














# New developments in PySDM and PySDM-examples v2: collisional breakup, immersion freezing, dry aerosol initialization, and adaptive time-stepping

Emily K. de Jong <sup>1</sup>, Clare E. Singer <sup>2</sup>, Sajjad Azimi <sup>2</sup>, Piotr Bartman <sup>3</sup>, Oleksii Bulenok <sup>3</sup>, Kacper Derlatka <sup>3</sup>, Isabella Dula<sup>2</sup>, Anna Jaruga <sup>2</sup>, J. Ben Mackay <sup>2,4</sup>, Ryan X. Ward <sup>2</sup>, and Sylwester Arabas <sup>3,5</sup>

<sup>1</sup> Department of Mechanical and Civil Engineering, California Institute of Technology, Pasadena, CA, United States of America <sup>2</sup> Department of Environmental Science and Engineering, California Institute of Technology, Pasadena, CA, United States of America <sup>3</sup> Faculty of Mathematics and Computer Science, Jagiellonian University, Kraków, Poland <sup>4</sup> Scripps Institution of Oceanography, San Diego, CA, United States of America <sup>5</sup> Department of Atmospheric Sciences, University of Illinois at Urbana-Champaign, Urbana, IL, United States of America

DOI: [10.21105/joss.04968](https://doi.org/10.21105/joss.04968)

## Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [David Hagan](#)  

## Reviewers:

- [@douglowe](#)
- [@emmasimp](#)

Submitted: 19 May 2022

Published: 19 April 2023

## License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](https://creativecommons.org/licenses/by/4.0/)).

## Summary

PySDM and the accompanying PySDM-examples packages are open-source modeling tools for computational studies of atmospheric clouds, aerosols, and precipitation. The project hinges on the particle-based microphysics modeling approach and Pythonic code design. The eponymous SDM refers to the Super Droplet Method – a Monte-Carlo algorithm introduced in Shima et al. (2009) to represent the coagulation of particles in modeling frameworks such as Large-Eddy Simulations (LES) of atmospheric flows. Recent efforts have culminated in the “v2” release line, which includes representation of a variety of new processes for both liquid and ice-phase particles, performance enhancements such as adaptive time-stepping, as well as a broadened suite of examples which demonstrate, test, and motivate the use of the SDM for cloud modeling research.

## Background and Statement of Need

The key motivation behind development of PySDM has been to offer the community an approachable, readily reusable software for users and developers who wish to contribute to the scientific progress of particle-based methods for simulating atmospheric clouds. To this end, we strive to maintain modularity of the PySDM building blocks, separation of functionality and examples, and extensive unit test coverage in the project. A user of the package can select top-level options such as the simulation environment, particle processes, and output attributes without a detailed grasp of the CPU and GPU backend code.

PySDM “v1” featured representation of the following processes: condensational growth/evaporation, collisional growth, aqueous sulfur chemistry, and coupling of particle transport and vapor/heat budget with grid-discretized fluid flow. This paper outlines subsequent developments in the “v2” releases of PySDM including representation of three new processes (collisional breakup, immersion freezing, and surface-partitioning of organic aerosol components), initialization framework for aerosol size and composition, enhanced support for adaptive time-stepping, and additional illustrative examples.

In the companion PySDM-examples package, we continue to expand and maintain a set of examples demonstrating project features through automated reproduction of results from

literature. The examples package serves multiple roles in the project. First, it guides users and developers through the package features. Second, PySDM-examples has been used as educational material, offering interactive Jupyter notebooks suitable for hands-on demonstrations of basic cloud-physics simulations. Third, inclusion of simulation scripts/notebooks pertaining to new research papers can streamline assessment of the results by reviewers. Running simulations described in a paper can be done independently on a cloud-computing platform such as Google Colab or mybinder.org. Finally, we require new examples introduced into PySDM-examples to be accompanied by a set of “smoke tests” in PySDM, which assert results against reference data to ensure that published results remain reproducible with future developments of PySDM.

## Summary of new features and examples in v2

For an example of running basic zero-dimensional simulations with PySDM, we refer to the project README.md file and Bartman, Bulenok, et al. (2022). The key building blocks of the PySDM API and class hierarchy are: “attributes”, “backends”, “dynamics”, “environments”, “products” and physics “formulae”. The following code snippets demonstrate new elements of PySDM API, which can be added or substituted into the “v1” API description to run simulations using the new features. Execution of code snippets from both the present “v2” and the previous “v1” papers is included in the PySDM continuous integration workflow.

