

Condensation/evaporation may be computed dynamically. For inorganic species, this is done by setting the parameter *ICUT* to 0. *ICUT* corresponds to the diameter (in $\mu$m) above which the dynamics condensation/evaporation of inorganic (and also organics for *soap_inorg*=1 ) is solved dynamically. Thermodynamic equilibrium may be assumed between the gas and particle phases. In that case, *ICUT* should be set to the maximum diameter, e.g. 10$\mu$m. For organic species, the parameter *ISOAPDYN* should be set to 0 to assume thermodynamic equilibrium and to 1 for a dynamical computation of condensation/evaporation.

If thermodynamic equilibrium is assumed, then the bulk equilibrium between the gas and the particle phases is first computed, and then the bulk aerosol concentrations are distributed amongst the different size sections.

The redistribution depends on the Kelvin effect, and for inorganics, it also depends on whether or not the sulfate is neutralized by positive inorganic ions (sodium and ammonium). The amount of ammonium that condenses to neutralise sulfate is determined separately from the amount of ammonium that condenses to neutralise other compounds, such as nitric acid. Then the amount of ammonium that neutralises sulfate is redistributed amongst the size section depending on the sulfate concentration of each size section. The amount of ammonium that condenses with other semi-volatile compounds such as nitric acid is redistributed amonst the size section using a formulation that depends on the condensation rate and the Kelvin effect. The condensation rate depends on the diffusivity of species and the diameters of particles. It is multiplied by a factor equals to (Kelvin coefficient -1) times the inorganic mass in the section.

To illustrate the role of the Kelvin effect versus the sulfate neutralisation, the urban condition test case of Seigneur et al. (1986) is modified by adding $NH_3$ as initial conditions, and by taking into account only condensation/evaporation processes. The initial particle concentration is assumed to be made of sulfate only, and the initial concentration of $NH_3$ is assumed to be 100 $\mu$g m$^{-3}$. The simulation is run for 12 h. The simulation that computes dynamically condensation/evaporation is run with the configuration file *namelist_condkelv_dyn_nh3.ssh*, the one that computes dynamically condensation/evaporation without taking into account the Kelvin effect is run with the configuration file *namelist_condnokelv_dyn_nh3.ssh*. Similarly, simulations assuming thermodynamic equilibrium are run with the configuration files *namelist_condkelv_eq_nh3.ssh* and *namelist_condnokelv_eq_nh3.ssh*. Simulations with *soap_inorg*=1 can also be run with the following configuration files *namelist_condkelv_eq_nh3-inorg.ssh* (with kelvin effect and the equilibrium approach) and *namelist_condkelv_dyn_nh3-soapinorg.ssh* (with kelvin effect and the dynamic ap-

proach) .

To compare the number and volume size distribution simulated with the four numerical algorithms, you can run the Python script *graph/dN_Vdlogd_condkelv_nh3.py*. The four numerical algorithms give similar number concentrations, as shown in Fig 7, because $NH_3$ condenses to neutralise the sulfate independently of the Kelvin effect.

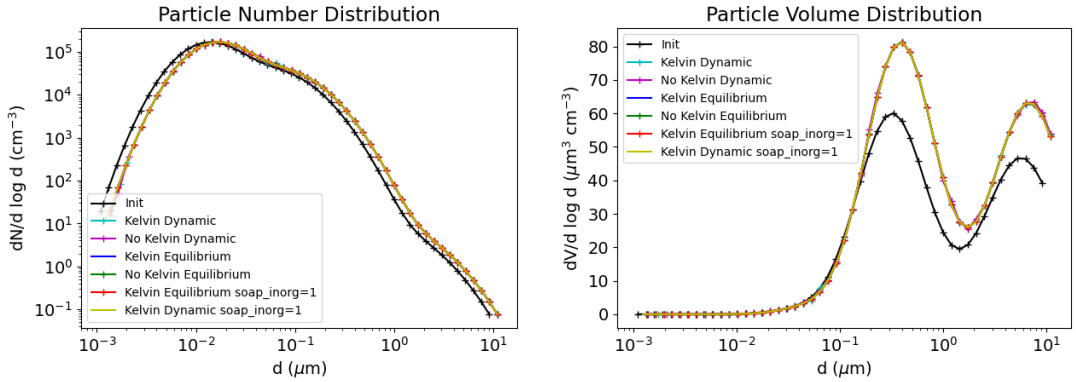


Figure 7: Condensation and thermodynamic equilibrium test case for the condensation of ammonia neutralising sulfate. Number (left panel) and volume (right panel) concentrations.

In the previous test cases above, about 40 $\mu$g m$^{-3}$ of ammonia is used to neutralise the sulfate and the remaining stays in the gas phase. To compare the modelling of the Kelvin effect with dynamical calculation and thermodynamic assumption, nitric acid is added in the initial gas-phase concentration. Nitric acid condenses with the remaining ammonia to form ammonium nitrate, as shown in Fig 8, which was obtained by running the configuration files *namelist_condnokelv_eq_inorg.ssh*, *namelist_condkelv_eq_inorg.ssh*, *namelist_condnokelv_dyn_inorg.ssh*, *namelist_condkelv_dyn_inorg.ssh*, *namelist_condkelv_eq_inorg-soapinorg.ssh*, *namelist_condkelv_dyn_inorg-soapinorg.ssh* and the python script *graph/dN_Vdlogd_condkelv_inorg.py* for the visualisation of the results. As seen in the comparison of the number size distribution, the particles of diameters below about 0.01 $\mu$m do not grow by the condensation of ammonium nitrate if the Kelvin effect is taken into account, while these particles grow to larger diameters by condensation of ammonium nitrate if the Kelvin effect is not taken into account. The algorithm used to redistribute ammonium nitrate amongst size section is effective at limiting this condensation on particles of small diameters.

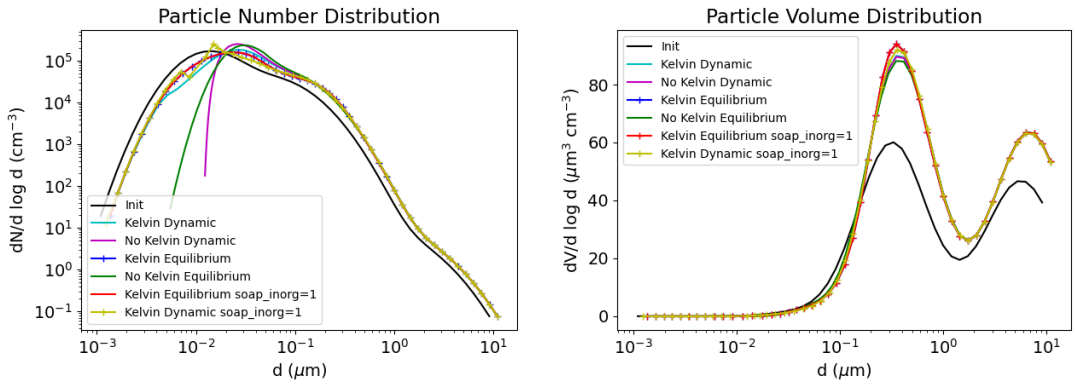


Figure 8: Condensation and thermodynamic equilibrium test case for the condensation of ammonia neutralising sulfate, and for the condensation of ammonium nitrate. Number (left panel) and volume (right panel) concentrations.

This test case if modified to illustrate the influence of the Kelvin effect on the condensation of organics. As for inorganics, the redistribution amongst size sections is done proportionally to the condensation rate multiplied by a factor equals to (Kelvin coefficient -1) times the organic mass in the section. In the urban condition test case of Seigneur et al. (1986), particles are now assumed to be made of non-volatile compounds (SOAlP). A semi-volatile compounds is added in the gas phase as initial conditions (SOAmP). The condensation of SOAmP onto the particles is studied with dynamical calculation or thermodynamic assumption and with or without taking into account the Kelvin effect, by running the configuration files *namelist_condnokelv_eq.ssh*, *namelist_condkelv_eq.ssh*, *namelist_condnokelv_dyn.ssh*, *namelist_condkelv_dyn.ssh* and the python script *graph/dN_Vdlogd_condkelvpy* for the visualisation of the results. As shown in Fig 9, the particles of diameters below about 0.03 $\mu$m do not grow by the condensation of organics if the Kelvin effect is taken into account, while these particles grow to larger diameters by condensation of organics if the Kelvin effect is not taken into account. The algorithm used to redistribute organics amongst size section is effective at limiting this condensation on particles of small diameters.

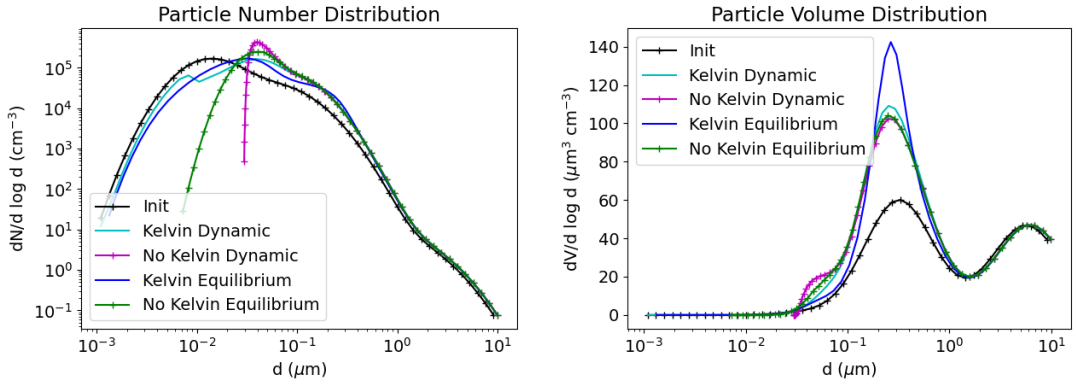


Figure 9: Condensation and thermodynamic equilibrium test case for the condensation of organics. Number (left panel) and volume (right panel) concentrations.

### 8.1.7 Ternary nucleation and influence of ELVOC on particle growth

To assess the ability of SSH-aerosol to deal with simultaneous strong coagulation and condensation/nucleation, the nucleation test case presented in Sartelet et al. (2006) is simulated. The initial distribution is the same as in the condensation of sulphuric acid test case of section 8.1.2. It corresponds to the hazy conditions of Seigneur et al. (1986). The sulphuric acid production rate is 0.825 $\mu$g m$^{-3}$ h$^{-1}$, the temperature is 288.15 K and the relative humidity is 60%. The particles are initially assumed to be made of 70% sulfate and 30% ammonium. The initial gas phase ammonia concentration is taken to be 8 $\mu$g m$^{-3}$. The concentrations of gas-phase ammonia and particulate-phase ammonium evolve with time due to both nucleation and condensation/evaporation The ternary nucleation of Napari et al. (2002) is used, but to avoid artificially large nucleation rates in the parameterization of Napari et al. (2002), a maximum nucleation rate of 1.d6 #particles cm$^{-3}$ is set. The simulation is run for 1 h with output every 60 s.

Two simulations are run: one where the processes (coagulation, condensation/evaporation and nucleation) are solved simultaneously (configuration file *namelist_nucl.ssh*), and one where the numerical resolution of coagulation is split from condensation/evaporation and nucleation (configuration file *namelist_nucl_split.ssh*).

To compare the number and volume size distribution simulated with the two numerical algorithms, you can run the Python script *graph/dN_Vdlogd_nucl.py*.

The two numerical algorithms give similar number concentrations, as shown in Fig 10.

The nucleated particles clearly grow to larger particles with time, as can be seen both in Fig 12 and by running the python script *graph/banana.py*.

Ultra-fine particles grow under the combined effect of coagulation, condensation of sulphate and nucleation. As shown in Figure 12, the nucleated particles clearly grow to larger particles with time. This growth can be accelerated by the presence of low-volatility compounds, as shown in Patoulias et al. (2018), underlying the importance of an accurate modelling of the formation of ELVOCs for the growth of ultra-fine particles. The simulation of Figures 10 and 12 is redone by adding 1 $\mu$g m$^{-3}$ of a SSH-aerosol surrogate, Monomer, which is an ELVOC formed from the oxidation of terpenes (configuration file *namelist_nucl_org.ssh*). To compare the number and volume size distribution simulated with/without the surrogate Monomer, you can run the python script *graph/dN_Vdlogd_nucl_org.py*. As shown in Figure 11, the comparison of the number concentration simulated with and without taking the surrogate Monomer into account shows that the particles have grown to larger diameters because of the presence of Monomer. The left panel of Figure 12 shows the time evolution of the number concentration for the different size sections without Monomer (script *graph/banana.py*) and the right panel of Figure 12 shows the same time evolution but with 1 $\mu$g m$^{-3}$ of the surrogate Monomer condensing (script *graph/banana_org.py*). The surrogate Monomer first condenses to existing particles around 0.01 $\mu$m, making them grow to larger diameters. Note that the condensation of the extremely low-volatility surrogate Monomer needs to be solved simultaneously to coagulation and nucleation to correctly represent this growth of ultra-fine particles (the parameter non_volatile needs to be set to 1 for the surrogate Monomer in the input file *species-list/species-list-aer.dat*).

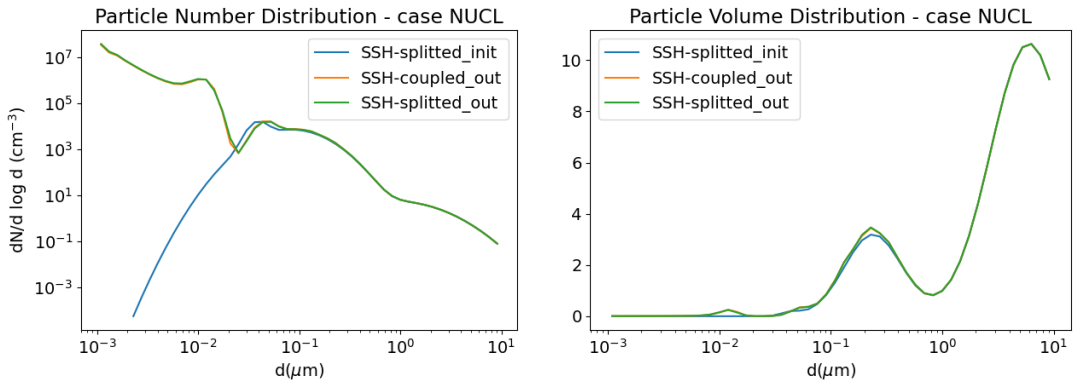


Figure 10: Nucleation test case. Number (left panel) and volume (right panel) concentrations.

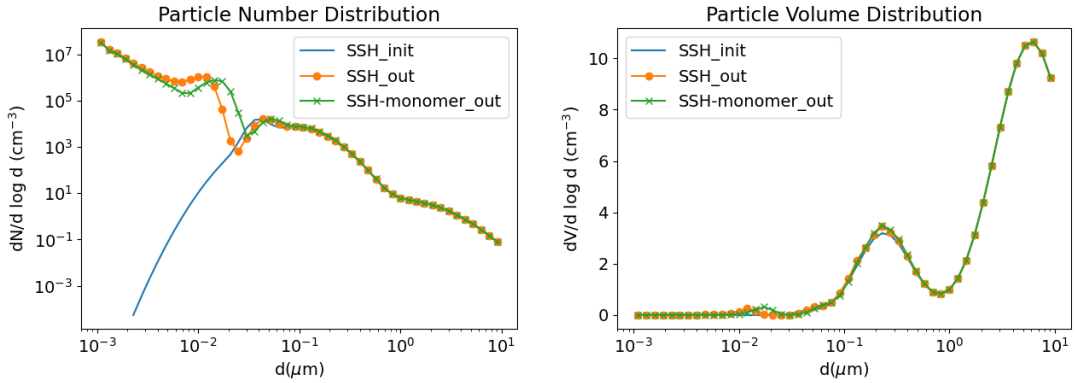


Figure 11: Nucleation test case with Monomer surrogate added. Number (left panel) and volume (right panel) concentrations.

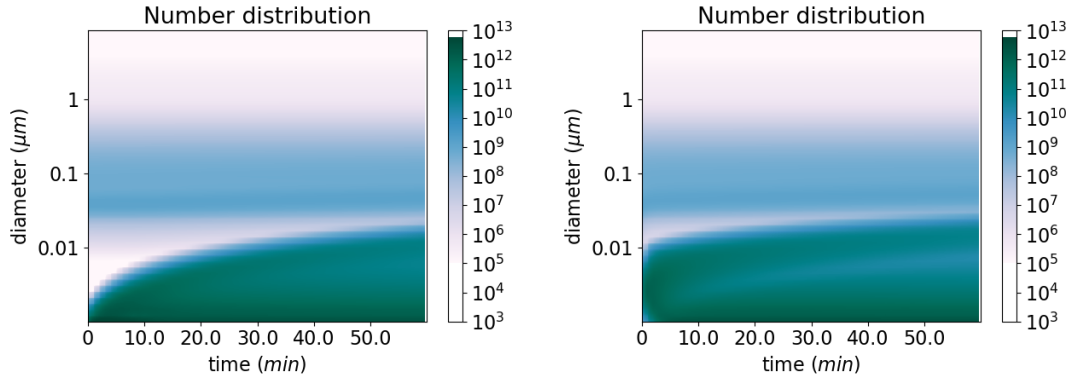


Figure 12: Nucleation test case. Time evolution of the number concentrations. Without organics (left panel), with organics (surrogate Monomer, right panel).

### 8.1.8 Heteromolecular nucleation and influence of the number of size sections on particle growth

Heteromolecular nucleation involves the nucleation of sulfuric acid and other organic compounds of extremely low volatility, for example, Monomer and Dimer formed from the autoxidation of monoterpenes Riccobono et al. (2014); Sartelet et al. (2022). To illustrate the influence of the heteromolecular nucleation, the initial distribution is the same as in the condensation of sulphuric acid test case of section 8.1.2. It corresponds to the hazy conditions of Seigneur et al. (1986). Particles are assumed to be made of sulphate, and an emission rate of 0.00001 $\mu$g m$^{-3}$ s$^{-1}$ is assumed for sulphuric acid (SULF), Monomer and Dimer. The simulations are run with either 12, 25, or 50 sections for 12 h. The configuration files are *namelist_nucl_elvoc_d12.ssh*, *namelist_nucl_elvoc_d25.ssh*, *namelist_nucl_elvoc_d50.ssh*. To compare the number and volume size distribution simulated with the different numbers of size sections, you can run the python scripts *graph/dN_Vdlogd_nucl-elvoc-d12.py*, *graph/dN_Vdlogd_nucl-elvoc-d25.py*, *graph/dN_Vdlogd_nucl-elvoc-d50.py*.

