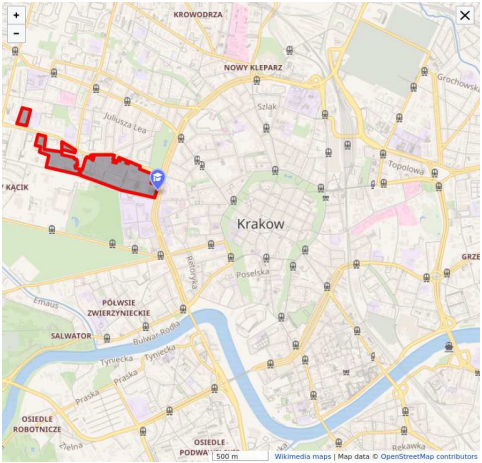
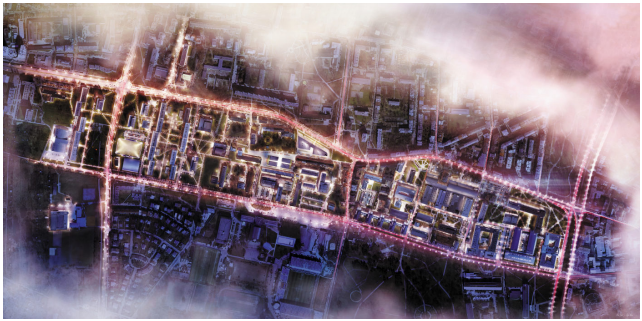
Adaptive timestepping for diffusional processes in particle-based aerosol-cloud microphysics

Sylwester Arabas[1], P. Bartman[2], P. Dziekan[3], T. Lüttmer[4], A. Makulska[3], A. Żaba[1]

1: agh.edu.pl, 2: uj.edu.pl, 3: uw.edu.pl, 4: uni-mainz.de

AGH University of Krakow





ca. 23 000 students & 2 400 academic/research staff

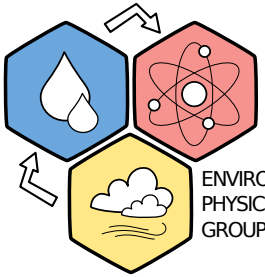
AGH

AGH Environmental Physics Group

AGH

(Wissenschaftlich-Technische Universität)

est. 1913 (by emperor Franz Joseph of Austria)
opened 1919 (by Polish government)



zfs.agh.edu.pl

AGH Environmental Physics Group: zfs.agh.edu.pl

AGH Environmental Physics Group EN

Team Field measurements Data access Laboratories Modelling & software Funding Publications Studies Doctorate Seminars Outreach

Field measurements

The AGH Environmental Physics Group is a leading national and internationally recognised center for environmental measurements, particularly using isotope methods. We present selected types of measurements performed by our group, along with references to relevant publications.

- AGH KASLAB, Kasprowy Wierch (1989 m amsl)

- AGH Campus 20m Tower (Kraków)

- Hydrological and hydrogeological studies

- Measurements at the 300-m BIK tower in North-East Poland

- UAV-borne hydrological and meteorological observations

- Meteo and air-quality profiling from a tethered balloon in Kraków

AGH Environmental Physics Group: zfs.agh.edu.pl

AGH Environmental Physics Group

EN

Team | Field measurements | Data access | Laboratories | Modelling & software | Funding | Publications | Studies | Doctorate | Seminars | Outreach

Modelling & software

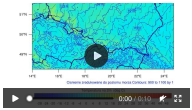
AGH Environmental Physics Group / Modelling & software

Computer simulations are an indispensable tool in our work. We specialize in the use of numerical models in environmental research, as well as in the engineering of open-source research software for use in environmental physics and, more broadly, in computational sciences. For high-performance computing, we employ AGH Cyfronet supercomputing resources. Methods and applications of numerical simulations and software engineering are part of our course offering. We leverage open-source software development and dissemination workflows for knowledge transfer across scientific disciplines and towards users outside of academia.

Operational weather prediction using WRF

contact: **Mirosław Zimnoch**

Our WRF-based daily workflow is set up to compute 48-hour predictions on a country-wide domain, with initial and boundary conditions fetched from the NOAA GFS output. The forecasts are publicly available in form of weather maps and meteograms at meteo.fis.agh.edu.pl website.



Open-source research software engineering

contact: **open-atmos-krk team**

Our team, with collaborators from partnering institutions as well as AGH students, develops and maintains several open-source Python packages:

numba-mpi | pypi package 1.0.1 | Anaconda.org 1.11.1 | AUR package 1.0.1

offers access to the Message Passing Interface (MPI) routines from Python code that uses the Numba just-in-time (JIT) compiler, enabling use of MPI communication in high-performance, multi-threaded, JIT-compiled Python code

use case: **py-pde** developed at Max Planck Institute for Dynamics and

github.com/open-atmos

open-atmos

What is open-atmos?

Open-atmos is a multi-institutional community hub for projects in atmospheric sciences. We support projects that are committed to an open development model. This includes:

- embracing open-source licenses
- developing code in public
- maintaining public issue trackers and discussion forums
- having automated cloud-hosted workflows executing thorough test suites
- being open to contributions from the entire community
- actively facilitating onboarding of new contributors by providing examples and tutorials
- supporting users with documentation maintained with the code
- empowering research users with reproducibility-oriented workflows for code dissemination and archiving, ensuring the ability to anonymously review and reuse the code and scientific results obtained with it
- adhering to best practices in software engineering

We welcome new members and projects to open-atmos!

Who is the admin team?

Sylwester Arabas (sylwester.arabas@agh.edu.pl)
Jeff Curtis (jcurtis2@illinois.edu)
Matt Dawson (matt@cohere-llc.com)
Oriol Jorba (oriol.jorba@bsc.es)
Nicole Riemer (nriemer@illinois.edu)
Matt West (mwest@illinois.edu)

People

Invite someone

Top languages

Python Fortran C++

github.com/open-atmos

PySDM Public

Pythonic particle-based (super-droplet) cloud microphysics (and aqueous-chemistry) package with box, parcel & 1D/2D prescribed-flow examples in Python, Julia and Matlab

Python ☆ 88 GPL-3.0 54 201 (2 issues need help) 40 Updated 2 days ago

PyPartMC Public

Python (and C++) interface to PartMC with Jupyter/Python, Julia and Matlab examples

C++ ☆ 31 GPL-3.0 16 55 (1 issue needs help) 14 Updated 3 days ago

PyMPDATA Public

Numba-accelerated Pythonic implementation of MPDATA with examples in Python, Julia, Rust and Matlab

Python ☆ 33 GPL-3.0 25 31 (1 issue needs help) 22 Updated 5 days ago

MOSAIC Public

Forked from pnnl/MOSAIC

Fork of MOSAIC maintained solely to support PartMC (www.github.com/compdyn/partmc) integration. This is not a general-purpose fork and may diverge from upstream in ways only meaningful within PartMC workflows. Upstream changes will be selectively incorporated or omitted based on relevance to PartMC compatibility.

Fortran ☆ 0 2 0 1 Updated 5 days ago

camp Public

Multi-phase chemistry treatment for atmospheric models

Fortran ☆ 16 MIT 10 6 2 Updated 3 weeks ago

focus: aerosol-cloud-precipitation interactions (numerics)

focus: aerosol-cloud-precipitation interactions

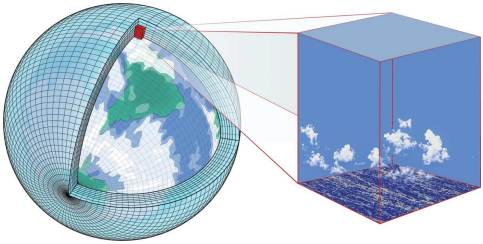
"Cloud and ship. Ukraine, Crimea, Black sea, view from Ai-Petri mountain"

(photo: Yevgen Timashov / National Geographic)

focus: aerosol-cloud-precipitation interactions

"Cloud and ship. Ukraine, Crimea, Black sea, view from Ai-Petri mountain"

(photo: Yevgen Timashov / National Geographic)

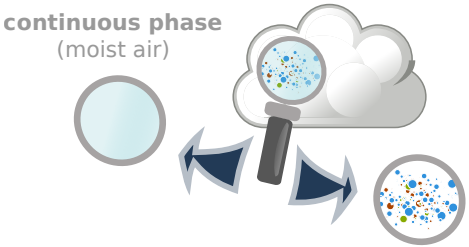


"Grid cells in a global climate model and a large-eddy simulation of shallow cumulus clouds at 5 m resolution"

(fig. from Schneider et al. 2017)

Eulerian vs. Lagrangian microphysics

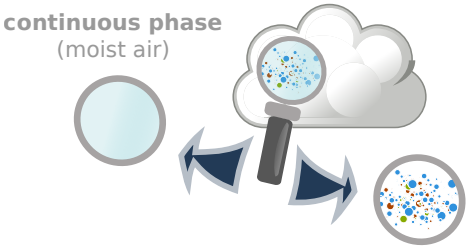
Eulerian vs. Lagrangian microphysics



dispersed phase

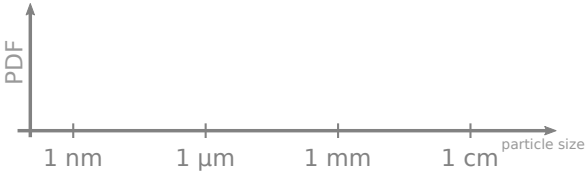
(aerosol particles, cloud droplets, drizzle, rain, snow, ...)

Eulerian vs. Lagrangian microphysics

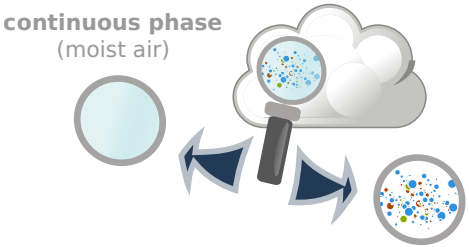


dispersed phase

(aerosol particles, cloud droplets, drizzle, rain, snow, ...)

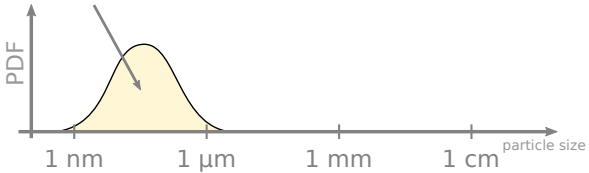


Eulerian vs. Lagrangian microphysics

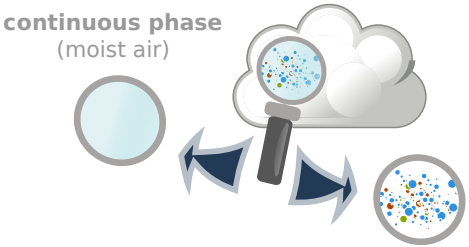


dispersed phase

(aerosol particles, cloud droplets, drizzle, rain, snow, ...)

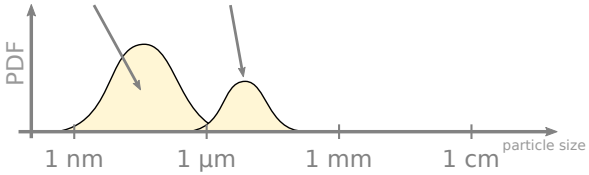


Eulerian vs. Lagrangian microphysics

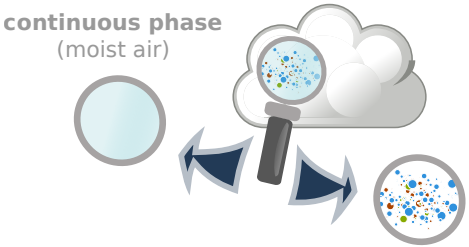


dispersed phase

(aerosol particles, cloud droplets, drizzle, rain, snow, ...)

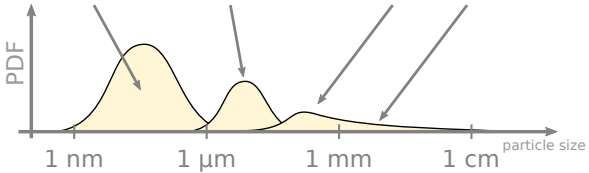


Eulerian vs. Lagrangian microphysics

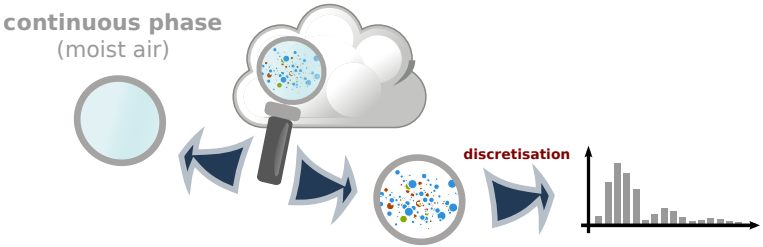


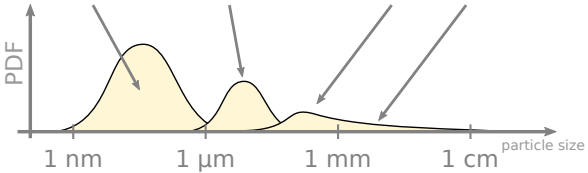
dispersed phase

(aerosol particles, cloud droplets, drizzle, rain, snow, ...)

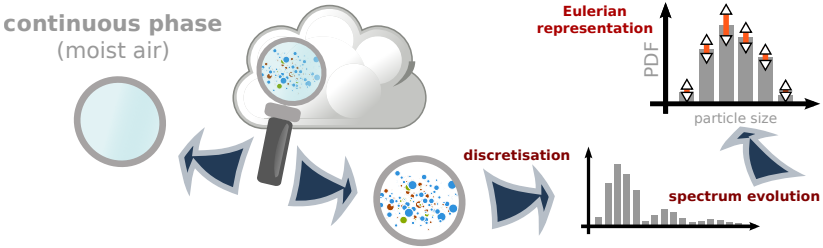


Eulerian vs. Lagrangian microphysics



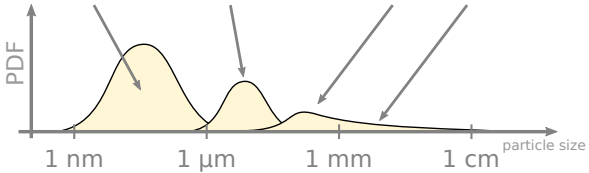


Eulerian vs. Lagrangian microphysics

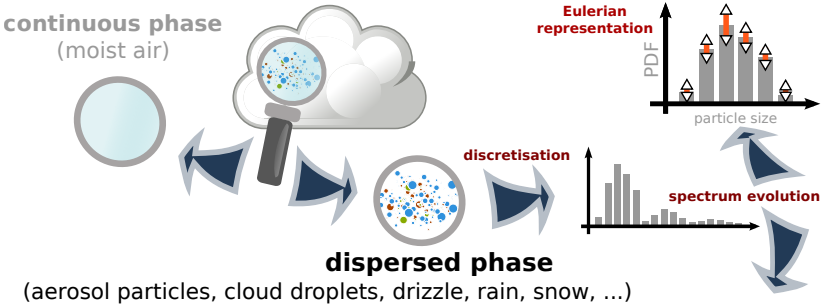


dispersed phase

(aerosol particles, cloud droplets, drizzle, rain, snow, ...)



Eulerian vs. Lagrangian microphysics



PDEs

ODEs

Lagrangian microphysics: early works (0D)

JOURNAL OF METEOROLOGY

THE GROWTH OF CLOUD DROPS IN UNIFORMLY COOLED AIR

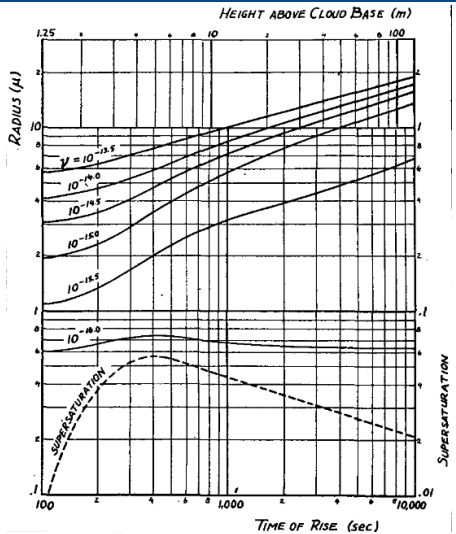
By Wallace E. Howell[1]

Blue Hill Meteorological Observatory, Harvard University[2]

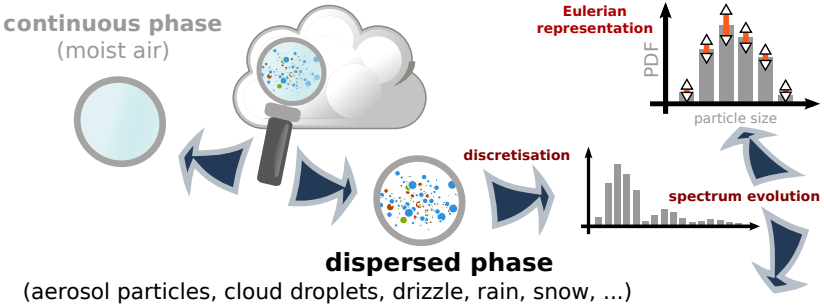
(Manuscript received 10 June 1948)

ABSTRACT

Recent studies of precipitation, aircraft icing, and visibility through fog have focussed attention on the physical constitution of clouds, a subject to which knowledge of the drop-size spectrum and its origin would be an important contribution. The drop-size spectrum resulting when air containing condensation nuclei is uniformly cooled may be computed, leading to a differential equation for the growth of a cloud drop which cannot be integrated analytically. A numerical method of integration is therefore employed.



