

The Aerosol Model Benchmarking Repository: a toolkit for model intercomparison

December 2025

Laura Fierce, Pacific Northwest National Laboratory (PNNL)
Jeffrey Johnson, Cohere Consulting LLC
Duncan Quevedo, University of California, Berkeley
Benjamin Murphy, EMEP MSC-W, Norwegian Meteorological Institute
Nicole Riemer, University of Illinois at Urbana–Champaign
Matthew Dawson, Cohere Consulting LLC



U.S. DEPARTMENT
of **ENERGY**

Prepared for the U.S. Department of Energy
under Contract DE-AC05-76RL01830

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor Battelle Memorial Institute, nor any of their employees, makes **any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights.** Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof, or Battelle Memorial Institute. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

PACIFIC NORTHWEST NATIONAL LABORATORY
operated by
BATTELLE
for the
UNITED STATES DEPARTMENT OF ENERGY
under Contract DE-AC05-76RL01830

Printed in the United States of America

Available to DOE and DOE contractors from
the Office of Scientific and Technical Information,
P.O. Box 62, Oak Ridge, TN 37831-0062

www.osti.gov

ph: (865) 576-8401

fox: (865) 576-5728

email: reports@osti.gov

Available to the public from the National Technical Information Service
5301 Shawnee Rd., Alexandria, VA 22312

ph: (800) 553-NTIS (6847)

or (703) 605-6000

email: info@ntis.gov

Online ordering: <http://www.ntis.gov>

The Aerosol Model Benchmarking Repository: a toolkit for model intercomparison

December 2025

Laura Fierce, Pacific Northwest National Laboratory (PNNL)
Jeffrey Johnson, Cohere Consulting LLC
Duncan Quevedo, University of California, Berkeley Benjamin
Murphy, EMEP MSC-W, Norwegian Meteorological Institute
Nicole Riemer, University of Illinois at Urbana–Champaign
Matthew Dawson, Cohere Consulting LLC

Prepared for
the U.S. Department of Energy
under Contract DE-AC05-76RL01830

Pacific Northwest National Laboratory
Richland, Washington 99354

Abstract

The Aerosol Model Benchmarking Repository and Standards (AMBRS) project was initiated to provide tools and to establish community standards for benchmarking aerosol models. This report describes a set of open-source tools for building, running, and analyzing aerosol box model simulations in a standardized framework. The framework consists of three core components: **AMBuilder**, a CMake-based build system that compiles supported models consistently; **AMBRS**, a Python module that defines unified numerical experiments and executes them with aligned inputs; and **part2pop**, an aerosol analysis package that standardizes output, computes diagnostics, and visualizes simulation results. Together, these tools enable reproducible intercomparison of aerosol schemes and support process-level evaluation of how model simplifications affect predictions of size distributions, cloud condensation nuclei activity, and other relevant properties relevant for the Earth-Energy system. Beyond its role in benchmarking, AMBRS provides a platform for studying aerosol processes across scales and can be used to generate training data for AI/ML applications in support of a broader hierarchical aerosol modeling strategy.

Summary

The AMBRS framework lowers the barrier for systematic evaluation of aerosol models by providing an integrated workflow:

- **Build:** AMBuilder compiles aerosol box models with consistent options and layout, ensuring reproducibility.
- **Run:** AMBRS executes unified ensembles of scenarios across models, harmonizing inputs and outputs.
- **Analyze:** part2pop interprets results, producing diagnostics and figures that allow direct comparison.

The tools are modular, extensible, and designed for community use. While AMBuilder and AMBRS were developed under this project, part2pop was originally created under other DOE-supported efforts and is broadly applicable beyond benchmarking. Together, these components provide a foundation for quantifying structural uncertainty, informing the development of next-generation aerosol models, and producing standardized datasets suitable for AI/ML model training.

Acknowledgments

This research was supported by the Earth and Biological Science Mission Seed Investment under the Laboratory Directed Research and Development (LDRD) Program at Pacific Northwest National Laboratory (PNNL). PNNL is a multi-program national laboratory operated for the U.S. Department of Energy (DOE) by Battelle Memorial Institute under Contract No. DE-AC05-76RL01830.

