PySDM: particle-based cloud modeling package with CPU and GPU number-crunching backends

Sylwester Arabas[1,2]

UIUC Atmospheric Sciences Seminar, 7 Sept. 2021

[1] postdoc @ Nicole Riemer's group here at atmos.illinois.edu
[2] on leave @ Faculty of Math. & CS at uj.edu.pl

PySDM: particle-based cloud modeling package with CPU and GPU number-crunching backends

Sylwester Arabas

co-authors & contributors:
@uj.edu.pl: **P. Bartman**, O. Bulenok, G. Łazarski, M. Olesik, ...
@caltech.edu: A. Jaruga, C. Singer, ...

Jagiellonian University, Kraków, Poland

Jagiellonian University, Kraków, Poland

- founded in 1364, among 20 world oldest (in cont. operation)

Jagiellonian University, Kraków, Poland

- founded in 1364, among 20 world oldest (in cont. operation)
- ca. 40 000 students, 7000 staff (4000 acad.), 16 faculties

Jagiellonian University, Kraków, Poland

- founded in 1364, among 20 world oldest (in cont. operation)
- ca. 40 000 students, 7000 staff (4000 acad.), 16 faculties
- American Studies since 1991

Jagiellonian University, Kraków, Poland

- founded in 1364, among 20 world oldest (in cont. operation)
- ca. 40 000 students, 7000 staff (4000 acad.), 16 faculties
- American Studies since 1991
- host to Smoluchowski Institute of Physics

Jagiellonian University, Kraków, Poland

- founded in 1364, among 20 world oldest (in cont. operation)
- ca. 40 000 students, 7000 staff (4000 acad.), 16 faculties
- American Studies since 1991
- host to Smoluchowski Institute of Physics
- 1917 Smoluchowski elected as Rector (professor since 1913)

Plan of the talk

PySDM: context

PySDM: statement of need & goals

PySDM: tour of the features

PySDM: demo (role play: reviewer)

PySDM: technological stack

context: aerosol-cloud-precipitation interactions (scales!)

"Cloud and ship. Ukraine, Crimea, Black sea, view from Ai-Petri mountain"
(photo: Yevgen Timashov / National Geographic)

Smoluchowski's coagulation equation (SCE)

concentration of particles of size x at time t: $c(x,t)$: $\mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}^+$
collision kernel: $a(x_1, x_2)$: $\mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}^+$

Smoluchowski's coagulation equation (SCE)

concentration of particles of size x at time t: $c(x,t)\colon \mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}^+$
collision kernel: $a(x_1, x_2)\colon \mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}^+$

$$\dot{c}(x) = \frac{1}{2}\int_0^x a(y, x-y)c(y)c(x-y)dy - \int_0^\infty a(y,x)c(y)c(x)dy \quad (1)$$

Smoluchowski's coagulation equation (SCE)

concentration of particles of size x at time t: $c(x,t)$: $\mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}^+$
collision kernel: $a(x_1, x_2)$: $\mathbb{R}^+ \times \mathbb{R}^+ \to \mathbb{R}^+$

$$\dot{c}(x) = \frac{1}{2}\int_0^x a(y, x-y)c(y)c(x-y)dy - \int_0^\infty a(y,x)c(y)c(x)dy \tag{1}$$

discretised particle concentration: $c_i = c(x_i)$ where $x_i = i \cdot x_0$

$$\dot{c}_i = \frac{1}{2}\sum_{k=1}^{i-1} a(x_k, x_{i-k})c_k c_{i-k} - \sum_{k=1}^{\infty} a(x_k, x_i)c_k c_i \tag{2}$$

cloud droplet collisional growth

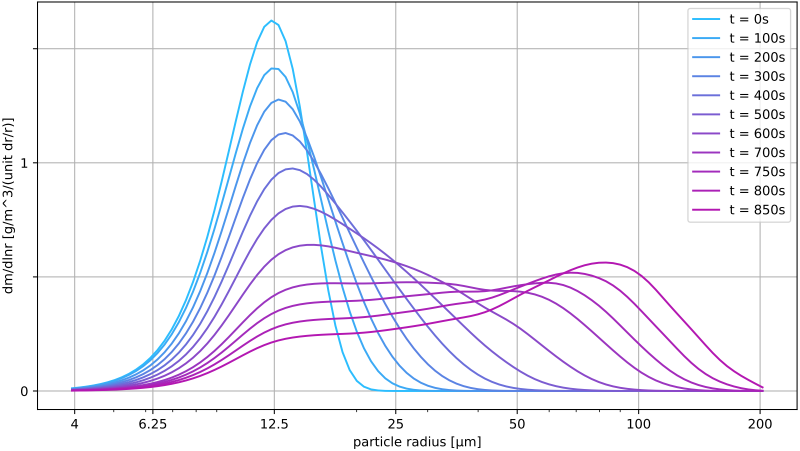


figure (PySDM simulation): Bartman, Arabas et al. 2021, LNCS (doi:10.1007/978-3-030-77964-1_2)

SCE: challenges/problems

- analytic solutions known only for simple kernels

SCE: challenges/problems

- analytic solutions known only for simple kernels
- numerical methods suffer from the curse of dimensionality
  when distinguishing particles of same size but different properties

SCE: challenges/problems

- analytic solutions known only for simple kernels
- numerical methods suffer from the curse of dimensionality
  when distinguishing particles of same size but different properties
- assumptions behind SCE difficult to meet in practice, e.g.:

SCE: challenges/problems

- analytic solutions known only for simple kernels
- numerical methods suffer from the curse of dimensionality
  when distinguishing particles of same size but different properties
- assumptions behind SCE difficult to meet in practice, e.g.:
  it is assumed that the system is large enough and the droplets inside are uniformly distributed, which in turn is only true for a small volume in the atmosphere

SCE: challenges/problems

- analytic solutions known only for simple kernels
- numerical methods suffer from the curse of dimensionality
  when distinguishing particles of same size but different properties
- assumptions behind SCE difficult to meet in practice, e.g.:
  it is assumed that the system is large enough and the droplets inside are uniformly distributed, which in turn is only true for a small volume in the atmosphere
- . . .

Monte-Carlo SCE alternatives: e.g., SDM by Shima et al.

Shima et al. 2009 (doi:10.1002/qj.441): warm-rain

Monte-Carlo SCE alternatives: e.g., SDM by Shima et al.

Shima et al. 2009 (doi:10.1002/qj.441): warm-rain

Shima et al. 2020 (doi:10.5194/gmd-13-4107-2020): mixed-phase

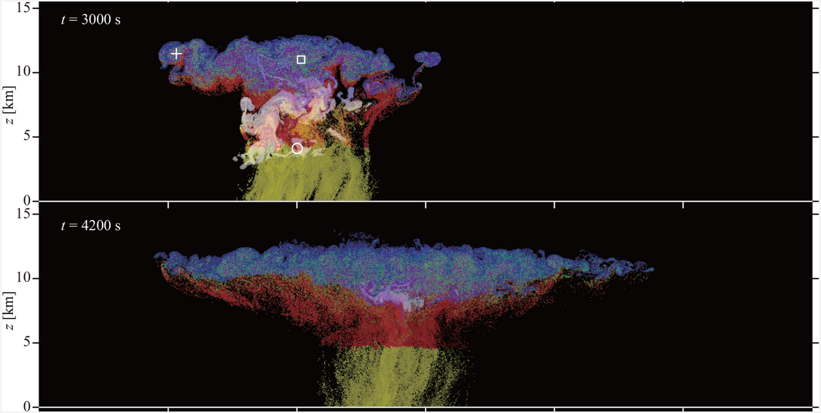


Figure 1. Typical realization of CTRL cloud spatial structures at $t = 2040$, 2460, 3000, 4200, and 5400 s. The mixing ratio of cloud water, rainwater, cloud ice, graupel, and snow aggregates are plotted in fading white, yellow, blue, red, and green, respectively. The symbols indicate examples of unrealistic predicted ice particles (Sects. 7.3 and 9.1). See also Movie 1 in the video supplement.

