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`;
Youngseob Kim, `youngseob.kim@enpc.fr`;
Zhizhao Wang, `zhizhao.wang@enpc.fr`;
Florian Couvidat, `florian.couvidat@ineris.fr`.

ssh-aerosol version: 0.7

Contents

1 Introduction

The ssh-aerosol model is 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 [?] 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 [?] is a thermodynamic model that compute 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 [?] 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 [?] 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 [?]).

ssh-aerosol is a free software. You can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation.

Hardware and software requirement

ssh-aerosol is written in the programming language FORTRAN and C++. It can run on PC or a cluster with both the gfortran and the GUN gcc Compiler and under a Linux system. Before the compilation, make sure the construction tool: SCONS has already been installed. If not you can obtain it through the following instruction of the site below: http://www.scons.org/wiki/SconsTutorial1

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

The NetCDF library may be required if you have precomputed coagulation repartition coefficients, you can download from the following site: http://www.unidata.ucar.edu/downloads/netcdf/index.jsp After all the required software and library are ready, the compilation can be done by typing a simple command: compile, within a terminal under the program main path.

2 Folder structure

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

2.1 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: the scons folder contains the files necessary for compiling the program and the include folder contains the source code, which is itself separated in different folders

- Module contains SCRAM subroutines to model aerosol dynamics
- SOAP contains SOAP subroutines to model organic aerosol thermodynamic
- CHEMISTRY contains H^2O subroutines to model gas-phase chemistry. These routines are generated from a list of reactions and species specified in the folder spack.
- spack contains the gas-phase chemical model generator.
- 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.

2.2 Input files

The main configuration file for the program ssh-aerosol is namelist.ssh. It requires several input data:

- The list of species and their properties. They are in the repertory species-list.
- The initial concentrations and emissions, which are in the repertory inputs
- The configuration files are in the repertory INIT

2.3 Output files

The simulation results are stored in the folder results. The folder graph contains a few python routines to postprocess the results and display. During the simulation, five subfolders and a report file are automatically generated/updated in the folder results:

- 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 list) is adopted as the file name (ex.*HNO3.txt*).
  In each file, a total of N + 1 rows of data are listed (N represents the number of iterations). The first line in the file records the initial concentration before the simulation, while line i (i = 2, 3,..., N + 1) records the gas phase mass concentration after i-1 time step in the simulation. All the concentration files files described in the folder results follow the same output format.
- Subfolder aero : this subfolder contains files that record the time variation of particulate mass concentration ($\mu g/m^3$) of each species in each size section. The file is named after the species name plusing the index of the size section (index = 1, 2,..., N_sizebin). For example, file *PNO3_1.txt* notes the time variation of nitrate particulate mass concentration in the first size section.
  The files that record the time variation of organic particles, inorganic particles, black carbon, dust, $PM_{2.5}$ and PM_{10} 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, PM2.5.txt* and *PM10.txt*, respectively.

- Subfolder TM : this subfolder contains files that record the time variation of total mass concentration ($\mu g/m^3$) and totoal aerosol mass concentration of all size sections ($\mu g/m^3$). The files is named after the aerosol species name, (plusing the name of its gas phase percursor for recording total mass) and plusing '_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 ($\#/m^3$) of each size section. The file (ex.*NUMBER_1.txt*) is named after 'NUMBER_' plusing the index of size section.
  The file *TNUM.txt* that records the time variation of total number concentration ($\#/m^3$) is located in this subfolder as well.

- Subfolder diameter : this subfolder contains files that record the time variation of the average diameter (μm) of each size section. The file (ex.*DIAMETER_1.txt*) is named after 'DIAMETER_' plusing the index of size section.

- Report file 'report.txt' : it records the main settings of the simulation.

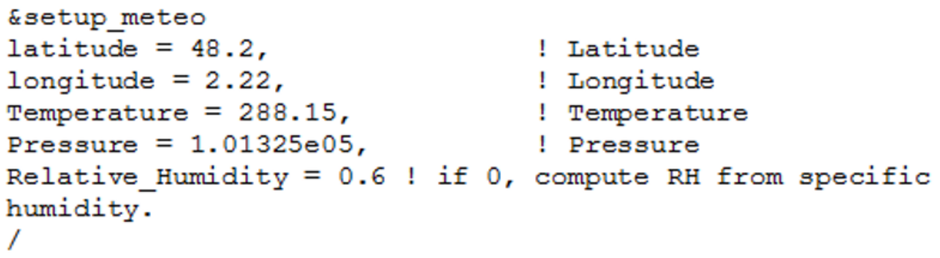
The user can also modify the name of the output folder (output_directory) or select the output file type (output_type = 1 for text outputs and = 2 for binary outputs) in the file *namelist.ssh.*

3 Main options

The different options are listed in the file namelist.ssh. They are grouped in different parts.