As shown in Fig. 13, the nucleated particles grow by condensation and coagulation. The depletion and growth of nucleated nanometric particles are well represented with 12 sections, although 25 sections are needed to better differentiate nucleated particles that have grown from those in the accumulation mode.

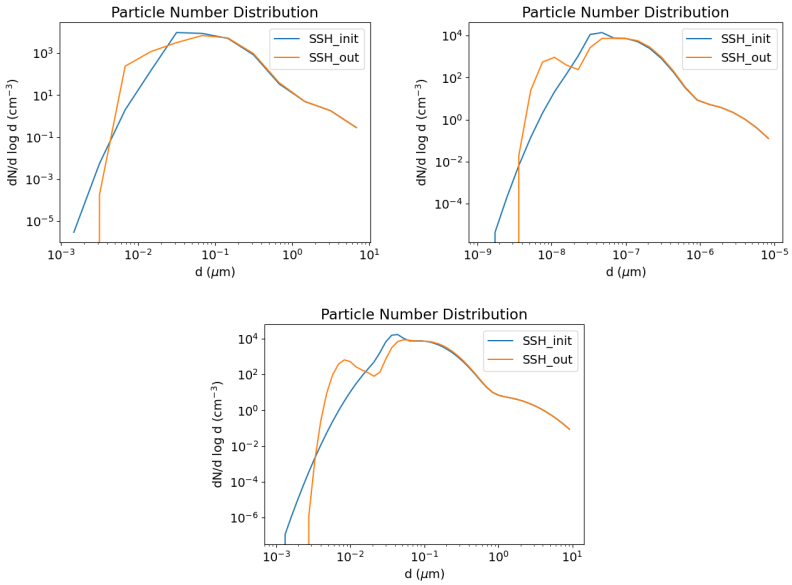


Figure 13: Heteromolecular nucleation test case. Number concentrations simulated with 12 sections (left panel), 25 sections (middle panel), and 50 sections (right panel).

### 8.1.9 Organic nucleation

Organic nucleation involves the nucleation of organic compounds of extremely low volatility, for example, PMonomer and PDimer formed from the autoxidation of monoterpenes Riccobono et al. (2014); Sartelet et al. (2022) in the H$^2$O mechanism. A constant $k$ to scale the nucleation rate $J$ and a power low constant $\alpha$ needs to be specified as input:

$$J = k\ [Organics]^{\alpha} \tag{1}$$

To illustrate the potential influence of the organic nucleation, as in the heteromolecular test case the initial distribution is the same as in the condensation of sulphuric acid test case of section 8.1.2. It corresponds to the hazy conditions of Seigneur et al. (1986). Particles are assumed to be made of sulphate, and an emission rate of 0.00001 $\mu$g m$^{-3}$ s$^{-1}$ is assumed for sulphuric acid (SULF), Monomer and Dimer. The simulations are run with 25 sections for 12 h. The configuration file is *namelist_nucl_org_only_d25.ssh*. To vizualize the simulation results, you can run the python script *graph/dN_Vdlogd_nucl-org_only-d25.py*. As shown in Fig. 14, with the input parameters arbitrarily chosen, the nucleated particles grow by condensation and coagulation.

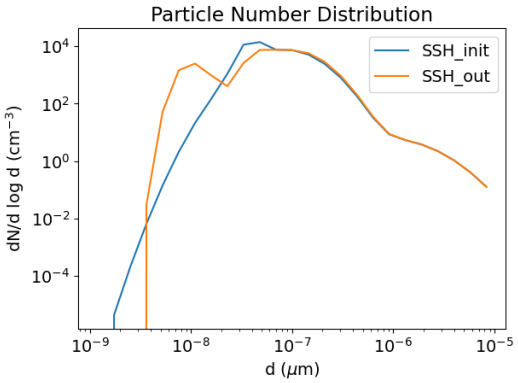


Figure 14: Organic nucleation test case. Number concentrations simulated with 25 sections.

## 8.2 Modelling of SOA formation

After emission into the atmosphere, the oxidation of volatile organic compounds (VOCs) leads to less volatile compounds than the precursors. These compounds condense more easily onto particles than their precursors and they contribute to the increase of particle mass. This section discusses the gas-to-particle partitioning of organic condensables under various absorption assumptions. The conditions used in the test cases of Sections 8.2.1 to 8.2.3are derived from an example of exhaust emissions aging, while the condition used in the test case of Section 8.2.4 represents a near-realistic condition (Sartelet et al. (2022)), as detailed in Section 8.6.2.

### 8.2.1 Formation of condensables

Platt et al. (2013) performed ageing experiments on emissions from an Euro 5 gasoline car. They monitored the total hydrocarbon mass (THC) as well as organic aerosol (OA) concentrations. The test case presented here corresponds to the simulation performed in Sartelet et al. (2018).

In our model, THC is assumed to be the sum of VOC and I/S VOCs (intermediate and semi-volatile VOCs). THC is initialised as measured in the experiments of Platt et al. (2013) before lights-on (1.4 ppmv + 0.9 ppmv of propene). IVOCs are estimated from VOC emissions using the ratio 0.17 estimated by Zhao et al. (2016), and NOx concentrations are initialised such as having a VOC/NOx ratio equal to 5.6 as in Platt et al. (2013). The speciation of VOCs to model species was done following Theloke and Friedrich (2007).

The configuration file for this test is *namelist_platt.ssh*.

Not only condensation/evaporation is considered by setting the variable of *namelist_platt.ssh with_cond* to 1, but also gaseous chemistry ( *tag_chem*=1). Predefined photolysis rates may be used, or tabulated values may be downloaded.

After running the model for 5 h, the evolution of the aerosol concentrations with time may be displayed by running the script *plot_platt_particles.py*. As in the experiment after correction for wall loss, the concentrations of particles is about 200 $\mu g\ m^{-3}$. More than half of the mass originates from the condensation/evaporation of aged I/S VOCs, as shown in Fig 15. The inorganic concentrations stay very low, as only 0.1 $\mu g\ m^{-3}$) of sulfate and ammonium were introduced.

Ammoniac emission was not considered in the previous test case. However, to reduce NOx emissions, ammoniac may also be emitted by recent vehicles Suarez-Bertoa et al. (2017). To illustrate the potential influence of $NH_3$ emission on secondary aerosol formation, the initial conditions of the test case above are modified to include $NH_3$ emissions, assumed to be 10% of NOx emissions Suarez-Bertoa et al. (2017). The configuration file for this test is *namelist_platt_nh3.ssh*. After running the model, the evolution of the aerosol concentrations with time may be displayed by running the script *plot_platt_particles_nh3.py*. As shown in Figure 15, $NH_3$ emission leads to a large increase in inorganic aerosol concentrations (from 0.1 $\mu$g $m^{-3}$ to about 37 $\mu$g $m^{-3}$).

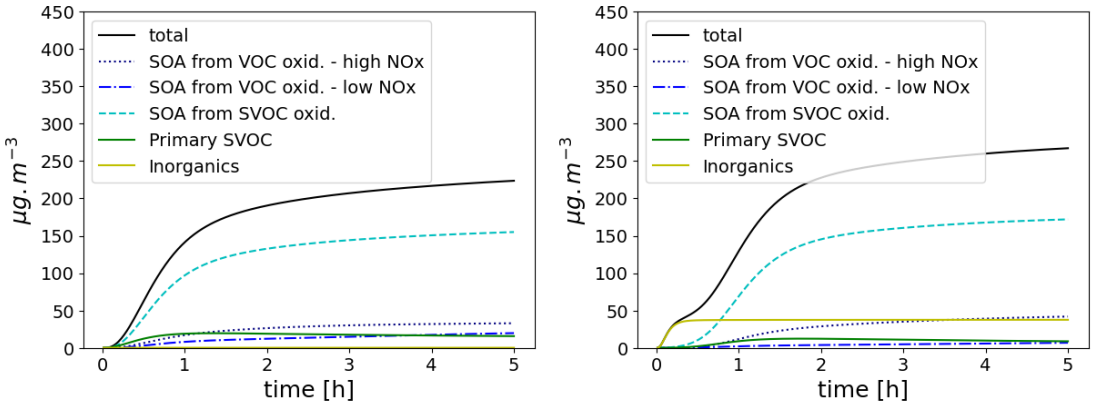


Figure 15: Time evolution of organic and inorganic concentrations for the Platt test case: without $NH_3$ emissions (left panel), with $NH_3$ emissions (right panel).

### 8.2.2 Absorption into an aqueous or an organic phase

The SOA compounds formed from the exhaust are assumed to be mostly hydrophobic, and therefore condense onto an organic phase. SOA compounds formed from biogenic precursors have a stronger affinity with water Couvidat et al. (2012); Couvidat and Seigneur (2011), and they may prefer condensing onto an aqueous phase, formed by hydrophilic organic compounds and water absorbed by inorganics and hydrophilic organic compounds Kim et al. (2019). To illustrate the influence of hygroscopicity, the exhaust test case of the previous section is simulated by adding 150 $\mu$g $m^{-3}$ of isoprene. In the case when $NH_3$ is not emitted in the exhaust, biogenic condensables are formed but mostly stay in the gas phase, because they are mostly hydrophilic and they can not condense efficiently on the hydrophobic particles formed from the exhaust emission. As shown in Figure 16, if $NH_3$ is added to the exhaust, inorganic aerosol concentrations largely increase, leading to an increase in isoprene SOA, which can then condense onto aqueous particles. Note that hygroscopicity is not computed for ideality to avoid artificially high concentrations of water in the organic phase (flag *hygroscopicity* is set to false in the file *SOAP/parameters.cxx*).

In the previous test case, the influence of activity coefficients on the SOA formation is

not taken into account. Depending on the user's choice, short-range or short, medium, and long-range activity coefficients can be modelled. If only short-range activity coefficients (interactions between uncharged molecules) are modelled, then they are computed using the UNIQUAC Functional-group Activity Coefficients (UNIFAC) thermodynamic model Fredenslund et al. (1975), which is based on a functional group method. If short, medium, and long-range activity coefficients are modelled, i.e. taking into account the influence of inorganic ions in the SOA aqueous phase, they are computed using the Aerosol Inorganic-Organic Mixtures Functional groups Activity Coefficients (AIOMFAC) model Zuend et al. (2008).

The configuration file for the test case of exhaust emissions with added isoprene emission is *namelist_platt_nh3_isop.ssh*. Two other simulations may also be run using UNIFAC to compute activity coefficients (*namelist_platt_nh3_isopu.ssh*) or AIOMFAC (*namelist_platt_nh3_isopa.ssh*).

The evolutions of the concentrations of isoprene SOA in particles as a function of time for these simulations may be seen in Figure 16, which can be plotted running the script *platt_isop.py*. As already shown in the 3D studies of Couvidat et al. (2012); Kim et al. (2019), and as shown here in Figure 16 for isoprene SOA, ideality strongly influences the partitioning between the gas and particle phase.

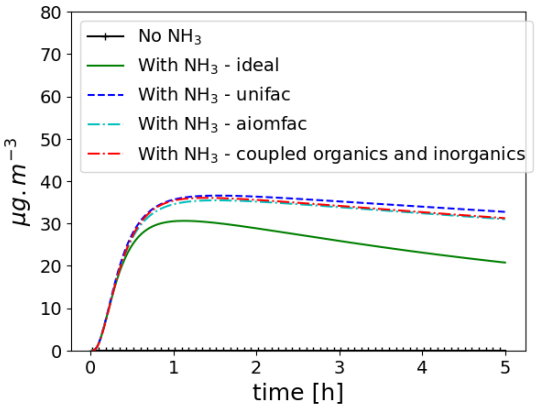


Figure 16: Time evolution of isoprene SOA in the particle phase, with concentrations approaching zero in the simulation without $NH_3$.

### 8.2.3 Absorption into both an aqueous and an organic phase

In the test case above with isoprene and $NH_3$ added to exhaust emissions, compounds were assumed to be either hydrophobic, or hydrophilic, following the properties described in the species list (file *species-list/species-list-aer.dat*), and in the namelist file, condensation on aqueous and organic phases was not coupled (variable *coupled_phases=0*. Here, the compounds are allowed to be both hydrophilic and hydrophobic. In the namelist *INIT/namelist_platt_nh3_isop_cpeq.ssh*, the variable *coupled_phases* is set to 1 and in the species list file *species-list/species-list-aer-cp.dat*, the partitioning column is set to *BOTH* for organic compounds.

The simulation can be run by typing *ssh-aerosol INIT/namelist_platt_nh3_isop_cpeq.ssh*. In this simulation, as well as in those of the previous section, thermodynamic equilibrium is assumed between the gas and particle phases. To evaluate the impact of this assumption, a simulation is run by letting the gas/particle partitioning evolve dynamically (*ISOAPDYN=1* in the namelist); this is run by typing *ssh-aerosol INIT/namelist_platt_nh3_isop_cpdyn.ssh*. The evolution of organic concentrations can be compared by going to the repertory *graph* and by running the python script *platt_isop_cp.py* (see Fig 17). The SOA concentrations are higher

when compounds are allowed to condense on both an organic and an aqueous phase (case coupled phase). If gas/particle partitioning evolves dynamically, the SOA concentrations are at first higher than when thermodynamic equilibrium is assumed, but then they get close to the equilibrium concentrations as time passes.

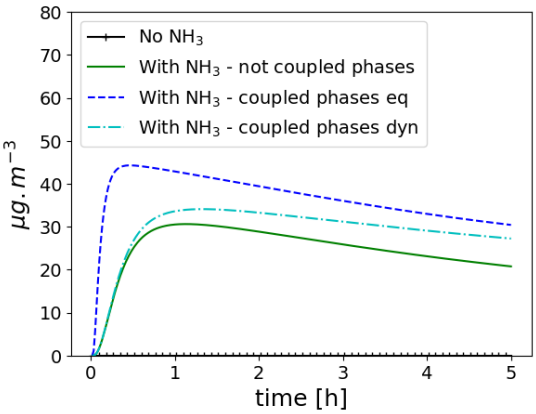


Figure 17: Time evolution of isoprene SOA in the particle phase by taking into account or not the fact that compounds may be both hydrophilic and hydrophobic.

### 8.2.4 Impact of absorption and activity modes on SOA formation

As discussed above, in SSH-aerosol, gas-phase condensable compounds can undergo gas-to-particle partitioning into either or both aqueous and organic particle phases, where thermodynamic interactions between organics and organic-inorganic compounds in the particle phase can be estimated by activity models. In the aerosol species list, each condensable compound can be flagged as "HPHO" to consider only hydrophobic partitioning, "HPHI" to consider only hydrophilic partitioning, and "BOTH" to account for partitioning into both phases. In the namelist, the activity model flag (*activity_model*) can be set to 1 for ideal conditions, 2 for the UNIFAC model, and 3 for both the UNIFAC and AIOMFAC models to calculate activity coefficients under non-ideal conditions.

A test case is presented to investigate the impact of different absorption and activity modes on SOA formation using SSH-aerosol. In this case, condensable gas-phase compounds formed using the H2O chemical mechanism are simulated to condense under either the default aerosol species list (*species-list/species-list-aer.dat*) or a modified list, with partitioning modes for most condensables set uniformly to HPHI (*species-list/species-list-aer-HPHI.dat*), HPHO (*species-list/species-list-aer-HPHO.dat*), or BOTH (*species-list/species-list-aer-BOTH.dat*), combined with each activity model setting (i.e., 1, 2, or 3). By combining four aerosol species lists with three activity models, this test case includes a total of 12 simulations.

In practice, these simulations are carried out using one basic namelist (*INIT/namelist_prt4act3.ssh*) and a shell script (*run_prt4act3.sh*) that modifies the namelist to apply different aerosol species lists, activity models, and output directories as required. Additional settings and initial gas-phase concentrations are derived from Sartelet et al. (2022), with relative humidity adjusted to 80%. High concentrations of gas-phase ammonia and nitrate (each at 20 $\mu g/m^3$) and particle-phase sulfate and ammonium (each at 5 $\mu g/m^3$) are included to ensure a sufficient aqueous phase for organics to condense on (See initial concentrations in inputs/inputs-prt4act3).

The time variations in SOA concentrations for this test case are plotted using the Python script *graph/plot_prt4act3.py*, with results organized by activity model into three panels, as

shown in Fig. 18. SOA concentrations vary significantly across different activity models, especially when calculating activity coefficients using both the UNIFAC and AIOMFAC models. In this case, the presence of inorganic ions is shown to inhibit the condensation of organics. The results also clearly demonstrate the importance of accounting for organic condensation into both particle phases. The SOA condensation process is complex and nonlinear, as the total SOA concentration resulting from partitioning into a single phase (either organic or aqueous) does not equal the SOA concentrations observed when condensation occurs in both phases simultaneously.

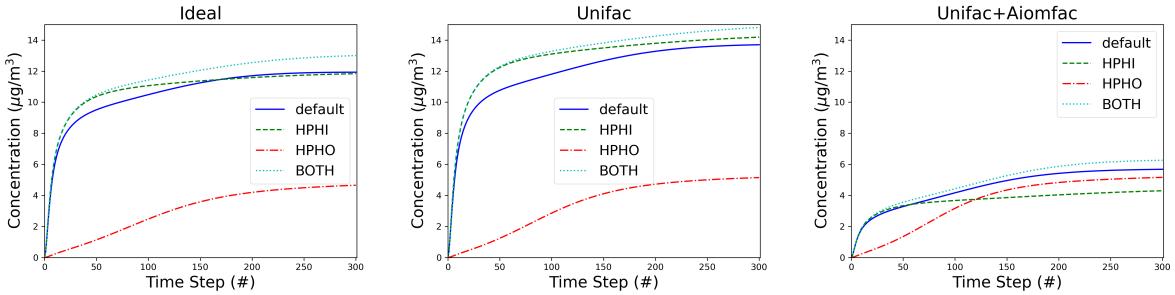


Figure 18: Time evolution of total SOA concentrations simulated with different absorption and activity modes.