Super Droplet Method vs. SCE: differences

method type	
mean-field, deterministic	Monte-Carlo, stochastic

considered pairs	
all (i,j) pairs	random set of $n_{sd}/2$ non-overlapping pairs, probability up-scaled by $(n_{sd}^2 - n_{sd})/2$ to $n_{sd}/2$ ratio

computation complexity	
$\mathcal{O}(n_{sd}^2)$	$\mathcal{O}(n_{sd})$

Super Droplet Method vs. SCE: differences

collisions triggered	
every time step	by comparing probability with a random number

collisions	
colliding a fraction of $\xi_{[i]}, \xi_{[j]}$	collide all of $\min\{\xi_{[i]}, \xi_{[j]}\}$ (all or nothing)

Super Droplet Method vs. SCE: differences

collisions triggered	
every time step	by comparing probability with a random number

collisions	
colliding a fraction of $\xi_{[i]}, \xi_{[j]}$	collide all of $\min\{\xi_{[i]}, \xi_{[j]}\}$ (all or nothing)

interpretation	
concentration "c_i" in size bin "i"	besides c_i, each "particle" i carries other physicochemical attributes incl. position in space (x_i, y_i, z_i)

Super Droplet Method vs. SCE: differences

collisions triggered	
every time step	by comparing probability with a random number

collisions	
colliding a fraction of $\xi_{[i]}, \xi_{[j]}$	collide all of $\min\{\xi_{[i]}, \xi_{[j]}\}$ (all or nothing)

interpretation	
concentration "c_i" in size bin "i"	besides c_i, each "particle" i carries other physicochemical attributes incl. position in space (x_i, y_i, z_i)

in aerosol community: DeVille, Riemer & West 2011:
Weighted Flow Algorithms (WFA) for stochastic particle coagulation

Confronting the Challenge of Modeling Cloud and Precipitation Microphysics

Hugh Morrison, Marcus van Lier-Walqui, Ann M. Fridlind, Wojciech W. Grabowski, Jerry Y. Harrington, Corinna Hoose, Alexei Korolev, Matthew R. Kumjian, Jason A. Milbrandt, Hanna Pawlowska, Derek J. Posselt, Olivier P. Prat, Karly J. Reimel, Shin-Ichiro Shima, Bastiaan van Diedenhoven, Lulin Xue

super-particles as an alternative to bulk or bin μ-phyics

Confronting the Challenge of Modeling Cloud and Precipitation Microphysics

Hugh Morrison, Marcus van Lier-Walqui, Ann M. Fridlind, Wojciech W. Grabowski, Jerry Y. Harrington, Corinna Hoose, Alexei Korolev, Matthew R. Kumjian, Jason A. Milbrandt, Hanna Pawlowska, Derek J. Posselt, Olivier P. Prat, Karly J. Reimel, Shin-Ichiro Shima, Bastiaan van Diedenhoven, Lulin Xue

AGU ADVANCING EARTH AND SPACE SCIENCE
Journal of Advances in Modeling Earth Systems 10.1029/2019MS001689

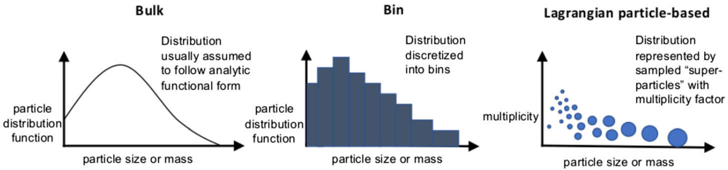


Figure 3. Representation of cloud and precipitation particle distributions in the three main types of microphysics schemes: Bulk (left), bin (center), and particle-based Lagrangian (right). The horizontal axes show particle diameter or mass, and the vertical axes show the number density distribution for the bulk and bin diagrams and "multiplicity" for the Lagrangian particle-based diagram, which is the actual number of particles that each super-particle represents. The size of the blue super-particles in this diagram represents the size or mass of a super-particle. Note that almost all current bulk schemes represent particle distributions using analytic functions, although some earlier schemes did not make any assumptions about the cloud particle distribution and only considered bulk cloud water content.

SDM

PySDM

Plan of the talk

PySDM: context

PySDM: statement of need & goals

PySDM: tour of the features

PySDM: demo (role play: reviewer)

PySDM: technological stack

Summer 2021 news

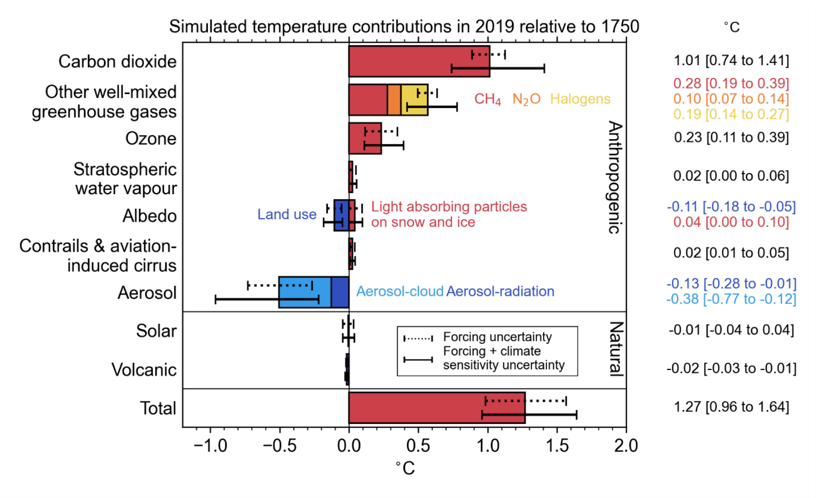


Figure 7.7: The contribution of forcing agents to 2019 temperature change relative to 1750 produced using the two-layer emulator (Supplementary Material 7.SM.2), constrained to assessed ranges for key climate metrics described in Cross-Chapter Box 7.1.

Summer 2021 news

IPCC AR6 WGI: Chapter 7

Final Government Draft Chapter 7 IPCC AR6 WGI

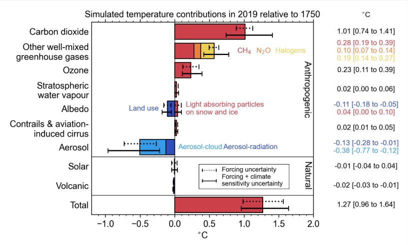


Figure 7.7: The contribution of forcing agents to 2019 temperature change relative to 1750 produced using the two-layer emulator (Supplementary Material 7.SM.2), constrained to assessed ranges for key climate metrics described in Cross-Chapter Box 7.1.