3.1 Meteorology

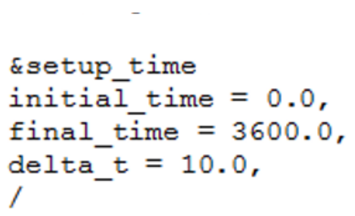
Input data concerning latitude (in degrees), longitude (in degrees), Temperature (in Kelvin), Pressure (in Pascal) and Relative Humidity (fraction) are listed in the group setup_meteo.



```
&setup_meteo
latitude = 48.2,                    ! Latitude
longitude = 2.22,                   ! Longitude
Temperature = 288.15,               ! Temperature
Pressure = 1.01325e05,              ! Pressure
Relative_Humidity = 0.6 ! if 0, compute RH from specific
humidity.
/
```

3.2 Time

The group setup_time lists the initial time of the simulation (in seconds from 1st January), the final time (in seconds from 1st 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,
final_time = 3600.0,
delta_t = 10.0,
/
```

3.3 Initial conditions

The group initial_condition lists the initial conditions of the simulation. The number of size section 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 μg m^{-3}, and the unit for number concentration is #particles m^{-3}.

```
&initial_condition
with_init_num = 0,                  ! 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 !! option 1 not yet available)
tag_dbd = 1,                        ! Method for defining particle
                                    ! size bounds (0 if they are
                                    ! generated, 1 if bounds are
                                    !read)
N_sizebin = 50,                     ! Number of size bin
init_gas_conc_file = "inputs/init_gas.dat",
init_aero_conc_mass_file = "inputs/init_aero.dat",
init_aero_conc_num_file = "inputs/init_num.dat",
/
```

```
&initial_diam_distribution
diam_input = 1.000000000000000E-03     1.202264434617413OE-03
1.445439770745928OE-03  1.737800828749376OE-03
2.089296130854039 9E-03  2.511886431509579 9E-03
3.019951720402017OE-03  3.630780547701014OE-03
4.365158322401660 1E-03  5.248074602497727 2E-03
6.309573444801933 7E-03  7.585775750291837 7E-03
9.120108393559098 5E-03  1.096478196143185 1E-02
1.318256738556406 9E-02  1.584893192461114 1E-02
1.905460717963248OE-02  2.290867652767774 1E-02
2.754228703338168 1E-02  3.311311214825911 3E-02
3.981071705534972 7E-02  4.786300923226385 9E-02
5.754399373371568 7E-02  6.918309709189365 8E-02
8.317637711026712 5E-02  1.000000000000000 1E-01
1.202264434617414OE-01  1.445439770745927 9E-01
1.737800828749375 1E-01  2.089296130854040OE-01
2.511886431509581 2E-01  3.019951720402017 1E-01
3.630780547701015 2E-01  4.365158322401662 1E-01
5.248074602497729 8E-01  6.309573444801938OE-01
7.585775750291844 4E-01  9.120108393559098 7E-01
1.096478196143186OE+00  1.318256738556407OE+00
1.584893192461114OE+00  1.905460717963249OE+00
2.290867652767774 9E+00  2.754228703338168 9E+00
3.311311214825912 1E+00  3.981071705534977 1E+00
4.786300923226384 9E+00  5.754399373371576 6E+00
6.918309709189369 3E+00  8.317637711026709OE+00
1.000000000000001 1E+01
/
```

3.4 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 discretized. It is set to 1 for internal mixing, because the composition of compounds is then not discretized. In 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
/
```

3.5 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. Note that the order of the species in this file should not be changed. It is set by the preprocessor of gas-phase chemical schemes.

```
&gas_phase_species
species_list_file = "./species-list/species-list-cb05en.dat"
/
```

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 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), accomodation coefficient (between 0 and 1) and density in μg μm^3. The categories to which species correspond are also listed. They should not be modified and they must correspond to those set in the routine ModuleInitialisation.f90 of ssh-aerosol.

```
&aerosol_species
aerosol_species_list_file = "./species-list/species-list-aer-
en.dat"
mineral_dust = "PMD",
black_carbon = "PBC", ! The number of species in each category
must correspond to those set in ModuleInitialisation
isorropia_species_name = "PNA", "PSO4", "PNH4", "PNO3","PHCL",
aec_species_name = "PBiA2D", "PBiA1D", "PBiA0D", "PAGLY",
"PAMGLY","PBiMT","PBiPER", "PBiDER", "PBiMGA", "PAnBlP",
"PAnBmP", "PBiBlP", "PBiBmP","PBiNGA", "PBiNIT3", "PBiNIT",
"PBiA3D", "PMonomer", "PDimer"
pankow_species_name = "PAnClP",
poa_species_name = "PSOAlP", "PSOAmP", "PSOAhP", "PPOAlP",
"PPOAmP", "PPOAhP"
/
```