Acronyms and Abbreviations

AMBRs – Aerosol Model Benchmarking Repository and Standards
AMBuilder – Aerosol Model Builder (CMake-based build system in AMBRs)
CCN – Cloud Condensation Nuclei
CAMP – Chemistry Across Multiple Phases
DOE – U.S. Department of Energy
E3SM – Energy Exascale Earth System Model
MAM4 – Modal Aerosol Model with four modes
MOSAIC – Model for Simulating Aerosol Interactions and Chemistry
PartMC – Particle-resolved Monte Carlo model
PNNL – Pacific Northwest National Laboratory
PPE – Perturbed Parameter Ensemble
RH – Relative Humidity

Contents

Abstract..... ii

Summary.....iii

Acknowledgments..... iv

Acronyms and Abbreviations v

1.0 Introduction 1

2.0 AMBuilder (Build) 2

3.0 AMBRS (Run) 3

4.0 part2pop (Analyze)..... 7

5.0 Next steps 10

6.0 Code availability 11

7.0 References..... 12

Figures

Figure 1. Size distribution comparison across models and time..... 6

Figure 2. CCN activation fraction as a function of supersaturation, corresponding to the
size distributions in Fig. 1 9

Tables

Table 1. AMBRS components and their roles in the build–run–analyze workflow..... 1

1.0 Introduction

Large-scale aerosol effects depend on particle-level characteristics, such as size, shape, and chemical composition, and there are a wide variety of approaches for representing aerosol in models (e.g., Stier et al. 2005; Liu et al. 2012; Pierce and Adams 2009; Bauer et al. 2008; Mann et al. 2014; Matsui et al. 2018; Tegen et al. 2019). In atmospheric sciences, global intercomparisons, such as AeroCom phase II, reveal a broad spread in aerosol burden and microphysics even under harmonized inputs (Mann et al. 2014; Tegen et al. 2019). However, these top-down intercomparison experiments cannot easily disentangle structural deficiencies in the aerosol representation from other uncertainties, such as meteorology and emissions. The few studies that have quantified structural uncertainty in aerosol representations have shown that differences in model assumptions about particle size-distributions can strongly affect aerosol radiative properties and cloud condensation nuclei activity (e.g., Ching et al. 2012; Fierce et al. 2016; Fierce et al. 2020; Matsui et al. 2018; Zheng et al. 2021), but studies have been isolated to idealized comparisons between a small number of model approximations. As a path forward, we introduce a toolkit to facilitate aerosol model benchmarking, intercomparison, and development.

The Aerosol Model Benchmarking Repository and Standards (AMBRS) project provides an integrated set of tools for building aerosol box models, running unified numerical experiments, and analyzing model output in a standardized framework. The goal is to make it straightforward to intercompare parameterizations across models and quantify structural uncertainty in reduced aerosol models.

This report documents the current state of the AMBRS tools, focusing on three main components:

1. AMBuilder — build system for consistent model compilation.
2. AMBRS — driver for running experiments and managing ensembles.
3. part2pop — analysis package for harmonized diagnostics and visualization.

Each component can be used on its own, but they are designed to interoperate so that models are built consistently, simulations are defined and executed comparably, and outputs feed a common analysis workflow. An overview of the AMBRS components is provided in Table 1; individual components and example use cases are provided in the remainder of this report.

Table 1. AMBRS components and their roles in the build–run–analyze workflow.

	Description	Primary Outputs
AMBuilder	CMake-based system that compiles and installs supported aerosol box models.	Standardized executables for each box model.
AMBRS	Python module that defines and executes unified numerical experiments across models with harmonized inputs.	Standardized model outputs with aligned variables, units, and metadata.
part2pop	General-purpose Python package for aerosol population analysis; linked here as the benchmarking analysis layer.	Derived diagnostics (size distributions, CCN, optics, freezing) and plots.

2.0 AMBuilder (Build)

AMBuilder is a CMake-based build system developed to provide a consistent and automated way to configure and compile aerosol box models. It manages compiler settings, dependencies, and chemistry options, ensuring that models are built with the same layout and can be executed in a standardized environment. This eliminates the ad hoc and error-prone process of manually compiling models, allowing researchers to focus on scientific experiments rather than build logistics. It also maximizes model compatibility with each scenario available in AMBRS. By design, AMBuilder enforces a directory layout and versioned installation scheme, making it easier to track which executables were used in benchmarking studies.