### Collisional Breakup

The collisional breakup process represents the splitting of two colliding superdroplets into multiple fragments. It can be specified as an individual Breakup “dynamic” or used within a unified Collision “dynamic”, in which the probability of breakup versus coalescence is sampled. The additional PySDM components used in the example below can be imported via:

```
from PySDM.dynamics.collisions import Collision
from PySDM.dynamics.collisions.collision_kernels import Golovin
from PySDM.dynamics.collisions.coalescence_efficiencies import ConstEc
from PySDM.dynamics.collisions.breakup_efficiencies import ConstEb
from PySDM.dynamics.collisions.breakup_fragmentations import ExponFrag
```

The rate of superdroplet collisions are specified by a collision kernel, and the breakup process requires three additional specifications: `coalescence_efficiencies` (probability of coalescence occurring), `breakup_efficiencies` (probability of breakup occurring if not coalescence), and `breakup_fragmentations` (the number of fragments formed in the case of a breakup event).

```
from PySDM import Builder
from PySDM.backends import CPU
from PySDM.environments import Box
from PySDM.physics import si
from PySDM.formulae import Formulae

formulae = Formulae(fragmentation_function="ExponFrag")
builder = Builder(backend=CPU(formulae), n_sd=100)
builder.set_environment(Box(dv=1 * si.m**3, dt=1 * si.s))
frag_scale = formulae.trivia.volume(radius=100 * si.micrometres)
builder.add_dynamic(Collision(
    collision_kernel=Golovin(b=1.5e3 / si.s),
    coalescence_efficiency=ConstEc(Ec=0.9),
    breakup_efficiency=ConstEb(Eb=1.0),
    fragmentation_function=ExponFrag(scale=frag_scale),
    adaptive=True,
))
```

In PySDM-examples, we introduced a set of notebooks reproducing figures from two publications. In Bieli et al. (2022), PySDM results from collisional coalescence and breakup were used as a calibration tool for learning microphysical rate parameters. In Jong et al. (in review), the physics of and algorithm for superdroplet breakup are described, and the impact of breakup on cloud properties is demonstrated with box and single-column simulations (the latter based on Shipway & Hill (2012)).

## Immersion Freezing

This release of PySDM introduces representation of immersion freezing, i.e. liquid-solid phase change contingent on the presence of insoluble ice nuclei immersed in supercooled water droplets. There are two alternative models implemented: the singular approach presented in Shima et al. (2020), and the time-dependent approach of Alpert & Knopf (2016). For the time-dependent model, the water Activity Based Immersion Freezing Model (ABIFM) of Knopf & Alpert (2013) is used. The Freezing “dynamic” is introduced by specifying whether a singular model is used, and additional particle attributes (either freezing temperature or immersed surface area) must be initialized accordingly.

```
from PySDM.dynamics import Freezing
builder.add_dynamic(Freezing(singular=False))
```

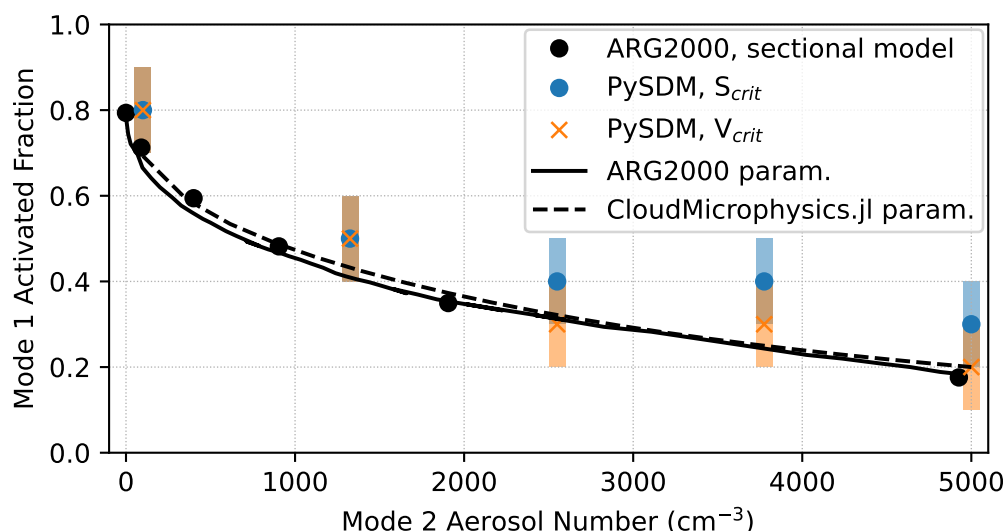
For validation of the the newly introduced immersion freezing models, a set of notebooks reproducing box-model simulations from Alpert & Knopf (2016) was introduced to the PySDM-examples package. A comparison of the time-dependent and singular models using a two-dimensional kinematic prescribed-flow framework was the focus of Arabas et al. (2023).