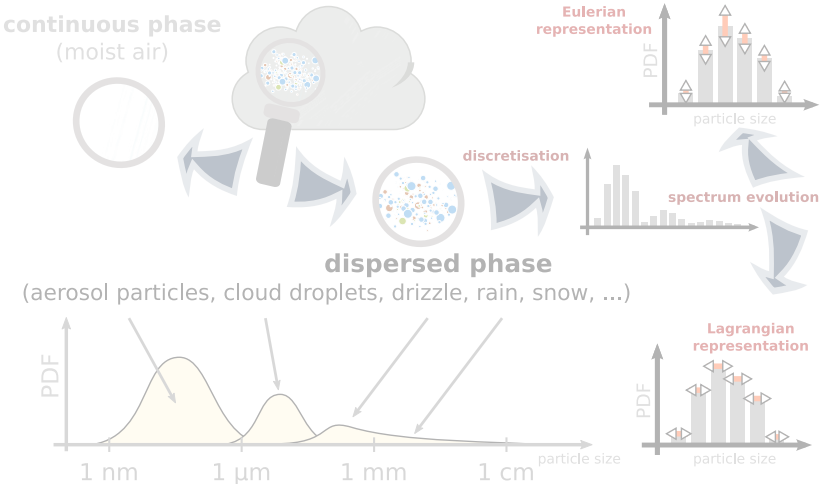
Eulerian vs. Lagrangian microphysics

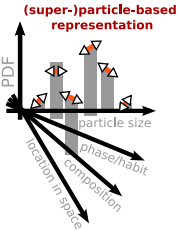


PDEs

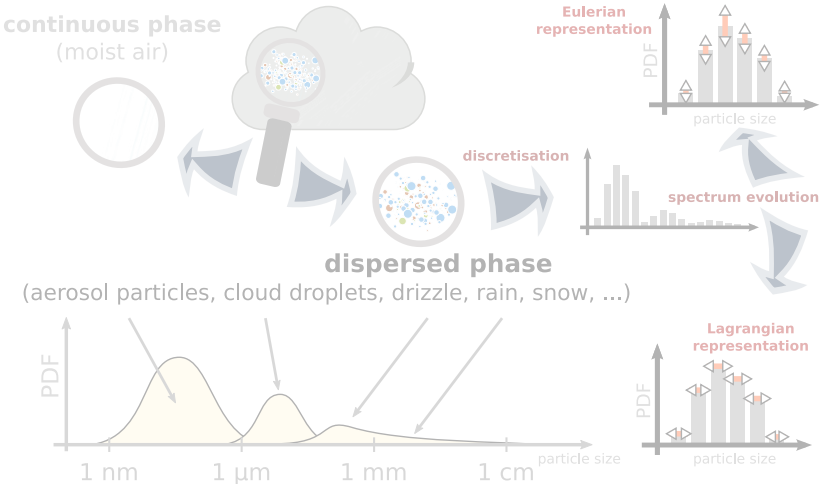
ODEs

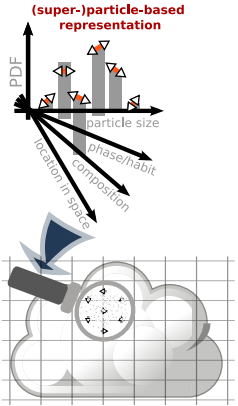
Eulerian vs. Lagrangian microphysics



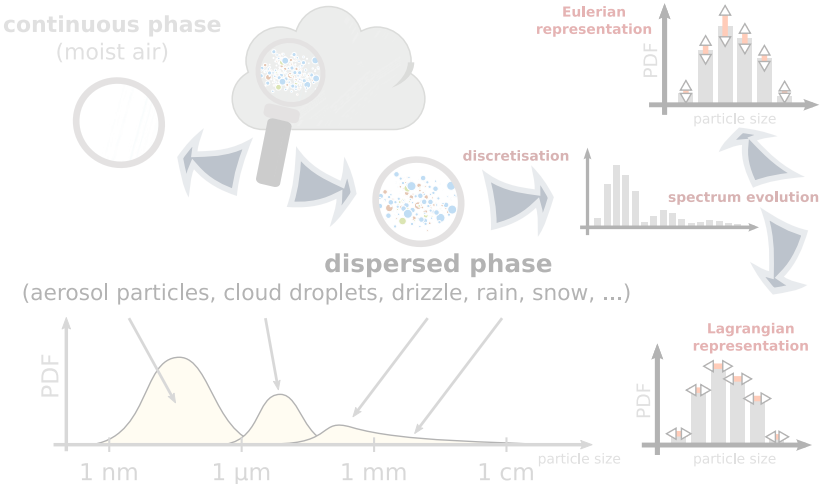


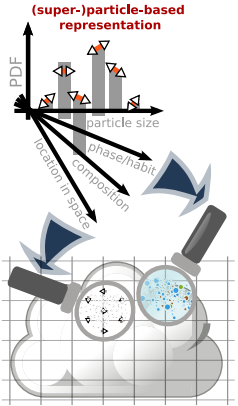
Eulerian vs. Lagrangian microphysics





Eulerian vs. Lagrangian microphysics





Lagrangian microphysics: early works (3D)

JOURNAL OF THE ATMOSPHERIC SCIENCES

A Numerical Experiment on Stochastic Condensation Theory

TERRY L. CLARK AND W. D. HALL

National Center for Atmospheric Research,[1] Boulder, CO 80307

(Manuscript received 30 August 1978, in final form 20 November 1978)

ABSTRACT

A three-dimensional numerical model is used to study the effect of small-scale supersaturation fluctuations on the evolving droplet distribution in the first 150 m above cloud base. The primary purpose of this research is to determine whether the irreversible coupling between the thermodynamics and dynamics due to finite phase relaxation time scales τ_s is sufficient to produce significant small-scale horizontal variations in supersaturation. Thus, the paper is concerned only with this internal source for thermodynamic variability. All other source terms, such as the downgradient flux of the variance of thermodynamic fields, have purposely been neglected.

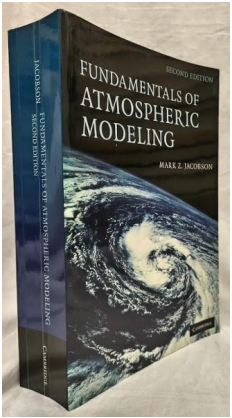
Lagrangian particle experiments were run in parallel with the basic Eulerian model. The purpose of these experiments is to relax some of the microphysical parameterization assumptions with respect to assumed distribution shape and as a result add credibility to the results of distribution broadening.

Eulerian vs. Lagrangian microphysics: a (probabilistic) breakthrough

pre-2009:

„advantage of the full-moving size structure is that core particle material is preserved during growth ... second advantage ... it eliminates numerical diffusion ... [but] nucleation, coagulation ... cause problems ... the full-moving structure is not used in three-dimensional models" [a]

„the use of a fixed grid allows for an easy implementation of collision processes, which is not possible for a moving grid (Lagrangian) approach" [b]



[a] Jacobson 2005: Fundamentals of Atmospheric Modeling
[b] Simmel & Wurzler 2006: Condensation and activation in sectional cloud microphysical models

Eulerian vs. Lagrangian microphysics: a (probabilistic) breakthrough

pre-2009:

„advantage of the full-moving size structure is that core particle material is preserved during growth ... second advantage ... it eliminates numerical diffusion ... [but] nucleation, coagulation ... cause problems ... the full-moving structure is not used in three-dimensional models" [a]

„the use of a fixed grid allows for an easy implementation of collision processes, which is not possible for a moving grid (Lagrangian) approach" [b]

[a] Jacobson 2005: Fundamentals of Atmospheric Modeling
[b] Simmel & Wurzler 2006: Condensation and activation in sectional cloud microphysical models

Shima 2009: Monte-Carlo particle-based collision algorithm for cloud simulations

Eulerian vs. Lagrangian microphysics: a (probabilistic) breakthrough

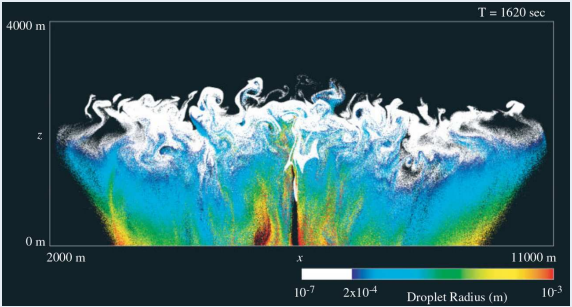
pre-2009:

„advantage of the full-moving size structure is that core particle material is preserved during growth ... second advantage ... it eliminates numerical diffusion ... [but] nucleation, coagulation ... cause problems ... the full-moving structure is not used in three-dimensional models" [a]

„the use of a fixed grid allows for an easy implementation of collision processes, which is not possible for a moving grid (Lagrangian) approach" [b]

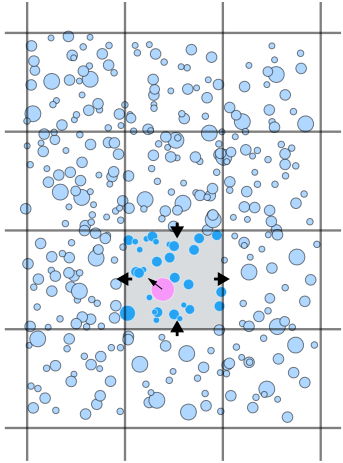
[a] Jacobson 2005: Fundamentals of Atmospheric Modeling
[b] Simmel & Wurzler 2006: Condensation and activation in sectional cloud microphysical models

Shima 2009: Monte-Carlo particle-based collision algorithm for cloud simulations



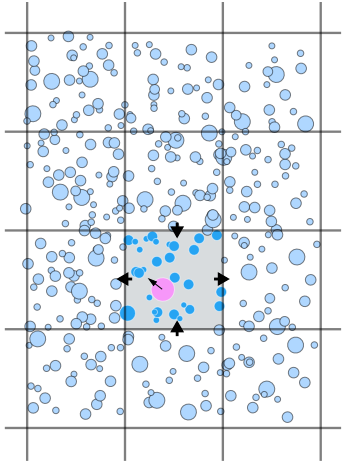
Super-droplet simulation of a shallow convective cloud (figure: Shima et al. 2009, QJRMS)

focus: aerosol representation in particle-based μ-physics models



focus: aerosol representation in particle-based μ-physics models

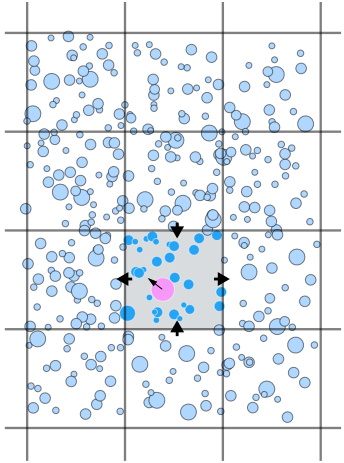
no aerosol budget:



focus: aerosol representation in particle-based μ-physics models

no aerosol budget:

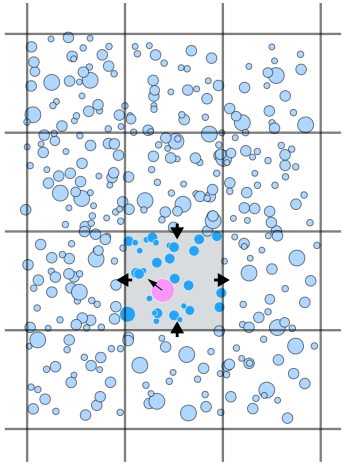
Riechelmann et al. '12 pre-existing arbitrary-sized droplets



focus: aerosol representation in particle-based μ-physics models

no aerosol budget:

Riechelmann et al. '12 pre-existing arbitrary-sized droplets
Naumann & Seifert '15 precip-sized particles only

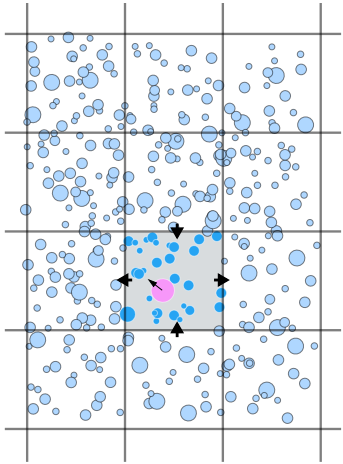


focus: aerosol representation in particle-based μ-physics models

no aerosol budget:

Riechelmann et al. '12 pre-existing arbitrary-sized droplets
Naumann & Seifert '15 precip-sized particles only

CCN activation (and deactivation) models:



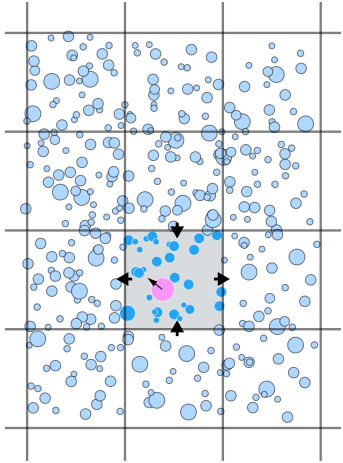
focus: aerosol representation in particle-based μ-physics models

no aerosol budget:

Riechelmann et al. '12 pre-existing arbitrary-sized droplets
Naumann & Seifert '15 precip-sized particles only

CCN activation (and deactivation) models:

Shima et al. '09 aerosol & droplets treated uniformly (Köhler terms)
⇝ CCN activation/deactivation resolved
⇝ finite CCN reservoir, precip sink vs. aerosol source!



focus: aerosol representation in particle-based μ-physics models

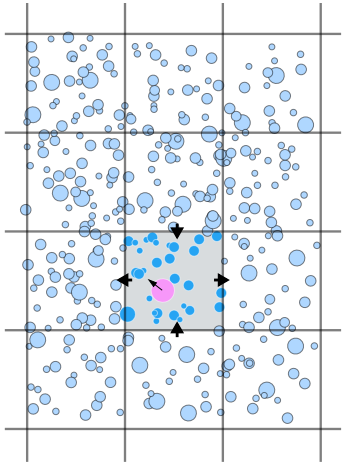
no aerosol budget:

Riechelmann et al. '12 pre-existing arbitrary-sized droplets
Naumann & Seifert '15 precip-sized particles only

CCN activation (and deactivation) models:

Shima et al. '09 aerosol & droplets treated uniformly (Köhler terms)
- ⇝ CCN activation/deactivation resolved
- ⇝ finite CCN reservoir, precip sink vs. aerosol source!

Grabowski et al. '18 Twomey activation (saturation activity spectrum)
- ⇝ infinite CCN reservoir
- ⇝ aerosol processing not resolvable
- ⇝ kinetic limitations not fully resolvable



focus: aerosol representation in particle-based μ-physics models

no aerosol budget:

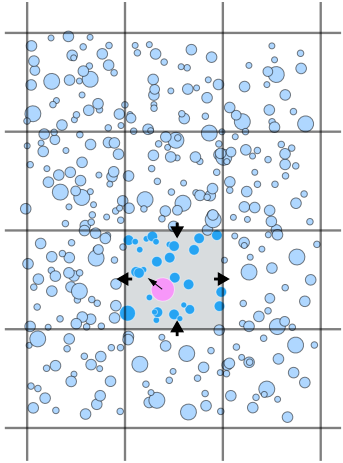
Riechelmann et al. '12 pre-existing arbitrary-sized droplets
Naumann & Seifert '15 precip-sized particles only

CCN activation (and deactivation) models:

Shima et al. '09 aerosol & droplets treated uniformly (Köhler terms)
- ⇝ CCN activation/deactivation resolved
- ⇝ finite CCN reservoir, precip sink vs. aerosol source!