Although AMBuilder is currently configured to support CARMA, MAM4, and PartMC, its modular structure makes it straightforward to add new models or extend options, for example by enabling CAMP chemistry or including MOSAIC. The system is designed for long-term extensibility and reproducibility, providing a foundation for all simulations carried out with AMBRS.

Typical usage:

```
cmake -S . -B build -DCMAKE_INSTALL_PREFIX=<prefix> [options]
cd build
make install
```

Install Python dependencies for AMBRS:

```
pip install -r requirements.txt
```

3.0 AMBRS (Run)

The AMBRS Python module provides the layer for defining and executing numerical experiments in a consistent way across different aerosol box models. It is the central driver of the benchmarking framework, translating shared experiment definitions into model-specific inputs and collecting outputs in a unified format. Experiments are defined in terms of species, processes, size distributions, and environmental conditions, which are then sampled to create perturbed parameter ensembles or single-scenario test cases. Once an ensemble is defined, AMBRS manages directories, input files, and parallel execution of each model. Outputs are collected into standardized files with aligned variables, units, and metadata, ensuring direct comparability across models. Currently, AMBRS supports single-box model experiments with PartMC and MAM4, but its structure is designed to be extensible: new models can be added through input/output adapters, and new experiment templates can be introduced as the scope of benchmarking grows. This component ensures that benchmarking is not only reproducible but also scalable to a larger library of models and scenarios.

An example for using the AMBRS driver is provided below, and the evolution of the size distribution for a similar example is shown in Fig. 1.

Simulation configuration (example):

```
n = 5                # ensemble members
n_part = 20000       # PartMC particles per run
dt = 60              # timestep [s]
nstep = 60           # 60 min run
p0 = 101325          # reference pressure [Pa]
h0 = 500             # reference height [m]
```

Species definitions (examples):

```
# aerosol species
so4 = ambrs.AerosolSpecies("SO4", molar_mass=96., density=1800.,
hygroscopicity=0.65)
pom = ambrs.AerosolSpecies("OC", molar_mass=12.01, density=1000.,
hygroscopicity=0.001)
soa = ambrs.AerosolSpecies("MSA", molar_mass=40., density=2600.,
hygroscopicity=0.53)
bc = ambrs.AerosolSpecies("BC", molar_mass=12.01, density=1800.,
hygroscopicity=0.0)
dst = ambrs.AerosolSpecies("OIN", molar_mass=135.065, density=2600.,
hygroscopicity=0.1)
na = ambrs.AerosolSpecies("Na", molar_mass=23., density=2200.,
hygroscopicity=0.53)
cl = ambrs.AerosolSpecies("Cl", molar_mass=35.5, density=2200.,
hygroscopicity=0.53)
ncl = na
h2o = ambrs.AerosolSpecies("H2O", molar_mass=18., density=1000.,
ions_in_soln=1)

# gas species
so2 = ambrs.GasSpecies("SO2", molar_mass=64.07)
h2so4 = ambrs.GasSpecies("H2SO4", molar_mass=98.079)
```

Processes:

```
processes = ambrs.AerosolProcesses(coagulation=True, condensation=True)
```