## 8.3 Mixing state

The previous test cases relied on the internal mixing assumption (one aerosol composition per aerosol size section). The internal mixing assumption relies on the assumption that particles from different sources mix instantaneously when they are present in the same air mass. Although this assumption may be realistic far from emission sources, it may be difficult to justify close to emission sources, where emitted particles can have compositions that are very different from background particles and from particles emitted from different sources. All aerosol dynamic processes may affect the mixing state of particles.

### 8.3.1 Coagulation

To illustrate how coagulation affects the mixing state of particles, the coagulation test case (see section 8.1.1) is revisited by assuming that the initial aerosol distribution is made of two low-volatility compounds of the same density (sulfate and another low-volatility compound). The configuration file for this test is *namelist_coag_ext.ssh*. In this coagulation test case, the composition is discretized using 4 sections by setting the variable *N_frac* to 4. The bound values of the fraction sections are specified using the variable *frac_input*.

Run the simulation by typing *ssh-aerosol INIT/namelist_coag_ext.ssh*. By summing the mass and number of particles of all size fraction sections, the size number and volume distribution are identical to those obtained with the internal mixing assumptions. This can be checked by going to the repertory *graph* and by running the python script *dN_Vdlogd_coag_ext.py* (see Fig 19). The python script *coag_2D.py* allows to create figures to visualize the initial and final mass concentrations as a function of size and fraction of sulfate. Hence, the evolution of the mixing-state of particles after 12 h of coagulation may be seen by comparing the two panels in Fig 20, which shows the number concentrations as a function of the size and fraction of sulfate, which is one of the two compounds of the particles.

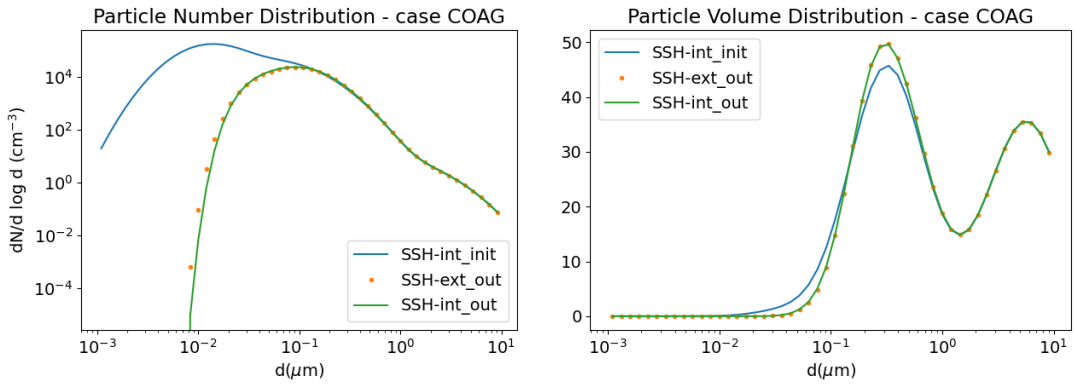


Figure 19: Coagulation test case with mixing-state modelling. Number (left panel) and volume (right panel) concentrations.

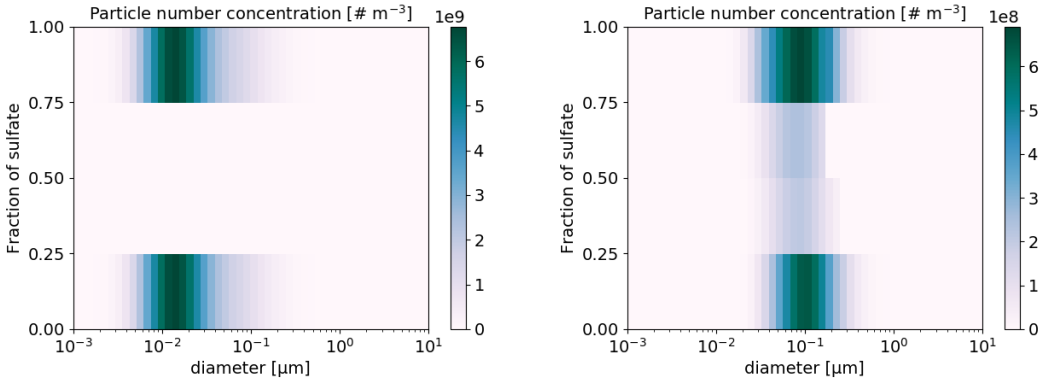


Figure 20: Number concentrations as a function of the size and fraction of one of the two compounds at the initial time (left panel) and after 12 h of simulation (right panel).

### 8.3.2 Condensation

The effect of condensation on the mixing state is assessed by revisiting the condensation test case (typical of a regional haze scenario, see section 8.1.2) by assuming that the initial aerosol distribution is made of two low-volatility compounds of the same density (sulfate and another low-volatility compound). As in Zhu et al. (2015), 10 composition fractions are used. The configuration file for this test is *namelist_cond_ext.ssh*. Run the simulation by typing *ssh-aerosol INIT/namelist_cond_ext.ssh*. By summing the mass and number of particles of all size fraction sections, the size number and volume distribution are identical to those obtained with the internal mixing assumptions. This can be checked by going to the repertory *graph* and by running the python script *dN_Vdlogd_cond_ext.py* (see Fig 21). The python script *cond_2D.py* allows to create figures to visualize the initial and final mass concentrations as a function of size and fraction of sulfate. Hence, the evolution of the mixing-state of particles after 12 h of condensation and coagulation may be seen by comparing the two panels in Fig 22, which shows the mass concentrations as a function of the size and fraction of sulfate, which is one of the two compounds of the particles. Sulphuric acid condenses to form sulfate. Because the condensation

rate is greater for particles of low diameters, the sulfate fraction is greater for those particles at the end of the simulation (Figure 22).

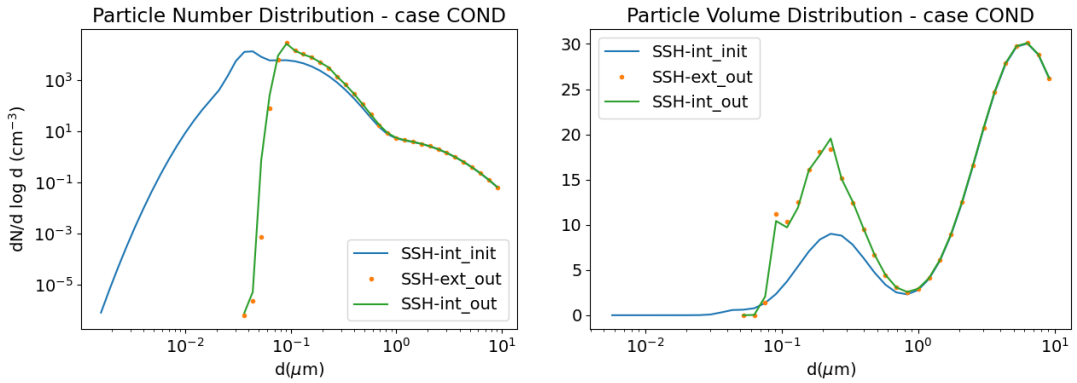


Figure 21: Regional haze test case with mixing-state modelling. Number (left panel) and volume (right panel) concentrations.

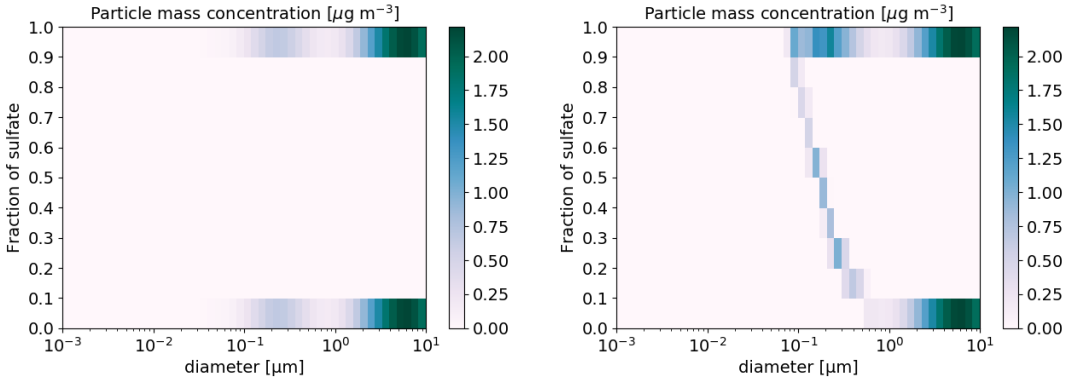


Figure 22: Mass concentrations as a function of the size and fraction of one of the two compounds at the initial time (left panel) and after 12 h of simulation (right panel).

## 8.4 Gas/particle partitioning and viscosity

The effect of viscosity on SOA formation can be achieved in two different ways. In the first method, the diffusion coefficient of organic compounds in the organic phase is fixed. The user selects the coefficient by putting the intended value in the parameter dorg in the section &physic_condensation of the namelist. In the second method, the viscosity is computed as a function of aerosol composition. This method is activated by fxing dorg to 0.

### 8.4.1 Fixed viscosity

The gas/particle mass transfer is strongly affected by particle viscosity. To illustrate this effect, the condensation of hydrophobic organic surrogates of different volatility is studied for different values of the viscosity, represented by different values of the organic-phase diffusion coefficients D ($10^{-18}$, $10^{-19}$, $10^{-20}$, $10^{-21}$, $10^{-22}$, $10^{-23}$, $10^{-24}$ m$^2$ s$^{-1}$). The value D = $10^{-12}$ m$^2$ s$^{-1}$

corresponds to an inviscid particle, while the value D = $10^{-24}$ m$^2$ s$^{-1}$ corresponds to a very viscous particle. Each particle is discretized into 5 layers to model the flux of diffusion of the organic compounds in the particle. The test cases presented here are similar to those of Figure 6 of Couvidat and Sartelet (2015). Three surrogates are successively studied: SOAlP, POAlP, and POAmP, which have partitioning coefficients $K_p$ equal to 100 m$^3$ $\mu$g$^{-1}$, 1 m$^3$ $\mu$g$^{-1}$, and 0.01 m$^3$ $\mu$g$^{-1}$. The surrogate that condenses is initially only in the gas phase, with a mass of 5 $\mu$g m$^{-3}$. It condenses on an organic phase of 5 $\mu$g m$^{-3}$ made of SOAlP, which has very low volatility ($K_p$ = 100 m$^3$ $\mu$g$^{-1}$). The test cases may be launched by launching the script *INIT/launch_visco.sh*. After computation of the condensation of the surrogates from different values of the viscosity, three graphs can be displayed by running the scripts *conc_visco_poalp.py*, *conc_visco_poamp.py*, *conc_visco_soalp.py* (one graph per surrogate). They are shown in Fig 23, which represents the time variations of particle-phase organic concentrations of the surrogate (POAlP in the upper left panel, POAmP in the upper right panel, and SOAlP in the lower panel). The surrogate of low volatility (SOAlP) condenses quickly onto the organic phase, independently of the viscosity (upper left panel of Fig 23). All the organic mass is then in the particle phase. Although the condensation of POAlP is influenced by the viscosity, a large fraction of the gas-phase concentration of POAlP condenses quickly (upper right panel of Fig 23). For POAmP, which is the most volatile of the three surrogates, high viscosity (low diffusion coefficient) strongly inhibits the condensation of POAmP (lower right panel of Fig 23).

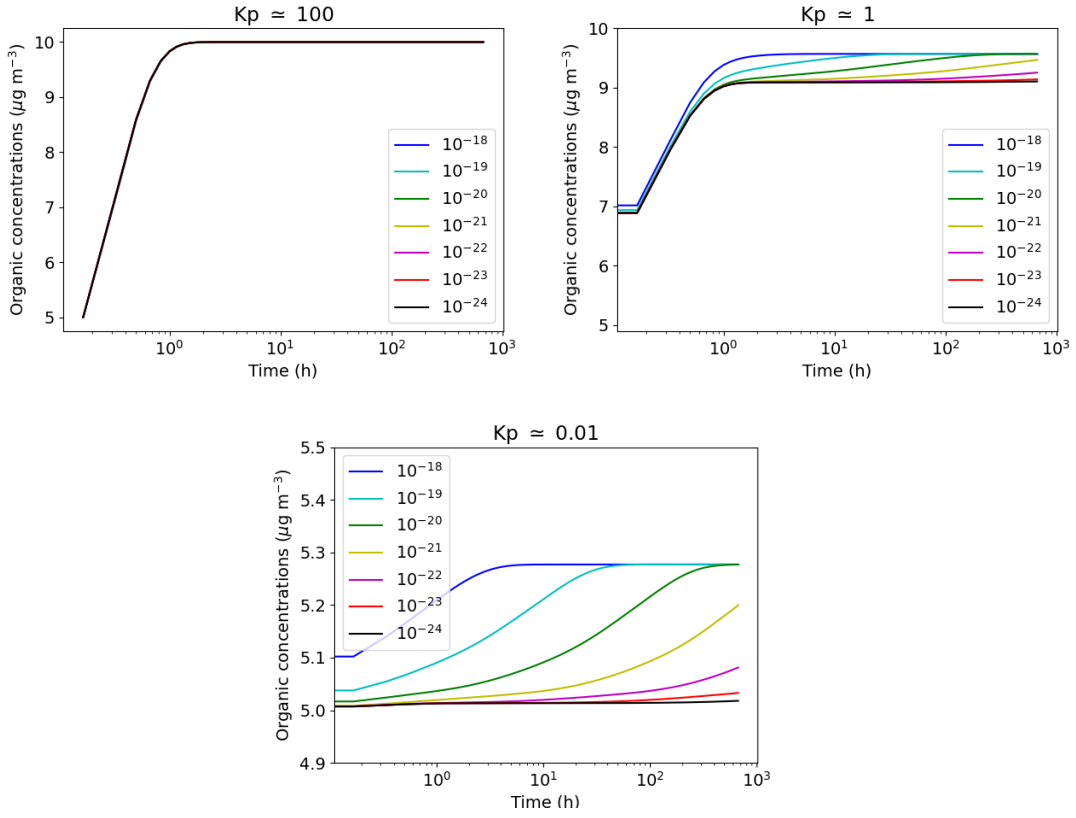


Figure 23: Viscosity test case: time evolution of organic concentrations (SOAlP in the upper left, POAlP in the upper right, POAmP in the lower left) for several organic-phase diffusivity: $10^{-18}$ (dark blue), $10^{-19}$ (light blue), $10^{-20}$ (green), $10^{-21}$ (yellow), $10^{-22}$ (magenta), $10^{-23}$ (red), $10^{-24}$ (black) m$^2$ s$^{-1}$.

### 8.4.2 Viscosity computed as a function of composition

In this method, the viscosity of the organic phase is computed with the AIOMFAC-VISC algorithm Gervasi et al. (2020). In this algorithm, the UNIFAC parameters are used to compute the viscosity. Therefore, the viscosity of a mixture of organic compounds can be computed as long as the decomposition in functional groups of the compounds is given (please refer to section 6.6). Compounds using the default structure (POAlP, POAmP, POAhP, BOAlP, BOAmP, BOAhP, SOAlP, SOAmP, SOAhP, BSOAlP, BSOAmP, BSOAhP) should therefore not be used with this method (except if they constitute a very small fraction of the organic phase. It should be noted that the aqueous-phase is still considered to be inviscid.

Following DeRieux et al. (2018), the AIOMFAC-VISC algorithm uses a fragility parameter $D_{fragility}$ to compute pure compound viscosity $\eta_0$ defined as:

$$log_{10}(\eta_0) = -5.0 + 0.434 \frac{D_{fragility} * T_0}{T - T_0} \tag{2}$$

with $T_0$ the Vogel temperature, or ideal glass transition temperature, computed with the following equation:

$$T_0 = \frac{T_g * 39.17}{D_{fragility} + 39.17} \tag{3}$$

with $T_g$ the glass transition temperature, a characteristic temperature (computed as a function of the number of N, C, and O atoms) at which the mixture changes from a glassy solid state to a semi-solid state.

Although $D_{fragility}$ is set to 10 in Gervasi et al. (2020), a parameterisation of $D_{fragility}$ as a function of the O/C ratio is used, following Schmedding et al. (2020). As the AIOMFAC-visc only computes viscosity, the parameterization of Maclean et al. (2021) is used to compute the diffusion coefficient of the compound $D_{diff_c}$ as the function of viscosity ($\eta$), the viscosity of pure water ($\eta_0$) and the diffusion coefficient in pure water $D_{diff,water}$:

$$D_{diff_c} = D_{diff,water} \times \left( \frac{\eta_0^{water}}{\eta} \right)^{\xi} \tag{4}$$

where $\xi$ is a parameter used to compare the radius of the diffusing molecule ($R_{diff}$) to the average radius of the compound mixture ($R_{mean}$, average of $R_{diff}$ ponderated with the molar fraction):

$$\xi = 1 - A\ exp\left( B \frac{R_{diff,c}}{R_{mean}} \right) \tag{5}$$

with A and B equal to 0.73 and 1.79, respectively Maclean et al. (2021).

$R_{diff}$ is calculated from the density and molar mass of the compound:

$$R_{diff} = \left( \frac{3 M_m}{4 N_a \rho_c * 1000} \right)^{\frac{1}{3}} \tag{6}$$

with $M_c$ the molar mass (in g/mol), $N_a$ the Avogadro number and $\rho_c$ the compound density (in kg/m$^3$).

$R_{mean}$ is calculated as the average value of $R_{diff}$ ponderated by molar fractions.

With this method, the diffusion coefficients are calculated for each species and each layer.

The effect of changing composition on the condensation of a semivolatile compound is illustrated in Fig. 24. The condensation of an organic compound (POAmP) is studied for different particle compositions and hence viscosity. Because the default structure is used to represent

POAmP, POAmP was selected as a condensing compound so that the condensation is not affected by non-ideality (the activity coefficient of compounds using the default structure has an activity coefficient equal to one). A low concentration of POAmP in the gas phase (0.01 $\mu$g/m$^3$) was set in order to not affect the viscosity of the organic phase.