Summer 2021 news

IPCC AR6 WGI: Chapter 7

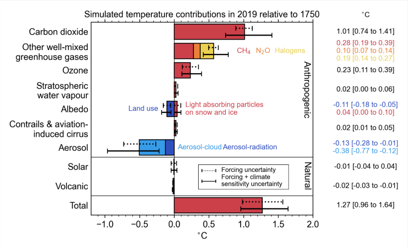


Figure 7.7: The contribution of forcing agents to 2019 temperature change relative to 1750 produced using the two-layer emulator (Supplementary Material 7.SM.2), constrained to assessed ranges for key climate metrics described in Cross-Chapter Box 7.1.

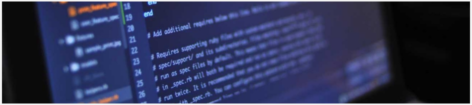
AMS Statement on Software Preservation, Stewardship and Reuse

AMS American Meteorological Society

MEMBERSHIP & GIVING | JOURNALS & PUBLICATIONS | MEETINGS & EVENTS | EDUCATION & CAREERS | POLICY PROGRAM | GET INVOLVED

Home › About AMS › AMS Statements › Statements of the AMS in Force › Software Preservation, Stewardship, and Reuse

Software Preservation, Stewardship, and Reuse



DOWNLOAD PDF

A Professional Guidance Statement of the American Meteorological Society

Adopted by the AMS Council on 16 July 2021

Motivation

Software is an essential component in driving scientific and technical advances in the atmospheric and oceanic sciences, leading to broader societal benefits. Society now relies upon software tools to assist in planning for daily life, improving the efficiency of economic activities, and saving lives when faced with pending natural disasters such as hurricanes. Modern numerical weather prediction (NWP) and ocean circulation models, which provide the foundation for environmental prediction, are essentially software products arising from decades of scientific research. As computational capacity and the complexity of observational networks increase, stewardship of software resulting from research is imperative in many cases. In order to build upon and further the knowledge that has been characterized within current software tools, the community is now expected to produce and curate software that is equitably accessible and easier to be reused by others. Equitable access to software that was used to discover the most recent research findings avoids wasteful duplication of efforts and provides an opportunity for any researcher to more easily build upon the work of others.

PySDM: goals (stimulating overlap with PartMC!)

Develop an implementation of the SDM algorithm:

PySDM: goals (stimulating overlap with PartMC!)

Develop an implementation of the SDM algorithm:

- applicable in research on aerosol-cloud-interactions (and beyond)
  KPI: reproduction of results from classic and recent literature

PySDM: goals (stimulating overlap with PartMC!)

Develop an implementation of the SDM algorithm:

- applicable in research on aerosol-cloud-interactions (and beyond)
  KPI: reproduction of results from classic and recent literature
- **easy to reuse**: code (Python), examples (Jupyter), extensibility (modular, high test coverage), interoperability (other languages, i/o), leveraging modern hardware (GPUs, multi-core CPUs)
  KPI: user feedback & contributions

PySDM: goals (stimulating overlap with PartMC!)

Develop an implementation of the SDM algorithm:

- applicable in research on aerosol-cloud-interactions (and beyond)
  KPI: reproduction of results from classic and recent literature
- **easy to reuse**: code (Python), examples (Jupyter), extensibility (modular, high test coverage), interoperability (other languages, i/o), leveraging modern hardware (GPUs, multi-core CPUs)
  KPI: user feedback & contributions
- **accessibility**: seamless Linux/macOS/Windows installation (pip)
  KPI: continuous integration on all targeted platforms

PySDM: goals (stimulating overlap with PartMC!)

Develop an implementation of the SDM algorithm:

- applicable in research on aerosol-cloud-interactions (and beyond)
  KPI: reproduction of results from classic and recent literature
- **easy to reuse**: code (Python), examples (Jupyter), extensibility (modular, high test coverage), interoperability (other languages, i/o), leveraging modern hardware (GPUs, multi-core CPUs)
  KPI: user feedback & contributions
- **accessibility**: seamless Linux/macOS/Windows installation (pip)
  KPI: continuous integration on all targeted platforms
- **curation**: open licensing (GPL), public versioned development (Github)
  KPI: instant and anonymous execution on commodity environment

Plan of the talk

PySDM: context

PySDM: statement of need & goals

PySDM: tour of the features

PySDM: demo (role play: reviewer)

PySDM: technological stack

github.com/atmos-clous-sim-uj

Atmospheric Cloud Simulation Group @ Jagiellonian University

Repositories 9 | Packages | People 11 | Teams 2 | Projects 1 | Settings

Pinned repositories

Customize pinned repositories

PySDM

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

Python ☆ 17 Forks 14

PyMPDATA

Forked from piotrbartman/PyMPDATA

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

Python ☆ 5 Forks 7

numba-mpi

Numba @njittable MPI wrappers tested on Linux, macOS and Windows

Python Forks 2

PySDM-examples

PySDM usage examples (mostly reproducing results from literature) depicting how to use PySDM in Python, in particular from Jupyter notebooks

Jupyter Notebook ☆ 1 Forks 4

PyMPDATA-examples

PyMPDATA usage examples (mostly reproducing results from literature) depicting how to use PyMPDATA in Python, in particular from Jupyter notebooks

Jupyter Notebook ☆ 1 Forks 3

PySDM: backends, dynamics & environments

PySDM: backends, dynamics & environments

"backends"

- CPU (Numba/LLVM)
- GPU (ThrustRTC/CUDA)

PySDM: backends, dynamics & environments

"backends"

- CPU (Numba/LLVM)
- GPU (ThrustRTC/CUDA)

"dynamics"

- coalescence (SDM + dt-adaptivity)
- condensation (dt-adaptive, bespoke semi-implicit ODE solver)
- displacement (incl. sedimentation)
- aqueous chemistry (Hoppel gap)
- immersion freezing (in progress)
- ...

PySDM: backends, dynamics & environments

“dynamics”

- coalescence (SDM + dt-adaptivity)
- condensation (dt-adaptive, bespoke semi-implicit ODE solver)
- displacement (incl. sedimentation)
- aqueous chemistry (Hoppel gap)
- immersion freezing (in progress)
- ...