Ensemble specification:

```

spec = ambrs.EnsembleSpecification(
    name=ensemble_name,
    aerosols=(so4, pom, soa, bc, dst, ncl, h2o),
    gases=(so2, h2so4),
    size=ambrs.AerosolModalSizeDistribution(modes=[
        ambrs.AerosolModeDistribution(
            name="accumulation",
            species=[so4, pom, soa, bc, dst, ncl],
            number=stats.uniform(1e7, 1e10),
            geom_mean_diam=stats.rv_discrete(values=([1.1e-7], [1.])),
            log10_geom_std_dev=log10(1.6),
            mass_fractions=[
                stats.rv_discrete(values=([1.], [1.])),
                stats.rv_discrete(values=([0.], [1.])),
                stats.rv_discrete(values=([0.], [1.])),
                stats.rv_discrete(values=([0.], [1.])),
                stats.rv_discrete(values=([0.], [1.])),
                stats.rv_discrete(values=([0.], [1.])),
                stats.rv_discrete(values=([0.], [1.])),
            ],
        ),
        ambrs.AerosolModeDistribution(
            name="aitken",
            species=[so4, soa, ncl],
            number=stats.uniform(1e7, 1e11),
            geom_mean_diam=stats.rv_discrete(values=([2.6e-8], [1.])),
            log10_geom_std_dev=log10(1.6),
            mass_fractions=[
                stats.rv_discrete(values=([1.], [1.])),
                stats.rv_discrete(values=([0.], [1.])),
                stats.rv_discrete(values=([0.], [1.])),
            ],
        ),
        ambrs.AerosolModeDistribution(
            name="coarse",
            species=[dst, ncl, so4, bc, pom, soa],
            number=stats.uniform(1e6, 1e7),
            geom_mean_diam=stats.rv_discrete(values=([2e-6], [1.])),
            log10_geom_std_dev=log10(1.8),
            mass_fractions=[
                stats.rv_discrete(values=([0.], [1.])),
                stats.rv_discrete(values=([0.], [1.])),
                stats.rv_discrete(values=([1.], [1.])), # SO4
                stats.rv_discrete(values=([0.], [1.])),
                stats.rv_discrete(values=([0.], [1.])),
                stats.rv_discrete(values=([0.], [1.])),
            ],
        ),
        ambrs.AerosolModeDistribution(
            name="primary carbon",
            species=[pom, bc],
            number=stats.rv_discrete(values=([0.], [1.])),
            geom_mean_diam=stats.loguniform(1e-8, 5e-8),
            log10_geom_std_dev=log10(1.8),
            mass_fractions=[
                stats.rv_discrete(values=([1.], [1.])), # POM
                stats.rv_discrete(values=([0.], [1.])), # BC
            ],
        ),
    ],
)

```

```

        ],
    ),
    ],
    gas_concs=tuple([stats.uniform(1e-10, 1e-8) for _ in range(2)]),
    flux=stats.uniform(1e-11, 1e-8),
    relative_humidity=stats.uniform(0, 0.99),
    temperature=stats.uniform(240, 310),
    pressure=p0,
    height=h0,
)

```

Create ensemble:

```

ensemble = ambrs.lhs(specification=spec, n=n)
scenario_names = [str(ii).zfill(1) for ii in range(1, len(ensemble.flux)+1)]

```

Running models:

```

# MAM4
mam4 = ambrs.mam4.AerosolModel(processes=processes)
mam4_inputs = mam4.create_inputs(ensemble=ensemble, dt=dt, nstep=nstep)
mam4_runner = ambrs.PoolRunner(model=mam4, executable="mam4", root=mam4_dir,
num_processes=1)
mam4_runner.run(mam4_inputs)

# PartMC
partmc = ambrs.partmc.AerosolModel(processes=processes, run_type="particle",
n_part=n_part, n_repeat=1)
partmc_inputs = partmc.create_inputs(ensemble=ensemble, dt=dt, nstep=nstep)
partmc_runner = ambrs.PoolRunner(model=partmc, executable="partmc",
root=partmc_dir, num_processes=1)
partmc_runner.run(partmc_inputs)