The time evolution of POAmP concentrations was computed for different RH and for different compositions of the aerosol onto which POAmP condenses. The different compositions are tested for different amounts of alcohol groups (OH), for different amounts of carbons, and with different types of functional groups (ketone, hydroxyl, acid, and nitrate). The simulations show that an aerosol composed of compounds with 16 carbons and 4 alcohol groups is very viscous at RH=10% and that condensation of POAmP hardly occurs. However, at RH=70%, the aerosol is inviscid and the condensation of POAmP is non-limited by viscosity. Adding just one OH, group has a strong effect on viscosity at RH=10%. With 16 carbons and 2 OH groups, POAmP needs around one hour to reach equilibrium. However, it needs more than 100 hours to reach equilibrium with an aerosol composed of 16 carbons and 3 OH groups. Adding, acid, hydroperoxide or nitrate groups affects even more significantly. Especially, just by adding one nitrate group, the aerosol switches from inviscid to extremely viscous. By comparison, increasing the number of carbons has only a small effect on viscosity. If the molecules have 2 OH groups, equilibrium is reached in less than 1 hour with less than 16 carbons. 20 carbons are necessary for the equilibrium to be reached in more than 100 hours.

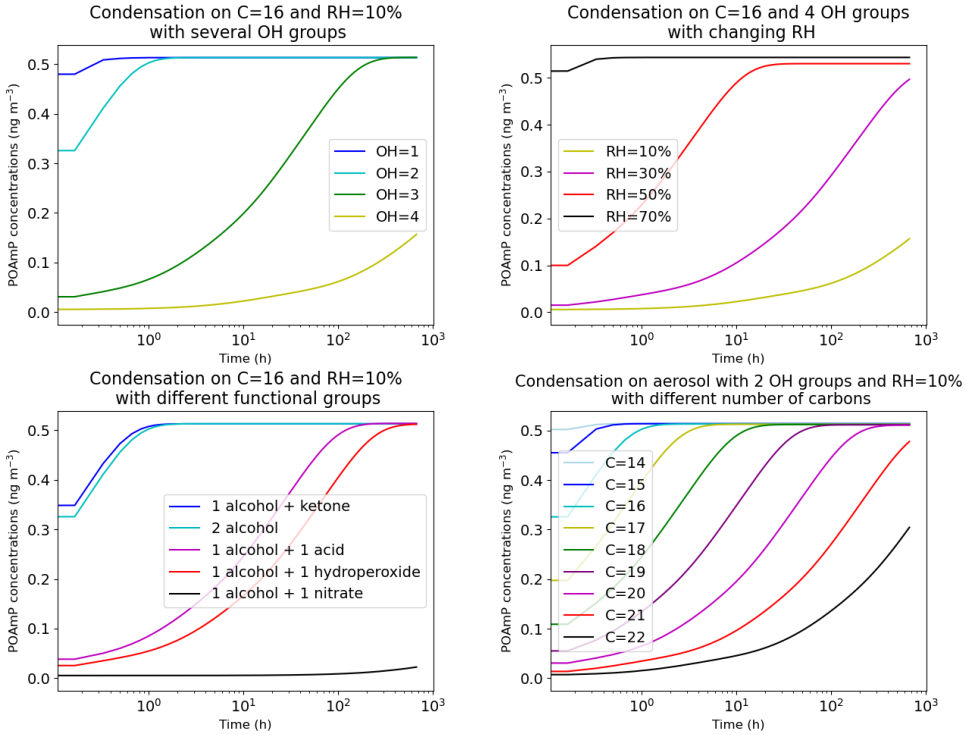


Figure 24: Concentration of POAmP condensing onto 5 $\mu$g/m$^3$ of a compound with 16 carbons $C_{16}$ as a function of time. The different graphs show the effect of changing the composition of the aerosol or humidity.

This test case can be reproduced by launching all *INIT/namelist_aiomfacvisc_oc*.ssh* namelists and by running the python script *conc_aiomfac-visc.py* in the *graph* repertory. Changing the composition was done by initializing the particle concentration of the desired species (OC0 to OC2NO3) defined in *species-list/species-list-aer-aiomfacvisc.dat*.

## 8.5 Particle-phase reactions

Some specific reactions can occur inside the particle. These reactions can be activated or deactivated by modifying the file indicated by *reaction_soap_file* in the section &physic_condensation of the namelist.

In this file, the several lines correspond to the different reactions accounted for. The first parameter corresponds to the reaction type.

### 8.5.1 Type 0: Irreversible 1st order reaction

For this type of reaction, a reaction that transforms X into Y is represented with a kinetic parameter k such as:

$$X \xrightarrow{k} Y \tag{7}$$

For example, this type of reaction can be activated by adding the following line in the file indicated by *reaction_soap_file* in the namelist:
0 ASOA1 ANVSOA 3.e-4 0 0 0

The first parameter after the kinetic type corresponds to the name of compound X. The second is the name of compound Y and the third is the kinetic k.

The fourth parameter is used to determine if the kinetic is catalyzed by water. In that case (parameter=1), the kinetic parameter is multiplied by the activity of water (equal to RH at equilibrium).

The fifth parameter is used to determine if the kinetic is catalyzed by pH. In that case (parameter=1), the kinetic parameter is multiplied by pH.

The final parameter, is used to consider if the reaction is mass conserving (=1) or mol conserving (=0).

An example of the transformation of semi-volatile into nonvolatile compounds is shown in Fig. 25 assuming a lifetime of semi-volatile in particles around 1 h, which corresponds to a kinetic rate of about 3 $10^{-4}$ $s^{-1}$, as done within CHIMERE by Cholakian et al. (2018). However, although the use of this kinetic rate may not be realistic, it may be used to try to implicitly represent the effect of viscosity.

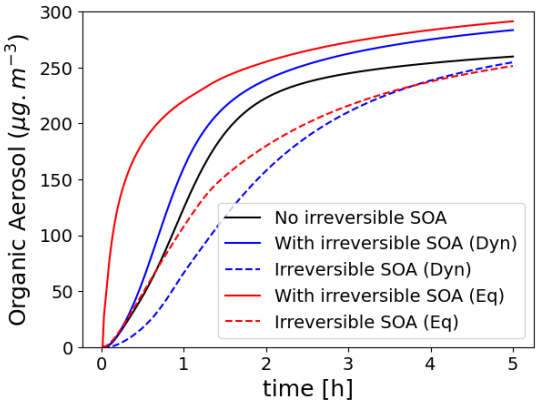


Figure 25: Time evolution (for the Platt test case with $NH_3$ emissions) of SOA concentrations with and without an irreversible transformation of SOA into non-volatile SOA. A kinetic of 3 × $10^{-4}$ $s^{-1}$ was used.

This type of reaction could also be used to represent hydrolysis (for example the hydrolysis of nitrate groups into hydroxyl groups).

This test case can be reproduced by using the namelists *INIT/namelist_platt_nh3_isopa_irreversible.ssh* (dynamic mode for the condensation of organic) and *INIT/namelist_platt_nh3_isopa_irreversible-eq.ssh* (equilibrium mode). The results should be compared with *INIT/namelist_platt_nh3_isopa.ssh*. The figure is done by *graph/platt_irr.py*.

### 8.5.2 Type 1: Bulk oligomerization

To activate "bulk oligomerization", an oligomer species should be first added to the aerosol namelist as described in the section "Oligomer species".

For example, this type of reaction can be activated by adding the following line in the file indicated by *reaction_soap_file* in the namelist:
1 BiA2D OligoBiA2D 2.2e-4 2 2.94 0
BiA2D is the name of the monomer species, OligoBiA2D is the name of the oligomer species, 2.2e-4 is the kinetic of oligomerization ($k_{oligo}^{max}$, 2 is the number of monomer blocks in the oligomer (=$m_{oligo}$), 2.94 is an equilibrium constant (=$K_{oligo}^{eq}$). The last parameter is used to determine if the kinetic is catalyzed by pH. In that case (parameter=1), the kinetic parameter is multiplied by pH.

In some modeling studies, a parameterization for oligomerization based on a simple first-order rate constant of oligomerization is used Carlton et al. (2008); Lemaire et al. (2016). However, a first-order complete reaction may not be appropriate to represent oligomerization. Indeed, oligomerization should involve second-order reversible reactions like esterification, hemiacetalization, aldolization, peroxyhemiacetalization. The equilibrium of these reactions is unfavored by humid conditions and the reaction is catalyzed under acidic conditions Couvidat et al. (2018b).

SSH-aerosol uses the bulk oligomerization parameterization developed by Couvidat et al. (2018b). This paramerization assumes that oligomerization is represented by a reversible process which is mainly due to mechanisms like esterification, unfavored by humid conditions. In the bulk oligomerization parameterization, oligomerization is represented through a single reaction to estimate if a compound is presented as a monomer or as an oligomer.

$$A \underset{k_{reverse}}{\overset{k_{oligo}}{\leftrightarrow}} \frac{1}{m_{oligo}} A_{oligo} \tag{8}$$

with A a monomer compound, $A_{oligo}$ the monomer blocks of compound A inside oligomers, $m_{oligo}$ the number of monomer blocks inside oligomers, $k_{oligo}$ the kinetic rate parameter of oligomerization and $k_{reverse}$ the kinetic rate parameter of the reverse reaction. In this parameterization, oligomers are represented by a single model species. A represents the monomer and $A_{oligo}$ any monomer blocks present inside the oligomers. It can be used to represent oligomerization in a mixture of compounds (for example a mixture of compounds A and B). If a monomer block from compound A reacts with another monomer block (the same compound A or another compound B), the compound is converted from monomer A to oligomer $A_{oligo}$ (independently of the reaction with A or B). The parameterization currently does not take into account different kinetic rate parameters between each combination of compounds due to a lack of data. It assumes that a compound A reacting with itself and a compound B reacting with itself have the same kinetic parameter as when compounds A and B react together. When more data become available, the parameterization could be improved to take into account different kinetic rate parameters of oligomerization per combination.

The net flux of oligomerization $J_{oligo}$ is computed using activities. Activity is often seen as

the "apparent concentration" of a compound in thermodynamics. It is linked to the chemical potential (molar Gibbs free energy of a particular component) by the following equation:

$$a_i = exp(\frac{\mu_i - \mu_i^0}{RT}) \tag{9}$$

with $a_i$ the activity of compound i, $\mu_i$ is the chemical potential of compound i and $\mu_i^0$ the chemical potential under standard conditions, R is the gas constant, T is thermodynamic temperature. Activities (calculated here on the mole fraction basis) are used instead of concentrations for two main reasons. First, chemical rates are more consistent with thermodynamic equilibrium by computing rates using activities. For example, in the case of a simple one product (A) giving one product (B) equilibrium reaction, if chemical reactions are written using concentrations, the net flux of reaction J would be computed with the following equation:

$$J = k_1 C_A - k_{-1} C_B \tag{10}$$

with $k_1$ the forward kinetic parameter, $k_{-1}$ the reverse kinetic parameter, $C_A$ the concentration of compound A and $C_B$ the concentration of compound B. At equilibrium, J would be equal to zero and the equilibrium constant would then correspond to the ratio of concentrations instead of a ratio of activities. This paradox can be lifted by using activities instead of concentrations. Moreover, some studies expressed the need to compute chemical rates using activities and showed that better results are obtained for non-ideal systems Madon and Iglesia (2000); Rahimpour (2004).

The net flux of oligomerization $J_{oligo}$ for the reduced parameterization is therefore computed with the following equations:

$$J_{oligo} = -\frac{dX_{A,monomer}}{dt} = k_{oligo} a_{A,monomer} - k_{reverse} a_{A,oligomer} \tag{11}$$

with $X_{A,monomer}$ the molar fraction of compound A, $a_{A,monomer}$ the activity on a molar fraction basis of compound A and $a_{A,oligomer}$ the activity on a molar fraction basis of the oligomer formed from compound A.

The kinetic rate of oligomerization $k_{oligo}$ is computed as follows:

$$k_{oligo} = k_{oligo}^{max} a_{monomer} \tag{12}$$

with $k_{oligo}^{max}$ the maximum kinetic rate parameter for oligomerization ($k_{oligo}^{max} = 2.2 \times 10^{-4}$ $s^{-1}$ following Couvidat et al. (2018a)), and $a_{monomer}$ the sum of monomer activities.

To calculate the reverse kinetic rate parameter, the computation is based on the equilibrium oligomerization constant $K_{oligo}^{eq}$, which is estimated to be 2.94 Couvidat et al. (2018a). The equilibrium constant for oligomerization (due to esterification or a similar oligomerization mechanism) may be written as

$$(K_{oligo}^{eq})^{m_{oligo}-1} = \frac{a_{A,oligomer}(a_{H_2O})^{m_{oligo}-1}}{a_{A,monomer}(a_{monomer})^{m_{oligo}-1}} \tag{13}$$

with $a_{H_2O}$ the activity of water on a molar basis.
At equilibrium, the rate of oligomerization is zero. Therefore,

$$\frac{k_{oligo}^{max}}{k_{reverse}} = \frac{a_{a,oligomer}/m_{oligo}}{a_{monomer} a_{a,monomer}} = \frac{(K_{oligo}^{eq})^{m_{oligo}-1}(a_{monomer})^{m_{oligo}-2}}{(a_{H_2O})^{m_{oligo}-1}} \tag{14}$$

For simplification purporses, $m_{oligo}$ is set to 2, and $k_{reverse}$ is estimated from Eq. 14.

The configuration file for the test case of exhaust emissions with added isoprene emissions and oligomerization (and activity coefficients computed with AIOMFAC) is *namelist_platt_nh3_isopa_oligo.ssh*.

The evolutions of the concentrations of isoprene SOA, monomers, and oligomers in the particles as a function of time for these simulations may be seen in Figure 26, which can be plotted running the script *platt_oligo.py*. Although, the SOA concentration do not change significantlty when oligomerization is activated, the SOA composition is strongly impacted. After 1 hour, almost all the SOA mass is constituted by monomer species. However, after 5 hours, monomer species constitute only 50% of isoprene SOA.

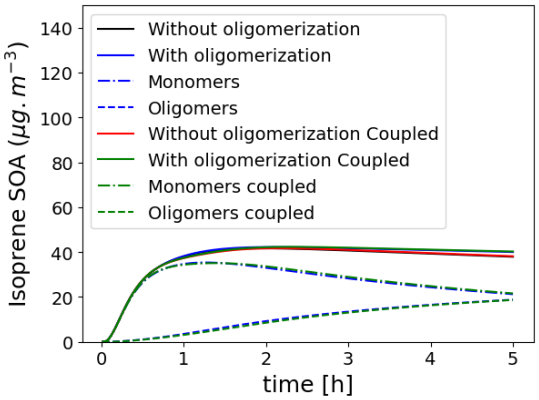


Figure 26: Time evolution of isoprene SOA, monomers, and oligomers in the particle phase.

### 8.5.3 Type 2: Hydratation of aldehydes

Aldehydes (and more generally carbonyl compounds) can undergo hydratation. In that case, the carbonyl group is transformed into a diol, and hydratation has to be represented as an equilibrium between two species, the aldehyde and the diol, using an hydratation constant $K_{hydratation}$ such as:

$$\frac{[diol]}{[aldehyde]} = K_{hydratation} \frac{\gamma^{\infty}_{diol}\gamma_{aldehyde}}{\gamma_{diol}\gamma^{\infty}_{aldehyde}} a_w \tag{15}$$

with $a_w$ the activity of water (equal to RH at equilibrium). $\gamma_{diol}$ and $\gamma_{aldehyde}$ corresponds to the activity coefficient of the diol and the aldehyde. They are normalized by the activity coefficient at infinite dilution $\gamma^{\infty}$ as the hydratation constant is generally measured in the aqueous phase.

The equilibrium constant between the diol and the aldehyde is generally quite low (0.57 for butanal according to GECKO-A Camredon et al. (2007)). However, glyoxal can be hydradated two times with a very high constant as illustrated in Fig. 27.

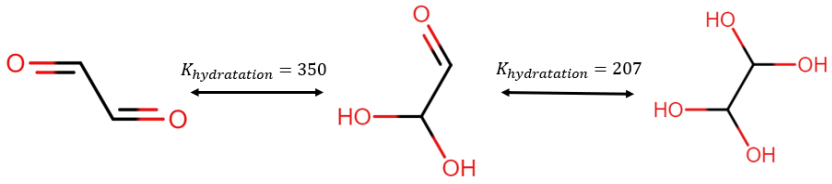


Figure 27: Illustration of glyoxal conversion into polyols due to hydration. The hydratation constants are taken from Ervens and Volkamer (2010).

While the concentrations of glyoxal in the particle phase without hydratation would be negligible (below $10^{-8}$ $\mu$g/m$^3$ in the Platt isoprene testcase with ammonia emissions), the total concentrations of glyoxal and its hydrate reach $1.8 \times 10^{-3}$ $\mu$g/m$^3$ in the *INIT/namelist_platt_nh3_isopagly.ssh* testcase (has to be launched with RACM2 to account for the formation of glyoxal).

Hydratation for glyoxal can be activated by adding the following lines in the file indicated by *reaction_soap_file* in the namelist:
2 GLY GLYOH 350.0
2 GLYOH GLYOHOH 207.0

### 8.5.4 Type 3: Reaction of organic compounds with inorganic ions

Organic compounds in the aqueous can react with inorganic ions to form other compounds. For example, the compound IEPOX (epoxide formed by the oxidation of isoprene) can react with $SO_4^{2-}$ and $HSO_4^-$ ions to form organosulfate Couvidat et al. (2013) or with $NO_3^-$ to form organonitrate.

The reaction of IEPOX with $NO_3^-$ can be activated by adding this line in the file indicated by *reaction_soap_file* in the namelist:
3 IEPOX NO3 BiNO3 2.e-4 1 0 0
with IEPOX the reacting organic, NO3 the reacting ion and BiNO3 the formed product. 2.e-4 is the kinetic of the reaction Eddingsaas et al. (2010). The following parameters are used to determine if the reaction is catalyzed by pH, if the reaction is catalyzed by water, and if the ion has to be consumed by the reaction or not (in that case the concentrations of the ion are unchanged).