## Initialization of multi-component internally or externally mixed aerosol

The new aerosol initialization framework introduced in PySDM “v2” allows flexible specification of multi-modal, multi-component aerosol. The DryAerosolMixture class takes a tuple of compounds and dictionaries specifying their molar masses, densities, solubilities, and ionic dissociation numbers. The user specifies the aerosol modes which are comprised of a kappa hygroscopicity value, calculated from the molecular components and their associated mass\_fractions, and a dry aerosol size spectrum. For example, a single-mode aerosol class (SimpleAerosol) can be defined as follows.

```
from PySDM.initialisation import spectra
from PySDM.initialisation.aerosol_composition import DryAerosolMixture
class SimpleAerosol(DryAerosolMixture):
    def __init__(self):
        super().__init__(
            compounds=("(NH4)2SO4", "NaCl"),
            molar_masses={"(NH4)2SO4": 132.14 * si.g / si.mole,
                          "NaCl": 58.44 * si.g / si.mole},
            densities={"(NH4)2SO4": 1.77 * si.g / si.cm**3,
                      "NaCl": 2.16 * si.g / si.cm**3},
            is_soluble={"(NH4)2SO4": True, "NaCl": True},
            ionic_dissociation_phi={"(NH4)2SO4": 3, "NaCl": 2},
        )
        self.modes = ({
            "kappa": self.kappa(
                mass_fractions={"(NH4)2SO4": 0.7, "NaCl": 0.3}),
            "spectrum": spectra.Lognormal(
                norm_factor=100 / si.cm**3,
                m_mode=50 * si.nm, s_geom=2
            ),
        },)
```

An aerosol object (instance of `DryAerosolMixture` subclass) is used during initialization to calculate the total number of superdroplets given a prescribed number per mode, sample the size spectrum from the aerosol spectrum property, and initialize the kappa times dry volume attribute using the hygroscopicity property kappa. The choice of kappa times dry volume as an extensive attribute ensures that, upon coalescence, the hygroscopicity of a resultant super-particle is the volume-weighted average of the hygroscopicity of the coalescing super-particles. The new aerosol initialization framework is used in several examples in `PySDM`-examples including a new example that reproduces results from Abdul-Razzak & Ghan (2000), comparing `PySDM` simulations against data retrieved from the publication as shown in Figure 1).



**Figure 1:** Activated aerosol fraction in Mode 1 as a function of aerosol number concentration in Mode 2, reproducing results from Abdul-Razzak & Ghan (2000). The figure shows the results from `PySDM` in color with two definitions of activated fraction based on the critical supersaturation threshold ( $S_{crit}$ ) or the critical volume threshold ( $V_{crit}$ ). For comparison, we include the parameterization developed in Abdul-Razzak & Ghan (2000) as formulated in their paper (solid line) and as implemented in a new Julia model (`CloudMicrophysics.jl`, dashed line), as well as the results from simulations reported in Abdul-Razzak & Ghan (2000) (black dots).

## Surface-partitioning of organics to modify surface tension of droplets

`PySDM` “v2” includes a new example demonstrating the available models for droplet surface tension. The four surface tension options included in `PySDM`, which define the droplet surface tension as a function of dry aerosol composition and wet radius, are: ‘Constant’, ‘CompressedFilmOvadnevaite’ (Ovadnevaite et al. (2017)), ‘CompressedFilmRuehl’ (Ruehl et al. (2016)), and ‘SzyszkowskiLangmuir’ following the Szyszkowski-Langmuir equation. Parameters for the three surface-partitioning models must be specified as shown below. A full comparison of the four surface tension models can be found in the `Singer_Ward` example.

```
from PySDM import Formulae
f = Formulae(
    surface_tension='CompressedFilmOvadnevaite',
    constants={
        'sgm_org': 35 * si.mN / si.m,
        'delta_min': 1.75 * si.nm
    }
)
```