“backends”

- CPU (Numba/LLVM)
- GPU (ThrustRTC/CUDA)

“environments”

- Box
- Parcel
- PyMPDATA-based:
  - Kinematic1D
  - Kinematic2D
- PySDMachine.jl (planned)

PySDM: 2D kinematic Sc test (Morrison & Grabowski '07)

2D flow field

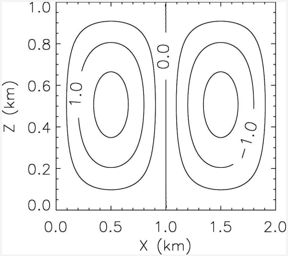
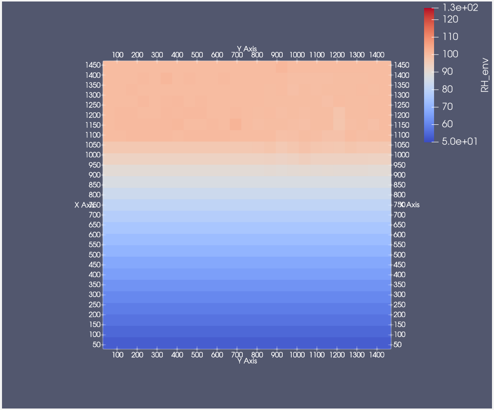
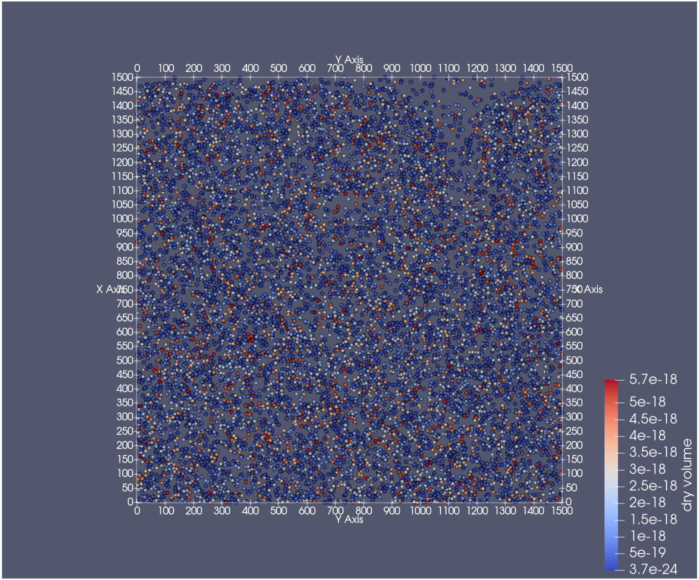


FIG. 1. Time-invariant vertical velocity for the stratocumulus case (contour interval is 0.5 m s^{-1}).

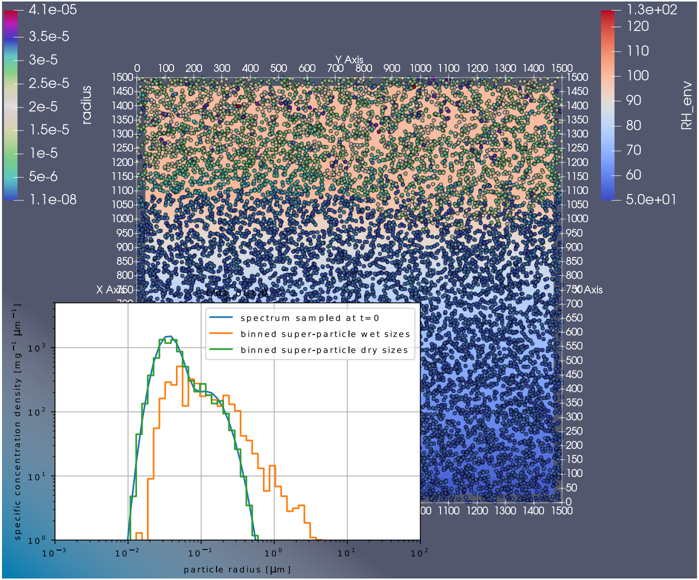
RH profile at t=0



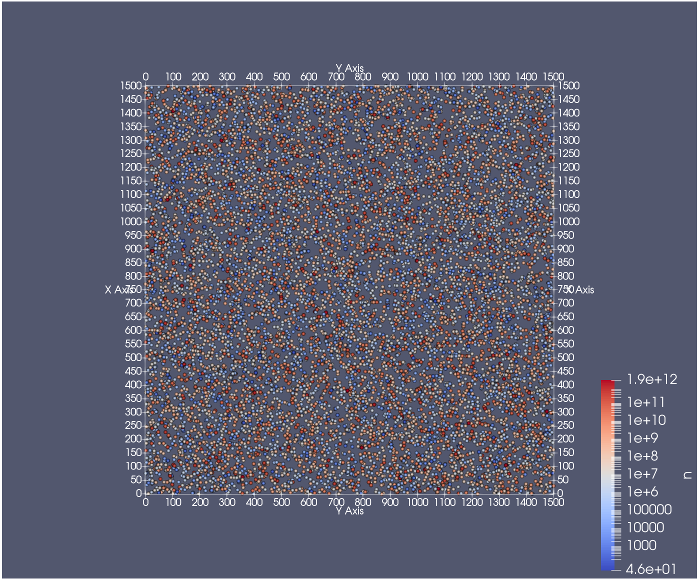
particle attribute initialisation: dry/wet volume



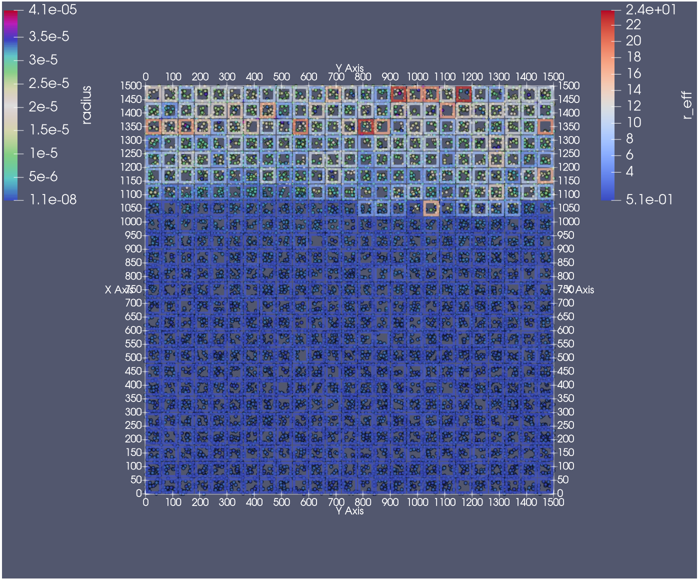
particle attribute initialisation: dry/wet volume



particle attribute initialisation: multiplicity

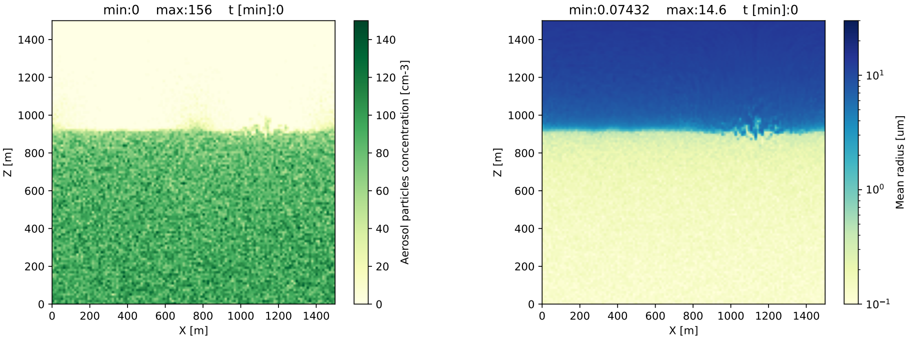


particle attribute evolution: droplet radius



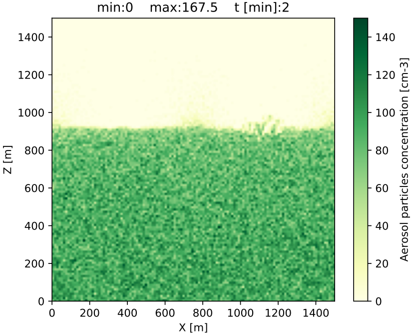
sample aerosol-cloud-precipitation interactions simulation