The Fig. 28 illustrates the formation of SOA due to antiparticle reactions for different amounts of ammonia (that affect the pH of the particle). In the absence of ammonia (very acidic particles), the formation of organosulfate is very important. For partially neutralized sulfate, the formation of organosulfate decreases but is compensated by the formation of methyltetrols due to the hydrolysis of IEPOX. When sulfate is almost totally neutralized, the concentrations of SOA formed by the hydrolysis of IEPOX decreases significantly and is strongly dominated by the formation of methyltetrols.

This testcase can be reproduced by launching the following namelists for soap_inorg=1: *INIT/namelist_iepox.ssh* (absence of ammonia), *INIT/namelist_iepox1.ssh* (partially neutralized) and *INIT/namelist_iepox2.ssh* (almost totally neutralized). The same namelist with soap_inorg=0 are also available: *INIT/namelist_iepox_iso.ssh*, *INIT/namelist_iepox1_iso.ssh* and *INIT/namelist_iepox2_iso.ssh* In this testcase, condensation/evaporation is treated at equilibrium. Two additional test cases are available to reproduce the simulation without ammonia with condensation/evaporation solved dynamically: *INIT/namelist_iepox_dyn.ssh* (soap_inorg=1) and *INIT/namelist_iepox_dyn_iso.ssh* (soap_inorg=0). The results are illustrated by Fig. 29.

The graph can be obtained by launching *graph/iepox.py*.

### 8.5.5 Type 4: Oligomerization of BiAOD

Concerning organic reactions in the particles, oligomerization of BiAOD (pinonaldehyde) may be considered. The oligomerization can be activated or deactivated by adding or removing this line in the file indicated by *reaction_soap_file* in the namelist:
4 BiA0D 0.1 1.91 6

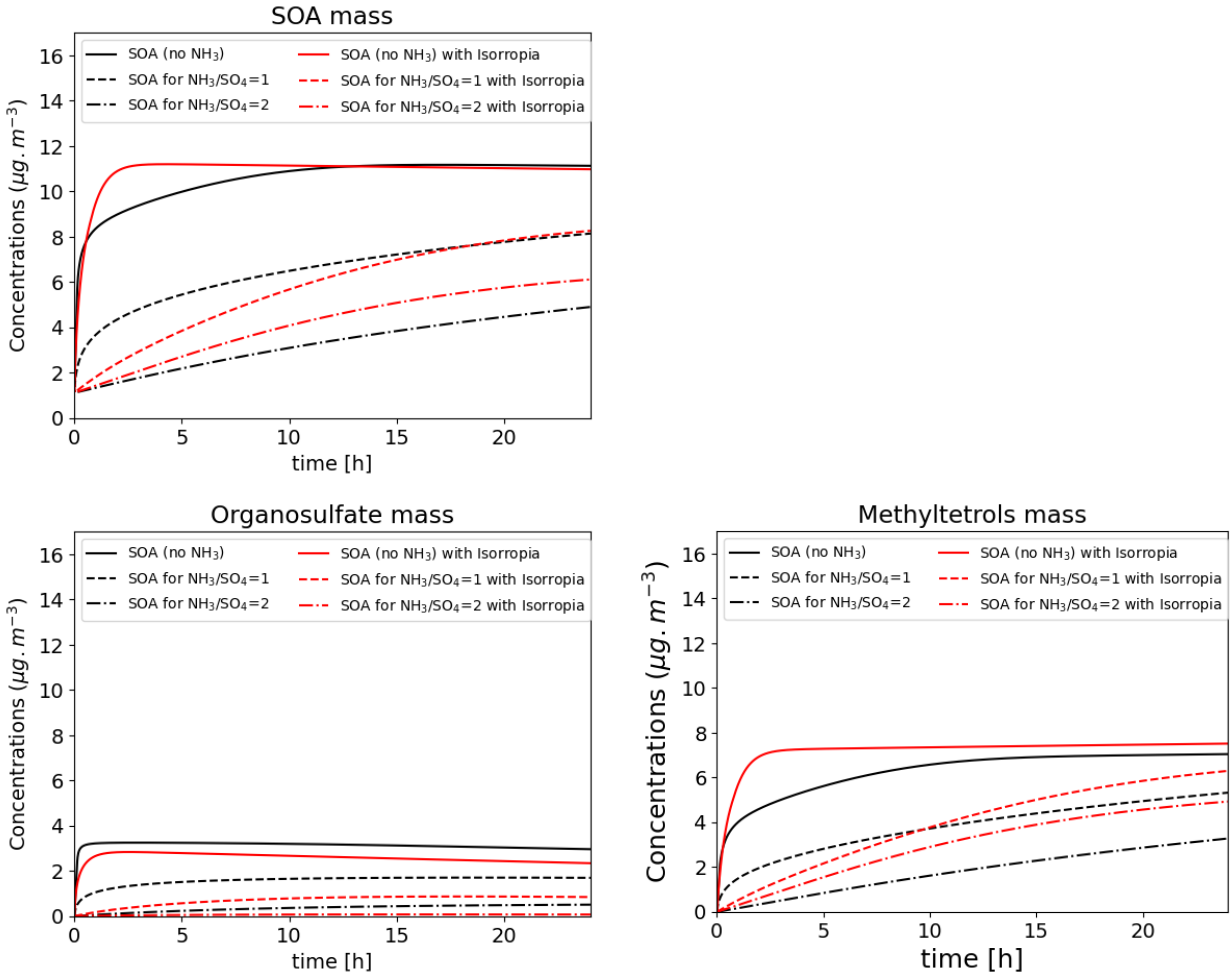


Figure 28: Formation of SOA (top), organosulfate (bottom left) and methyltetrols (bottom right) due to the reaction of 10 $\mu$g/m$^3$ IEPOX in the particle phase (in the presence of 10 $\mu$g/m$^3$ of $HNO_3$ at the beginning of the reaction and 3 $\mu$g/m$^3$ of sulfate) for a different ratio of ammonia to sulfate with soap_inorg=0 (Isorropia for inorganic thermodynamic) or soap_inorg=1 (SOAP for inorganic thermodynamic).

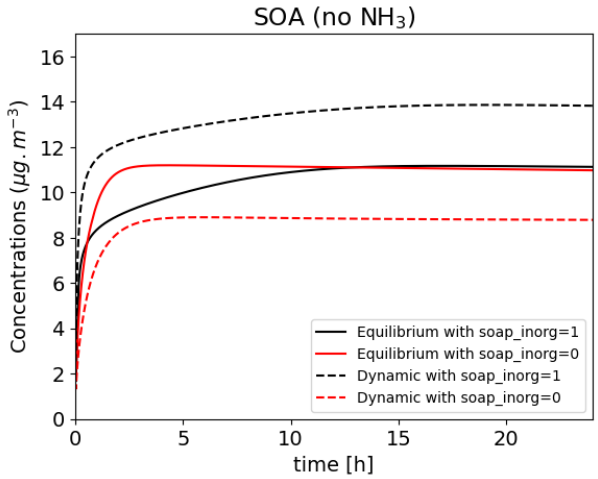


Figure 29: Formation of SOA (top) due to the reaction of 10 $\mu$g/m$^3$ IEPOX in the particle phase (in the presence of 10 $\mu$g/m$^3$ of $HNO_3$ at the beginning of the reaction and 3 $\mu$g/m$^3$ of sulfate) without ammonia with soap_inorg=0 and soap_inorg=1 when the equilibrium or dynamic approch for condensation/evaporation is used.

## 8.6 Modelling of complex VOC chemical mechanisms

The degradation of primary VOCs in the troposphere results in low volatile organic compounds that can condense on particles under favorable conditions and form SOA. To address the influence of detailed gas-phase chemical processes of VOC oxidation on the formation of SOA, SSH-aerosol provides an option to simulate SOA formation with very detailed VOC chemical mechanisms, such as a near-explicit Master Chemical Mechanism (MCM) Jenkin et al. (1997). If a near-explicit mechanism is used for some VOCs, the inorganic reactions leading to the formation of oxidants do not always need to be included in the reaction list, since their concentrations can be fixed by the input constant concentration profiles (detailed in Sect.6.3), or they can be estimated using a specific chemical scheme, such as CB05 or RACM2. Two examples are given here: one is the example of the MCM beta-caryophyllene scheme with fixed oxidant concentrations and the other one is a chemical scheme built using CB05 and chemical schemes from Wang et al. (2023) from monoterpenes and sesquiterpene, and from Sartelet et al. (2024) for toluene.

### 8.6.1 Complex VOC mechanisms: example of beta-caryophyllene

An example of simulating the degradation scheme of beta-caryophyllene (BCARY) in MCM v3.3.1 Jenkin et al. (2012) is given as a test case. In SSH-aerosol, the chemistry flag to use the BCARY gas-phase chemical mechanism is ***mcm***. The corresponding reaction list (*BCARYorg.reaction*) and species list (*BCARYorg.species*) is located in the directory: *src/include/CHEMISTRY/mcm*, along with the Peroxy radicals ($RO_2$) species list (*BCARYorg.RO2*), and a file containing the species properties (*BCARYorg.mol*).

Additionally, the aerosol species lists used in this test case are provided in the folder *species-list/* with the keyword *-mcm-bcary-*:

- *species-list-aer-mcm-bcary-smiles.dat*. It provides aerosol SMILES structures that can be directly read by SSH-aerosol.
- *species-list-aer-mcm-bcary-fgl.dat*. It provides aerosol UNIFAC functional group lists that can be directly read by SSH-aerosol.

- *species-list-aer-mcm-bcary.dat*. It does not provide the chemical structures of generic species. This information can be read from a separate file (*aerosol-structure-mcm-bcary.dat*).

Additionally, the species properties file and reaction list can be used in the GENOA model Wang et al. (2022) to generate reduced semi-explicit SOA mechanisms, which can be applied to SSH-aerosol as well.

The user can then run the test case with either of the two namelists: *INIT/namelist_mcm-wSMILES.ssh* or *namelist_mcm-wFGL.ssh*. Although the two namelists read the chemical structures of the generic aerosol species differently, their simulation results should be identical.

**Treatment of $RO_2$-$RO_2$ reactions** Under low-NOx conditions, SOA is formed as a result of $RO_2$ species reacting with hydroperoxy radicals ($HO_2$) and other $RO_2$ radicals. In SSH-aerosol, $RO_2$-$RO_2$ reactions can be treated using the concept of a "$RO_2$ pool". For example, a $RO_2$-$RO_2$ reaction can be written as follows:

```
NBCO2 -> 3.000E-01 BCBNO3 + 7.000E-01 BCAL //
     + 7.000E-01 NO2
%9.20E-14*RO2
KINETIC RO2 1 ARR 9.200E-14
```

where "RO2 1" indicates that species NBCO2 reacts with the total concentration of all $RO_2$ species in the $RO_2$ pool with index "1". The number following "RO2" is required and specifies which $RO_2$ pool the species reacts with. It can be adjusted to refer to a different $RO_2$ pool if multiple pools are considered (the numbering starts from 1). All $RO_2$ species are defined in an input RO2 species file, where each line lists an RO2 species name and its corresponding $RO_2$ pool number, separated by a space.

The $RO_2$-$RO_2$ reaction can be treated in four different ways by adopting different options (*tag_RO2*) in the namelist:

- tag_RO2=0, compute no $RO_2$-$RO_2$ reaction. A zero kinetic rate is given to the reaction with "TB RO2'.
- tag_RO2=1, compute $RO_2$-$RO_2$ reactions with only the produced $RO_2$ concentrations. The total concentrations of all $RO_2$ species recorded in the $RO_2$ species list are added up to compute the kinetic rates.
- tag_RO2=2, compute $RO_2$-$RO_2$ reactions with only the background $RO_2$ concentrations. For this option to work, the background concentrations should be provided for the gas-phase species *RO2pool* in the input constant concentration file.
- tag_RO2=3, compute $RO_2$-$RO_2$ reactions with both the produced and background $RO_2$ concentrations. In this case, the kinetic rate is calculated based on a sum of the background concentrations from the constant concentration file and the produced $RO_2$ concentrations of all $RO_2$ species.

In order to record the output concentrations of the $RO_2$ pool, a specific gas-phase species, *RO2pool*, has been added to the gas-phase species list. There should be no other use for this *RO2pool* species. Here is an example of how to set up the namelist for $RO_2$-$RO_2$ reactions:

```
! use two-step chemical solver
tag_twostep = 1,
```

```
! RO2 species list
RO2_list_file = "./src/include/CHEMISTRY/mcm/BCARYorg.RO2",
! RO2-RO2 reaction option:
! 0 no RO2 reaction, 1 only generated RO2, 2 only background RO2
! 3 background + generated RO2
tag_RO2 = 3,
```

Shell scripts *INIT/launch_mcm_diffRO2.sh* and *INIT/launch_mcm_diffRO2_smiles.sh* are provided in SSH-aerosol, which allow the user to test different *tag_RO2* options. The results of the simulation should be located under the folder *results* with the keyword *mcm-wFGL-tag_RO2-* or *mcm-wSMILES-tag_RO2-*, if the script is successfully launched with

```
bash INIT/launch_mcm_diffRO2.sh
```

With the Python script *graph/mcm.py*, the plots of the simulated BCARY-SOA yields can be generated, as shown in Fig. 30.

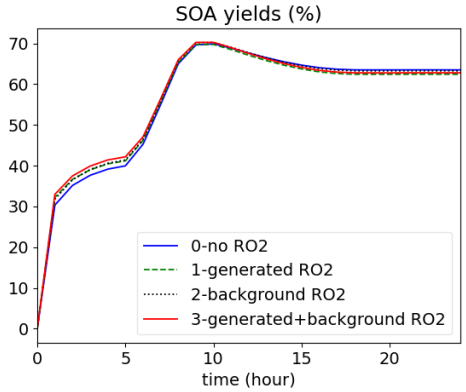


Figure 30: Time evolution of BCARY-SOA yields simulation with the BCARY MCM chemical mechanism and different $RO_2$-$RO_2$ reaction options.

### 8.6.2 Chemical schemes with complex VOC mechanisms

In this test case, chemical schemes of different complexities are used and compared with respect to their simulated SOA concentrations for different $NO_2$ levels. Initial gas concentrations are taken as representative of atmospheric conditions, as simulated in Sartelet et al. (2022) over Greater Paris in July 2009. The initial $NO_2$ concentration is taken equal to 26 $\mu$g m$^{-3}$, and the simulated OH concentration varies between 1.2 and 7.6 x $10^{-4}$ $\mu$g m$^{-3}$. Simulations are performed with initial $NO_2$ concentrations multiplied by 2 (52 $\mu$g m$^{-3}$) or divided by 2 (13 $\mu$g m$^{-3}$). Low values are taken for initial organic and inorganic aerosol concentrations, to serve as a condensation core. To keep the particle dynamic as simple as possible, only one size section is used. It is fixed to be of diameters in the range 0.1585 $\mu$m-0.4 $\mu$m. The simulation is run for about 5 hours with outputs every minute.

Chemical schemes are built using CB05 and chemical schemes of different complexities for monoterpenes, toluene, naphthalene, and sesquiterpene. For all these SOA precursors, the $H^2O$ mechanism built from chamber measurements is used. For monoterpenes, the $H^2O$ mechanism is supplemented with a simple scheme representing autoxidation Chrit et al. (2017). For monoterpenes, the near-explicit scheme is from MCM and PRAM, and the GENOA reduced scheme from Wang et al. (2023). For sesquiterpene (*beta*-caryophyllene), the near-explicit scheme is

from MCM and the GENOA reduced scheme is from Wang et al. (2024). For toluene, the near-explicit scheme is from Lannuque et al. (2023). It is supplemented with the molecular rearrangement as implemented in Sartelet et al. (2024), and the GENOA reduced scheme is from Sartelet et al. (2024). For naphthalene, the near-explicit scheme is from Lannuque and Sartelet (2024). The schemes are detailed in the repertory *src/include/CHEMISTRY/individual_schemes/*.

The different chemical schemes are combined into the CB05 mechanism for oxidant and $O_3$ formation. Four combinations are studied here. In each combination, the scheme CB05 is supplemented with different SOA schemes for monoterpenes, sesquiterpene, toluene, and naphthalene. In the first combination, the SOA schemes correspond to the near-explicit ones. In the second combination, GENOA-reduced schemes are used for monoterpenes, sesquiterpene, and toluene SOA. In the third and fourth combinations, the $H^2O$ mechanism is considered for the SOA precursor, while autoxidation of monoterpenes is ignored in the fourth combination. To combine the chemical schemes with the CB05 mechanism for oxidant and $O_3$ formation, the links to the path of the chemical schemes and to the species list are given in the files *species_matching_expl.dat* and *user_defined_scheme_expl.cfg* for the first combination using the explicit schemes, in the files *species_matching_rdc.dat* and *user_defined_scheme_rdc.cfg* for the second combination using the reduced schemes, in the files *species_matching_h2o.dat* and *user_defined_scheme_h2o.cfg* for the third combination using the $H^2O$ scheme, in the files *species_matching_h2o-noautox.dat* and *user_defined_scheme_h2o-noautox.cfg* for the fourth combination using the $H^2O$ scheme without autoxidation for monoterpenes. These files may be found in the directory *src/include/CHEMISTRY/cb05-ozone*.

To generate the overall chemical mechanism with the explicit schemes (first combination) and generate the code executable, please type

```
./clean -a
./compile -b=user -c=cb05-ozone -u=user_defined_scheme_expl.cfg -m=species_matching_expl.dat
```

During the compilation, the files describing the reactions and species list are automatically placed in the repertory "src/include/CHEMISTRY/cb05-ozone/". They are also available for reference in the repertory "src/include/CHEMISTRY/generated_mechanisms/expl/".