## Adaptive time-stepping

In PySDM “v2”, the Condensation, Collision, and Displacement “dynamics” all support adaptive time-stepping logic, which involves sub-stepping within the user-specified time step used for coupling with the “environment”. Adaptivity is enabled by default and can be disabled by passing `False` as the value of optional adaptive keyword to the given dynamic. This adaptive time-stepping applies separately in each grid box of a multidimensional environment, and includes a load-balancing logic as described in Bartman & Arabas (2023). In the case of collisions, the time-step adaptivity is aimed at eliminating errors associated with multiple coalescence events within a timestep. In the case of condensation, the time-step adaptivity is aimed at reducing computational load by coupling the time-step length choice with ambient supersaturation leading to using longer time-steps in cloud-free regions and shorter time-steps in regions where droplet [de]activation or rain evaporation occurs. In the case of displacement, the time-step adaptivity is aimed at obeying a given tolerance in integration of the super-particle trajectories, and the error measure is constructed by comparing implicit- and explicit-Euler solutions.

## Relevant recent open-source developments

PySDM supports a PyMPDATA-based (Bartman, Banaśkiewicz, et al., 2022) reimplementations of the 1D kinematically-driven test framework in a recently-published intercomparison of microphysics methods (Hill et al., 2023). The authors are unaware of recent SDM algorithm implementations in open-source packages beyond those mentioned in (Bartman, Bulenok, et al., 2022) and the related list of links in the PySDM README file. Furthermore, none of these implementations include superdroplet-count-conserving collisional breakup, organic surface partitioning or adaptive time-stepping for coagulation. The aerosol initialization method described in PySDM v2 is similar to that of pyrce1 (Rothenberg & Wang, 2017). Leveraging the availability of PyPartMC - a new Python interface to the PartMC particle-resolved Monte-Carlo aerosol simulation code (D’Aquino et al., 2023), PySDM test suite has been extended with automated checks against PartMC.

## Author contributions

EdJ led the formulation and implementation of the collisional breakup scheme with contributions from JBM. CES added the aerosol initialization framework. CES contributed the new surface tension models and relevant examples, in consultation with RXW. SAz contributed to extensions and enhancement of the one-dimensional kinematic framework environment. PB led the formulation and worked with SAR on implementation of the adaptive time-stepping schemes. KD contributed to setting up continuous integration workflows for the GPU backend. OB implemented breakup handling within the GPU backend and contributed code refactors and new tests for both CPU and GPU backends. ID, CES, and AJ contributed to the aerosol activation examples. The immersion freezing representation code was developed by SAR. Maintenance of the project have been carried out by SAR, CES, and EdJ.

## Acknowledgments

We thank Shin-ichiro Shima (University of Hyogo, Japan) for his continuous help and support in implementing SDM. Part of the outlined developments was supported by the generosity of Eric and Wendy Schmidt (by recommendation of Schmidt Futures). Development of ice-phase microphysics representation has been supported through grant no. DE-SC0021034 by the Atmospheric System Research Program and Atmospheric Radiation Measurement Program sponsored by the U.S. Department of Energy (DOE). EdJ’s contributions were made possible by support from the Department of Energy Computational Sciences Graduate Research Fellowship.

SAr, OB and KD acknowledge support from the Polish National Science Centre (grant no. 2020/39/D/ST10/01220).