Grabowski et al. '18 Twomey activation (saturation activity spectrum)
- ⇝ infinite CCN reservoir
- ⇝ aerosol processing not resolvable
- ⇝ kinetic limitations not fully resolvable

INP freezing (and melting) models:



focus: aerosol representation in particle-based μ-physics models

no aerosol budget:

Riechelmann et al. '12 pre-existing arbitrary-sized droplets
Naumann & Seifert '15 precip-sized particles only

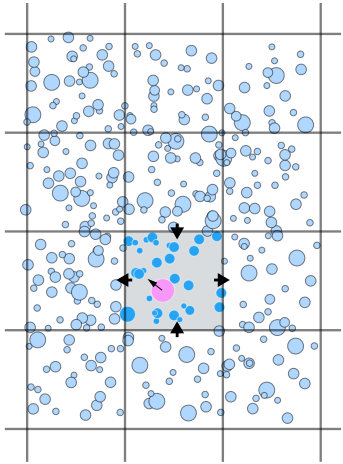
CCN activation (and deactivation) models:

Shima et al. '09 aerosol & droplets treated uniformly (Köhler terms)
- ⇝ CCN activation/deactivation resolved
- ⇝ finite CCN reservoir, precip sink vs. aerosol source!

Grabowski et al. '18 Twomey activation (saturation activity spectrum)
- ⇝ infinite CCN reservoir
- ⇝ aerosol processing not resolvable
- ⇝ kinetic limitations not fully resolvable

INP freezing (and melting) models:

Shima et al. '20 "singular" (temp. spectrum, akin to Twomey CCN)
- ⇝ involves assumed cooling rates
- ⇝ INP processing not fully resolvable



focus: aerosol representation in particle-based μ-physics models

no aerosol budget:

Riechelmann et al. '12 pre-existing arbitrary-sized droplets
Naumann & Seifert '15 precip-sized particles only

CCN activation (and deactivation) models:

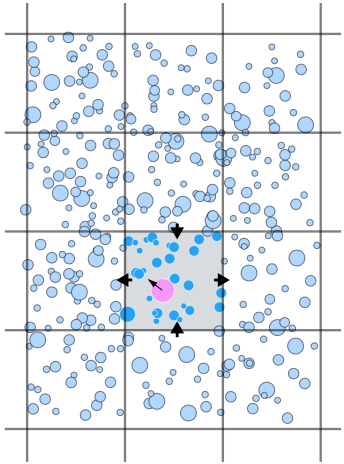
Shima et al. '09 aerosol & droplets treated uniformly (Köhler terms)
- ⇝ CCN activation/deactivation resolved
- ⇝ finite CCN reservoir, precip sink vs. aerosol source!

Grabowski et al. '18 Twomey activation (saturation activity spectrum)
- ⇝ infinite CCN reservoir
- ⇝ aerosol processing not resolvable
- ⇝ kinetic limitations not fully resolvable

INP freezing (and melting) models:

Shima et al. '20 "singular" (temp. spectrum, akin to Twomey CCN)
- ⇝ involves assumed cooling rates
- ⇝ INP processing not fully resolvable

Arabas et al. '25 "time-dependent" (solution effects resolvable)



focus: aerosol representation in particle-based μ-physics models

no aerosol budget:

Riechelmann et al. '12 pre-existing arbitrary-sized droplets
Naumann & Seifert '15 precip-sized particles only

CCN activation (and deactivation) models:

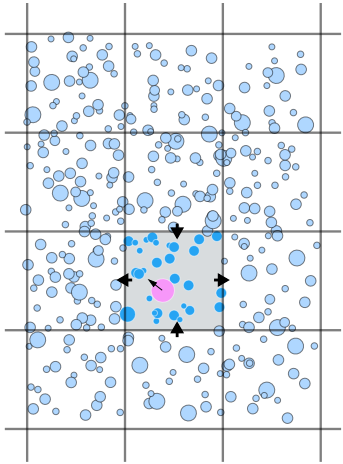
Shima et al. '09 aerosol & droplets treated uniformly (Köhler terms)
⇝ CCN activation/deactivation resolved
⇝ finite CCN reservoir, precip sink vs. aerosol source!

Grabowski et al. '18 Twomey activation (saturation activity spectrum)
⇝ infinite CCN reservoir
⇝ aerosol processing not resolvable
⇝ kinetic limitations not fully resolvable

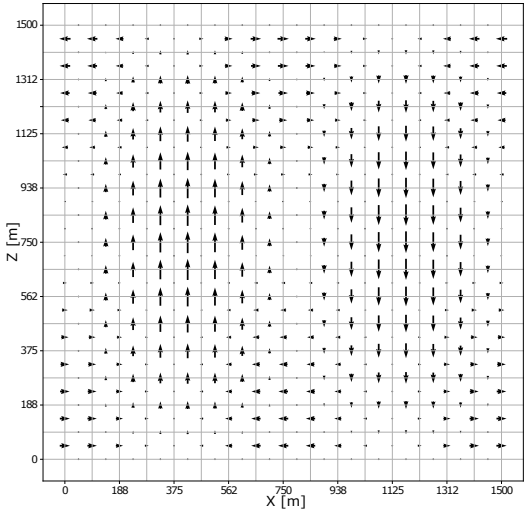
INP freezing (and melting) models:

Shima et al. '20 "singular" (temp. spectrum, akin to Twomey CCN)
⇝ involves assumed cooling rates
⇝ INP processing not fully resolvable

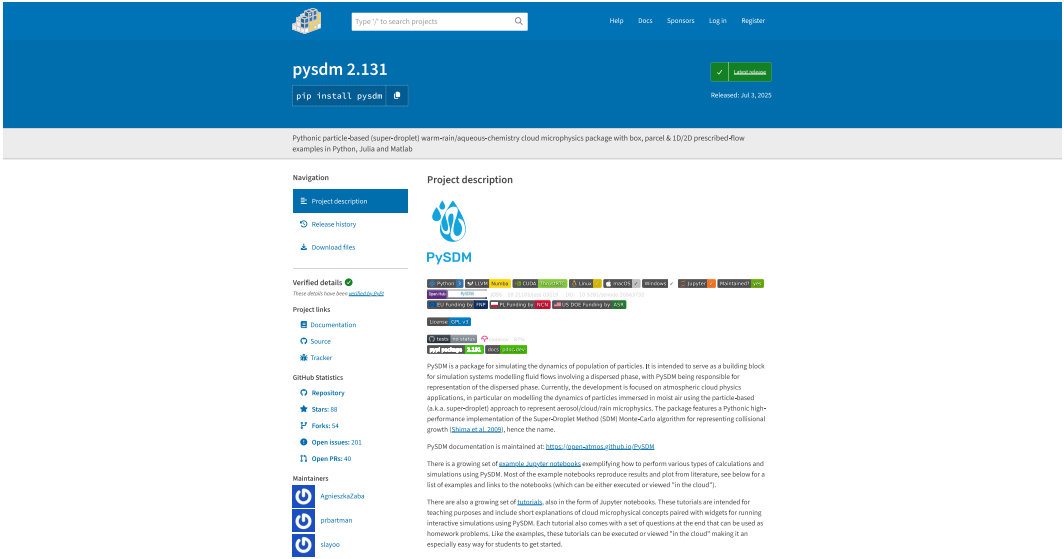
Arabas et al. '25 "time-dependent" (solution effects resolvable)



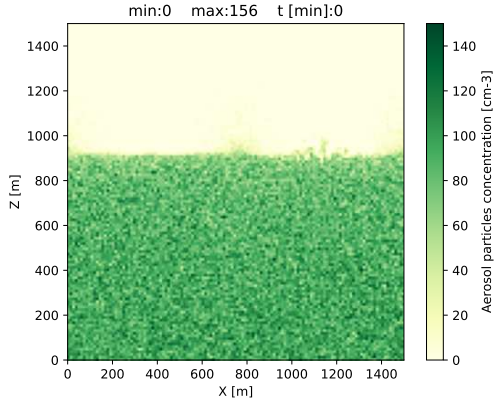
Kessler kinematic test (particle-based μ-physics setup as in Arabas et al. '15)

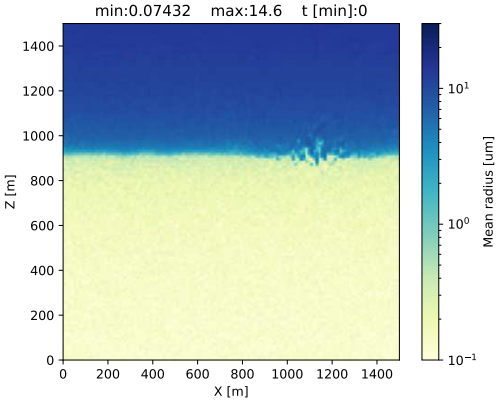


pypi.org/p/PySDM: JIT-compiled (CPU/GPU) particle-based μ-physics



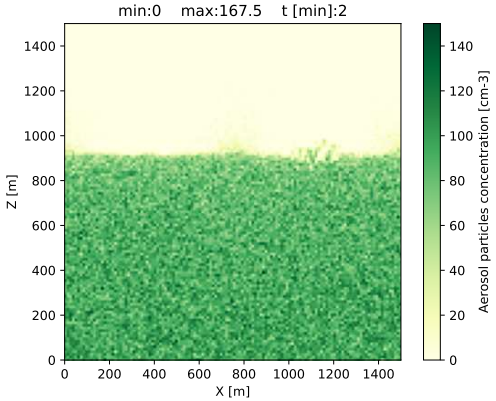
prescribed-flow Sc (incl. virga/CCN resuspension) – PySDM

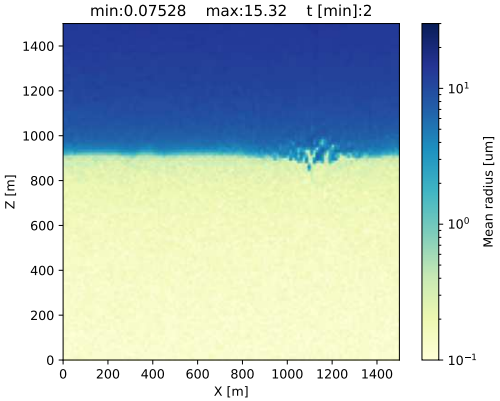




Computational grid: 128x128
Computational particles: 2^{21}

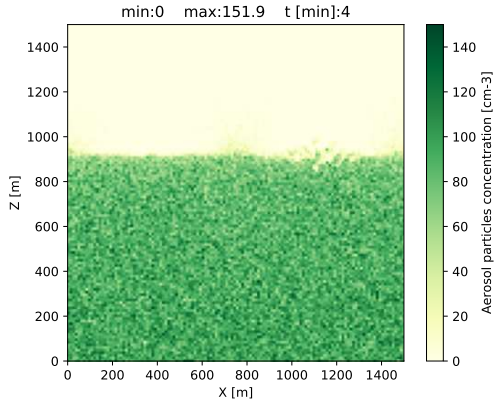
prescribed-flow Sc (incl. virga/CCN resuspension) – PySDM

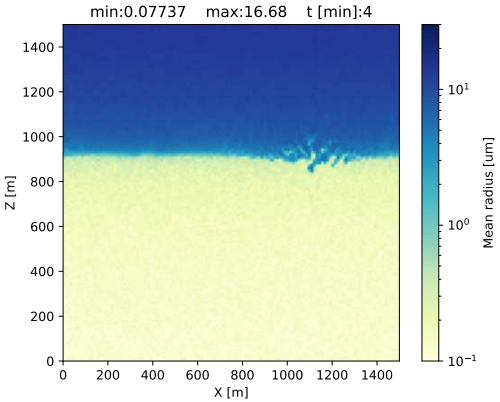




Computational grid: 128x128
Computational particles: 2^{21}

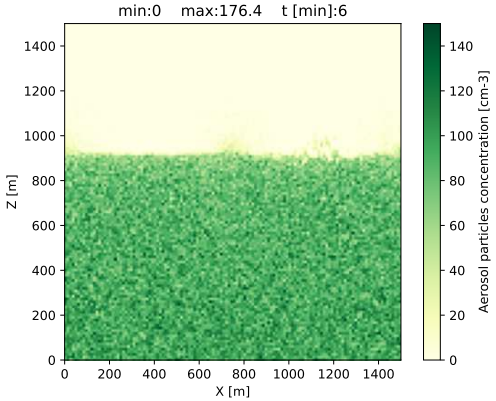
prescribed-flow Sc (incl. virga/CCN resuspension) – PySDM

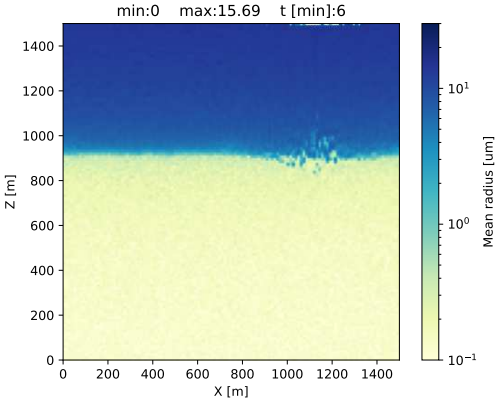




Computational grid: 128x128
Computational particles: 2^{21}

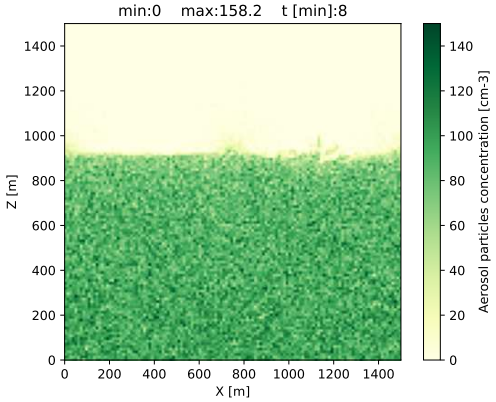
prescribed-flow Sc (incl. virga/CCN resuspension) – PySDM

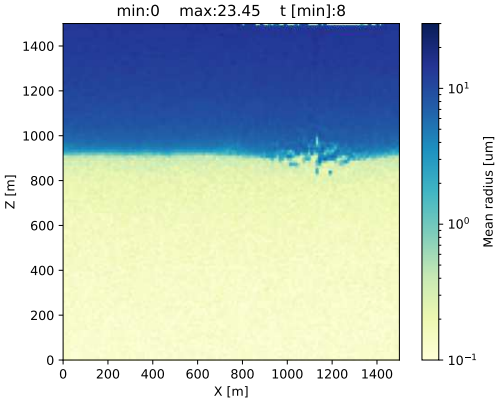




Computational grid: 128x128
Computational particles: 2^{21}

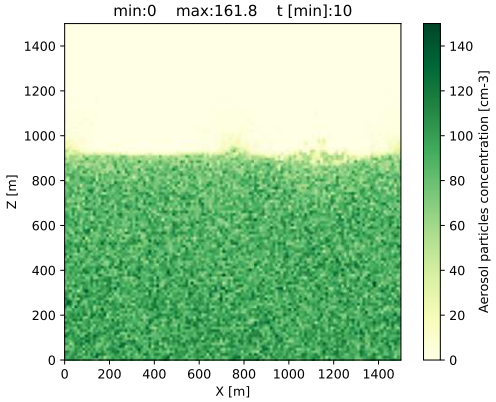
prescribed-flow Sc (incl. virga/CCN resuspension) – PySDM

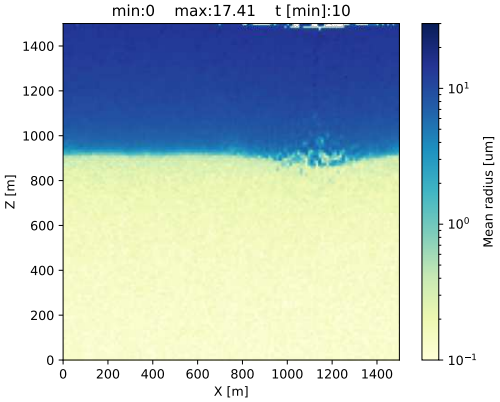




Computational grid: 128x128
Computational particles: 2^{21}

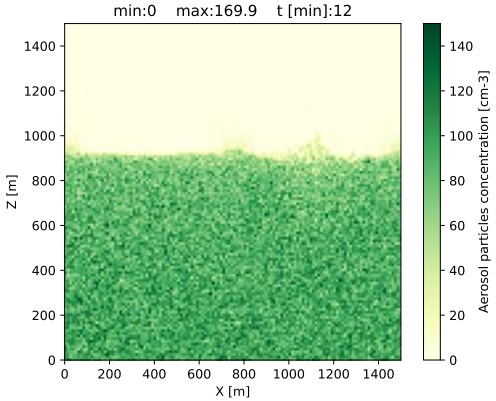
prescribed-flow Sc (incl. virga/CCN resuspension) – PySDM