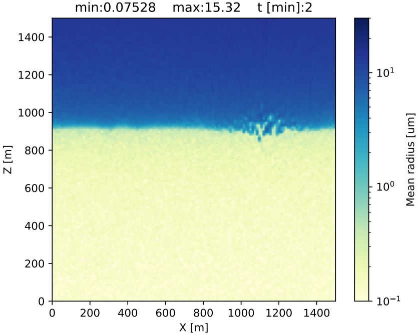
Computational grid: 128x128
Computational particles: 2^{21}



sample aerosol-cloud-precipitation interactions simulation

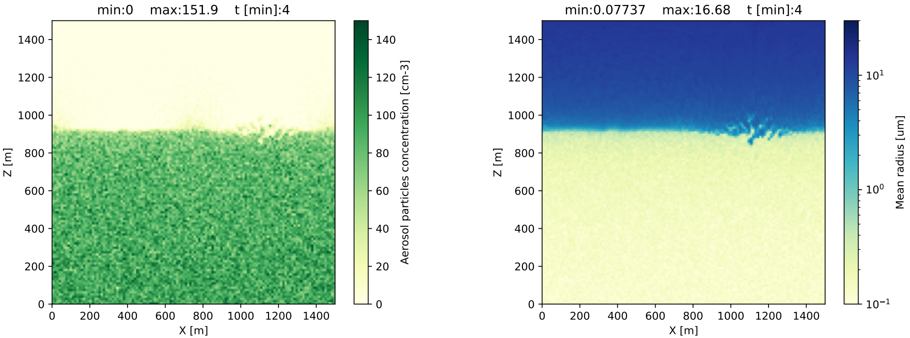
Computational grid: 128x128
Computational particles: 2^{21}





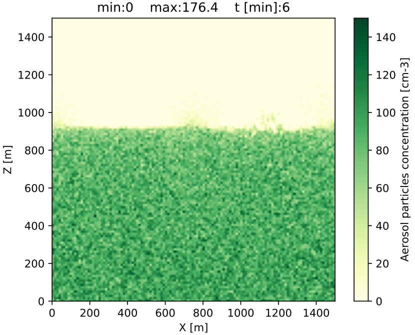
sample aerosol-cloud-precipitation interactions simulation

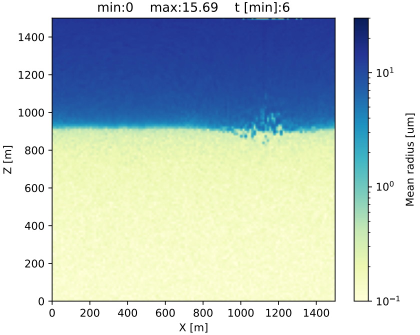
Computational grid: 128x128
Computational particles: 2^{21}



sample aerosol-cloud-precipitation interactions simulation

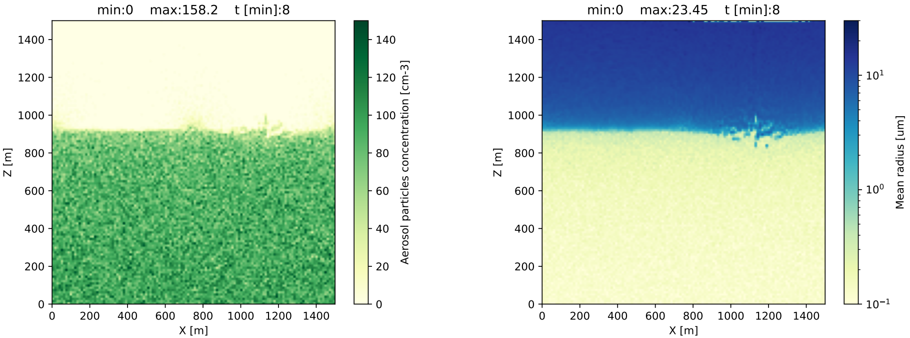
Computational grid: 128x128
Computational particles: 2^{21}





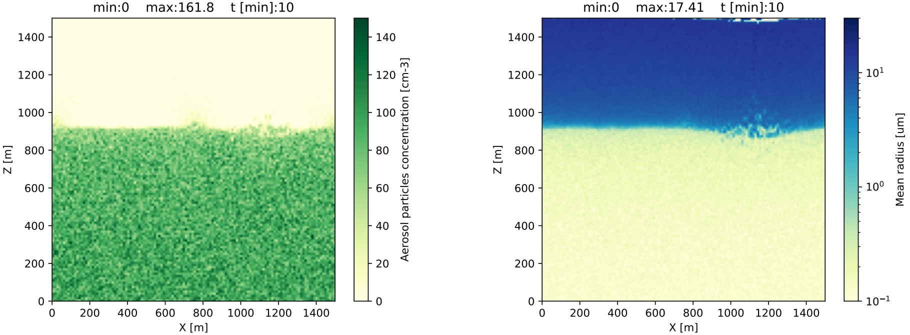
sample aerosol-cloud-precipitation interactions simulation

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



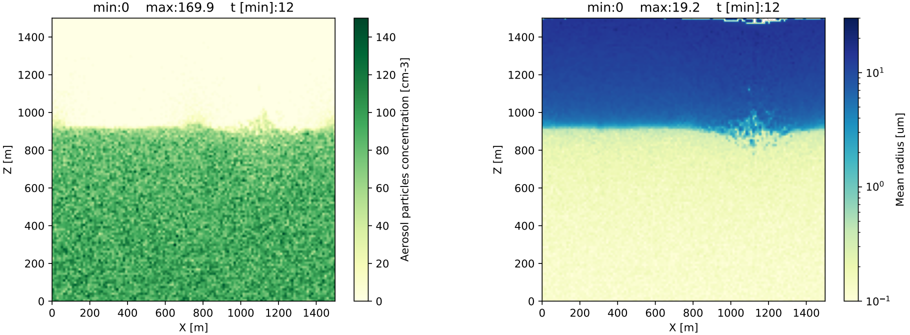
sample aerosol-cloud-precipitation interactions simulation

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



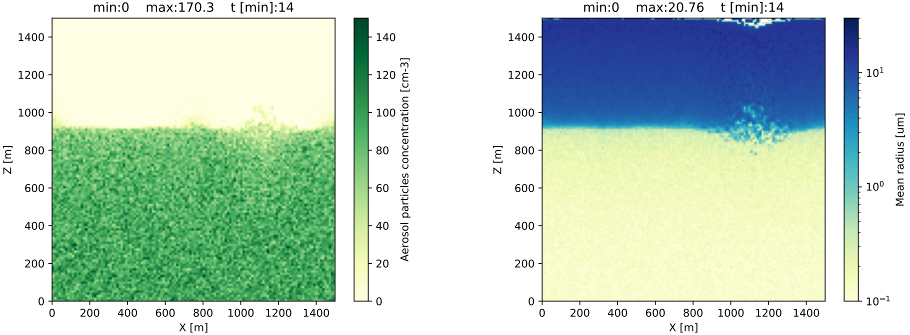
sample aerosol-cloud-precipitation interactions simulation