The simulations with the near-explicit schemes and the different $NO_2$ levels can then be run by typing the following commands:

```
./ssh-aerosol INIT/namelist_vocox_expl.ssh
./ssh-aerosol INIT/namelist_vocox_expl_dNO2.ssh
./ssh-aerosol INIT/namelist_vocox_expl_mNO2.ssh
```

To generate the overall chemical mechanism with the GENOA reduced schemes (second combination) and generate the code executable, please type

```
./clean -a
./compile -b=user -c=cb05-ozone -u=user_defined_scheme_rdc.cfg -m=species_matching_rdc.dat
```

During the compilation, the files describing the reactions and species list are automatically placed in the repertory "src/include/CHEMISTRY/cb05-ozone/". They are also available for reference in the repertory "src/include/CHEMISTRY/generated_mechanisms/rdc/".

The simulations with the GENOA reduced schemes and the different $NO_2$ levels can then be run by typing the following commands:

```
./ssh-aerosol INIT/namelist_vocox_rdc.ssh
./ssh-aerosol INIT/namelist_vocox_rdc_dNO2.ssh
./ssh-aerosol INIT/namelist_vocox_rdc_mNO2.ssh
```

To generate the overall chemical mechanism with the $H^2O$ schemes (third combination) and generate the code executable, please type

```
./clean -a
./compile -b=user -c=cb05-ozone -u=user_defined_scheme.cfg
-m=species_matching_h2o.dat
```

The simulations with the $H^2O$ schemes and the different $NO_2$ levels can then be run by typing the following commands:

```
./ssh-aerosol INIT/namelist_vocox.ssh
./ssh-aerosol INIT/namelist_vocox_dNO2.ssh
./ssh-aerosol INIT/namelist_vocox_mNO2.ssh
```

To generate the overall chemical mechanism with the $H^2O$ schemes without monoterpene autoxidation (fourth combination) and generate the code executable, please type

```
./clean -a
./compile -b=user -c=cb05-ozone -u=user_defined_scheme_h2o-noautox.cfg
    -m=species_matching_h2o-noautox.dat
```

The simulations with the $H^2O$ schemes and the different $NO_2$ levels can then be run by typing the following commands:

```
./ssh-aerosol INIT/namelist_vocox-h2onoautox.ssh
./ssh-aerosol INIT/namelist_vocox-h2onoautox_dNO2.ssh
./ssh-aerosol INIT/namelist_vocox-h2onoautox_mNO2.ssh
```

The toluene SOA concentrations simulated with the different schemes and in the different $NO_2$ conditions may be compared by running the scripts *graph/plot_vocox_toluene_h2o.py* and *graph/plot_vocox_toluene_rdc.py*. As shown in Fig. 31, the toluene SOA concentrations increase with time. Using the near-explicit schemes, the toluene SOA concentrations are lower when the $NO_2$ concentrations are decreased by a factor 2. If the $NO_2$ concentrations are increased by a factor 2, the toluene SOA concentrations are lower than the reference in the first 3 hours of the simulations, but they are higher after. As shown in the left panel of Fig. 31, the toluene SOA concentrations of the reduced and near-explicit schemes are very similar. As shown in the left panel of Fig. 31, the toluene SOA concentrations of the $H^2O$ scheme are lower than those of the near-explicit schemes. This was already obserbed in Sartelet et al. (2024), and it is probably due to the influence of molecular rearrangement in the explicit scheme. The evolution of the toluene SOA concentrations is similar between $H^2O$ and the near-explicit schemes when $NO_2$ concentrations are reduced: the toluene SOA concentrations decrease. They also decrease with the $H^2O$ scheme when $NO_2$ concentrations are doubled. This decrease persists through the whole simulation, contrary to the near-explicit scheme.

The monoterpene SOA concentrations simulated with the different schemes and in the different $NO_2$ conditions may be compared by running the scripts *graph/plot_vocox_monoterp_h2o.py*, *graph/plot_vocox_monoterp_rdc.py*, *graph/plot_vocox_monoterp_h2o-noautox.py*. As shown in Fig.32, the monoterpene SOA concentrations increase through the simulation. They also increase as $NO_2$ levels decrease, as already noted in Wang et al. (2024). As shown in the left panel of Fig.32, the SOA concentrations simulated with the reduced schemes and their evolution with $NO_2$ levels are similar between the reduced and the near-explicit schemes. As shown in the lower panel of Fig.32, the monoterpene SOA concentrations are significantly lower with the $H^2O$ mechanism

that ignores autoxidation. Furthermore, monoterpene SOA concentrations decrease as $NO_2$ levels decrease in opposite to the simulation with the neat-explicit mechanism. Taking into account autoxidation in the $H^2O$ mechanism leads to higher monoterpene SOA concentrations (upper left panel of Fig32). The monoterpene SOA concentrations are then higher with lower $NO_2$ levels, as simulated with the near-explicit scheme. Furthermore, they are also lower than the reference with higher $NO_2$ levels in the first 1.5 hours of the simulation. However, they become higher than the reference with higher $NO_2$ levels in the remaining time of the simulation, in opposition to the simulation with the near-explicit scheme.

The naphthalene SOA concentrations simulated with the different schemes and in the different $NO_2$ conditions may be compared by running the script *graph/plot_vocox_naph.py*. As shown in the left panel of Fig.33, the naphtalene SOA concentrations increase through the simulation. They also increase as the $NO_2$ levels increase. The naphtalene SOA concentrations simulated with the $H^2O$ scheme are much lower than those simulated with the near-explicit scheme, except when the $NO_2$ levels are doubled.

The sesquiterpene SOA concentrations simulated with the different schemes and in the different $NO_2$ conditions may be compared by running the script *graph/plot_vocox_sesqui.py*. As shown in the right panel of Fig.33, the SOA concentrations increase through the simulation. During the first two hours of the simulations, the sesquiterpene SOA concentrations increase with increasing $NO_2$ levels. This is not the case in the simulation with the $H^2O$ mechanism, where the SOA concentrations are higher with the reference $NO_2$ levels. These test cases illustrate the importance to near-explicit chemical schemes, which integrate the different chemical pathways for SOA formation. Simple schemes based on chamber experiments may estimate different evolution with changing environments than the near-explicit schemes. Reduced schemes built with GENOA are able in the examples provided here to reproduce the evolution predicted by the near-explicit schemes.

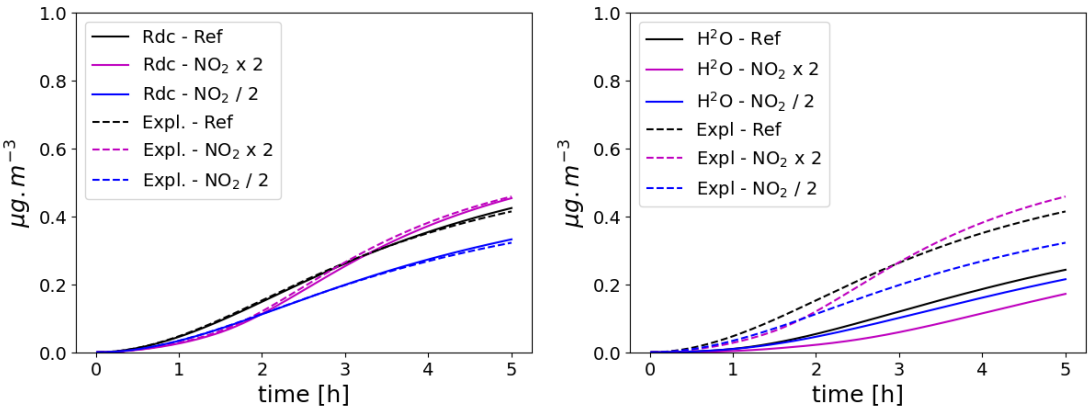


Figure 31: Time evolution of toluene SOA mass with different chemical schemes and for different $NO_2$ initial concentrations. In the right panel, the near-explicit chemical scheme of Lannuque et al. (2023) supplemented by molecular rearrangement Sartelet et al. (2024) is compared to the $H^2O$ mechanism. In the left panel, the near-explicit scheme is compared to the reduced scheme of Sartelet et al. (2024).

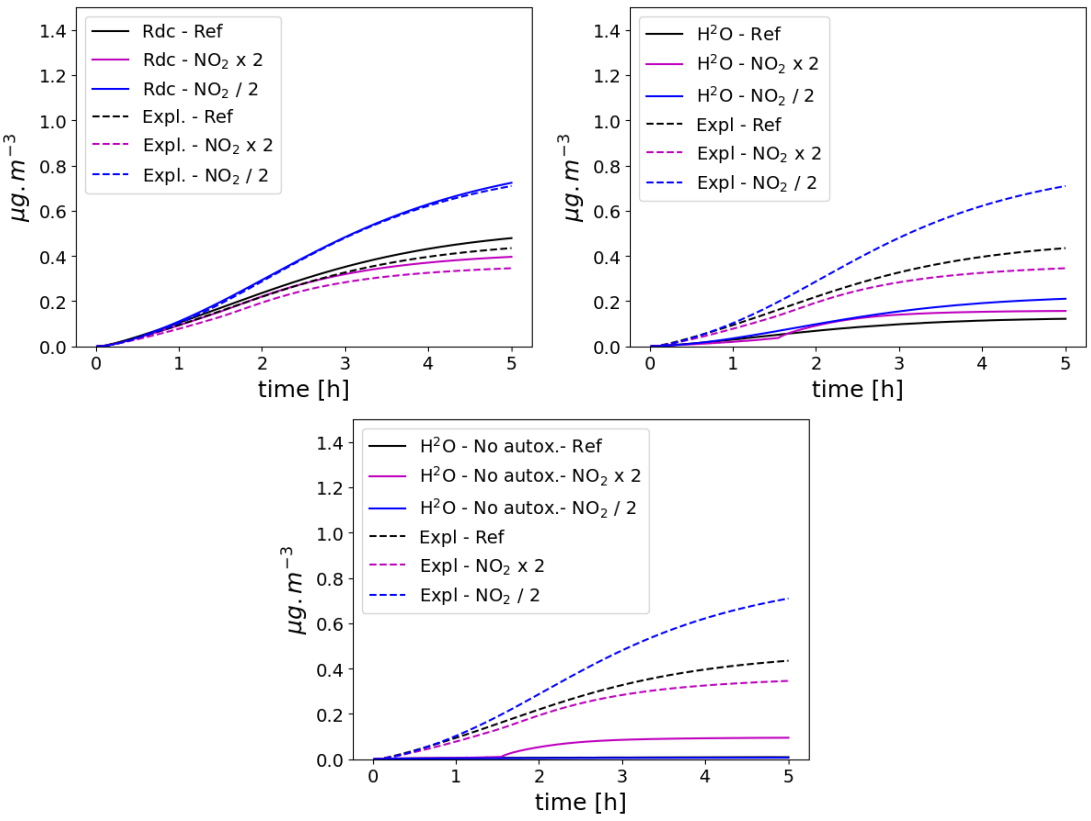


Figure 32: Time evolution of monoterpene SOA mass with different chemical schemes and for different $NO_2$ initial concentrations. In the upper left panel, the near-explicit chemical scheme of MCM and PRAM is compared to the reduced scheme of Wang et al. (2023). The near-explicit scheme is compared to the $H^2O$ mechanism in the upper right panel and to the $H^2O$ mechanism without autoxidation in the lower panel.

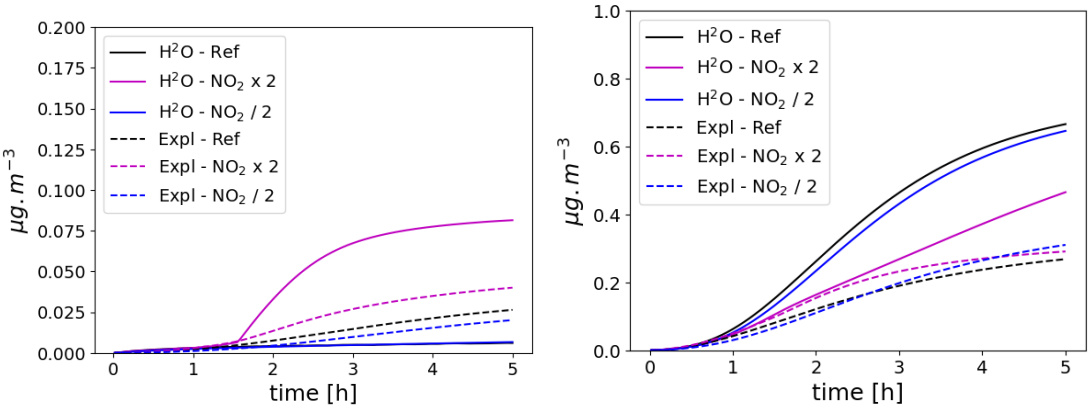


Figure 33: Time evolution of naphthalene (left panel) and $\beta$-caryophyllene (right panel) SOA mass with different chemical schemes and for different $NO_2$ initial concentrations. The near-explicit chemical scheme of Lannuque and Sartelet (2024) is compared to the $H^2O$ mechanism.

## 8.7 Comparisons to flow tube or chamber experiments

### 8.7.1 Wall losses

Wall losses of particles and organic vapors can be accounted for in the model.

**Wall losses of organic vapors:** A reversible losses of organic vapors is accounted for SVOC species defined in the *aerosol_species_list_file* with the following reaction from Huang et al. (2018):

$$\frac{dC_g}{dt} = -k_w^g (C_g - \frac{C_w}{K_w C_{wall}}) \tag{16}$$

with $C_g$ the gas phase concentrations, $k_w^g$ the wall losses kinetic of organic vapors, $C_w$ the equivalent wall concentration and $K_w$ the wall partitioning constant.
$C_{wall}$ is the equivalent wall concentration (in $\mu g/m^3$) that has to be defined in the namelist. If $C_{wall}$ is not defined or equal to 0, wall deposition of organic vapors will not be taken into account in the model. One possible method to estimate $C_{wall}$ and $K_w$ is to use the method of Huang et al. (2018):

$$C_{wall} = 10\ 800\ \frac{S}{V} \tag{17}$$

with S/V the surface on volume ratio of the chamber, and

$$K_w = \frac{K_p}{\gamma^{wall}} \times \frac{M_m}{200} \tag{18}$$

with $M_m$, the molar mass, $\gamma^{wall}$ the wall activity coefficient, $K_p$ the ideal partitioning constant computed such as:

$$K_p = \frac{760 \times 8.202 \times 10^{-5}}{10^6 M_m P_i^0(T)} \tag{19}$$

with $P_i^0(T)$ the saturation vapor pressure at temperature T (calculated from the properties defined in *aerosol_species_list_file*. The wall activity coefficient is calculated by this formula calculated for Teflon film (fluorinated ethylene propylene):

$$\gamma^{wall} = 10^{3.299} K_p^{0.6407} \tag{20}$$

$k_w^g$ can be directly provided by the user by defining the value of the parameter *kwall_gas* in the namelist. Alternatively, $k_w^g$ can be calculated based on the eddy diffusion coefficient. For that, the eddy diffusion coefficient $k_e$ (parameter *eddy_turbulence* in the namelist) and the S/V ratio (parameter *surface_volume_ratio* in the namelist) have to be provided by the user. It will then be calculated with the following formula:

$$k_w^g = \frac{S}{V} \left( \frac{\pi}{2\sqrt{k_e D_g}} + \frac{4}{\alpha_w \bar{c}} \right)^{-1} \tag{21}$$

with $D_g$ the gas-phase diffusion coefficient, $\bar{c}$ the mean speed velocity of the compound and $\alpha_w$ the accommodation coefficient for condensation onto the wall Lannuque et al. (2023).

$$\alpha_w = 10^{-2.744} K_p^{-1.407} \tag{22}$$

**Wall losses of particles:** The wall losses of particles are irreversible and are represented as:

$$\frac{dC_p}{dt} = -k_w^p C_p \tag{23}$$

with $C_p$ the particle phase concentrations, $k_w^p$ the wall losses kinetic of particles.

$k_w^p$ can either be directly provided by the user by defining the value of the parameter *kwall_particle* in the namelist. An alternative is to compute the wall losses of particles for each size bin. For that, the eddy diffusion coefficient $k_e$ (parameter *eddy_turbulence* in the namelist), the S/V ratio (parameter *surface_volume_ratio* in the namelist), the radius of the chamber $R_{chamber}$ (*radius_chamber* in the namelist) and the minimal wall loss rate $k_{wp0}$ (*kwp0* in the namelist in $s^{-1}$) due to electrostatic forces.
$k_w^p$ is calculated based on Pierce et al. (2008):

$$k_w^p = k_{wp0} + \frac{6\sqrt{k_e D}}{\pi R_{chamber}} D_1 \left( \frac{\pi v_s}{2\sqrt{k_e D}} \right) + \frac{v_s}{4R_{chamber}/3} \tag{24}$$

with $D_1$ is the Debye function, D the Brownian diffusivity of the particle (dependent on the aerosol diameter) and the settling velocity of the particle (dependent on the aerosol diameter).

**Test case for wall losses:** The namelist *namelist_platt_nh3_isopa_wlosses_gp.ssh* reproduces the *namelist_platt_nh3_isopa.ssh* test case but with values for wall losses of particles and organic vapors.

```
Cwall=10000,
surface_volume_ratio=2.22d0,
eddy_turbulence=0.1d0,
kwp0=8.e-5,
radius_chamber=1.,
```

The values selected have typical wall loss rate: $k_w^p$ around $10^{-4}$ $s^{-1}$ and $k_w^g$ around $3 \times 10^{-4}$ $s^{-1}$.

The namelists *namelist_platt_nh3_isopa_wlosses_g.ssh* and *namelist_platt_nh3_isopa_wlosses_p.ssh* only consider vapor and particle losses, respectively. Fig. 34 shows the effect of considering wall losses of gas and particles.

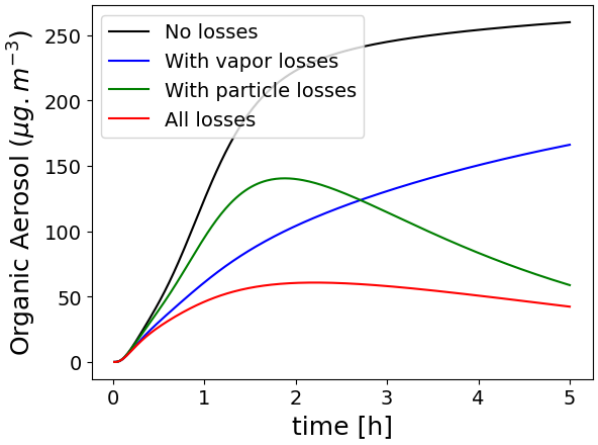


Figure 34: Effect of wall losses of organic vapors and particles in the Platt test case.