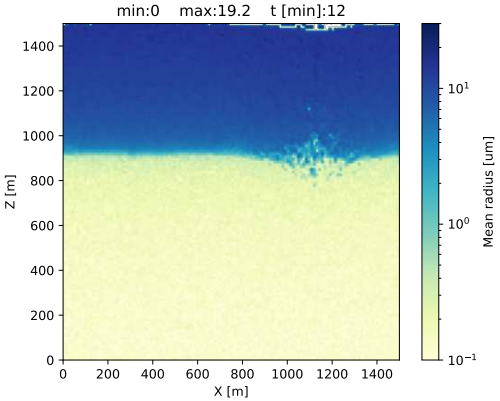




Computational grid: 128x128
Computational particles: 2^{21}

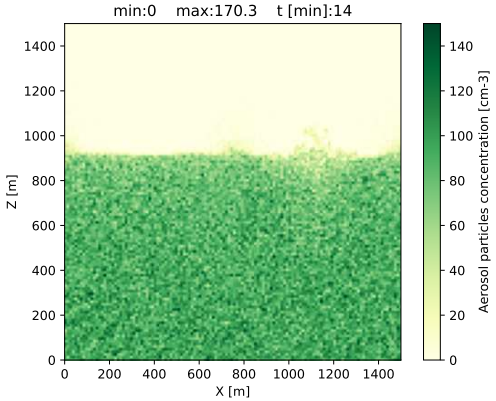
prescribed-flow Sc (incl. virga/CCN resuspension) – PySDM

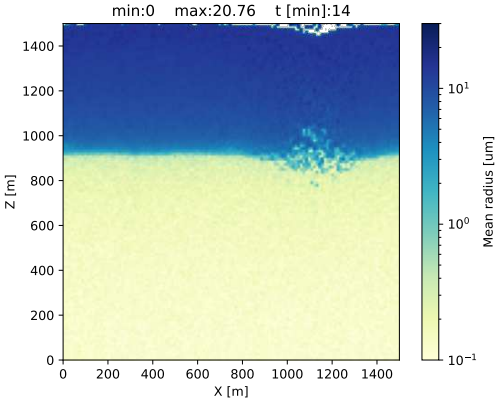




Computational grid: 128x128
Computational particles: 2^{21}

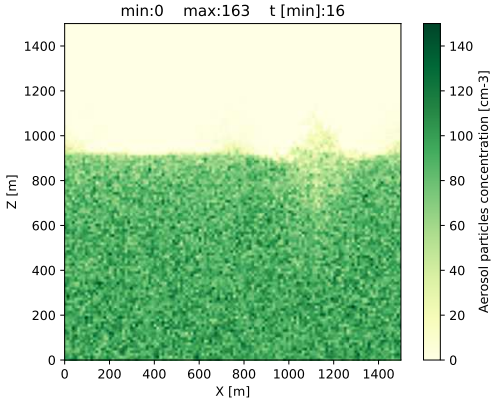
prescribed-flow Sc (incl. virga/CCN resuspension) – PySDM

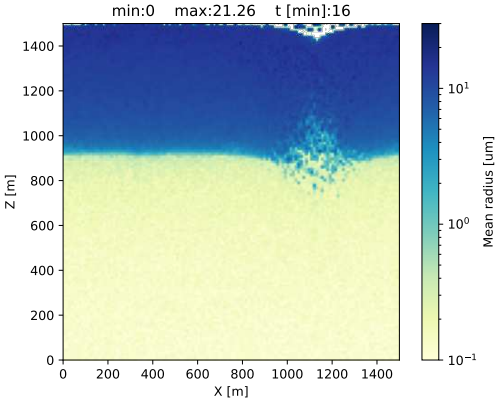




Computational grid: 128x128
Computational particles: 2^{21}

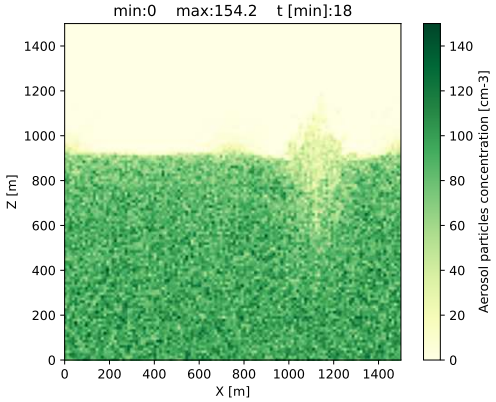
prescribed-flow Sc (incl. virga/CCN resuspension) – PySDM

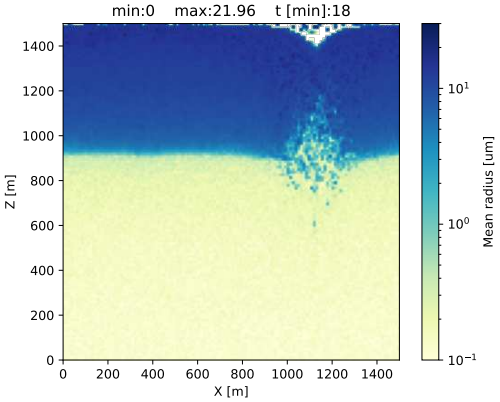




Computational grid: 128x128
Computational particles: 2^{21}

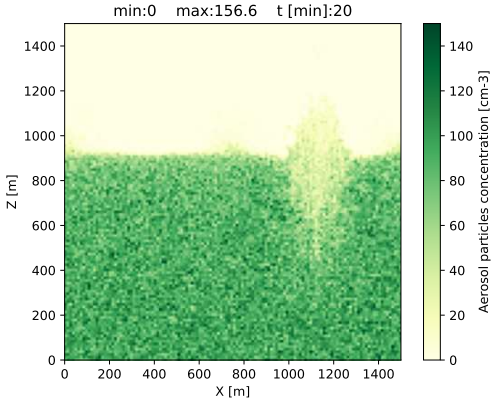
prescribed-flow Sc (incl. virga/CCN resuspension) – PySDM

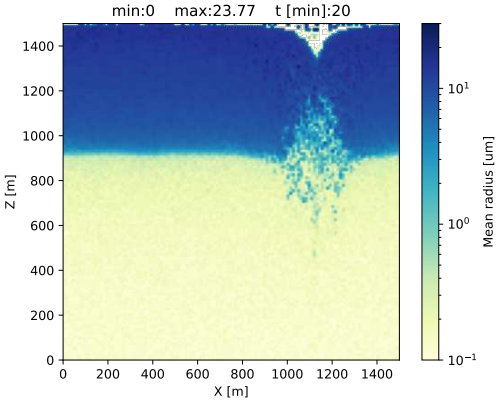




Computational grid: 128x128
Computational particles: 2^{21}

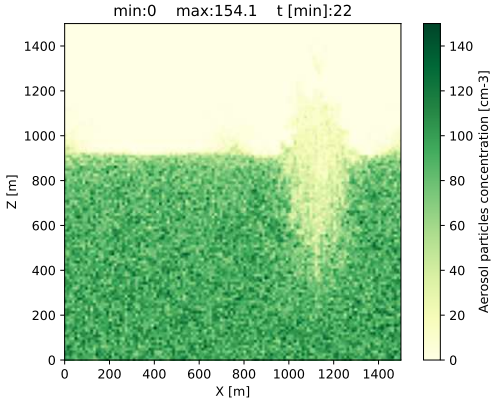
prescribed-flow Sc (incl. virga/CCN resuspension) – PySDM

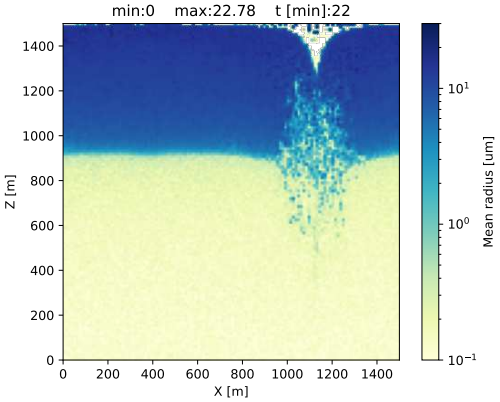




Computational grid: 128x128
Computational particles: 2^{21}

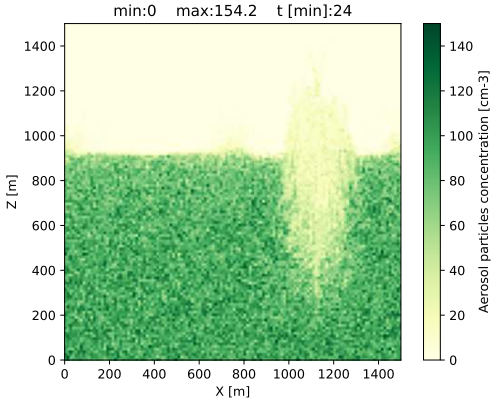
prescribed-flow Sc (incl. virga/CCN resuspension) – PySDM

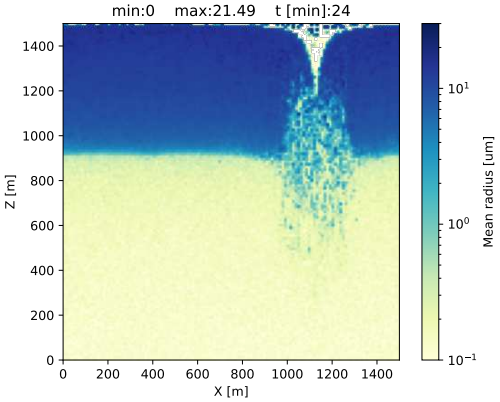




Computational grid: 128x128
Computational particles: 2^{21}

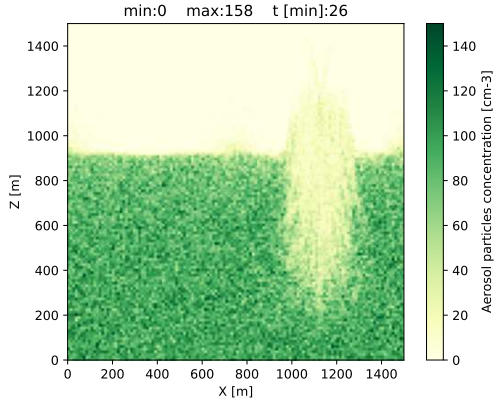
prescribed-flow Sc (incl. virga/CCN resuspension) – PySDM

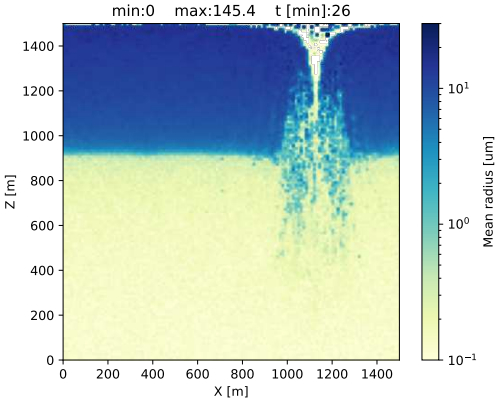




Computational grid: 128x128
Computational particles: 2^{21}

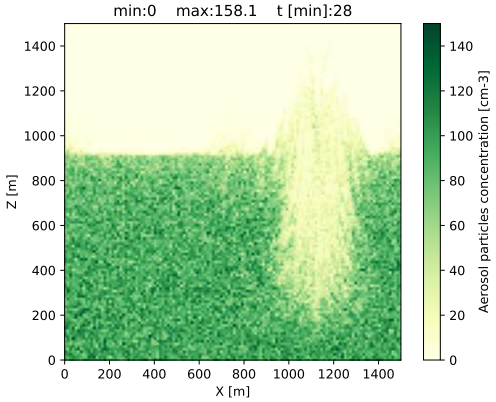
prescribed-flow Sc (incl. virga/CCN resuspension) – PySDM

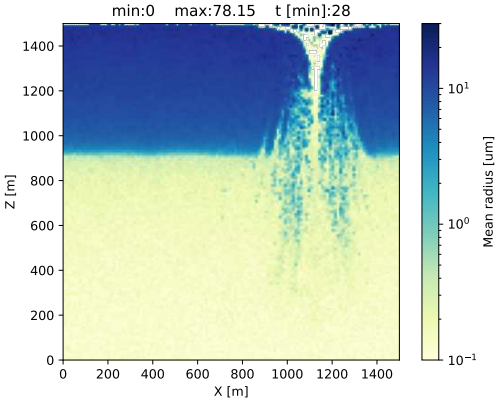




Computational grid: 128x128
Computational particles: 2^{21}

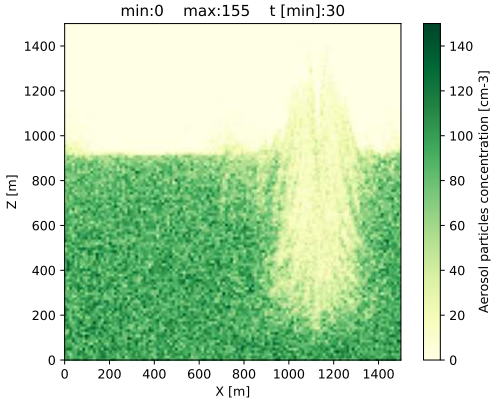
prescribed-flow Sc (incl. virga/CCN resuspension) – PySDM

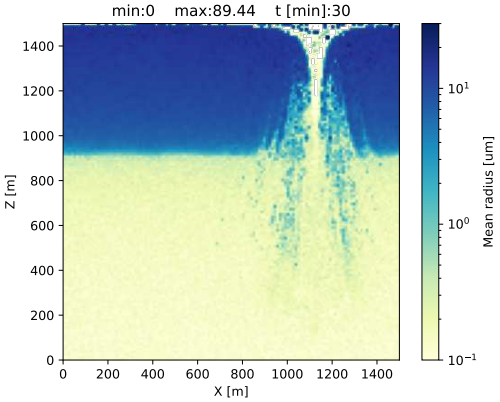




Computational grid: 128x128
Computational particles: 2^{21}

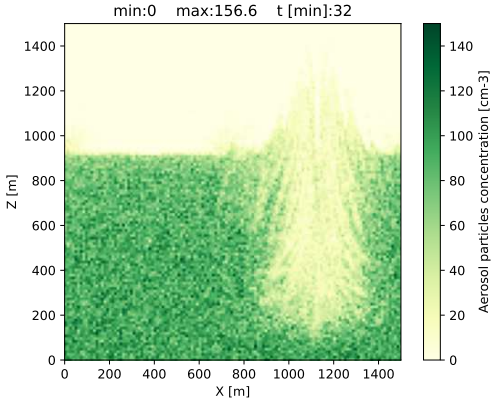
prescribed-flow Sc (incl. virga/CCN resuspension) – PySDM

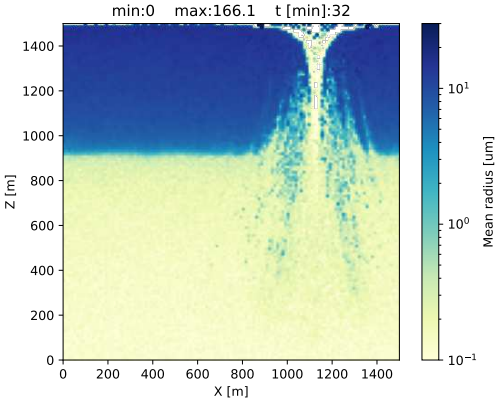




Computational grid: 128x128
Computational particles: 2^{21}

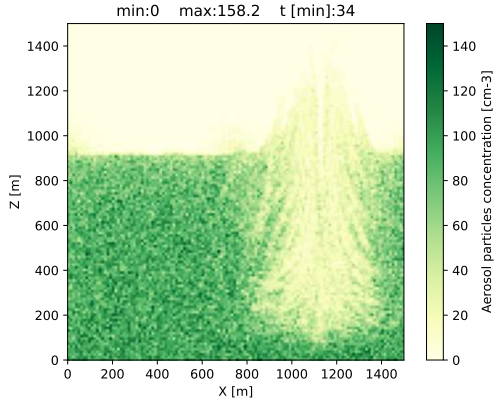
prescribed-flow Sc (incl. virga/CCN resuspension) – PySDM