## References

- Abdul-Razzak, H., & Ghan, S. J. (2000). A parameterization of aerosol activation: 2. Multiple aerosol types. *J. Geophys. Res.* <https://doi.org/10.1029/1999JD901161>
- Alpert, P. A., & Knopf, D. A. (2016). Analysis of isothermal and cooling-rate-dependent immersion freezing by a unifying stochastic ice nucleation model. *Atmos. Chem. Phys.* <https://doi.org/10.5194/acp-16-2083-2016>
- Arabas, S., Curtis, J. H., Silber, I., Fridlind, A., Knopf, D. A., West, M., & Riemer, N. (2023). On probabilistic particle-based modeling of immersion freezing. *103rd American Meteorological Society Annual Meeting*. <https://ams.confex.com/ams/103ANNUAL/meetingapp.cgi/Paper/420160>
- Bartman, P., & Arabas, S. (2023). Adaptive time-stepping for particle-based cloud microphysics: Super-droplet transport, collisions and condensational growth. *103rd American Meteorological Society Annual Meeting*. <https://ams.confex.com/ams/103ANNUAL/meetingapp.cgi/Paper/419078>
- Bartman, P., Banaśkiewicz, J., Drenda, S., Manna, M., Olesik, M., Rozwoda, P., Sadowski, M., & Arabas, S. (2022). PyMPDATA v1: Numba-accelerated implementation of MPDATA with examples in Python, Julia and Matlab. In *J. Open Source Soft.* <https://doi.org/10.21105/joss.03896>
- Bartman, P., Bulenok, O., Górski, K., Jaruga, A., Łazarski, G., Olesik, M. A., Piasecki, B., Singer, C. E., Talar, A., & Arabas, S. (2022). PySDM v1: Particle-based cloud modelling package for warm-rain microphysics and aqueous chemistry. *J. Open Source Soft.* <https://doi.org/10.21105/joss.03219>
- Bieli, M., Dunbar, O. R. A., Jong, E. K. de, Jaruga, A., Schneider, T., & Bischoff, T. (2022). An efficient bayesian approach to learning droplet collision kernels: Proof of concept using "cloudy", a new n-moment bulk microphysics scheme. *J. Adv. Model. Earth Syst.* <https://doi.org/10.1029/2022MS002994>
- D'Aquino, Z., Arabas, S., Curtis, J. H., Vaishnav, A., Choi, J., Riemer, N., & West, M. (2023). PyPartMC: A Pythonic interface to a particle-resolved Monte-Carlo aerosol simulation framework. *103rd American Meteorological Society Annual Meeting*. <https://ams.confex.com/ams/103ANNUAL/meetingapp.cgi/Paper/421645>
- Hill, A. A., Lebo, Z. J., Andrejczuk, M., Arabas, S., Dziekan, P., Field, P., Gettelman, A., Hoffmann, F., Pawlowska, H., Onishi, R., & Vie, B. (2023). Toward a numerical benchmark for warm rain processes. *J. Atmos. Sci.* <https://doi.org/10.1175/JAS-D-21-0275.1>
- Jong, E. de, Mackay, J. B., Jaruga, A., & Arabas, S. (in review). Breakups are complicated: An efficient representation of collisional breakup in the superdroplet method. *Geosci. Model Dev.* <https://doi.org/10.5194/egusphere-2022-1243>
- Knopf, D. A., & Alpert, P. A. (2013). A water activity based model of heterogeneous ice nucleation kinetics for freezing of water and aqueous solution droplets. *Faraday Discuss.* <https://doi.org/10.1039/c3fd00035d>
- Ovadnevaite, J., Zuend, A., Laaksonen, A., Sanchez, K. J., Roberts, G., Ceburnis, D., Decesari, S., Rinaldi, M., Hodas, N., Facchini, M. C., Seinfeld, J. H., & O'Dowd, C. (2017). Surface tension prevails over solute effect in organic-influenced cloud droplet activation. *Nature*. <https://doi.org/10.1038/nature22806>



- Rothenberg, D., & Wang, C. (2017). An aerosol activation metamodel of v1.2.0 of the pyrcel cloud parcel model: Development and offline assessment for use in an aerosol–climate model. *Geosci. Model. Dev.* <https://doi.org/10.5194/gmd-10-1817-2017>
- Ruehl, C. R., Davies, J. F., & Wilson, K. R. (2016). An interfacial mechanism for cloud droplet formation on organic aerosols. *Science*. <https://doi.org/10.1126/science.aad4889>
- Shima, S., Kusano, K., Kawano, A., Sugiyama, T., & Kawahara, S. (2009). The super-droplet method for the numerical simulation of clouds and precipitation: A particle-based and probabilistic microphysics model coupled with a non-hydrostatic model. *Q. J. Royal Meteorol. Soc.* <https://doi.org/10.1002/qj.441>
- Shima, S., Sato, Y., Hashimoto, A., & Misumi, R. (2020). Predicting the morphology of ice particles in deep convection using the super-droplet method: Development and evaluation of SCALE-SDM 0.2.5-2.2.0, -2.2.1, and -2.2.2. *Geosci. Model Dev.* <https://doi.org/10.5194/gmd-13-4107-2020>
- Shipway, B. J., & Hill, A. A. (2012). Diagnosis of systematic differences between multiple parametrizations of warm rain microphysics using a kinematic framework. *Q. J. Royal Meteorol. Soc.* <https://doi.org/10.1002/qj.1913>