### 8.7.2 SOA formation from toluene oxidation under experimental conditions

This test case is based on the work of Lannuque et al. (2023) on toluene oxidation. The 3 simulations of the test represent the formation of SOA in an UV-lamp irradiated aerosol flow tube into which ammonium sulfate seeds and a high amount of gaseous toluene and isopropylnitrite (IPN) are injected. Toluene oxidation is here represented with the detailed mechanism of Lannuque et al. (2023), considering wall losses for stables gaseous species and irreversible condensation pathway for glyoxal and methylglyoxal. Radical equilibrium, IPN chemistry and experimental conditions representation are described in Lannuque et al. (2021). The organic compounds can here condense on both organic and aqueous phase (*partitioning* = BOTH in *species-list-aer-TOLexp.dat*) which are coupled (*coupled_phases*=1 in the namelist). The three simulations are run considering the gas-particle partitioning evolve dynamically (*ISOAPDYN*=1 in the namelist). The differences came from the interactions in the particulate phase with one ideal case (*activity_model*=1 in the namelist), one including interactions between uncharged molecules only (UNIFAC, *activity_model*=2) and one considering the previous interaction and those with inorganic ions in aqueous phase (AIOMFAC, *activity_model*=3). The simulations can be run by typing *ssh-aerosol INIT/namelist_toluene_aft_ideal.ssh*, *ssh-aerosol INIT/namelist_toluene_aft_unifac.ssh* and *ssh-aerosol INIT/namelist_toluene_aft_aiomfac.ssh*. After running the 3 simulations, SOA evolutions may be displayed by running the script *toluene-aft.py* in the *graph* repository. In the test case conditions, considering interactions between uncharged molecules enhances the condensation while considering those with inorganic ions in the aqueous phase limit it, almost offsetting first interactions. Note that results are different than in Lannuque et al. (2023) due to the consideration of wall losses and glyoxal/methylglyoxal irreversible condensation pathways and to a change in gaseous diffusivity calculation which now includes the species molar mass. The SOA concentrations simulated after about 13 min compare reasonably well to the observed one, i.e. 16 $\mu$g m$^{-3}$.

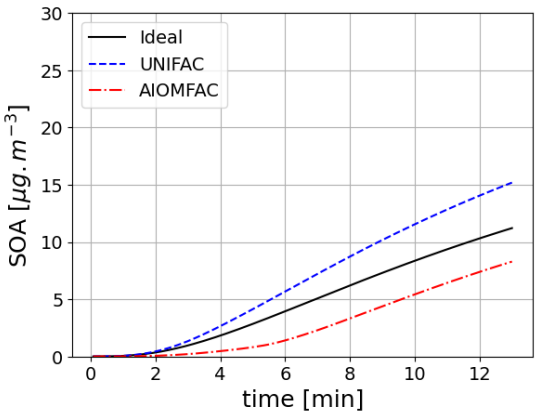


Figure 35: Time evolution of toluene SOA mass under experimental conditions taking into account or not interactions in condensed phases.

The near-explicit toluene SOA mechanism of Lannuque et al. (2023) is replaced by reduced toluene SOA mechanisms generated with GENOA Sartelet et al. (2024). Two reduced toluene SOA mechanisms are considered. The first mechanism is reduced from MCM ("SART24-1"). The second one ("SART24-2") is the reduced mechanism of Lannuque et al. (2023), including the methylglyoxal irreversible condensation pathway, to which is added the molecular rearrangement with ring opening of a bicyclic peroxy radical (BPR) with an O-O bridge (ipso-BPR, Iyer et al.

(2023)). In the second reduced mechanism, as shown in Figure 36, SOA from the ipso-BPR pathway represents 18% of the toluene SOA.

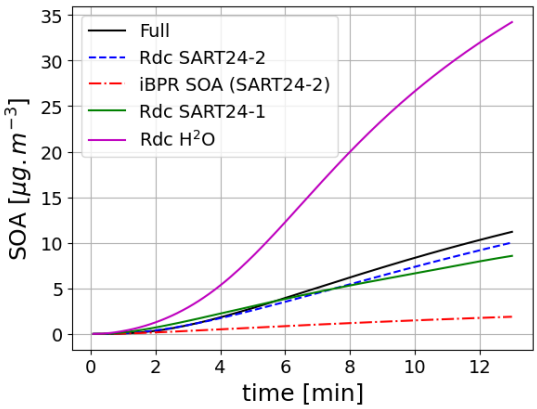


Figure 36: Time evolution of toluene SOA mass under experimental conditions with two different reduced mechanisms: the first mechanism is reduced from MCM ("SART24-1"), the second mechanism ("SART24-2") is reduced from the mechanism of Lannuque et al. (2023), including the methylglyoxal irreversible condensation pathway, to which is added the molecular rearrangement with ring opening of a bicyclic peroxy radical (BPR) with an O-O bridge (Iyer et al. (2023)).

**Note on irreversible condensation pathway of methylglyoxal and glyoxal:** Lannuque et al. (2023) and Lannuque and Sartelet (2024) detailed chemical mechanisms for toluene and naphthalene include empirical reactions to represent irreversible glyoxal and methylglyoxal condensation pathways. Under this name of irreversible pathway lies the process of aqueous phase oxidation of dicarbonyls by OH (as opposed to the classical reversible pathways impacted by hydration and isomerization). The parameterization has been greatly improved since that presented in Lannuque et al. (2023) for methylglyoxal. In concrete terms, this irreversible condensation pathway is represented by a reaction:

$$DICARB. \xrightarrow{k_{irdi}} IRDICARB. \tag{25}$$

where DICARB. is the semi-volatile species studied (glyoxal or methylglyoxal) and IRDICARB. its non-volatile counterpart via irreversible partitioning. The kinetics of this reaction follows the parameterization of Curry et al. (2018):

$$k_{irdi} = \frac{1}{4} v \times \gamma \times A_{surf} \tag{26}$$

with:

$$v = \sqrt{\frac{8RT}{\pi M_{dicarb}}} \tag{27}$$

$$\frac{1}{\gamma} = \frac{1}{\alpha} + \frac{v}{4RTH^{eff}_{dicarb}\sqrt{k^l D_{aq}}} \times \frac{1}{\coth q - 1/q} \tag{28}$$

$$q = \frac{R_p}{\sqrt{\frac{D_{aq}}{k^l}}} \tag{29}$$

fixing the aqueous-phase diffusion coefficient $D_{aq}$ to $10^{-9}$ $m^2$ $s^{-1}$, typical for small organics (Bird et al., 2006), and the mass accommodation coefficient $\alpha$ to 0.02, similar to that of formaldehyde uptake to water (Jayne et al., 1992). Henry constant of dicarb. are adapted from Sander (2015) to match with Hu et al. (2022) observations and the Henry constant of OH is adapted from Sander (2015) leading to OH aqueous concentration in agreement with typical mean value in the atmosphere from Herrmann et al. (2010). Details and description of the parameters are available in Curry et al. (2018).

### 8.7.3 SOA formation from naphthalene oxidation under experimental conditions

This test case is based on the work of Lannuque and Sartelet (2024) on naphthalene oxidation. The 6 simulations of the test represent the formation of SOA with the same UV-lamp irradiated aerosol flow tube as presented in the toluene test case (see Lannuque et al. (2023)) into which ammonium sulfate seeds and gaseous naphthalene and isopropylnitrite (IPN) are injected. Naphthalene oxidation is here represented with the detailed mechanism of Lannuque and Sartelet (2024), considering wall losses for stables gaseous species and irreversible condensation pathway for glyoxal (see section 8.7.2). Experiments are presented in detail in Lannuque et al. (2023) and Lannuque and Sartelet (2024). IPN chemistry and experimental conditions representation are described in Lannuque et al. (2021). The organic compounds can here condense on both organic and aqueous phase (*partitioning* = BOTH in *species-list-aer-NAPHexp.dat*) which are coupled (*coupled_phases*=1 in the namelist).

The simulations are run considering the gas-particle partitioning evolve dynamically (*ISOAP-DYN*=1 in the namelist). The differences came from (i) the representation of the particulate phase with 3 cases: one ideal case (*activity_model*=1 in the namelist), one considering the interactions between uncharged molecules and those with inorganic ions in the aqueous phase (AIOMFAC, *activity_model*=3) and one including interactions and considering viscosity with 5 layers in the particles and (ii) the amount of $NO_X$ during the simulation with 2 cases: with or without an initial injection of 200 ppb of $NO_2$. The simulations can be run by running *ssh-aerosol* with the following namelists in INIT repertory as arguments:

```
namelist_naphthalene_aft_ideal.ssh
namelist_naphthalene-no2_aft_ideal.ssh
namelist_naphthalene_aft_aiomfac.ssh
namelist_naphthalene-no2_aft_aiomfac.ssh
namelist_naphthalene_aft_aiomfac_visc.ssh
namelist_naphthalene-no2_aft_aiomfac_visc.ssh
```

After running the simulations, SOA evolutions may be displayed by running the script *naphthalene-aft.py* in the *graph* repository. In the test case conditions, considering the interactions and/or the viscosity in the particle limits the condensation of semi-volatile compounds as well as the initial injection of $NO_2$ leading to the formation of less SOA.

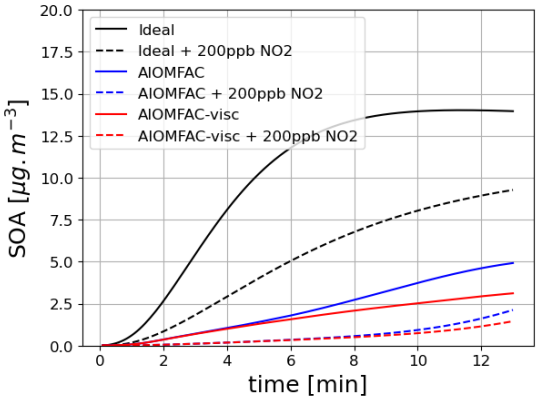


Figure 37: Time evolution of naphthalene SOA mass under different experimental conditions taking into account or not interactions and viscosity in condensed phases.

### 8.7.4 SOA formation from monoterpene oxidation

This test case is based on the work presented in Wang et al. (2023). The two simulations represent the formation of SOA in an idealized oxidative flow reactor (OFR) Xavier et al. (2019). The oxidation of $\alpha$-pinene is simulated using a reference mechanism and a reduced mechanism.

The simulations can be run after the program is compiled using the genoa flag `-g=fast`

```
./compile -g=fast
./ssh-aerosol INIT/namelist_mt_ref.ssh 1
./ssh-aerosol INIT/namelist_mt_rdc.ssh 1
```

The script `monoterpene.py` can be used to display the simulation results. An image file `monoterpene.png` is generated.

```
cd graph
python3 monoterpene.py
```

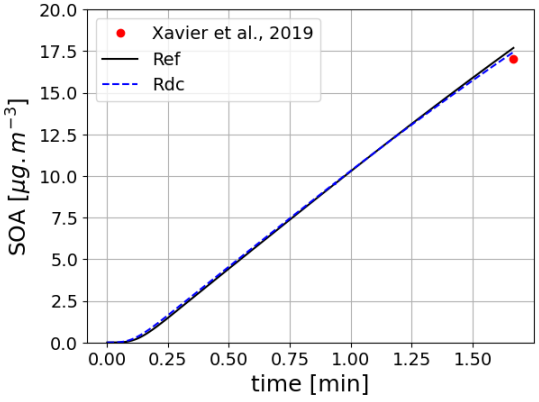


Figure 38: Time evolution of monoterpene SOA mass using the reference mechanism (`Ref`) and the reduced mechanism (`RdC`). The results are compared to the result reported in Xavier et al. (2019).

## 8.8 Coupled organic-inorganic thermodynamic module

The standard configuration of SSH-aerosol is based on two separate thermodynamic modules: SOAP and ISORROPIA. ISORROPIA is called first to compute water, pH, and inorganics. Then, SOAP is called to compute the concentrations of organics. In this system, organics do not affect the computation of pH or the concentrations of inorganics. To solve this, organics and inorganics must be solved together. This can be done by adding the flag *soap_inorg=1* (in *&physic_condensation*) in the namelist.

With *soap_inorg=1*, SOAP will be used to solve the equilibrium and the dynamics of inorganic aerosol. Organics can affect the formation of inorganics by:

1. Influencing the condensation of water, which in turn affects the absorption of HNO3 and NH3.

2. Influencing the computation of pH as the dissociation of acid organics is accounted for.

3. Influencing the computation of molalities (concentrations of ions in mol/kg). In ISORROPIA, molalities are computed as mol/kg of water. In SOAP, all solvent molecules are accounted for (water but also organics).

SOAP is based on the AIOMFAC thermodynamic module that accounts for numerous ions, $H^+$, $OH^-$, $NO_3^-$, $HSO_4^-$, $SO_4^{2-}$, $HCO_3^-$, $CO_3^{2-}$, $Cl^-$, $Na^+$, $K^+$, $Mg^{2+}$, $Ca^{2+}$.

In that mode, *ICUT* will also be used to distinguish particles at equilibrium (diameter under *ICUT*) to use an equilibrium approach from particles that evolved dynamically (diameter above *ICUT*). To solve the dynamic of inorganics with SOAP, it is recommended to use the flag *imethod=1* in *&physic_condensation* that activates implicit numerical approaches in SOAP. Those approaches are more suited for the condensation of inorganic aerosol.

The namelist *INIT/namelist namelist_platt_nh3_isopa-inorg.ssh* can be used to reproduce the *INIT/namelist namelist_platt_nh3_isopa.ssh* test case with *soap_inorg=1*. This case is illustrated in Fig. 16.

Another test case is available in *INIT/namelist_caco3-dyn-hno3.ssh*, *INIT/namelist_caco3-dyn.ssh*, *INIT/namelist_caco3-eq-hno3.ssh*, *INIT/namelist_caco3-eq.ssh*. In this case, SSH-aerosol simulates the evolution of $CaCO_3$ in the presence or in the absence of $HNO_3$ with an equilibrium approach or a dynamic approach. This case is illustrated by Fig. 39.

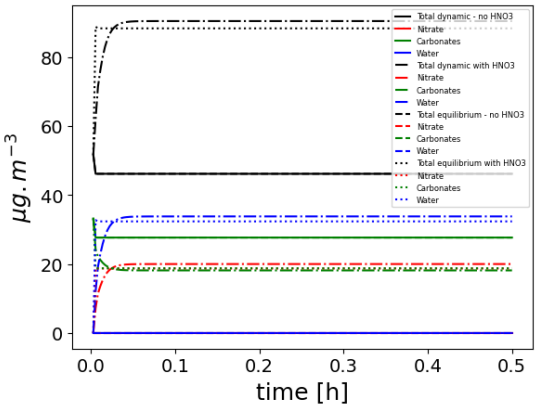


Figure 39: Time evolution of $CaCO_3$ particles in presence or in absence of $HNO_3$ with an equilibrium approach or a dynamic approach.

As shown in this figure, when $CaCO_3$ is not exposed to $HNO_3$ the particle does not evolve. Especially, it doesn't absorb water. This is due to the fact that $CaCO_3$ is currently the only particle for which delisquescence is accounted for due to its very low saturation value that would correspond to a delisquescence relative humidity close to 100%. The code considered that when [$Ca^{2+}$ (mol/kg of solvents)][$CO_3^{2-}$ (mol/kg of solvents)]>K (with K the equilibrium constant), a solid $CaCO_3$ is formed. When exposed to $HNO_3$, around 2 molecules of $NO_3^-$ will replace 1 molecule of $CO_3^{2-}$ to maintain electroneutrality. It leads to the presence of the nitrate and water inside the particle, and part of $Ca^{2+}$ that was previously associated $CO_3^{2+}$ will be transferred to the aqueous phase. Therefore, due to the condensation of $HNO_3$, the particle grew bigger in size and became hygroscopic.

# 9 FAQ

1. Q. What does the following error mean?

   ```
   scons: *** [include/Talos/_talos.so] Source file: include/Talos/talos.i is
   static and is not compatible with the shared target: include/Talos/_talos.so
   ```

   A. Library SWIG is missing. If you have already installed it, please check the path to swig in your ssh-aerosol/src/include/Talos/SConstruct

2. Q. What are the models coupled to SSH-aerosol?

   SSH-aerosol can be called by as a module inside other models using the interfaces described in section ....

   SSH-aerosol is currently coupled to:

   - Chemistry Transport Models CHIMERE (`https://www.lmd.polytechnique.fr/chimere/`) Wang et al. (2024) and Polyphemus (`https://cerea.enpc.fr/polyphemus/`) Sartelet et al. (2022)
   - CFD models Code-Saturne (`https://www.code-saturne.org/cms/web/`) and OpenFOAM (`https://www.openfoam.com/`) Lin et al. (2023)
   - Urban canopy model MUNICH `https://cerea.enpc.fr/munich/` Kim et al. (2022)

3. Q. What is new in SSH-aerosol v2.0 compared to v1.3

   New features in v2.0 include

   - Aerosol dynamics and thermodynamics
     - Consideration of the Kelvin effect to determine size distribution of condensing and evaporating species when assuming thermodynamic equilibrium
     - Faster computation of coagulation partition coefficients
     - For condensation/evaporation, computation of diffusion coefficients based on the viscosity (AIOMFAC-visc algorithm)
     - More nucleation schemes
     - Intra-particle reaction module (oligomerization, hydrolysis, reactions between organic and inorganic ions)
     - Automatic decomposition of species in UNIFAC functional groups when the SMILES code is provided
     - Coupled inorganic-organic thermodynamic module

- Gas-phase chemistry
  - Coupling to the GENerator of Reduced Organic Aerosol Mechanisms (GENOA) and Master Chemical Mechanism
  - Option to take into account peroxy radical pools for chemical mechanisms
  - Treatment of specific MCM and GECKO-A kinetic rates
  - New organic aerosol mechanisms for toluene, naphtalene, terpenes and sesquiterpenes
- Addition of a wall loss module and comparisons to chamber experiments

# References


Binkowski, F. S. and Roselle, S. J. (2003). Models-3 community multiscale air quality (cmaq) model aerosol component 1. model description. *J. Geophys. Res.: Atmospheres*, ,08(D6).