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



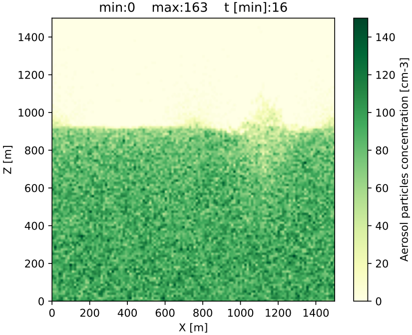
sample aerosol-cloud-precipitation interactions simulation

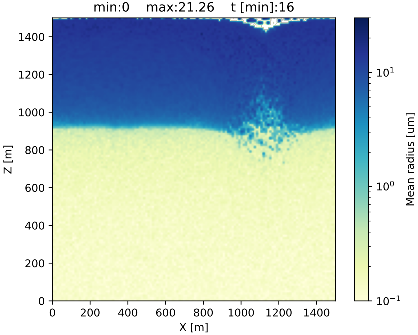
Computational grid: 128x128
Computational particles: 2^{21}



sample aerosol-cloud-precipitation interactions simulation

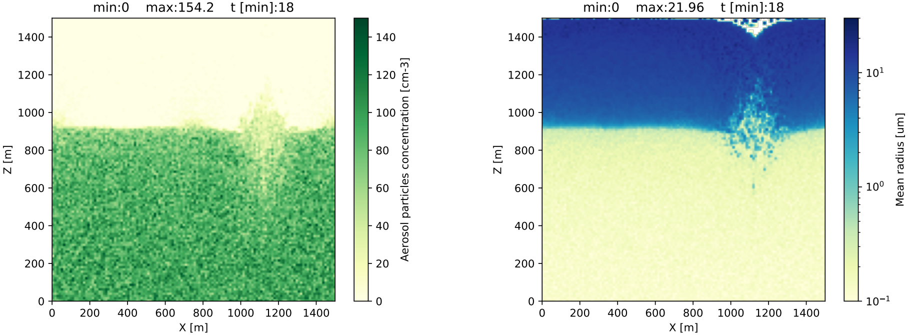
Computational grid: 128x128
Computational particles: 2^{21}





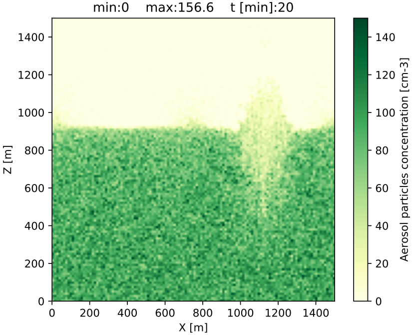
sample aerosol-cloud-precipitation interactions simulation

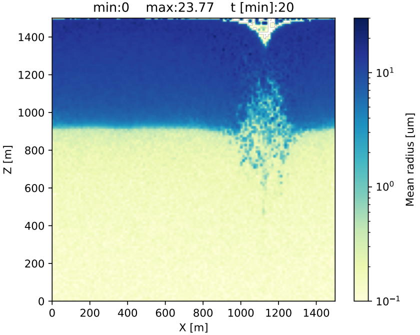
Computational grid: 128x128
Computational particles: 2^{21}



sample aerosol-cloud-precipitation interactions simulation

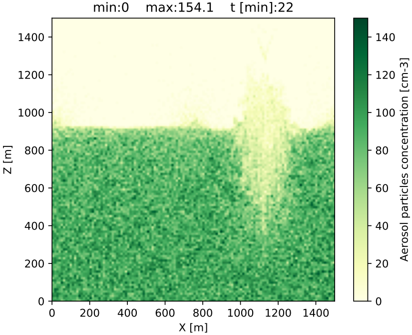
Computational grid: 128x128
Computational particles: 2^{21}

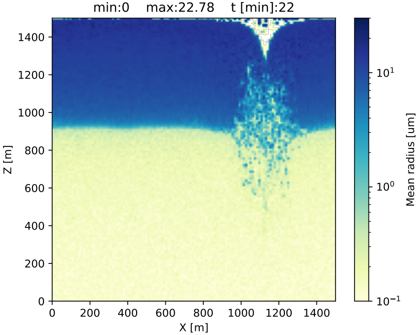




sample aerosol-cloud-precipitation interactions simulation

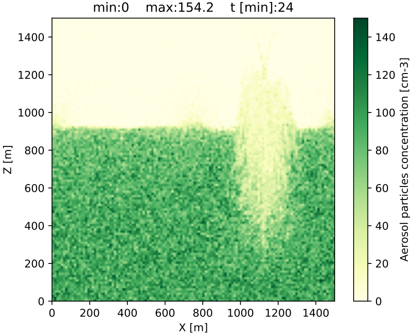
Computational grid: 128x128
Computational particles: 2^{21}

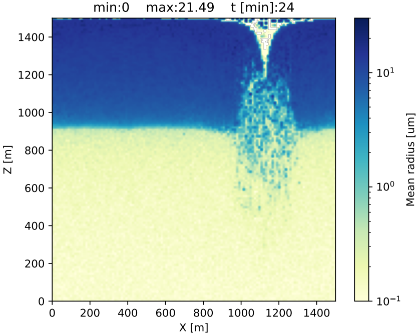




sample aerosol-cloud-precipitation interactions simulation

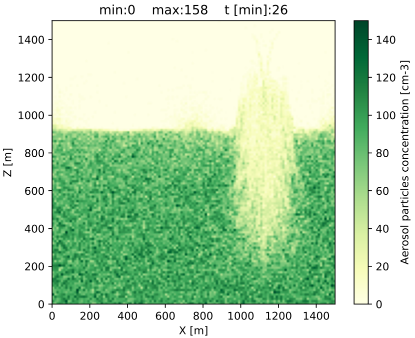
Computational grid: 128x128
Computational particles: 2^{21}

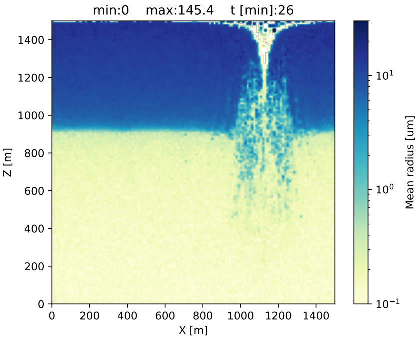




sample aerosol-cloud-precipitation interactions simulation

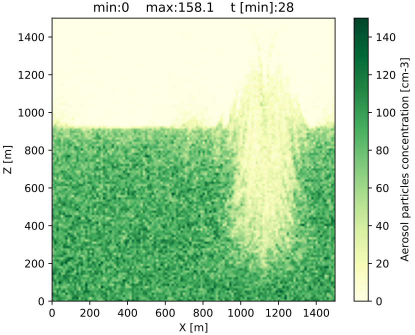
Computational grid: 128x128
Computational particles: 2^{21}