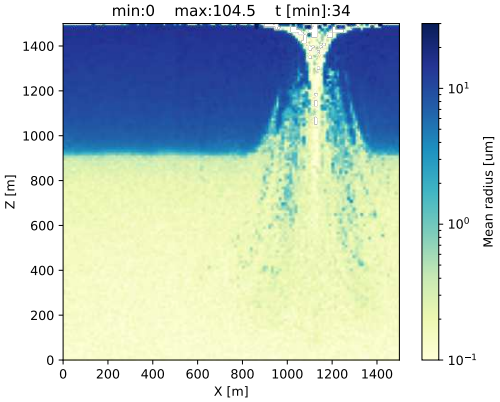




Computational grid: 128x128
Computational particles: 2^{21}

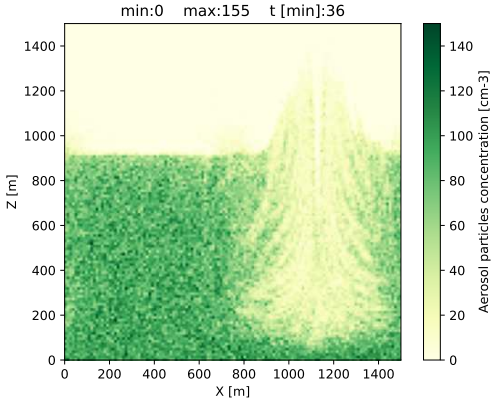
prescribed-flow Sc (incl. virga/CCN resuspension) – PySDM

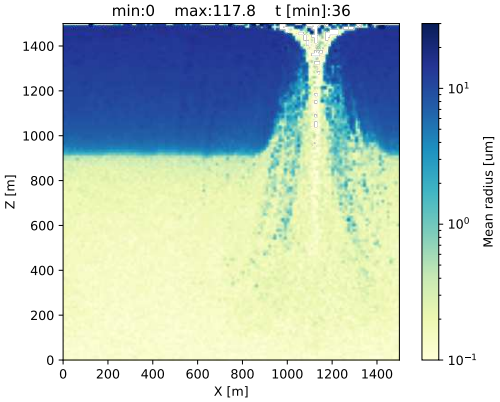




Computational grid: 128x128
Computational particles: 2^{21}

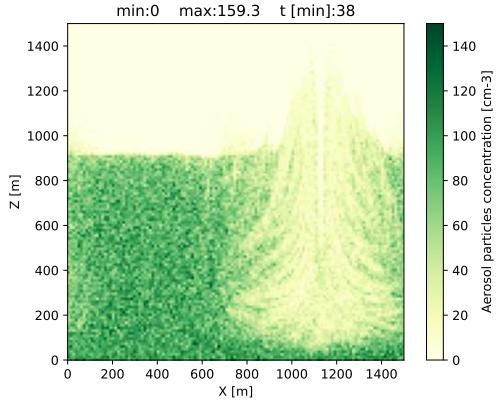
prescribed-flow Sc (incl. virga/CCN resuspension) – PySDM

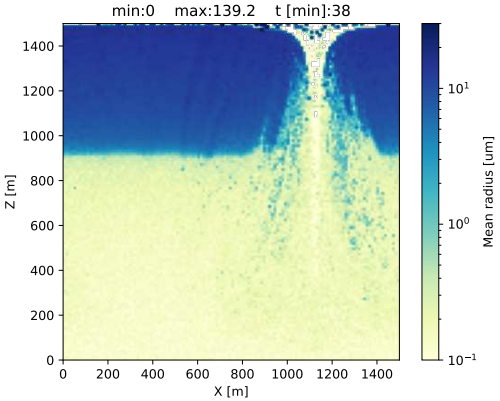




Computational grid: 128x128
Computational particles: 2^{21}

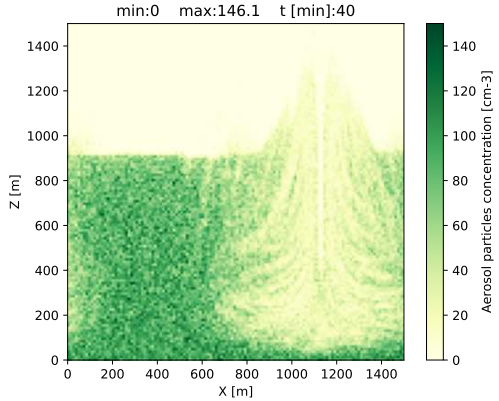
prescribed-flow Sc (incl. virga/CCN resuspension) – PySDM

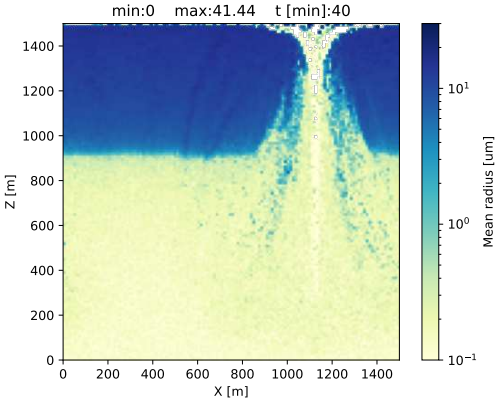




Computational grid: 128x128
Computational particles: 2^{21}

prescribed-flow Sc (incl. virga/CCN resuspension) – PySDM





Computational grid: 128x128
Computational particles: 2^{21}

(super-)particle growth via diffusion and its challenges

(super-)particle growth/evap. modelled via vapour & heat diffusion in air

governing drop-growth law (Howell 1949, Mason 1971, ...):

$$\dot{x} = \frac{\mathrm{d}x}{\mathrm{d}r}\dot{r} = \frac{\mathrm{d}x}{\mathrm{d}r}\,\frac{1}{r}\,\frac{\mathrm{RH}(q,\theta,\rho) - 1 - \frac{a}{r} + \frac{b}{r^3}}{F(q,\theta,\rho)}$$

r – radius of droplet, x – diffusion solver coordinate, RH – relative humidity

a=a(T), b=κr_d^3 - aerosol parameters (e.g., κ-Köhler curve)

F^{-1} - effective diffusion coefficient (incl. ventilation and latent heat effects)

q, θ, ρ – thermodynamic state variable triplet

(super-)particle growth/evap. modelled via vapour & heat diffusion in air

governing drop-growth law (Howell 1949, Mason 1971, ...):

$$\dot{x} = \frac{\mathrm{d}x}{\mathrm{d}r}\dot{r} = \frac{\mathrm{d}x}{\mathrm{d}r}\,\frac{1}{r}\,\frac{\mathrm{RH}(q,\theta,\rho) - 1 - \frac{a}{r} + \frac{b}{r^3}}{F(q,\theta,\rho)}$$

r – radius of droplet, x – diffusion solver coordinate, RH – relative humidity

a=a(T), b=κr_d^3 - aerosol parameters (e.g., κ-Köhler curve)

F^{-1} - effective diffusion coefficient (incl. ventilation and latent heat effects)

q, θ, ρ – thermodynamic state variable triplet

coupled with supersaturation evolution: $\dot{\mathrm{RH}} = f(\dot{r}, \ldots)$

quick demo: hello-world with pypi.org/p/ccnact

Type "/" to search projects

Help Docs Sponsors Log in Register

ccnact 0.0.7

```
pip install ccnact
```

Latest release

Released: May 8, 2026

KISS SciPy-based CCN activation model

Navigation

- Project description
- Release history
- Download files

Verified details

These details have been verified by PyPI

Project links

- Source
- Tracker

GitHub Statistics

- Repository
- Stars: 1
- Forks: 0
- Open issues: 18
- Open PRs: 4

Project description

KISS SciPy-based CCN activation model

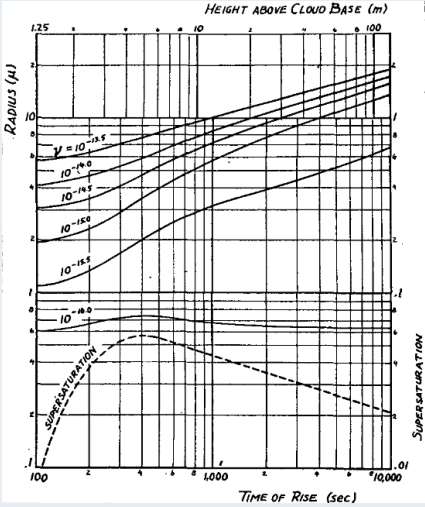
pypi package 0.0.7

overview

`ccnact` is a simple, yet complete, adiabatic/hydrostatic air-parcel framework employing moving-sectional/particle-resolved aerosol-cloud microphysics, featuring:

- integration using SciPy's interface to LSODA stiff ODE solver
- ODE system based on Arabas & Shima 2017 (extended to polydisperse aerosol size spectrum)
- exköhler wet radii equilibration of input dry-size spectrum with SciPy's elementwise root finder
- capability of resolving aerosol activation, deactivation, drop growth, evaporation and ripening
- multi-modal lognormal (using SciPy's stats routines) spectrum specification (concentration at STP)
- portable across platforms and architectures (CI on Linux, macOS & Windows, on Intel & ARM CPUs)
- single-function interface allowing to modify every single constant, and returning a tuple of:
  - concentration of activated droplets (at STP)
  - maximal supersaturation
- effective interfacing options for Matlab, IDL, Julia, etc
- unit-aware implemetation using Pint (dimensional analysis enabled for tests only)
- subsecond execution times for common parameter settings
- KISS design: depends on SciPy, NumPy & Pint only; single ~500 LOC file (physics + setup + tests)

Howell 1949



numerical issues and selected developments

Arnason & Brown '71

numerical analysis of the (stiff) problem

numerical issues and selected developments

Arnason & Brown '71

numerical analysis of the (stiff) problem

Robinson '84

adaptive timesteppig (RK4) to limit workload out of [de]activation zones

numerical issues and selected developments

Arnason & Brown '71

numerical analysis of the (stiff) problem

Robinson '84

adaptive timesteppig (RK4) to limit workload out of [de]activation zones

Kogan '91‘

advective forcing / μ-physical response tendency chunking (bin μ-physics)

numerical issues and selected developments

Arnason & Brown '71

numerical analysis of the (stiff) problem

Robinson '84

adaptive timesteppig (RK4) to limit workload out of [de]activation zones

Kogan '91‘

advective forcing / μ-physical response tendency chunking (bin μ-physics)

Shima et al. '09

implicit-in-size/explicit-in-thermodynamics scheme with $x = r^2$ to alleviate stiffness (particle-local substepping)

numerical issues and selected developments

Arnason & Brown '71

numerical analysis of the (stiff) problem

Robinson '84

adaptive timesteppig (RK4) to limit workload out of [de]activation zones

Kogan '91‘

advective forcing / μ-physical response tendency chunking (bin μ-physics)

Shima et al. '09

implicit-in-size/explicit-in-thermodynamics scheme with $x = r^2$ to alleviate stiffness (particle-local substepping)

Ching et al. '12 (PartMC)

formulation with an analytic Jacobian

numerical issues and selected developments

Arnason & Brown '71

numerical analysis of the (stiff) problem

Robinson '84

adaptive timesteppig (RK4) to limit workload out of [de]activation zones

Kogan '91‘

advective forcing / μ-physical response tendency chunking (bin μ-physics)

Shima et al. '09

implicit-in-size/explicit-in-thermodynamics scheme with $x = r^2$ to alleviate stiffness (particle-local substepping)

Ching et al. '12 (PartMC)

formulation with an analytic Jacobian

Dziekan et al. '19 (libcloudph++/UWLCM)

semi-Lagrangian thermodynamics for diffusional growth (particle-local substepping)

numerical issues and selected developments

Arnason & Brown '71

numerical analysis of the (stiff) problem

Robinson '84

adaptive timesteppig (RK4) to limit workload out of [de]activation zones

Kogan '91‘

advective forcing / μ-physical response tendency chunking (bin μ-physics)

Shima et al. '09

implicit-in-size/explicit-in-thermodynamics scheme with $x = r^2$ to alleviate stiffness (particle-local substepping)

Ching et al. '12 (PartMC)

formulation with an analytic Jacobian

Dziekan et al. '19 (libcloudph++/UWLCM)

semi-Lagrangian thermodynamics for diffusional growth (particle-local substepping)

Shima et al. '20 (SCALE-SDM)

operator splitting for mixed-phase system

numerical issues and selected developments

Arnason & Brown '71

numerical analysis of the (stiff) problem

Robinson '84

adaptive timesteppig (RK4) to limit workload out of [de]activation zones

Kogan '91‘

advective forcing / μ-physical response tendency chunking (bin μ-physics)

Shima et al. '09

implicit-in-size/explicit-in-thermodynamics scheme with $x = r^2$ to alleviate stiffness (particle-local substepping)

Ching et al. '12 (PartMC)

formulation with an analytic Jacobian

Dziekan et al. '19 (libcloudph++/UWLCM)

semi-Lagrangian thermodynamics for diffusional growth (particle-local substepping)

Shima et al. '20 (SCALE-SDM)

operator splitting for mixed-phase system

Bartman '20 (MSc thesis introducing PySDM)

$x = \ln(\text{mass})$ to ensure positive-definite mass/size (resuspension, ripening),

no-mem-overhead adaptivity with cell-local substepping

numerical issues and selected developments

Arnason & Brown '71

numerical analysis of the (stiff) problem

Robinson '84

adaptive timesteppig (RK4) to limit workload out of [de]activation zones

Kogan '91'

advective forcing / μ-physical response tendency chunking (bin μ-physics)

Shima et al. '09

implicit-in-size/explicit-in-thermodynamics scheme with $x = r^2$ to alleviate stiffness (particle-local substepping)

Ching et al. '12 (PartMC)

formulation with an analytic Jacobian

Dziekan et al. '19 (libcloudph++/UWLCM)

semi-Lagrangian thermodynamics for diffusional growth (particle-local substepping)

Shima et al. '20 (SCALE-SDM)

operator splitting for mixed-phase system

Bartman '20 (MSc thesis introducing PySDM)

$x = \ln(\text{mass})$ to ensure positive-definite mass/size (resuspension, ripening),

no-mem-overhead adaptivity with cell-local substepping

Matsushima et al. '23 (SCALE-SDM), Bayley '25 (CLEO)

revisits of the adaptive timestep criteria

numerical issues and selected developments

Arnason & Brown '71

numerical analysis of the (stiff) problem

Robinson '84

adaptive timesteppig (RK4) to limit workload out of [de]activation zones

Kogan '91'

advective forcing / μ-physical response tendency chunking (bin μ-physics)

Shima et al. '09

implicit-in-size/explicit-in-thermodynamics scheme with $x = r^2$ to alleviate stiffness (particle-local substepping)

Ching et al. '12 (PartMC)

formulation with an analytic Jacobian

Dziekan et al. '19 (libcloudph++/UWLCM)

semi-Lagrangian thermodynamics for diffusional growth (particle-local substepping)

Shima et al. '20 (SCALE-SDM)

operator splitting for mixed-phase system

Bartman '20 (MSc thesis introducing PySDM)

$x = \ln(\text{mass})$ to ensure positive-definite mass/size (resuspension, ripening),

no-mem-overhead adaptivity with cell-local substepping

Matsushima et al. '23 (SCALE-SDM), Bayley '25 (CLEO)

revisits of the adaptive timestep criteria

Lüttmer '26 (PySDM)

substepping for ice-phase deposition growth/sublimation/WBF

crux of the problem?