3.6 Emissions

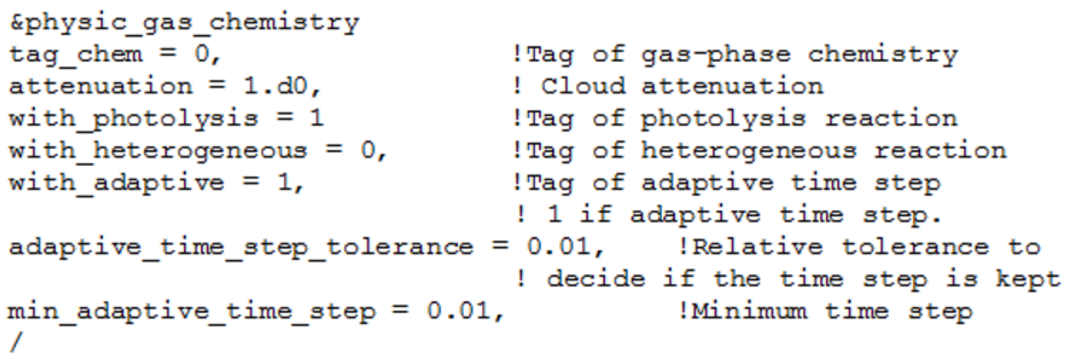
The group emissions defines options linked to emissions. The variable tag_emis defines whether emissions are used (tag_emis = 1) or not (tag_emis = 0). Emissions are assumed to be internally mixed. Gas-phase emissions and/or particle-phase emissions can be specified. Number concentrations at emission 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 μg 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 = 1,          ! 0 Without emissions,
                       ! 1 with internally-mixed emissions
with_emis_num = 0,     ! 0 estimated from mass and diameter;
                       ! 1  read
emis_gas_file = "./inputs/emis_gas.dat",
emis_aero_mass_file = "./inputsemis_aero.dat",
emis_aero_num_file = "./inputs/emis_aero_num.dat"
/
```

3.7 Numerical and physical options

3.7.1 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.



```
&physic_gas_chemistry
tag_chem = 0,                   !Tag of gas-phase chemistry
attenuation = 1.d0,             ! Cloud attenuation
with_photolysis = 1             !Tag of photolysis reaction
with_heterogeneous = 0,         !Tag of heterogeneous reaction
with_adaptive = 1,              !Tag of adaptive time step
                                ! 1 if adaptive time step.
adaptive_time_step_tolerance = 0.01,    !Relative tolerance to
                                ! decide if the time step is kept
min_adaptive_time_step = 0.01,          !Minimum time step
/
```

3.7.2 Numerical issues related to aerosols

The group physic_particle_numerical_issues lists options related to numerical issues when solving aerosol dynamics. The variable DTAEROMIN specfies 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 μg μm^{-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 recommanded to set splitting to 1.

```
&physic_particle_numerical_issues
DTAEROMIN = 1E-5,               !Minimum time step
redistribution_method = 12,     !Redistibution methods
                                 !0 without redistribution
with_fixed_density = 1,         !1 if density is fixed - 0 else
fixed_density = 1.56D-06,
splitting = 1,              ! 0 if coagulation and
(condensation/evaporation+nucleation) are splitted - 1 if they
are not
/
```

3.7.3 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). Rapartition 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 discretization 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
Coefficient_file = "coef_s1_f1_b6.nc",  ! Repart coefficient file
Nmc = 2000                ! Number of Monte Carlo points to compute
                          ! repartition coefficients.
/
```

3.7.4 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 recommanded 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 larger diameter to assume thermodynamic equilibrium. For the condensation/evaporation of organic compounds, 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, but condensation/evaporation of all particles is solved dynamically if (ISOAPDYN = 1) and tequilibrium is set to a small value (e.g. 1.d-15). In the current version of the code, the diffusion coefficient dorg in the organic phase is assumed constant. Typical values would be 1.d-12 for non viscous particles and 1.d-23 for very viscous particles. The variable coupled_phases should be set to 1 if the resolution os 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).

```
&physic_condensation
with_cond = 1,                ! Tag of condensation/evaporation
Cut_dim = 0.0,                ! Diameter under which equilibrium
                              ! is assumed for inorganics
ISOAPDYN = 0,                 ! 0 = equilibrium, 1 = dynamic
nlayer = 1,
with_kelvin_effect = 1,       ! 1 if kelvin effect account.
tequilibrium = 1.d-15,        ! time under which equilibrium
                              ! is assumed for organics.
dorg = 1.d-12,                ! diffusion coefficient
                              ! in the organic phase.
coupled_phases = 1,           ! 1 if aqueous and organic phases
                              ! are coupled
activity_model = 1,           ! 1. "ideal", 2. "unifac",
                              ! 3. "aiomfac"
/
```

3.7.5 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. Three nucleation models are implemented.