```

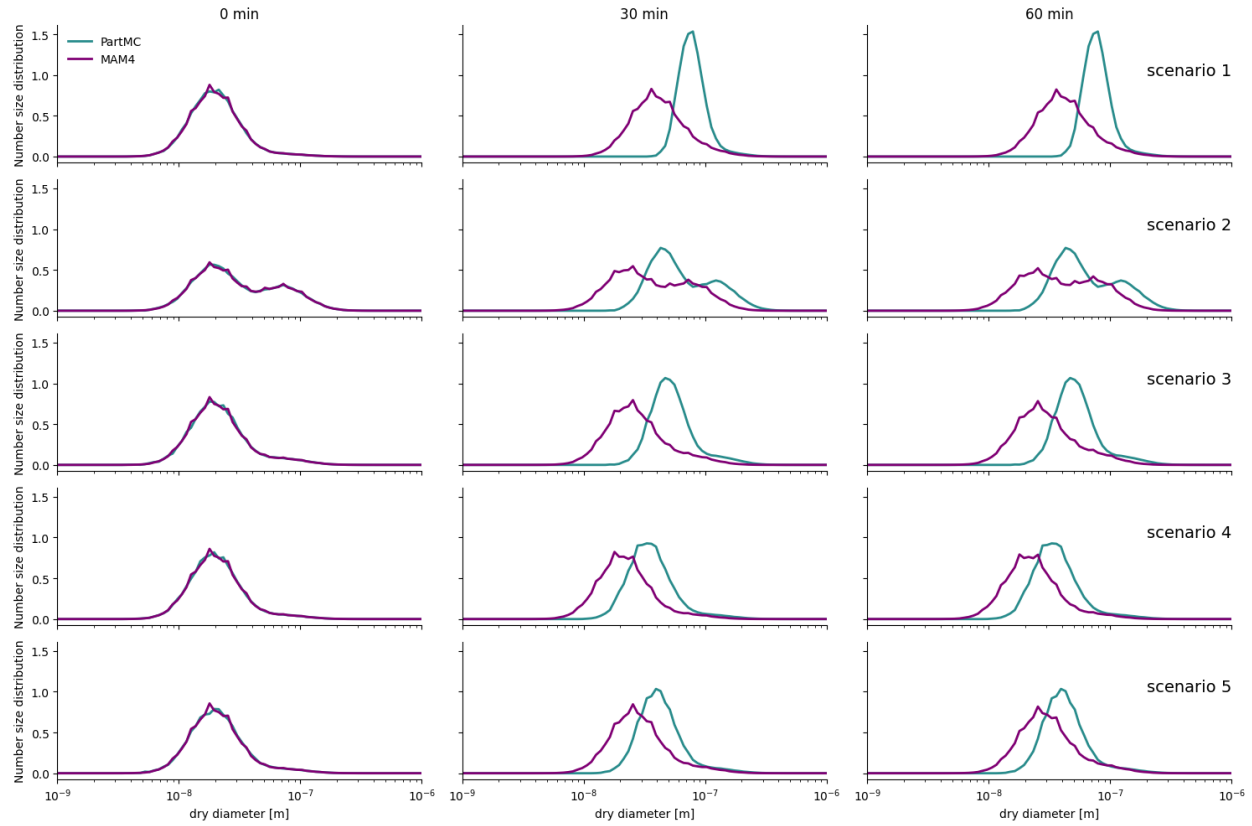


Figure 1. Size distribution comparison across models and time.
 Normalized aerosol size distributions (dN/dlnD) for PartMC and MAM4 across ensemble scenarios (rows) and timesteps (columns: 0, 30, 60 min).

4.0 part2pop (Analyze)

part2pop is a Python analysis package that provides standardized diagnostics and visualization of aerosol populations. It was originally developed as a general-purpose tool under other project funding to support a wide range of aerosol science applications, including optics, cloud condensation nuclei (CCN), and freezing diagnostics. Under the AMBRs project, part2pop has been linked as the analysis layer for benchmarking experiments, but its scope extends well beyond this application. Within AMBRs, part2pop provides a consistent way to build particle populations from model outputs, compute derived quantities, and generate figures for direct comparisons. For example, the part2pop package provides size distributions, CCN activity, or mixing state indices across models and visualizes the results. Because it was designed for flexibility and extensibility, new diagnostics and visualization modules can be added without disrupting existing workflows, making it well-suited to support both AMBRs and other aerosol modeling efforts.