Camredon, M., Aumont, B., Lee-Taylor, J., and Madronich, S. (2007). The soa/voc/no$_x$ system: an explicit model of secondary organic aerosol formation. *Atmospheric Chemistry and Physics*, 7(21):5599–5610.

Carlton, A. G., Turpin, B. J., Altieri, K. E., Seitzinger, S. P., Mathur, R., Roselle, S. J., and Weber, R. J. (2008). Cmaq model performance enhanced when in-cloud secondary organic aerosol is included: Comparisons of organic carbon predictions with measurements. *Environ. Sci. and Technol.*, 42(23):8798–8802.

Cholakian, A., Beekmann, M., Colette, A., Coll, I., Siour, G., Sciare, J., Marchand, N., Couvidat, F., Pey, J., Gros, V., Sauvage, S., Michoud, V., Sellegri, K., Colomb, A., Sartelet, K., Langley DeWitt, H., Elser, M., Prévot, A., Szidat, S., and Dulac, F. (2018). Simulation of fine organic aerosols in the western Mediterranean area during the ChArMEx 2013 summer campaign. *Atmos. Chem. Phys.*, 18:7287–7312.

Chrit, M., Sartelet, K., Sciare, J., Pey, J., Marchand, N., Couvidat, F., Sellegri, K., and Beekmann, M. (2017). Modelling organic aerosol concentrations and properties during charmex summer campaigns of 2012 and 2013 in the western mediterranean region. *Atmos. Chem. Phys.*, 17(20):12509–12531.

Couvidat, F., Bessagnet, B., Garcia-Vivanco, M., Real, E., Menut, L., and Colette, A. (2018a). Development of an inorganic and organic aerosol model (chimere 2017$\beta$ v1.0): seasonal and spatial evaluation over europe. *Geosci. Model Dev.*, 11(1):165–194.

Couvidat, F., Debry, E., Sartelet, K., and Seigneur, C. (2012). A hydrophilic/hydrophobic organic ($H^2O$) aerosol model: Development, evaluation and sensitivity analysis. *J. Geophys. Res.: Atmospheres*, 117(D10).

Couvidat, F. and Sartelet, K. (2015). The Secondary Organic Aerosol Processor (SOAP v1. 0) model: a unified model with different ranges of complexity based on the molecular surrogate approach. *Geosci. Model Dev.*, 8(4):1111–1138.

Couvidat, F., Sartelet, K., and Seigneur, C. (2013). Investigating the impact of aqueous-phase chemistry and wet deposition on organic aerosol formation using a molecular surrogate modeling approach. *Environ. Sci. and Technol.*, 47(2):914–922.

Couvidat, F. and Seigneur, C. (2011). Modeling secondary organic aerosol formation from isoprene oxidation under dry and humid conditions. *Atmos. Chem. Phys.*, 11:893–909.

Couvidat, F., Vivanco, M. G., and Bessagnet, B. (2018b). Simulating secondary organic aerosol from anthropogenic and biogenic precursors: comparison to outdoor chamber experiments, effect of oligomerization on soa formation and reactive uptake of aldehydes. *Atmos. Chem. Phys.*, 18:15743–15766.

Curry, L., Tsui, W., and McNeill, V. (2018). Technical note: Updated parameterization of the reactive uptake of glyoxal and methylglyoxal by atmospheric aerosols and cloud droplets. *Atmos. Chem. Phys.*, 18(13):9823–9830.

DeRieux, W.-S. W., Li, Y., Lin, P., Laskin, J., Laskin, A., Bertram, A. K., Nizkorodov, S. A., and Shiraiwa, M. (2018). Predicting the glass transition temperature and viscosity of secondary organic material using molecular composition. *Atmos. Chem. Phys.*, 18:6331–6351.

Devilliers, M., Debry, E., Sartelet, K., and Seigneur, C. (2013). A new algorithm to solve condensation/evaporation for ultra fine, fine, and coarse particles. *J. Atmos. Sci.*, 55:116–136.

Eddingsaas, N. C., VanderVelde, D. G., and Wennberg, P. O. (2010). Kinetics and products of the acid-catalyzed ring-opening of atmospherically relevant butyl epoxy alcohols. *J. Phys. Chem. A*, 114(31):8106–8113.

Ervens, B. and Volkamer, R. (2010). Glyoxal processing by aerosol multiphase chemistry: towards a kinetic modeling framework of secondary organic aerosol formation in aqueous particles. *Atmospheric Chemistry and Physics*, 10(17):8219–8244.

Fredenslund, A., Jones, R., and Prausnitz, J. (1975). Group-contribution estimation of activity coefficients in nonideal liquid mixtures. *AIChE J.*, 21:1086–1099.

Gervasi, N. R., Topping, D. O., and Zuend, A. (2020). A predictive group-contribution model for the viscosity of aqueous organic aerosol. *Atmos. Chem. Phys.*, 20:2987–3008.

Herrmann, H., Hoffmann, D., Schaefer, T., Brauer, P., and Tilgner, A. (2010). Tropospheric aqueous-phase free-radical chemistry: Radical sources, spectra, reaction kinetics and prediction tools. *ChemPhysChem*, 11:3796–3822.

Hu, J., Chen, Z., Qin, X., and Dong, P. (2022). Reversible and irreversible gas–particle partitioning of dicarbonyl compounds observed in the real atmosphere. *Atmos. Chem. Phys.*, 22(10):6971–6987.

Huang, Y., Zhao, R., Charan, S. M., Kenseth, C. M., Zhang, X., and Seinfeld, J. H. (2018). Unified theory of vapor–wall mass transport in teflon-walled environmental chambers. *Environ. Sci. and Technol.*, 52(4):2134–2142.

Iyer, S., Kumar, A., Savolainen, A., Barua, S., Daub, C., Pichelstorfer, L., Roldin, P., Garmash, O., Seal, P., Kurtén, T., and Rissanen, M. (2023). Molecular rearrangement of bicyclic peroxy radicals is a key route to aerosol from aromatics. *Nature Comm.*, 14:4984.

Jenkin, M., Wyche, K., Evans, C., Carr, T., Monks, P., Alfarra, M., Barley, M., McFiggans, G., Young, J., and Rickard, A. (2012). Development and chamber evaluation of the mcm v3. 2 degradation scheme for $\beta$-caryophyllene. *Atmos. Chem. Phys.*, 12(11):5275–5308.

Jenkin, M. E., Saunders, S. M., and Pilling, M. J. (1997). The tropospheric degradation of volatile organic compounds: a protocol for mechanism development. *Atmos. Environ.*, 31(1):81–104.

Kim, Y., Lugon, L., Maison, A., Sarica, T., Roustan, Y., Valari, M., Zhang, Y., André, M., and Sartelet, K. (2022). MUNICH v2.0: a street-network model coupled with SSH-aerosol (v1.2) for multi-pollutant modelling. *Geosci. Model Dev.*, 15(19):7371–7396.

Kim, Y., Sartelet, K., and Couvidat, F. (2019). Modeling the effect of non-ideality, dynamic mass transfer and viscosity on soa formation in a 3-d air quality model. *Atmos. Chem. Phys.*, 19(2):1241–1261.

Kittelson, D., Watts, W., and Johnson, J. (2006). On-road and laboratory evaluation of combustion aerosols - part1: Summary of diesel engine results. *J. Atmos. Sci.*, 37(8):913–930.

Kuang, C., McMurry, P., McCormick, A., and Eisele, F. (2008). Dependence of nucleation rates on sulfuric acid vapor concentration in diverse atmospheric locations. *J. Geophys. Res.*, 113(D10209).

Lannuque, V., D'Anna, B., Couvidat, F., Valorso, R., and Sartelet, K. (2021). Improvement in modeling of oh and ho2 radical concentrations during toluene and xylene oxidation with racm2 using mcm/gecko-a. *Atmosphere*, 12(6):732.

Lannuque, V., D'Anna, B., Kostenidou, E., Couvidat, F., Martinez-Valiente, A., Eichler, P., Wisthaler, A., Müller, M., Temime-Roussel, B., Valorso, R., and Sartelet, K. (2023). Gas-particle partitioning of toluene oxidation products: an experimental and modeling study. *Atmos. Chem. Phys.*, 23(24):15537–15560.

Lannuque, V. and Sartelet, K. (2024). Development of a detailed gaseous oxidation scheme of naphthalene for secondary organic aerosol (soa) formation and speciation. *Atmos. Chem. Phys.*, 15(24):8589–8606.

Lemaire, V., Coll, I., Couvidat, F., Mouchel-Vallon, C., Seigneur, C., and Siour, G. (2016). Oligomer formation in the troposphere: from experimental knowledge to 3-d modeling. *Geosci. Model Dev.*, 9(4):1361–1382.

Lin, C., Wang, Y., Ooka, R., Flageul, C., Kim, Y., Kikumoto, H., Wang, Z., and Sartelet, K. (2023). Modeling of street-scale pollutant dispersion by coupled simulation of chemical reaction, aerosol dynamics, and CFD. *Atmos. Chem. Phys.*, 23(2):1421–1436.

Maclean, A. M., Smith, N. R., Li, Y., Huang, Y., Hettiyadura, A. P. S., Crescenzo, G. V., Shiraiwa, M., Laskin, A., Nizkorodov, S. A., and Bertram, A. K. (2021). Humidity-dependent viscosity of secondary organic aerosol from ozonolysis of beta-caryophyllene: Measurements, predictions, and implications. *ACS Earth and Space Chemistry*, 5(2):305–318.

Madon, R. J. and Iglesia, E. (2000). Catalytic reaction rates in thermodynamically non-ideal systems. *Journal of Molecular Catalysis A: Chemical*, 163:189–204.

Merikanto, J., Napari, I., Vehkamäki, H., Anttila, T., and Kulmala, M. (2007). New parameterization of sulfuric acid-ammonia-water ternary nucleation rates at tropospheric conditions. *J. Geophys. Res.*, 112(D15207).

Merikanto, J., Napari, I., Vehkamäki, H., Anttila, T., and Kulmala, M. (2009). Correction to "new parameterization of sulfuric acid-ammonia-water ternary nucleation rates at tropospheric conditions". *J. Geophys. Res.*, 114(D09206).

Napari, I., Noppel, M., Vehkamaki, H., and Kulmala, M. (2002). Parametrization of ternary nucleation rates for $h_2so_4$-$nh_3$-$h_2o$ vapors. *J. Geophys. Res.*, 107 (D19).

Patoulias, D., Fountoukis, C., Riipinen, I., Asmi, A., Kulmala, M., and Pandis, S. (2018). Simulation of the size-composition distribution of atmospheric nanoparticles over Europe. *Atmos. Chem. Phys.*, 18:13639–13654.

Pierce, J. R., Engelhart, G. J., Hildebrandt, L., Weitkamp, E. A., Pathak, R. K., Donahue, N. M., Robinson, A. L., Adams, P. J., and Pandis, S. N. (2008). Constraining particle evolution from wall losses, coagulation, and condensation-evaporation in smog-chamber experiments: Optimal estimation based on size distribution measurements. *Aer. Sci. and Technol.*, 42(12):1001–1015.

Platt, S. M., Haddad, I. E., Zardini, A., Clairotte, M., Astorga, C., Wolf, R., Slowik, J., Temime-Roussel, B., Marchand, N., Ježek, I., Drinovec, L., Močnik, G., Möhler, O., Richter, R., Barmet, P., Bianchi, F., Baltensperger, U., and Prévôt, A. (2013). Secondary organic aerosol formation from gasoline vehicle emissions in a new mobile environmental reaction chamber. *Atmos. Chem. Phys.*, 13(18):9141–9158.

Rahimpour, M. (2004). A non-ideal rate-based model for industrial urea thermal hydrolyser. *Chemical Engineering and Processing: Process Intensification*, 43(10):1299–1307.

Riccobono, F., Schobesberger, S., Scott, C., Dommen, J., Ortega, I., Rondo, L., Almeida, J., Amorim, A., Bianchi, F., Breitenlechner, M., David, A., Downard, A., Dunne, E., Duplissy, J., Ehrhart, S., Flagan, R., Franchin, A., Hansel, A., Junninen, H., Kajos, M., Keskinen, H., Kupc, A., Kürten, A., Kvashin, A., Laaksonen, A., Lehtipalo, K., Makhmutov, V., Mathot, S., Nieminen, T., Onnela, A., Petäjä, T., Praplan, A., Santos, F., Schallhart, S., Seinfeld, J., Sipilä, M., Spracklen, D., Stozhkov, Y., Stratmann, F., Tomé, A., Tsagkogeorgas, G., Vaattovaara, P., Viisanen, Y., Vrtala, A., Wagner, P., Weingartner, E., Wex, H., Wimmer, D., Carslaw, K., Curtius, J., Donahue, N., Kirkby, J., Kulmala, M., Worsnop, D., and Baltensperger, U. (2014). Oxidation products of biogenic emissions contribute to nucleation of atmospheric particles. *Science*, 16(344):717–721.

Sander, R. (2015). Compilation of henry's law constants (version 4.0) for water as solvent. *Atmos. Chem. Phys.*, 15(8):4399–4981.

Sartelet, K., Couvidat, F., Wang, Z., Flageul, C., and Kim, Y. (2020). SSH-Aerosol v1.1: A Modular Box Model to Simulate the Evolution of Primary and Secondary Aerosols. *Atmosphere*, 11:525.

Sartelet, K., Hayami, H., Albriet, B., and Sportisse, B. (2006). Development and preliminary validation of a modal aerosol model for tropospheric chemistry: MAM. *Aer. Sci. and Technol.*, 40(2):118–127.

Sartelet, K., Kim, Y., Couvidat, F., Merkel, M.and Petäjä, T., Sciare, J., and Wiedensohler, A. (2022). Influence of emission size distribution and nucleation on number concentrations over Greater Paris. *Atmos. Chem. Phys.*, 22:8579–8596.

Sartelet, K., Wang, Z., Lannuque, V., Iyer, S., Couvidat, F., and Sarica, T. (2024). Modelling molecular composition of SOA from toluene photo-oxidation at urban and street scales. *Environ. Sci.: Atmos.*, pages –.

Sartelet, K., Zhu, S., Moukhtar, S., André, M., André, J., Gros, V., Favez, O., Brasseur, A., and Redaelli, M. (2018). Emission of intermediate, semi and low volatile organic compounds from traffic and their impact on secondary organic aerosol concentrations over greater paris. *Atmos. Environ.*, 180:126–137.

Schmedding, R., Rasool, Q. Z., Zhang, Y., Pye, H. O. T., Zhang, H., Chen, Y., Surratt, J. D., Lopez-Hilfiker, F. D., Thornton, J. A., Goldstein, A. H., and Vizuete, W. (2020). Predicting secondary organic aerosol phase state and viscosity and its effect on multiphase chemistry in a regional-scale air quality model. *Atmos. Chem. Phys.*, 20:8201–8225.

Seigneur, C., Hudischewskyj, A. B., Seinfeld, J. H., Whitby, K. T., Whitby, E. R., Brock, J. R., and Barnes, H. M. (1986). Simulation of aerosol dynamics: A comparative review of mathematical models. *Aer. Sci. and Technol.*, 5(2):205–222.

Suarez-Bertoa, R., Mendoza-Villafuerte, P., Riccobono, F., Vojtisek, M., Pechout, M., Perujo, A., and Astorga, C. (2017). On-road measurement of NH3 emissions from gasoline and diesel passenger cars during real world driving conditions. *Atmos. Environ.*, 166:488–497.

Theloke, J. and Friedrich, R. (2007). Compilation of a database on the composition of anthropogenic voc emissions for atmospheric modeling in europe. *Atmos. Environ.*, 41(19):4148–4160.

Vehkamaki, H., Kulmala, M., Napari, I., K.E.J., L., Timmreck, C., Noppel, M., and Laaksonen, A. (2002). An improved parameterization for sulfuric acid-water nucleation rates for tropospheric and stratospheric conditions. *J. Geophys. Res.*, 107(D22):4622.

Wang, Z., Couvidat, F., and Sartelet, K. (2022). Generator of reduced organic aerosol mechanism (genoa v1. 0): an automatic generation tool of semi-explicit mechanisms. *Geosci. Model Dev.*, 15:8957–8982.

Wang, Z., Couvidat, F., and Sartelet, K. (2023). Implementation of a parallel reduction algorithm in the GENerator of reduced Organic Aerosol mechanisms (GENOA v2.0): Application to multiple monoterpene aerosol precursors. *J. Aerosol Sci.*, 174:106248.

Wang, Z., Couvidat, F., and Sartelet, K. (2024). Response of biogenic secondary organic aerosol formation to anthropogenic nox emission mitigation. *Sci. Total Environ.*, 927:172142.

Xavier, C., Rusanen, A., Zhou, P., Dean, C., Pichelstorfer, L., Roldin, P., and Boy, M. (2019). Aerosol mass yields of selected biogenic volatile organic compounds – a theoretical study with nearly explicit gas-phase chemistry. *Atmos. Chem. Phys.*, 19(22):13741–13758.

Zhang, Y., Seigneur, C., Seinfeld, J. H., Jacobson, M. Z., and Binkowski, F. S. (1999). Simulation of aerosol dynamics: A comparative review of algorithms used in air quality models. *Aer. Sci. and Technol.*, 31(6):487–514.

Zhao, Y., Nguyen, N. T., Presto, A. A., Hennigan, C. J., May, A. A., and Robinson, A. L. (2016). Intermediate volatility organic compound emissions from on-road gasoline vehicles and small off-road gasoline engines. *Environ. Sci. and Technol.*, 50(8):4554–4563.

Zhu, S., Sartelet, K. N., and Seigneur, C. (2015). A size-composition resolved aerosol model for simulating the dynamics of externally mixed particles: SCRAM (v 1.0). *Geosci. Model Dev.*, 8(6):1595–1612.

Zuend, A., Marcolli, C., Luo, B., and Peter, T. (2008). A thermodynamic model of mixed organic-inorganic aerosols to predict activity coefficients. *Atmos. Chem. Phys.*, 8:4559–4593.