- binary: water and sulfate with the parameterisation of [?] (nucl_model = 0)
- ternary: water, sulfate and ammonium with the parameterisation of [?] (nucl_model = 1) or with the parameterisation of [?, ?] (nucl_model = 2). To avoid artificially large nucleation rates in the parameterisation of [?], a maximum nucleation rate of 1.d6 #particles cm^{-3} is set.

```
&physic_nucleation
with_nucl = 1,          !Tag of nucleation
                        !Need to have lower diameter about 1nm.
nucl_model = 1,         !Nucleation model (0 = binary of Vekhamaki,
                   !1= ternary of Napari, 2= ternary of Merikanto)
/
```

3.7.6 Organics

Concerning organic reactions in the particles, oligomerization of pinonaldehyde may be considered (with_oligomerization = 1) or not.

```
&physic_organic
with_oligomerization = 1
/
```

3.8 Output

The output directory may be specified, as well as the format of the files (text if output_type = 1, and binary if output_type = 2).

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

3.9 Coupling with external tools

SSHaerosol can be coupled with external (3D) tools using the shared library (`ssh-aerosol.so`) available in the `src` folder after compilation. A prototype of the typical workflow is described hereafter.

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;
}
```

Initialization

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 wether SSHaerosol 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, SSHaerosol 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);
}
```

Time advancement

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

- Sets the time step in SSHaerosol using `api_set_sshaerosol_dt`
- Sets the Pressure, Temperature, pH, ... in SSHaerosol using `api_set_sshaerosol_temperature` and similar functions
- Sets the gaseous concentrations in SSHaerosol using `api_set_sshaerosol_gas_concentration`
- Sets the aerosol concentrations in SSHaerosol using `api_set_sshaerosol_aero_concentration`
- Sets the aerosol numbers in SSHaerosol using `api_set_sshaerosol_aero_number`
- Computes one time 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 SSHaerosol using the `api_get_*` functions and use them in the external code