LES timesteps vs. activation timescale

Hoffmann '16

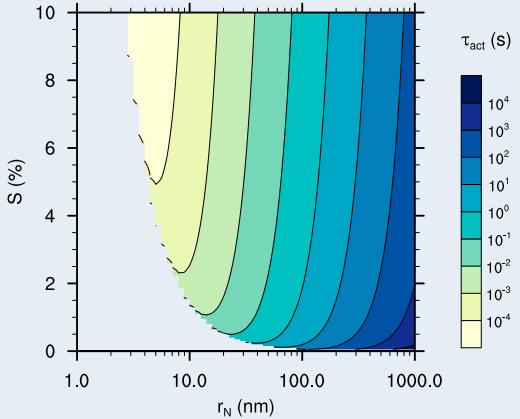


FIG. 2. The activation time scale τ_{act} as a function of dry aerosol radius r_N and supersaturation S. For values of $S < S_{crit}$ (white areas), τ_{act} does not exist.

adaptivity scheme in PySDM

adaptive time stepping for cond/evap in PySDM

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} x_{[i]} \\ \vdots \\ \rho_\mathrm{d} \\ q \\ \theta \end{bmatrix} = \begin{bmatrix} \dot{x}(x_{[i]}, \theta, q, \rho_\mathrm{d}) \\ \vdots \\ \dot{\rho}_\mathrm{d} \\ \dot{q}_\mathrm{cond} + \dot{q}_\mathrm{env} \\ \dot{\theta}_\mathrm{cond} + \dot{\theta}_\mathrm{env} \end{bmatrix}$$

$$m_{[i]} = e^{x_{[i]}}$$

adaptive time stepping for cond/evap in PySDM

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} x_{[i]} \\ \vdots \\ \rho_\mathrm{d} \\ q \\ \theta \end{bmatrix} = \begin{bmatrix} \dot{x}(x_{[i]}, \theta, q, \rho_\mathrm{d}) \\ \vdots \\ \dot{\rho}_\mathrm{d} \\ \dot{q}_\mathrm{cond} + \dot{q}_\mathrm{env} \\ \dot{\theta}_\mathrm{cond} + \dot{\theta}_\mathrm{env} \end{bmatrix}$$

$$m_{[i]} = e^{x_{[i]}}$$

grid-cell tendencies:

$$\dot{\rho}_\mathrm{d} \approx 0$$

$$\dot{q}_{cond} = -\frac{1}{\rho_\mathrm{d}} \frac{1}{\Delta V} \sum_i \xi_{[i]} \frac{dm_{[i]}}{dt}$$

adaptive time stepping for cond/evap in PySDM

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} x_{[i]} \\ \vdots \\ \rho_\mathrm{d} \\ q \\ \theta \end{bmatrix} = \begin{bmatrix} \dot{x}(x_{[i]}, \theta, q, \rho_\mathrm{d}) \\ \vdots \\ \dot{\rho}_\mathrm{d} \\ \dot{q}_\mathrm{cond} + \dot{q}_\mathrm{env} \\ \dot{\theta}_\mathrm{cond} + \dot{\theta}_\mathrm{env} \end{bmatrix}$$

$$m_{[i]} = e^{x_{[i]}}$$

grid-cell tendencies:

$$\dot{\rho}_\mathrm{d} \approx 0$$

$$\dot{q}_{cond} = -\frac{1}{\rho_\mathrm{d}} \frac{1}{\Delta V} \sum_i \xi_{[i]} \frac{dm_{[i]}}{dt}$$

$$\dot{\theta}_\mathrm{cond} = -\frac{l_\mathrm{v} \theta \dot{q}_{cond}}{c_p T}$$

adaptive time stepping for cond/evap in PySDM

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} x_{[i]} \\ \vdots \\ \rho_\mathrm{d} \\ q \\ \theta \end{bmatrix} = \begin{bmatrix} \dot{x}(x_{[i]}, \theta, q, \rho_\mathrm{d}) \\ \vdots \\ \dot{\rho}_\mathrm{d} \\ \dot{q}_\mathrm{cond} + \dot{q}_\mathrm{env} \\ \dot{\theta}_\mathrm{cond} + \dot{\theta}_\mathrm{env} \end{bmatrix}$$

$$m_{[i]} = e^{x_{[i]}}$$

grid-cell tendencies:

$$\dot{\rho}_\mathrm{d} \approx 0$$

$$\dot{q}_{cond} = -\frac{1}{\rho_\mathrm{d}} \frac{1}{\Delta V} \sum_i \xi_{[i]} \frac{dm_{[i]}}{dt}$$

$$\dot{\theta}_\mathrm{cond} = -\frac{l_\mathrm{v} \theta \dot{q}_{cond}}{c_p T}$$

$$\dot{q}_\mathrm{env} = -\rho_\mathrm{d}^{-1} \nabla \cdot (\vec{u} \rho_\mathrm{d} q_\mathrm{v})$$

adaptive time stepping for cond/evap in PySDM

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} x_{[i]} \\ \vdots \\ \rho_\mathrm{d} \\ q \\ \theta \end{bmatrix} = \begin{bmatrix} \dot{x}(x_{[i]}, \theta, q, \rho_\mathrm{d}) \\ \vdots \\ \dot{\rho}_\mathrm{d} \\ \dot{q}_\mathrm{cond} + \dot{q}_\mathrm{env} \\ \dot{\theta}_\mathrm{cond} + \dot{\theta}_\mathrm{env} \end{bmatrix}$$

$$m_{[i]} = e^{x_{[i]}}$$

grid-cell tendencies:

$$\dot{\rho}_\mathrm{d} \approx 0$$

$$\dot{q}_{cond} = -\frac{1}{\rho_\mathrm{d}} \frac{1}{\Delta V} \sum_i \xi_{[i]} \frac{dm_{[i]}}{dt}$$

$$\dot{\theta}_\mathrm{cond} = -\frac{l_\mathrm{v} \theta \dot{q}_{cond}}{c_p T}$$

$$\dot{q}_\mathrm{env} = -\rho_\mathrm{d}^{-1} \nabla \cdot (\vec{u} \rho_\mathrm{d} q_\mathrm{v})$$

$$\dot{\theta}_\mathrm{env} = -\rho_\mathrm{d}^{-1} \nabla \cdot (\vec{u} \rho_\mathrm{d} \theta_\mathrm{d})$$

adaptive time stepping for cond/evap in PySDM

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} x_{[i]} \\ \vdots \\ \rho_\mathrm{d} \\ q \\ \theta \end{bmatrix} = \begin{bmatrix} \dot{x}(x_{[i]}, \theta, q, \rho_\mathrm{d}) \\ \vdots \\ \dot{\rho}_\mathrm{d} \\ \dot{q}_\mathrm{cond} + \dot{q}_\mathrm{env} \\ \dot{\theta}_\mathrm{cond} + \dot{\theta}_\mathrm{env} \end{bmatrix}$$

$$m_{[i]} = e^{x_{[i]}}$$

grid-cell tendencies:

$$\dot{\rho}_\mathrm{d} \approx 0$$

$$\dot{q}_{cond} = -\frac{1}{\rho_\mathrm{d}} \frac{1}{\Delta V} \sum_i \xi_{[i]} \frac{dm_{[i]}}{dt}$$

$$\dot{\theta}_\mathrm{cond} = -\frac{l_\mathrm{v} \theta \dot{q}_{cond}}{c_p T}$$

$$\dot{q}_\mathrm{env} = -\rho_\mathrm{d}^{-1} \nabla \cdot (\vec{u} \rho_\mathrm{d} q_\mathrm{v})$$

$$\dot{\theta}_\mathrm{env} = -\rho_\mathrm{d}^{-1} \nabla \cdot (\vec{u} \rho_\mathrm{d} \theta_\mathrm{d})$$

adiabatic/hydrostatic parcel tendencies:

$$\dot{q}_\mathrm{env} = 0 \qquad \dot{\theta}_\mathrm{env} = 0 \qquad \dot{\rho}_\mathrm{d} \neq 0$$

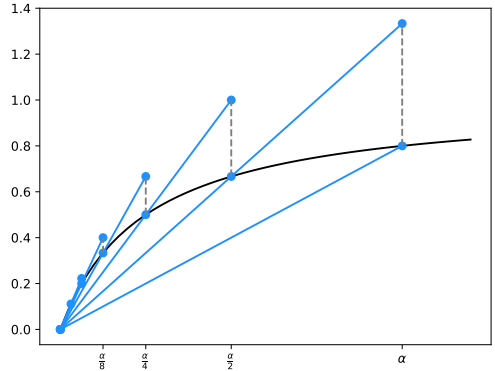
adaptive time stepping for cond/evap in PySDM

In each major time step n, the number of time substeps is adjusted iteratively searching for such $m \in \mathbb{N}$, and as a consequence $\alpha = \frac{1}{2^m}$, for which the following condition holds:

$$\left| \underbrace{\left(\theta^{n,2\alpha} - \theta^{n}\right)}_{\Delta\theta} - 2 \underbrace{\left(\theta^{n,\alpha}\big|_{\alpha} - \theta^{n}\right)}_{\Delta\theta|_{\alpha}} \right| < r_\theta \theta^n$$

where r_θ is the relative tolerance. Note that if only θ is differentiable in the limit of $m \to \infty$, the left-hand side $|\,\Delta\theta|_{2\alpha} - 2\,\Delta\theta|_{\alpha}\,| \to 0$ assuring convergence.

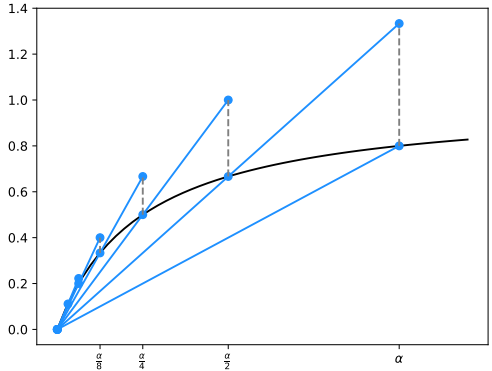
adaptive time stepping for cond/evap in PySDM



conceptual view (Y axis could be θ or supersaturation)

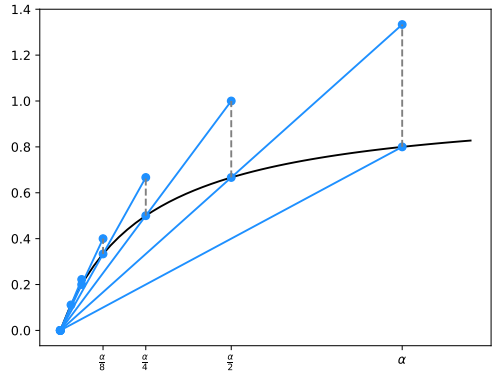
adaptive time stepping for cond/evap in PySDM

- implicit-in-ln(mass) / explicit-in-thermo



conceptual view (Y axis could be θ or supersaturation)

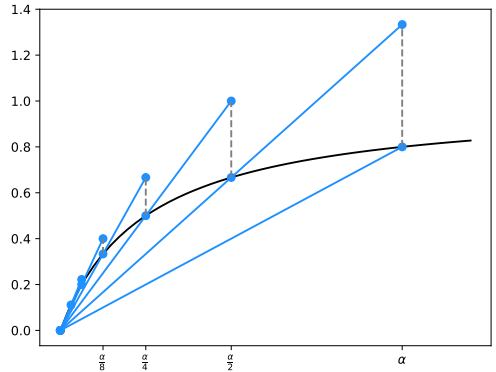
adaptive time stepping for cond/evap in PySDM



conceptual view (Y axis could be θ or supersaturation)

- implicit-in-ln(mass) / explicit-in-thermo
- condensation step adapted:
  - once per major time step
  - in each cell independently (for all particles in the cell)

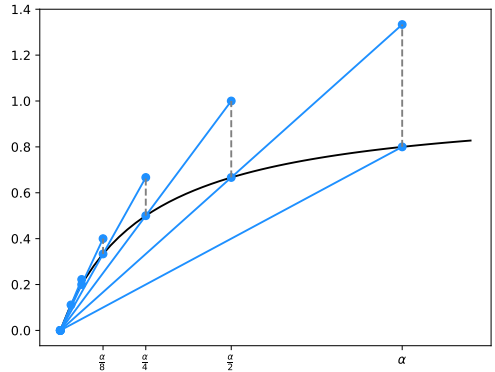
adaptive time stepping for cond/evap in PySDM



conceptual view (Y axis could be θ or supersaturation)

- implicit-in-ln(mass) / explicit-in-thermo
- condensation step adapted:
  - once per major time step
  - in each cell independently (for all particles in the cell)
- no additional memory is required

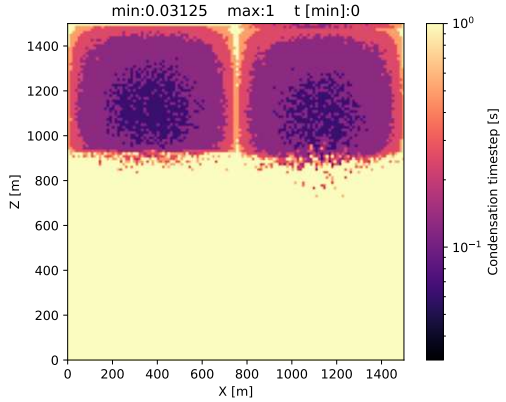
adaptive time stepping for cond/evap in PySDM

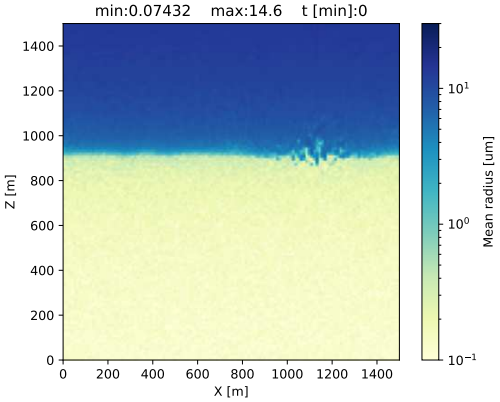


conceptual view (Y axis could be θ or supersaturation)

- implicit-in-ln(mass) / explicit-in-thermo
- condensation step adapted:
  - once per major time step
  - in each cell independently (for all particles in the cell)
- no additional memory is required
- on CPU: cells sorted by prior Δt: $\rightsquigarrow$ avoiding idling threads
- on GPU: global Δt

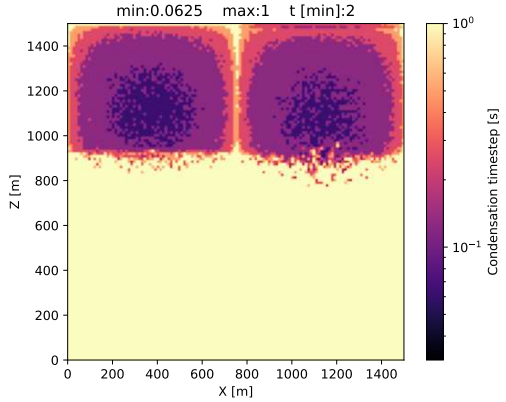
adaptive time stepping for cond/evap in PySDM

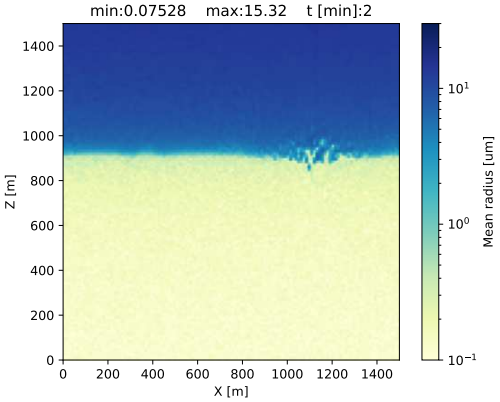




Computational grid: 128x128
Computational particles: 2^{21}

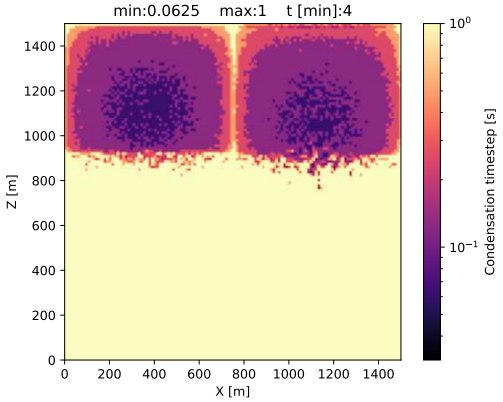
adaptive time stepping for cond/evap in PySDM

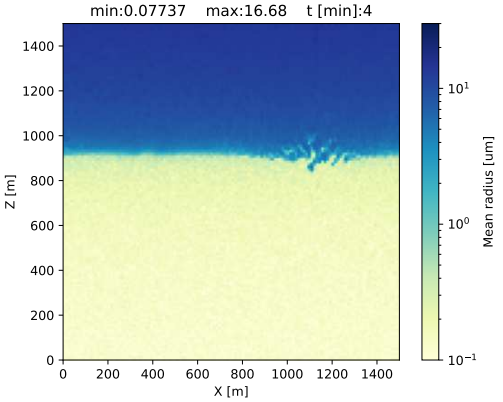




Computational grid: 128x128
Computational particles: 2^{21}

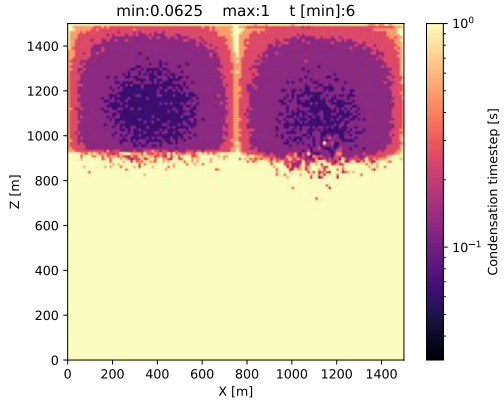
adaptive time stepping for cond/evap in PySDM

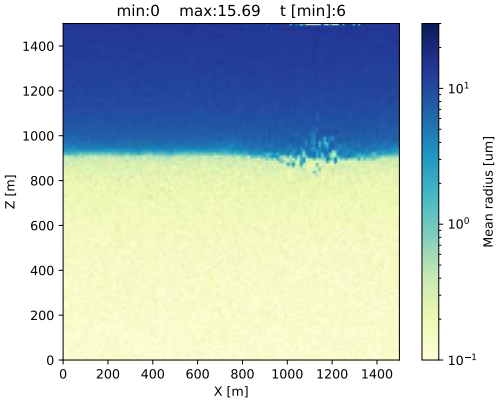




Computational grid: 128x128
Computational particles: 2^{21}

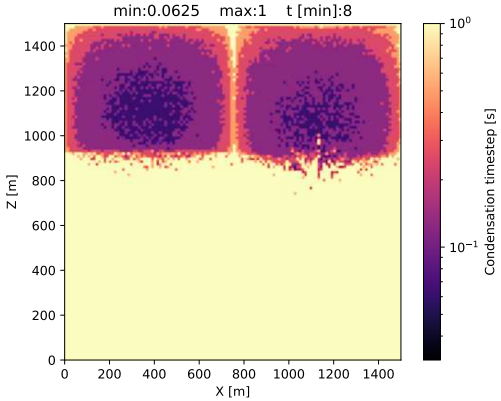
adaptive time stepping for cond/evap in PySDM