Example: plot $dN/d\ln D$ (Fig. 1) and CCN activation fraction (Fig. 2) from AMBRs outputs.

```
import numpy as np
import matplotlib.pyplot as plt
from part2pop.viz.style import StyleManager, Theme
from part2pop.viz.builder import build_plotter

timesteps = [1, 31, 61]
scenarios = scenario_names

# --- Figure 1: dN/dlnD -----
---
fig, ax = plt.subplots(len(scenarios), len(timesteps), figsize=(15, 10),
                        squeeze=False, sharex=True, sharey=True)

for i, scenario_name in enumerate(scenarios):
    for j, t in enumerate(timesteps):
        partmc_output = ambrs.partmc.retrieve_model_state(
            scenario_name=scenario_name,
            scenario=ensemble.member(int(scenario_name) - 1),
            timestep=t, repeat_num=1, species_modifications={},
            ensemble_output_dir=partmc_dir,
        )
        mam4_output = ambrs.mam4.retrieve_model_state(
            scenario_name=scenario_name,
            scenario=ensemble.member(int(scenario_name) - 1),
            timestep=t, repeat_num=1, species_modifications={},
            ensemble_output_dir=mam4_dir,
        )

        mgr = StyleManager(Theme(), deterministic=False)
        series = [
            {"key": "partmc", "population":
partmc_output.particle_population, "label": "PartMC"},
            {"key": "mam4", "population": mam4_output.particle_population,
"label": "MAM4"},
        ]
        line_styles = mgr.plan("line", [s["key"] for s in series])
        line_styles["partmc"]["color"] = "#298c8c"
```

```

line_styles["mam4"]["color"]    = "#800074"

# Shared diameter grid (SI units)
N_bins = 50
D_min, D_max = 1e-9, 1e-6
D_grid = np.logspace(np.log10(D_min), np.log10(D_max), N_bins)

for s in series:
    cfg = {
        "varname": "dNdlnd",
        "var_cfg": {"D": D_grid, "normalize": True},
        "style": line_styles[s["key"]],
    }
    plotter = build_plotter("state_line", cfg)
    plotter.plot(s["population"], ax[i, j], label=s["label"])

ax[i, j].set_xscale('log')
ax[i, j].spines['top'].set_visible(False)
ax[i, j].spines['right'].set_visible(False)
if i == 0: ax[i, j].set_title(f'{t-1} min')
if i == 0 and j == 0: ax[i, j].legend(frameon=False, loc='upper
left')
    if j == len(timesteps) - 1: ax[i, j].text(2e-7, 1., f'scenario
{scenario_name}', fontsize=12)
    if j > 0: ax[i, j].set_ylabel('')
    if i < len(scenarios) - 1: ax[i, j].set_xlabel('')

# --- Figure 2: CCN activation fraction -----
--
fig, ax = plt.subplots(len(scenarios), len(timesteps), figsize=(15, 10),
                        squeeze=False, sharex=True, sharey=True)

for i, scenario_name in enumerate(scenarios):
    for j, t in enumerate(timesteps):
        partmc_output = ambrs.partmc.retrieve_model_state(
            scenario_name=scenario_name,
            scenario=ensemble.member(int(scenario_name) - 1),
            timestep=t, repeat_num=1, species_modifications={},
            ensemble_output_dir=partmc_dir,
        )
        mam4_output = ambrs.mam4.retrieve_model_state(
            scenario_name=scenario_name,
            scenario=ensemble.member(int(scenario_name) - 1),
            timestep=t, repeat_num=1, species_modifications={},
            ensemble_output_dir=mam4_dir,
        )

        mgr = StyleManager(Theme(), deterministic=False)
        series = [
            {"key": "partmc", "population":
partmc_output.particle_population, "label": "PartMC"},
            {"key": "mam4", "population": mam4_output.particle_population,
"label": "MAM4"},
        ]
        line_styles = mgr.plan("line", [s["key"] for s in series])
        line_styles["partmc"]["color"] = "#298c8c"
        line_styles["mam4"]["color"]    = "#800074"

```

```

for s in series:
    cfg = {
        "varname": "frac_ccn",
        "var_cfg": {"s_grid": np.logspace(-2, 1., 50)},
        "style": line_styles[s["key"]],
    }
    plotter = build_plotter("state_line", cfg)
    plotter.plot(s["population"], ax[i, j], label=s["label"])

    ax[i, j].set_xscale('log')
    ax[i, j].spines['top'].set_visible(False)
    ax[i, j].spines['right'].set_visible(False)
    if i == 0: ax[i, j].set_title(f'{t-1} min')
    if i == 0 and j == 0: ax[i, j].legend(frameon=False, loc='upper
left')
    if j == len(timesteps) - 1: ax[i, j].text(1.5, 0.2, f'scenario
{scenario_name}', fontsize=12)
    if j > 0: ax[i, j].set_ylabel('')
    if i < len(scenarios) - 1: ax[i, j].set_xlabel('')