Finalization

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);
```

Parallelism

If the external tool is running with MPI, each MPI process 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`).

4 Test cases

This section presents a few test-cases to demonstrate how the model works.

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-en.dat* and *species-list/species-list-cb05en*. 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.

4.1 Dynamic of coagulation and condensation

In the literature, to test the overall behavior of PM models, the following basic tests of condensation and coagulation of sulfate are often considered [?, ?, ?]. 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 sulfuric acid production rate of 9.9 μg m^{-3}, and those of urban conditions for the coagulation test [?, ?] , 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 [?] obtained with another models.

4.1.1 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 1.

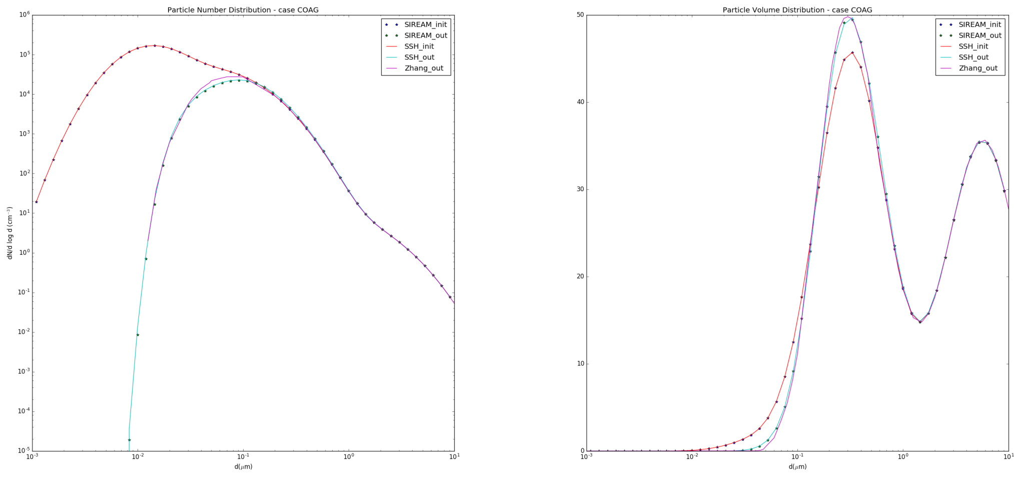


Figure 1: Coagulation test case. Number (left panel) and volume (right panel) concentrations.

4.1.2 Condensation of sulfate

The condensation test with hazy conditions is very stringent, because the high condensation rate lead 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 2.

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 3). 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 [**?**] , 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 accomodation coefficient in the file *species-list/species-list-aer-en.dat*.

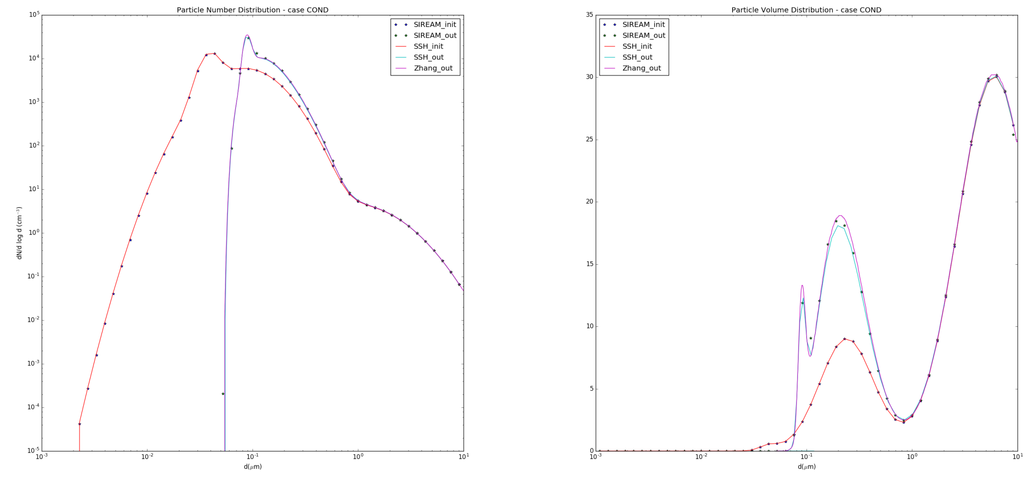


Figure 2: Condensation test case without redistribution. Number (left panel) and volume (right panel) concentrations.

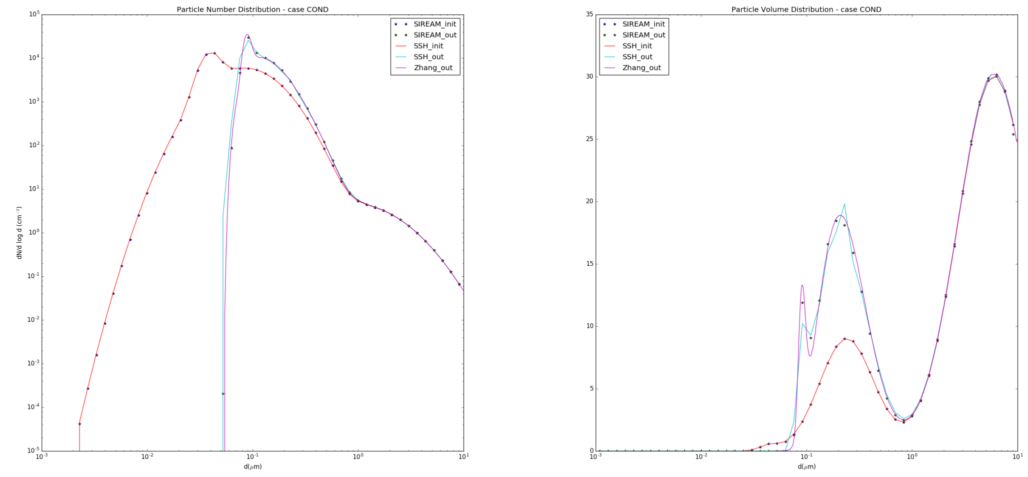


Figure 3: Condensation test case with Euler-coupled redistribution. Number (left panel) and volume (right panel) concentrations.

4.1.3 Condensation of low-volatility organics