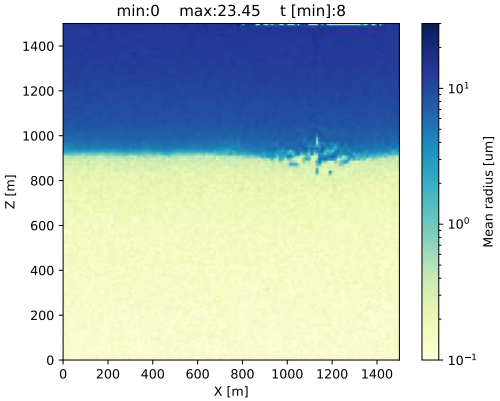




Computational grid: 128x128
Computational particles: 2^{21}

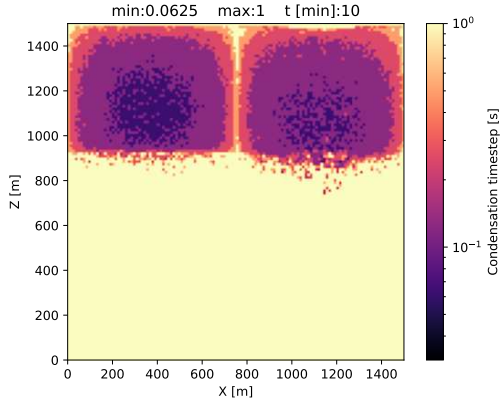
adaptive time stepping for cond/evap in PySDM

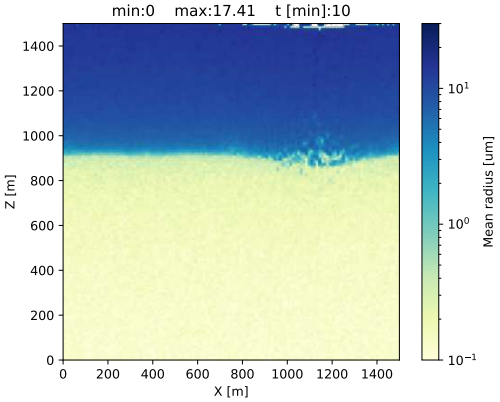




Computational grid: 128x128
Computational particles: 2^{21}

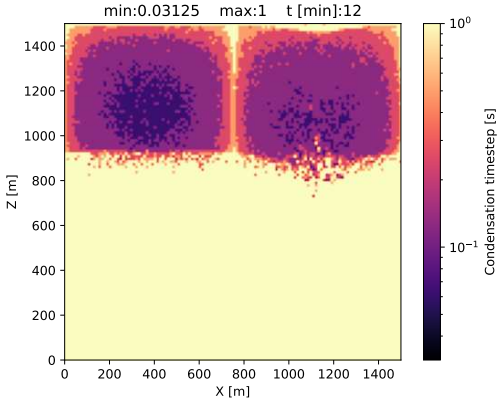
adaptive time stepping for cond/evap in PySDM

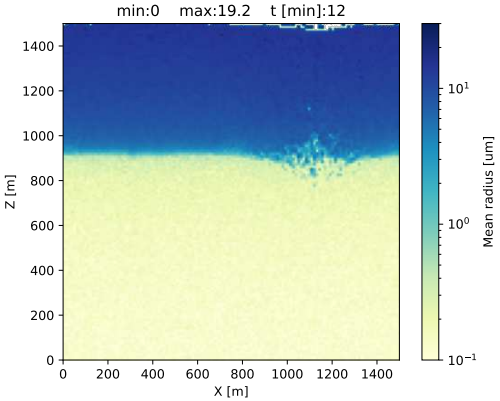




Computational grid: 128x128
Computational particles: 2^{21}

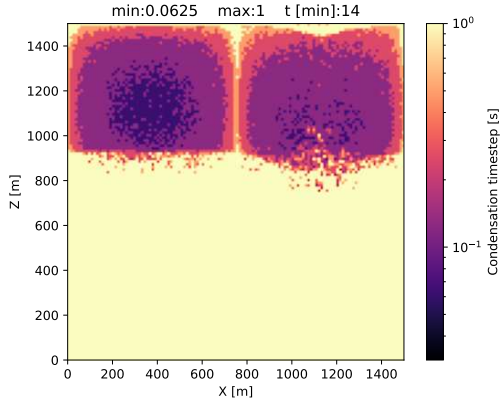
adaptive time stepping for cond/evap in PySDM

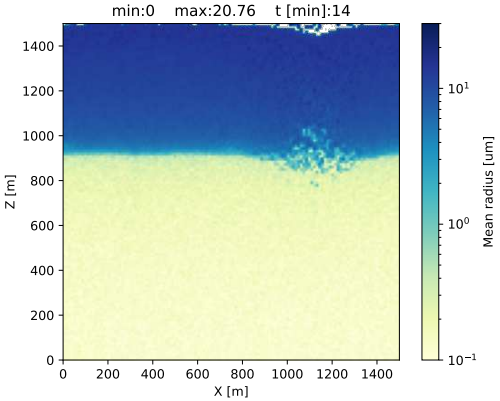




Computational grid: 128x128
Computational particles: 2^{21}

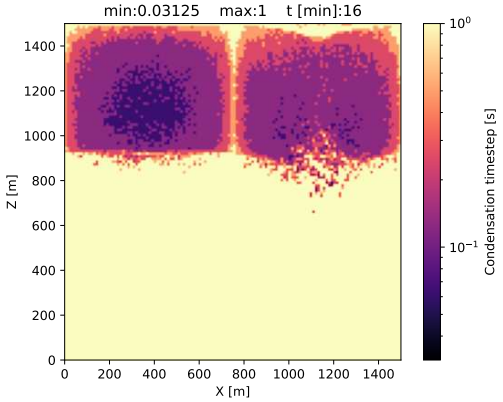
adaptive time stepping for cond/evap in PySDM

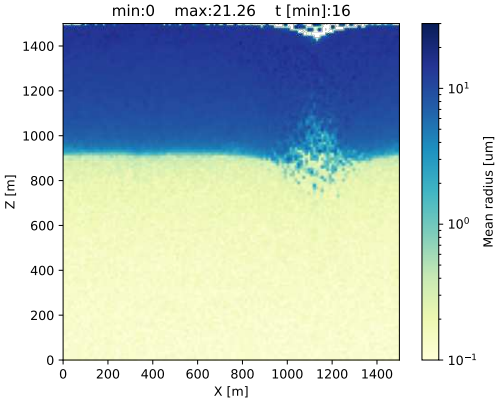




Computational grid: 128x128
Computational particles: 2^{21}

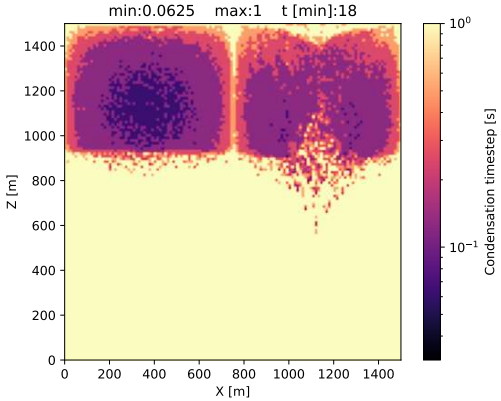
adaptive time stepping for cond/evap in PySDM

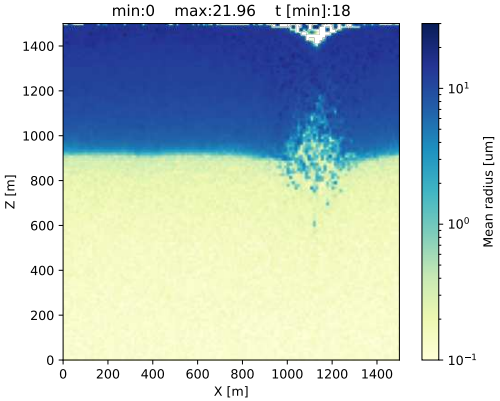




Computational grid: 128x128
Computational particles: 2^{21}

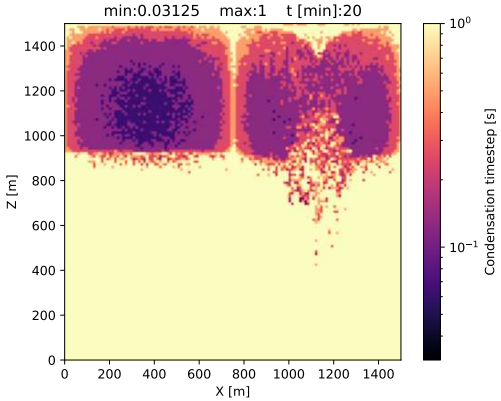
adaptive time stepping for cond/evap in PySDM

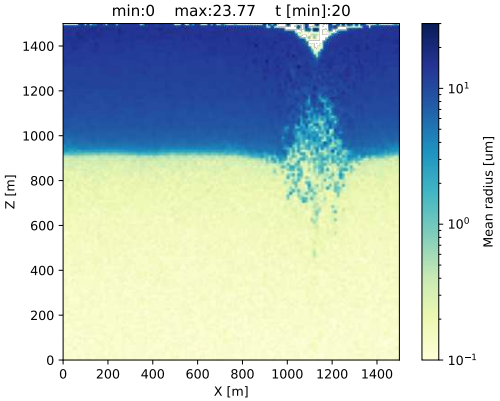




Computational grid: 128x128
Computational particles: 2^{21}

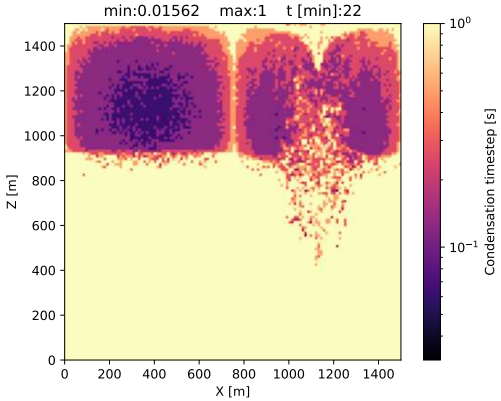
adaptive time stepping for cond/evap in PySDM

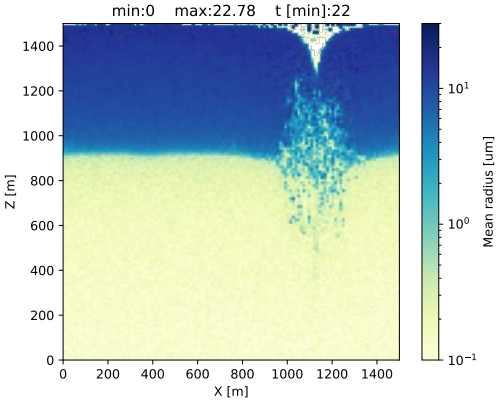




Computational grid: 128x128
Computational particles: 2^{21}

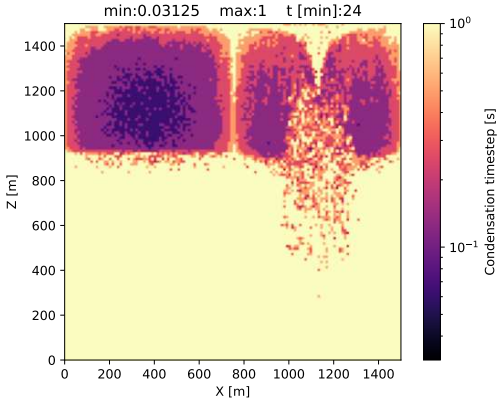
adaptive time stepping for cond/evap in PySDM

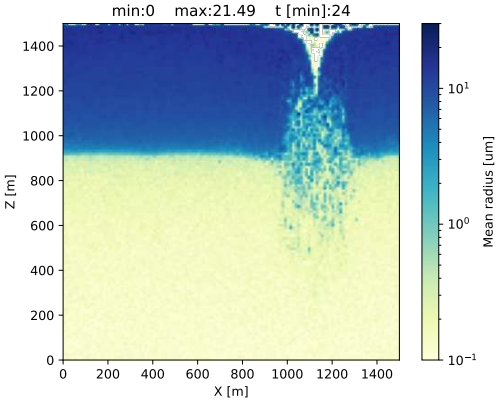




Computational grid: 128x128
Computational particles: 2^{21}

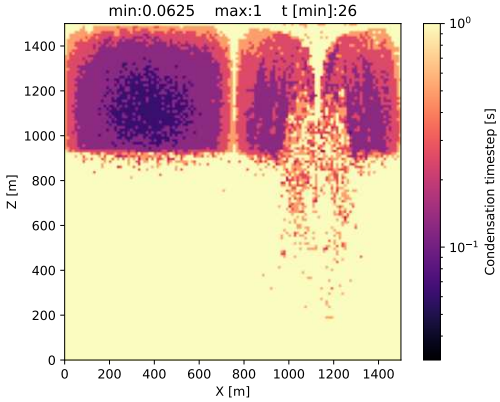
adaptive time stepping for cond/evap in PySDM

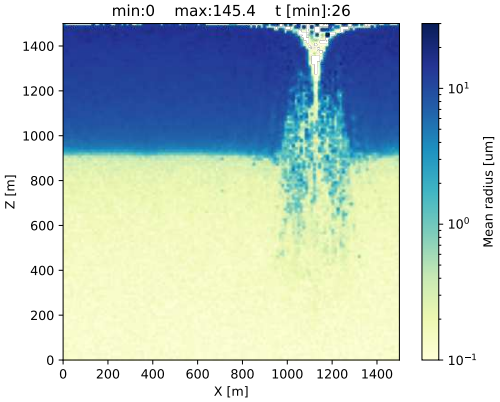




Computational grid: 128x128
Computational particles: 2^{21}

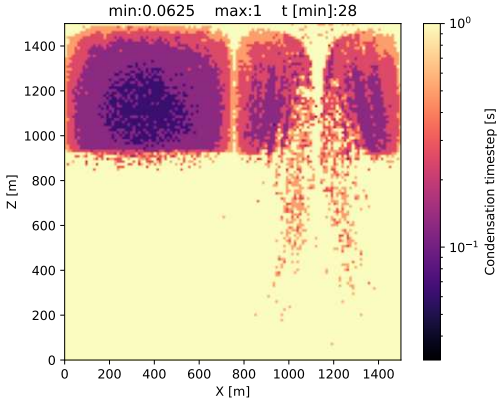
adaptive time stepping for cond/evap in PySDM

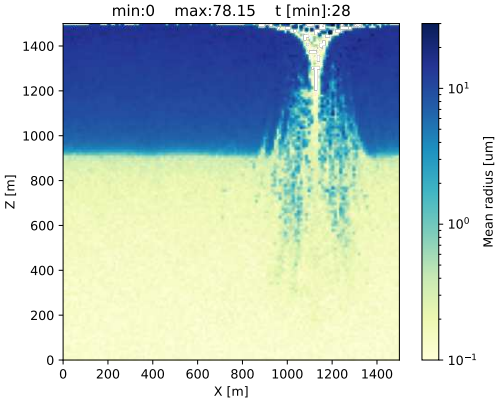




Computational grid: 128x128
Computational particles: 2^{21}

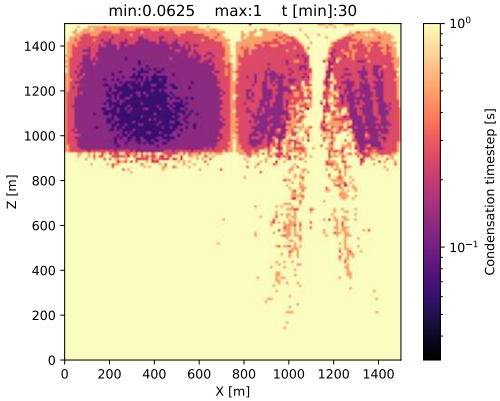
adaptive time stepping for cond/evap in PySDM