```

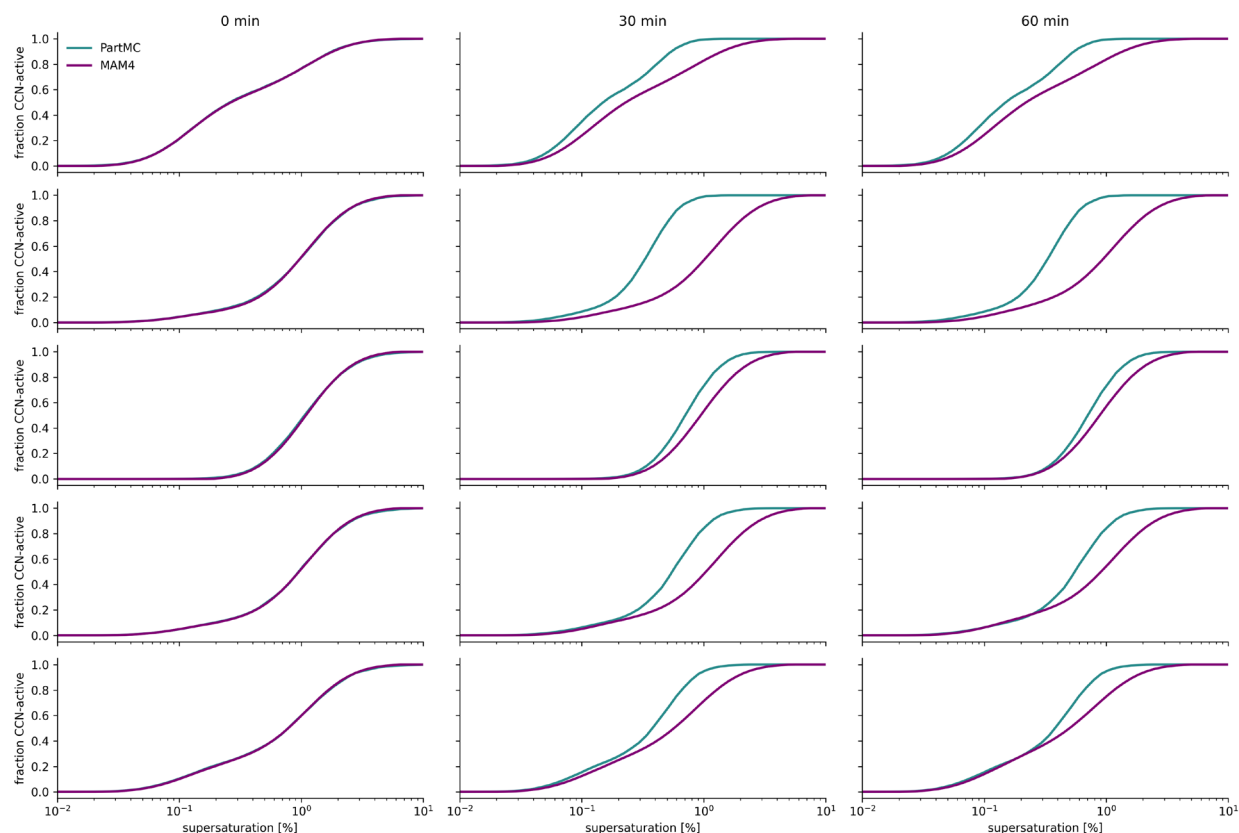


Figure 2. CCN activation fraction as a function of supersaturation, corresponding to the size distributions in Fig. 1

Fraction of particles activating as CCN for PartMC and MAM4 across ensemble scenarios (rows) and timesteps (columns: 0, 30, 60 min). Figure was generated from AMBRS model output using part2pop.

5.0 Next steps

The AMBRS workflow will continue to evolve as a resource for the Department of Energy's broader mission in Energy-Earth System Modeling. AMBRS provides a hierarchical modeling framework that links high-fidelity particle-resolved simulations with the simplified aerosol models, enabling systematic evaluation of how model approximations impact model predictions of aerosol properties relevant to the Energy-Earth system.

Future work will extend AMBRS in several directions. Through engagement with the aerosol community, we will expand the library of supported models, test cases, and simulated processes. By connecting outputs to observational datasets, AMBRS can also help bridge verification and validation layers in DOE's hierarchical modeling strategy.

Another key opportunity is the use of AMBRS to generate standardized datasets for artificial intelligence and machine learning (AI/ML). Ensembles of simulations spanning wide parameter ranges can be used to train surrogate models and emulators, accelerating model development and enabling reduced-cost exploration of aerosol–climate interactions. This capability will support DOE's long-term vision for integrating AI/ML with high-performance computing to advance predictive Earth system science.