Extremely-low volatility organic compounds (ELVOCs) are formed from the ozonolysis of monoterpenes [?] . Similarly to sulfate, these compounds have a very low saturation vapor 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-monomer/species-list-aer-en.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.

4.1.4 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 [**?**] . This test case simulates the test case of the highly polluted day of 25 June 2001 of [**?**] . Measurements of PM and gaseous species made in Tokyo (Japan) are taken as initial conditions. Because the data were averaged continuously during 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 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.*

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 4, the gas-phase concentrations quickly reach thermodynamic equilibrium.

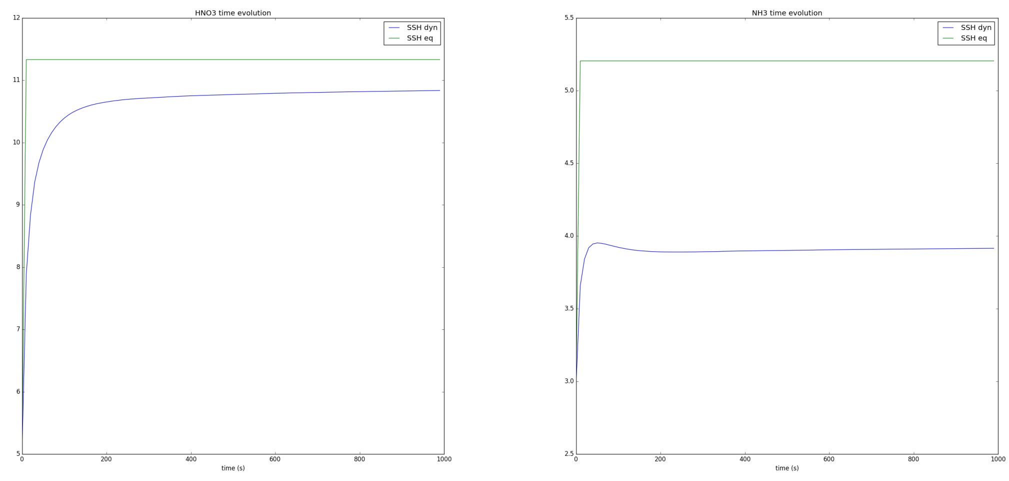


Figure 4: Condensation/évaporation of inorganic test case. Time evolution of HNO_3 concentrations (left panel) and NH_3 concentrations (right panel).

4.1.5 Kelvin effect

To illustrate the importance of the Kelvin effect for the growth of ultrafine particles, the test case of [**?**] concerning the growth of ultrafine particles emitted from the exhaust of a diesel engine was simulated. As in [?], a typical diesel engine emission initial distribution from [?] is used here to study the gas/particle conversion of nonadecane (C19H40). It has a reference saturation vapor 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 5 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.

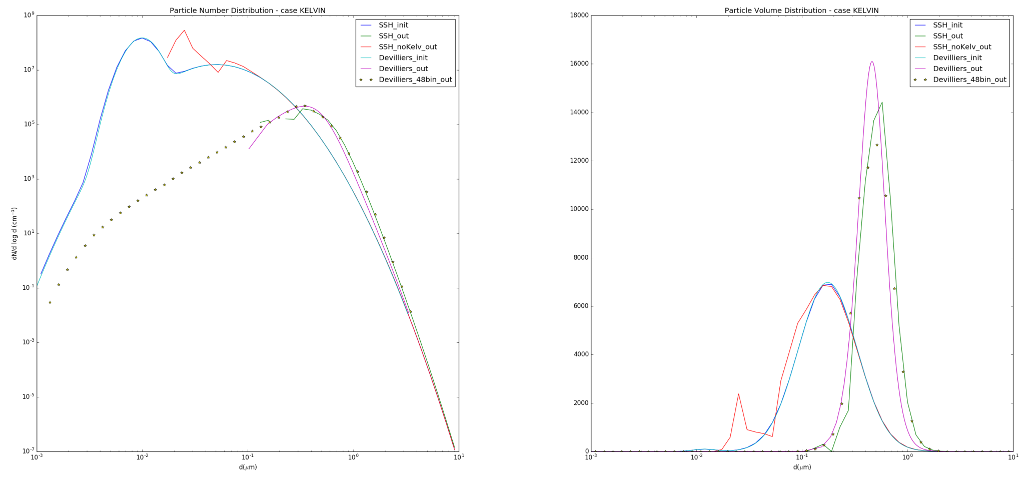


Figure 5: Kelvin test case. Number (left panel) and volume (right panel) concentrations.

4.1.6 Nucleation