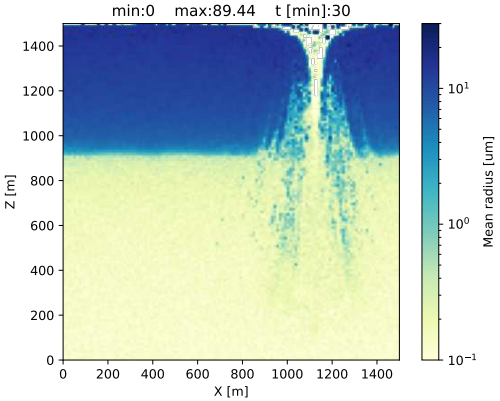




Computational grid: 128x128
Computational particles: 2^{21}

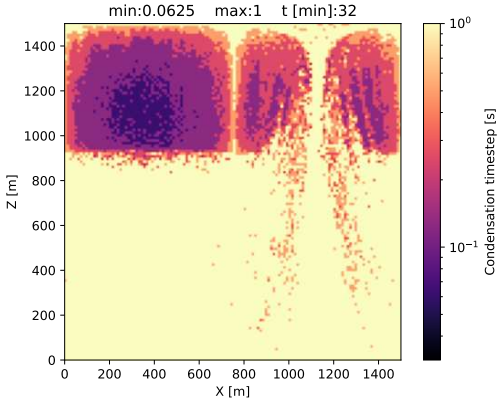
adaptive time stepping for cond/evap in PySDM

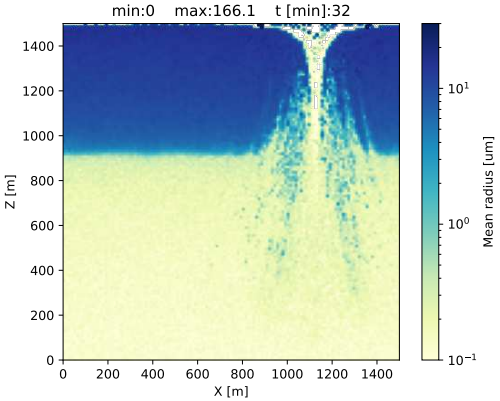




Computational grid: 128x128
Computational particles: 2^{21}

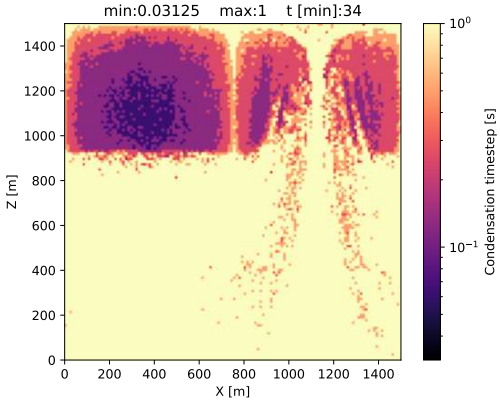
adaptive time stepping for cond/evap in PySDM

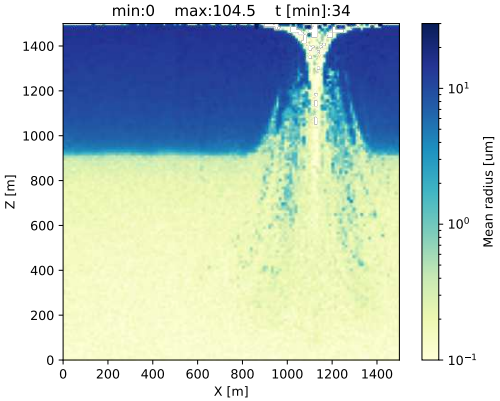




Computational grid: 128x128
Computational particles: 2^{21}

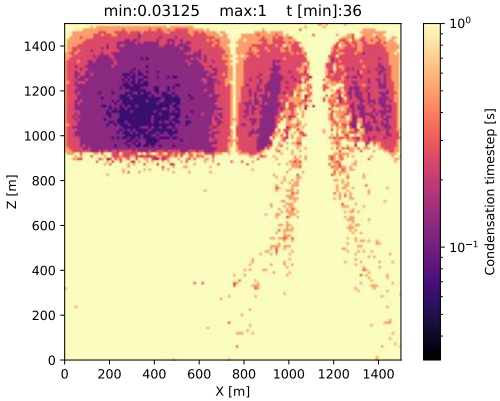
adaptive time stepping for cond/evap in PySDM

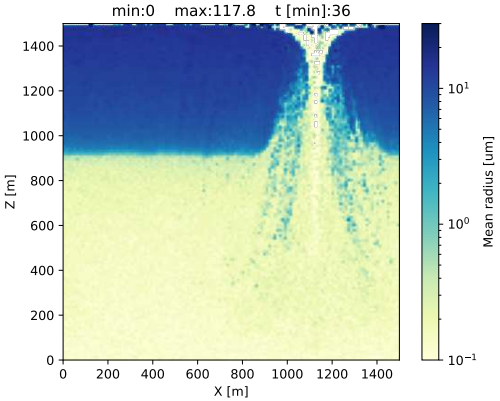




Computational grid: 128x128
Computational particles: 2^{21}

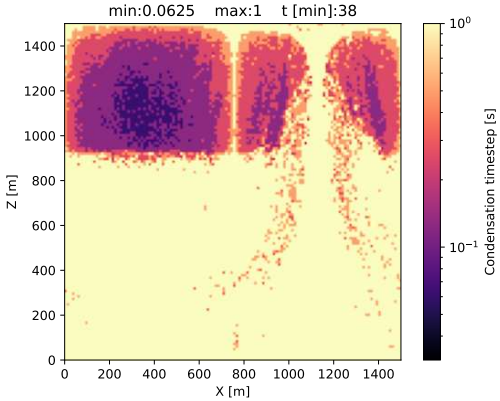
adaptive time stepping for cond/evap in PySDM

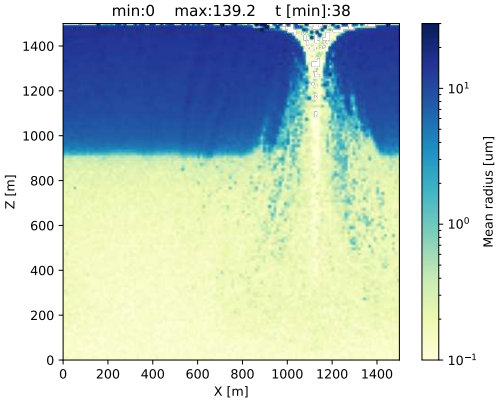




Computational grid: 128x128
Computational particles: 2^{21}

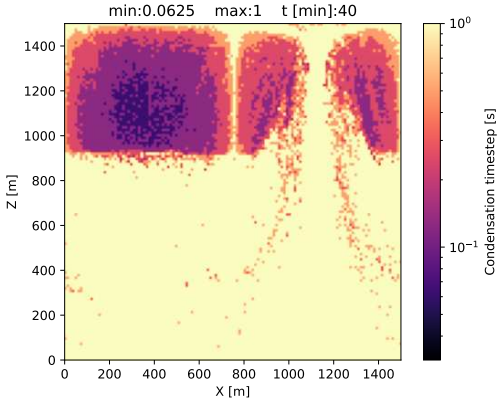
adaptive time stepping for cond/evap in PySDM

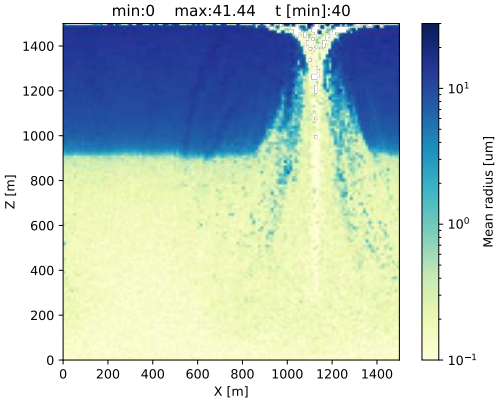




Computational grid: 128x128
Computational particles: 2^{21}

adaptive time stepping for cond/evap in PySDM





Computational grid: 128x128
Computational particles: 2^{21}

related developments

Shima et al. 2020: operator splitting scheme for super-droplets

Table 1 in Shima et al. '20

Table 1. An example of the calculation order when updating the system state from t to $t + \Delta t$. We first calculate the fluid dynamics and then calculate cloud microphysics. Each process is integrated one time step forward at a time. Processes lagging in time are calculated preferentially.

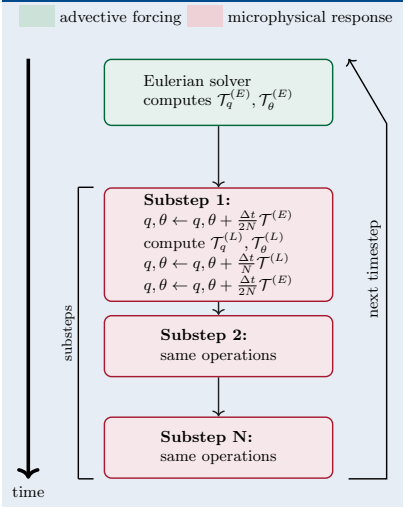
	$t \longrightarrow t + \Delta t$					
Fluid dynamics	1	2	3	4	5	6
Advection	7					
Freezing/melting	8					
Condensation/evaporation	9	12	13	15	17	19
Deposition/sublimation	10		14		18	
Collision–coalescence/ –riming/–aggregation	11			16		

Let Δt_{dyn} be the time step for moist air fluid dynamics.

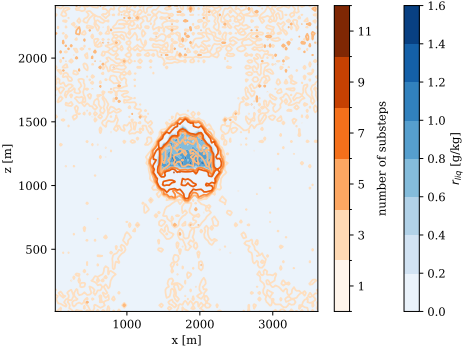
These process time steps are all divisors of the common time step Δt.

We first calculate fluid dynamics without the coupling terms from particles to moist air (Eqs. 76–81), and update moist air from $\boldsymbol{G}_{lmn}(t)$ to $\boldsymbol{G}'_{lmn}(t)$. Then, we update super-particles $\{\{\xi_i, \boldsymbol{x}_i, \boldsymbol{a}_i\}\}$ from t to $t+\Delta t$. We select one elementary cloud microphysics process, integrate it forward by one time step, and then move on to the next process.

Kogan '91 tendency "chunking" (as in UWLCM & PySDM)



cond/evap adaptivity in UWLCM (Piotr Dziekan & Agnieszka Makulska)



- particle-local substepping (i.e. timesteps vary within a cell)
- several adaptivity criteria (incl. option to toggle it by proximity to critical radius)
- tests with 2D warm-bubble setup ✓
- semi-Lagrangian thermodynamics option

https://hanami-project.com/2026/05/20/advancing-cloud-modelling-hanami-microphysics-hpc-optimisation-uwlcm/

Appendix B: Condensation substepping algorithm

Consider condensation of SDs within cell i at time step n. The vector of thermodynamic conditions in that cell at the moment right before condensation is calculated is denoted by $\boldsymbol{\psi}_i^{[n]} = \left(\theta^{[n]}, q_\mathrm{v}^{[n]}\right)_i$. The number of time steps is denoted by S_c and substeps are indexed by ν, starting at $\nu = 1$. Superdroplets within cell i are numbered by μ. The vector of thermodynamic conditions that a given SD experiences at substep ν is denoted by $\check{\boldsymbol{\psi}}_\mu^{[\nu]}$. Using this notation, the substepping algorithm is

$$\check{\boldsymbol{\psi}}_\mu^{[\nu+1/2]} = \check{\boldsymbol{\psi}}_\mu^{[\nu]} + \frac{\boldsymbol{\psi}_i^{[n]} - \check{\boldsymbol{\psi}}_\mu^{[\nu=1]}}{S_\mathrm{c}}, \tag{B1}$$

$$r_\mu^{2[\nu+1]} = r_\mu^{2[\nu]} + \frac{\Delta t}{S_\mathrm{c}} \frac{\mathrm{d}r^2}{\mathrm{d}t}\bigg|_{r_\mu^{2[\nu+1]}, \check{\boldsymbol{\psi}}_\mu^{[\nu+1/2]}}, \tag{B2}$$

$$\check{\boldsymbol{\psi}}_\mu^{[\nu+1]} = \check{\boldsymbol{\psi}}_\mu^{[\nu+1/2]} + \boldsymbol{A} \frac{4}{3} \frac{\pi \rho_\mathrm{w} V}{\rho_\mathrm{d}^\mathrm{r}} \sum_{\mu=1}^{\mu=N_i^{[n]}} \xi_\mu \left[\left(r_\mu^{2[\nu+1]}\right)^{3/2} - \left(r_\mu^{2[\nu]}\right)^{3/2} \right], \tag{B3}$$

where r_μ^2 is the square of the wet radius of the μth SD, $N_i^{[n]}$ is the number of SDs in cell i at time step n and $\boldsymbol{A} = \left(\theta^\mathrm{c} l_\mathrm{v} / \left(c_\mathrm{pd} T^\mathrm{c}\right), -1\right)$. The sum in Eq. (B3) is calculated over all SDs in cell i at time step n.

The initial value $\check{\boldsymbol{\psi}}_\mu^{[\nu=1]}$ is equal to the thermodynamic conditions after condensation finished in the previous time step. Two ways of defining $\check{\boldsymbol{\psi}}_\mu^{[\nu=1]}$ are considered that differ regarding the spatial cell from which this initial condition is diagnosed:

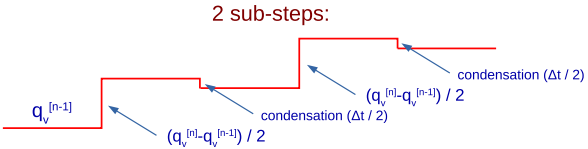
$$\check{\boldsymbol{\psi}}_\mu^{[\nu=1]} = \left(\boldsymbol{\psi}^{[n-1]} + R_\mathrm{c}^{[n-1]}\right)_{i(n-1)}, \tag{B5}$$

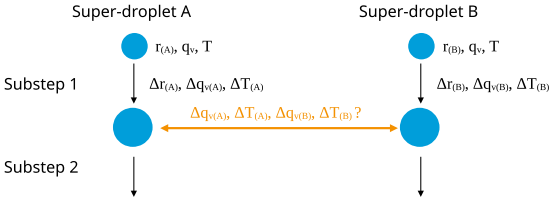
referred to as "per-particle" substepping, and

$$\check{\boldsymbol{\psi}}_\mu^{[\nu=1]} = \left(\boldsymbol{\psi}^{[n-1]} + R_\mathrm{c}^{[n-1]}\right)_{i(n)}, \tag{B6}$$

a procedure we call "per-cell" substepping. The notation $i\,(n)$ stands for the index of the cell in which the μth SD was at time step n. The per-cell substepping is less accurate, but requires less computational time and uses less memory. The reason for this is that in the per-cell method, all SDs in a given cell have the same values of $\check{\boldsymbol{\psi}}_\mu^{[\nu]}$.

the issue of "mixing" (Piotr Dziekan & Agnieszka Makulska)





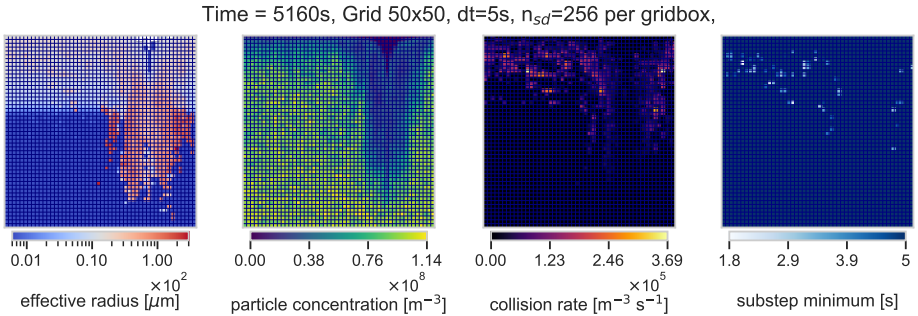
- mean distance between droplets: ∼2 mm
- turbulent mixing timescale at a 2 mm distance: ∼0.1 s
- turbulent mixing timescale at a 1 cm distance: ∼0.5 s
- no clear answer if it's better to "mix" (homogenize) cell every substep or every step?

what about coagulation?

Ware et al. 2025 (just accepted in JAMES!): arXiv:2509.05536 – PySDM

Adaptive time-stepping for the Super-Droplet Method Monte Carlo collision-coalescence scheme

Emma Ware[1*✉], Piotr Bartman-Szwarc[2], Adele L. Igel[1], and Sylwester Arabas[3]



Thank you for your attention!

sylwester.arabas@agh.edu.pl

`open-atmos-krk.github.io`
(incl. ccnact – contributions welcome!)