Taken together, these next steps position AMBRS as more than a benchmarking platform: it is a tool for accelerating development of the next generation of aerosol models, strengthening DOE's Energy–Earth System Modeling portfolio, and enabling new science at the interface of multiscale modeling, high-fidelity simulations, and AI/ML.

6.0 Code availability

All AMBRS components (AMBuilder, AMBRS driver, and part2pop) are available under permissive, OSI-approved licenses (e.g., BSD-3-Clause or MIT). Source code is maintained in public repositories with issue tracking and open contribution via pull requests.

- AMBuilder: <https://github.com/AMBRS-project/ambuilder>
- AMBRS driver: <https://github.com/AMBRS-project/ambrs>
- part2pop: <https://github.com/pnnl/part2pop>

7.0 References

- Bauer, S. E., D. L. Wright, D. Koch, E. R. Lewis, R. McGraw, L-S. Chang, S. E. Schwartz, and R. Ruedy. "MATRIX (Multiconfiguration Aerosol Tracker of mIXing state): an aerosol microphysical module for global atmospheric models." *Atmospheric Chemistry and Physics* 8, no. 20 (2008): 6003-6035.
- Ching, J., N. Riemer, and M. West. "Impacts of black carbon mixing state on black carbon nucleation scavenging: Insights from a particle-resolved model." *Journal of Geophysical Research: Atmospheres* 117, no. D23 (2012).
- Fierce, L., T. C. Bond, S. E. Bauer, F. Mena, and N. Riemer. 2016. "Black carbon absorption at the global scale is affected by particle-scale diversity in composition." *Nature communications* 7, no. 1 (2016): 12361.
- Fierce, L., T. B. Onasch, C. D. Cappa, C. Mazzoleni, S. China, J. Bhandari, P. Davidovits, D. A. Fischer, T. J. Helgestad, A. T. Lambe, A. J. Sedlacek III, G. D. Smith, and L. Wolff. "Radiative absorption enhancements by black carbon controlled by particle-to-particle heterogeneity in composition." *Proceedings of the National Academy of Sciences* 117, no. 10 (2020): 5196-5203.
- Liu, X., P-L. Ma, H. Wang, S. Tilmes, B. Singh, R. C. Easter, S. J. Ghan, and P. J. Rasch. "Description and evaluation of a new four-mode version of the Modal Aerosol Module (MAM4) within version 5.3 of the Community Atmosphere Model." *Geoscientific Model Development* 9, no. 2 (2016): 505-522.
- Mann, G. W., K. S. Carslaw, C. L. Reddington, K. J. Pringle, M. Schulz, A. Asmi, H. Tost, et al. 2014. "Intercomparison and evaluation of global aerosol microphysical properties among AeroCom models of a range of complexity." *Atmospheric Chemistry and Physics* 14, no. 9 (2014): 4679-4713.
- Matsui, H., M. Koike, Y. Kondo, N. Moteki, J. D. Fast, and R. A. Zaveri. 2013. "Development and validation of a black carbon mixing state resolved three-dimensional model: Aging processes and radiative impact." *Journal of Geophysical Research: Atmospheres* 118, no. 5 (2013): 2304-2326.
- Pierce, J. R., and P. J. Adams. "A computationally efficient aerosol nucleation/condensation method: Pseudo-steady-state sulfuric acid." *Aerosol Science and Technology* 43, no. 3 (2009): 216-226.
- Sartelet, K., F. Couvidat, Z. Wang, C. Flageul, and Y. Kim. "SSH-aerosol v1. 1: A modular box model to simulate the evolution of primary and secondary aerosols." *Atmosphere* 11, no. 5 (2020): 525.
- Stier, P., J. Feichter, S. Kinne, S. Kloster, E. Vignati, J. Wilson, L. Ganzeveld, et al. "The aerosol-climate model ECHAM5-HAM." *Atmospheric Chemistry and Physics* 5, no. 4 (2005): 1125-1156.
- Zheng, Z., M. West, L. Zhao, P-L. Ma, X. Liu, and N. Riemer. "Quantifying the structural uncertainty of the aerosol mixing state representation in a modal model." *Atmospheric Chemistry and Physics* 21, no. 23 (2021): 17727-17741.

Pacific Northwest National Laboratory

902 Battelle Boulevard
P.O. Box 999
Richland, WA 99354

1-888-375-PNNL (7665)

www.pnnl.gov