To assess the ability of ssh-aerosol to deal with simultaneous strong coagulation and condensation/nucleation, the nucleation test case presented in [?] is simulated. The initial distribution is the same as in the condensation of sulfuric acid test case of section 4.1.2. It corresponds to the hazy conditions of [?]. The sulfuric acid production rate is 0.825 μg 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 μg 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 [?] is used, but to avoid artificially large nucleation rates in the parameterisation of [?], 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 splitted 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, except for particles of diameter below 2 nm, which are over-estimated when coagulation is splitted from condensation/evaporation. For such small particles, the combined effects of coagulation, nucleation and condensation is important, as shown in Fig 6.

The nucleated particles clearly grow to larger particles with time, as can be seen both in Fig 7 and by running the python script *graph/banana.py*.

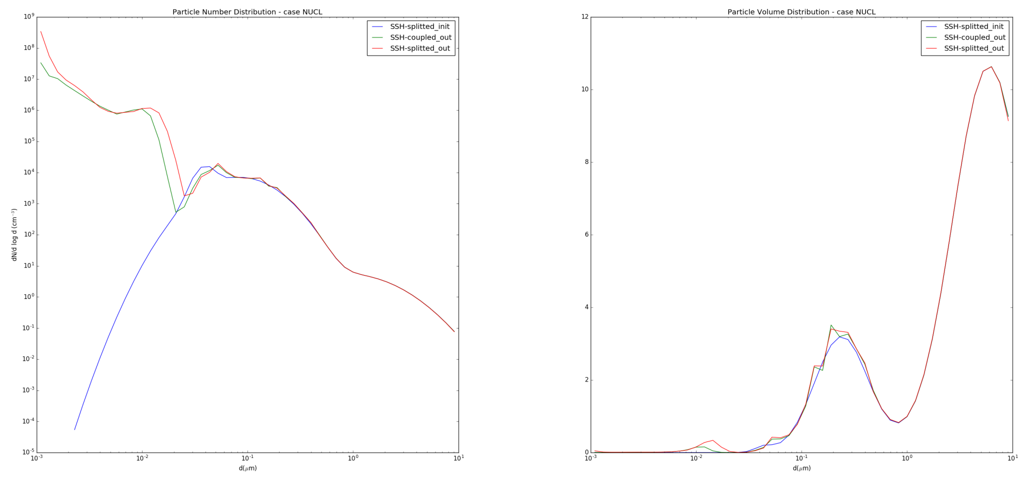


Figure 6: Nucleation test case. Number (left panel) and volume (right panel) concentrations.

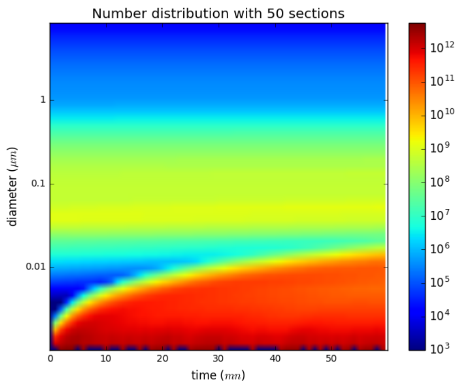


Figure 7: Nucleation test case. Time evolution of the number concentrations.

4.2 Modelling of aerosol 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.

[?] 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 [?].

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

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). Photolysis rates are predefined in this version of ssh, and a more accurate representation of those (similar to what is used in 3D) will be implemented soon.

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 origins from the condensation/evaporation of aged I/S VOCs, as shown in Fig 8. The inorganic concentrations stay low (5 $\mu g\ m^{-3}$ of sulfate was introduced initially as a seed and does not vary during the simulation; and nitrate stays below 9 $\mu g\ m^{-3}$).

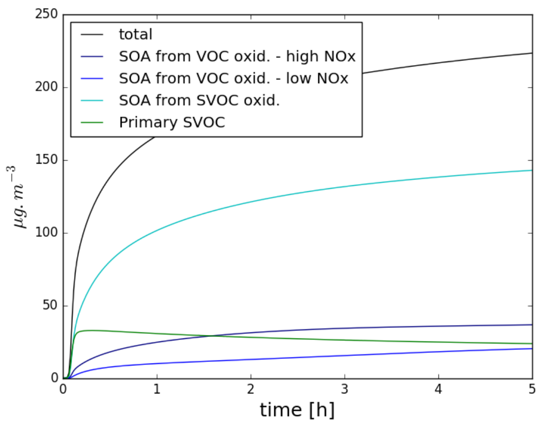


Figure 8: Platt test case. Time evolution of organic concentrations.

4.3 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.

4.3.1 Coagulation

To illustrate how coagulation affects the mixing state of particles, the coagulation test case (see section 4.1.1) is revisited by assuming that the initial aerosol distribution is made of two low-volatility compounds of 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 9). The evolution of the mixing-state of particles after 12 h of coagulation may be seen by comparing the two panels in Fig 10, which shows the number concentrations as a function of the size and fraction of one sulfate, which is one of the two compounds of the particles.

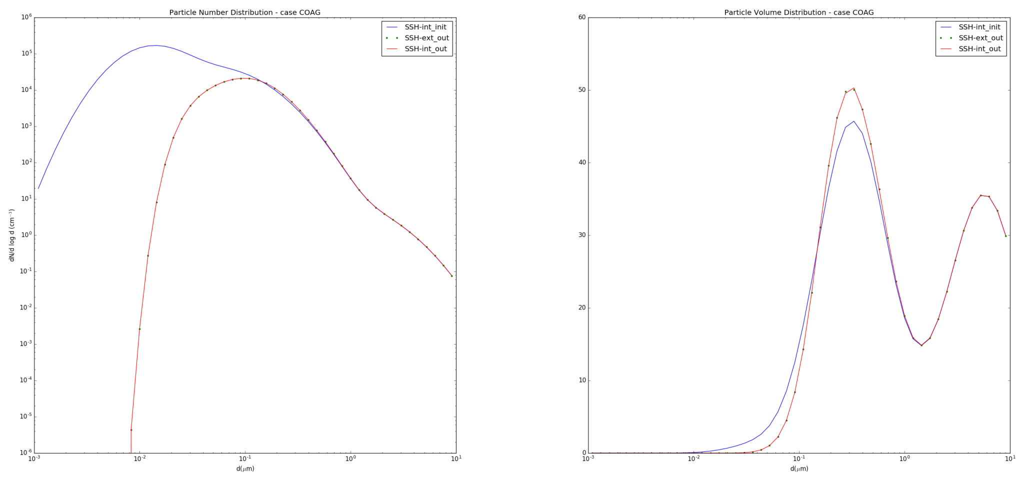


Figure 9: Coagulation test case with mixing-state modelling. Number (left panel) and volume (right panel) concentrations.

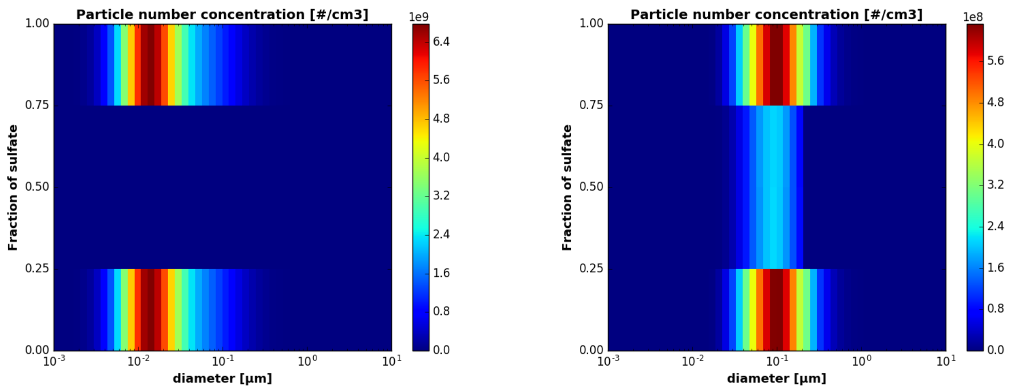
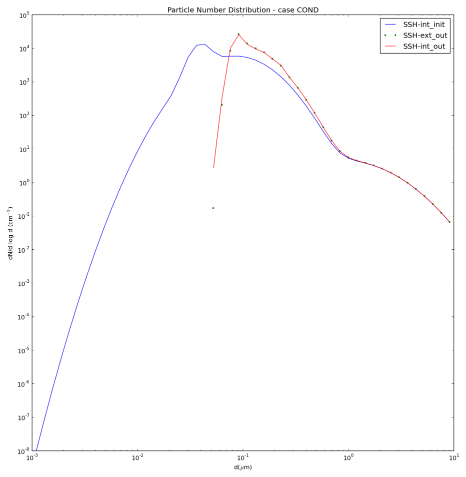


Figure 10: Number concentrations as a function of the size and fraction of one of the two compounds at initial time (left panel) and after 12 h of simulation (right panel).

4.3.2 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 4.1.2) by assuming that the initial aerosol distribution is made of two low-volatility compounds of same density (sulfate and another low-volatility compound). As in [?], 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 11). 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 12, which shows the mass concentrations as a function of the size and fraction of one sulfate, which is one of the two compounds of the particles. Sulfuric 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 12).



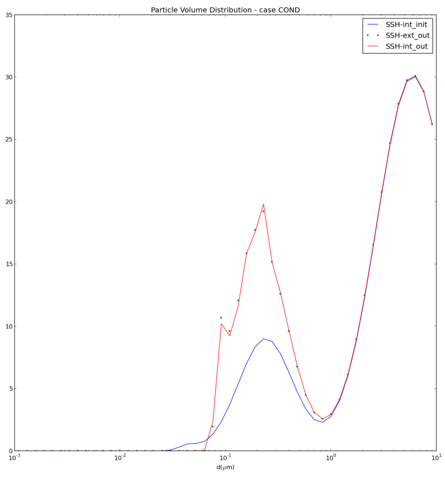
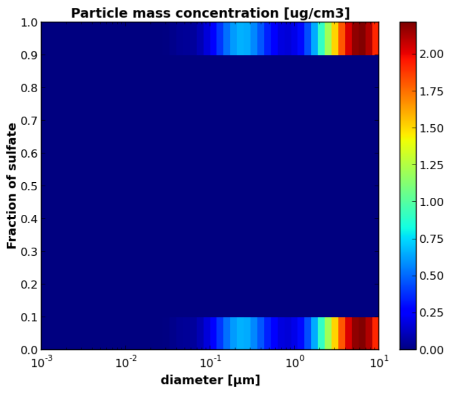


Figure 11: Regional haze test case with mixing-state modelling. Number (left panel) and volume (right panel) concentrations.



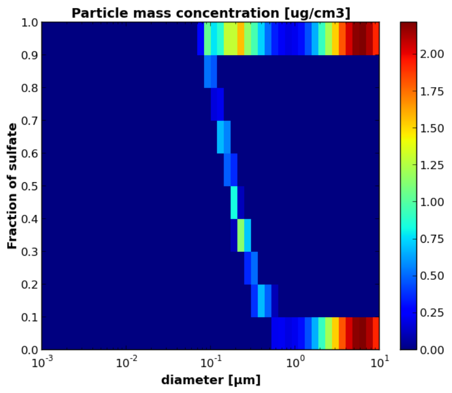


Figure 12: Mass concentrations as a function of the size and fraction of one of the two compounds at initial time (left panel) and after 12 h of simulation (right panel).