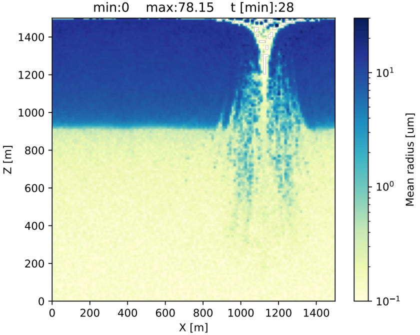




sample aerosol-cloud-precipitation interactions simulation

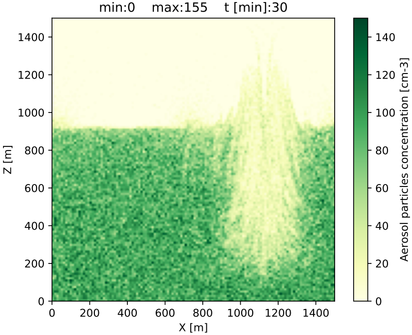
Computational grid: 128x128
Computational particles: 2^{21}

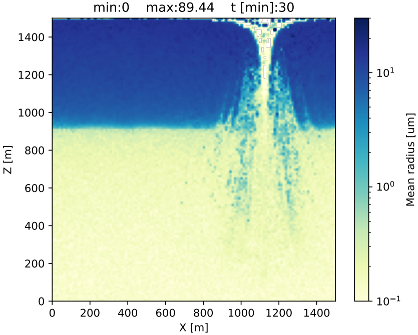




sample aerosol-cloud-precipitation interactions simulation

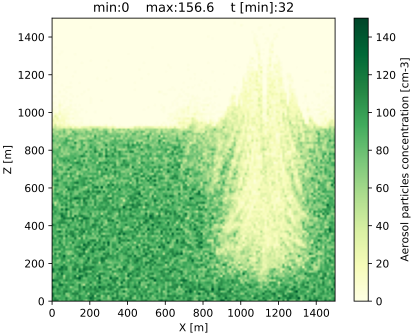
Computational grid: 128x128
Computational particles: 2^{21}

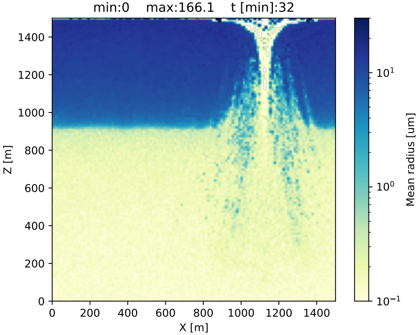




sample aerosol-cloud-precipitation interactions simulation

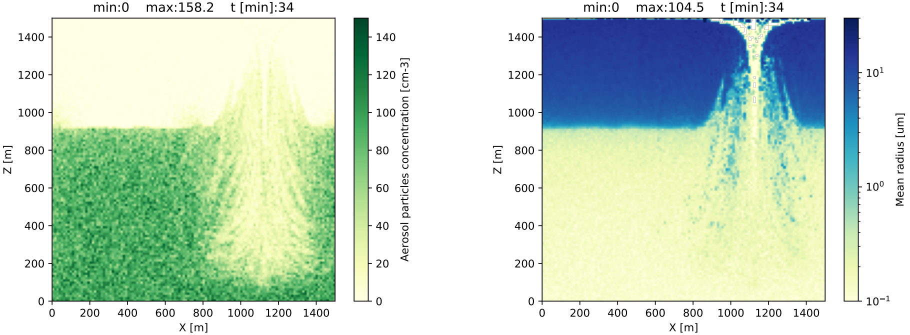
Computational grid: 128x128
Computational particles: 2^{21}





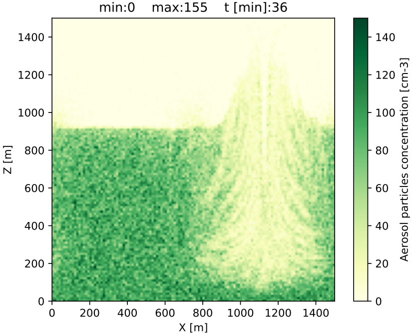
sample aerosol-cloud-precipitation interactions simulation

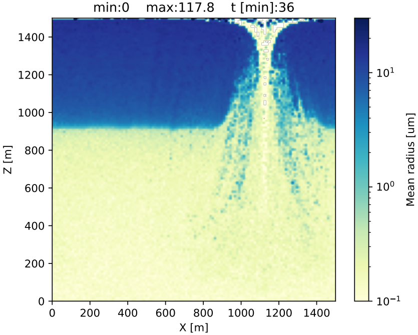
Computational grid: 128x128
Computational particles: 2^{21}



sample aerosol-cloud-precipitation interactions simulation

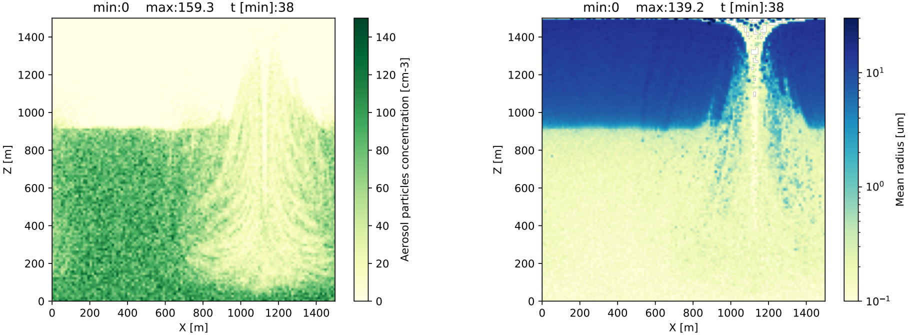
Computational grid: 128x128
Computational particles: 2^{21}





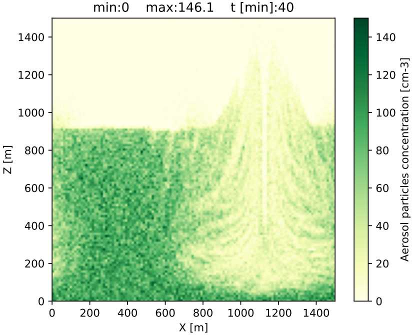
sample aerosol-cloud-precipitation interactions simulation

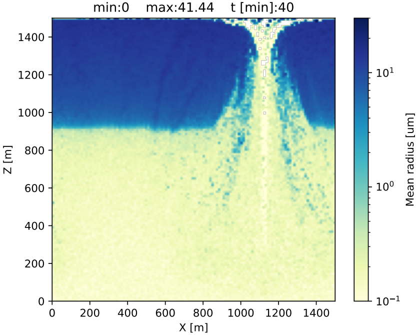
Computational grid: 128x128
Computational particles: 2^{21}



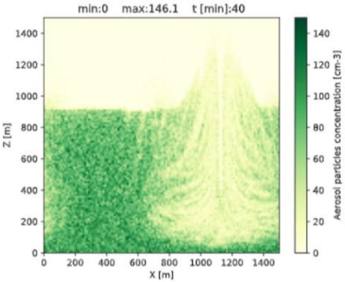
sample aerosol-cloud-precipitation interactions simulation

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

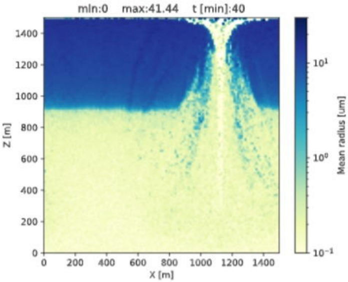




PySDM:



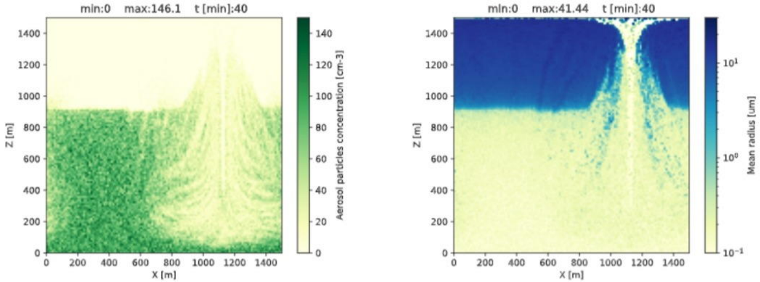
min:0 max:146.1 t [min]:40
Z [m]
X [m]
Aerosol particles concentration [cm-3]



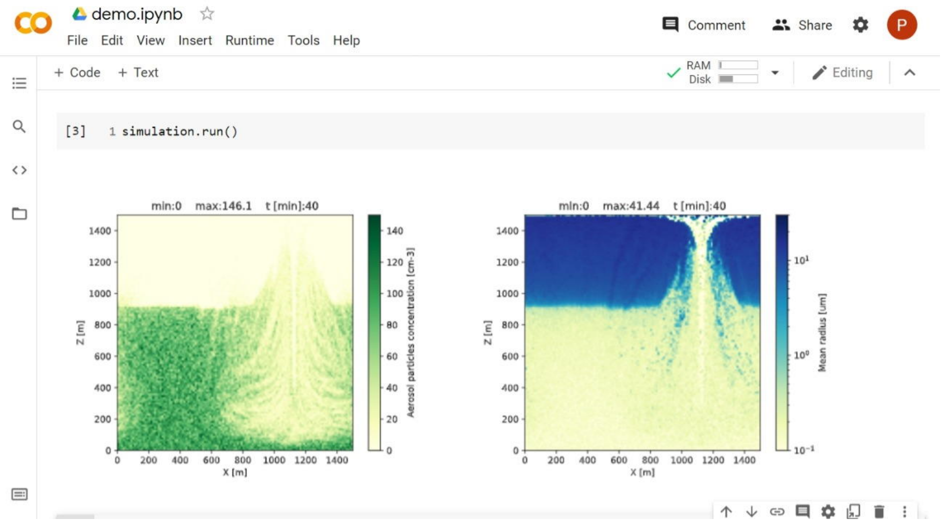
min:0 max:41.44 t [min]:40
Z [m]
X [m]
Mean radius [um]

PySDM: Pythonic

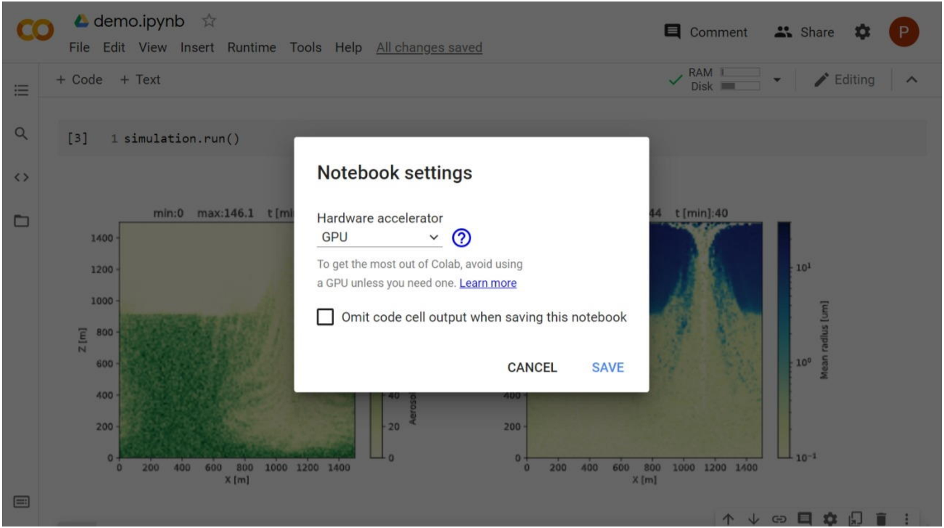
```
[3]   1 simulation.run()
```



PySDM: Pythonic, Jupyter-friendly



PySDM: Pythonic, Jupyter-friendly, GPU-enabled



Plan of the talk

PySDM: context

PySDM: statement of need & goals

PySDM: tour of the features

PySDM: demo (role play: reviewer)

PySDM: technological stack

PySDM-examples: pypi.org/p/PySDM-examples

Search projects

Help Sponsors Log in Register

PySDM-examples 1.6

```
pip install PySDM-examples
```

Latest version

Released: Sep 6, 2021

Navigation

- Project description
- Release history
- Download files

Project links

- Homepage

Statistics

View statistics for this project via Libraries.io, or by using our public dataset on Google BigQuery

Meta

License: GPL-3.0

Project description

This repository stores example files for `PySDM` depicting usage of `PySDM` from Python via Jupyter. For information on the `PySDM` package itself and examples of usage from Julia and Matlab, see PySDM README.md file.

Please use the PySDM issue-tracking and dicsussion infrastructure for `PySDM-examples` as well.

0D box-model coalescence-only examples:

- Shima et al. 2009 (Box model, coalescence only, test case employing Golovin analytical solution):
  - Fig. 2: launch binder Open in Colab
- Berry 1967 (Box model, coalescence only, test cases for realistic kernels):
  - Figs. 5, 8 & 10: launch binder Open in Colab

0D parcel-model condensation only examples:

- Arabas & Shima 2017 (monodisperse size spectrum activation/deactivation test case):
  - Fig. 5: launch binder Open in Colab
- Yang et al. 2018 (polydisperse size spectrum activation/deactivation test case):
  - Fig. 2: launch binder Open in Colab
- Lowe et al. 2019 (externally mixed polydisperse size spectrum with surface-active organics case):
  - Fig. 1: launch binder Open in Colab
  - Fig. 2: launch binder Open in Colab

nature COMMUNICATIONS

https://doi.org/10.1038/s41467-019-12982-0 OPEN

Key drivers of cloud response to surface-active organics

S.J. Lowe[1,2], D.G. Partridge[3], J.F. Davies[4], K.R. Wilson[5], D. Topping[6] & I. Riipinen[1,2,7*]

Aerosol-cloud interactions constitute the largest source of uncertainty in global radiative forcing estimates, hampering our understanding of climate evolution. Recent empirical evidence suggests surface tension depression by organic aerosol to significantly influence the formation of cloud droplets, and hence cloud optical properties. In climate models, however, surface tension of water is generally assumed when predicting cloud droplet concentrations. Here we show that the sensitivity of cloud microphysics, optical properties and shortwave radiative effects to the surface phase are dictated by an interplay between the aerosol particle size distribution, composition, water availability and atmospheric dynamics. We demonstrate that accounting for the surface phase becomes essential in clean environments in which ultrafine particle sources are present. Through detailed sensitivity analysis, quantitative constraints on the key drivers – aerosol particle number concentrations, organic fraction and fixed updraft velocity – are derived for instances of significant cloud microphysical susceptibilities to the surface phase.

PySDM-examples: Lowe et al. 2019

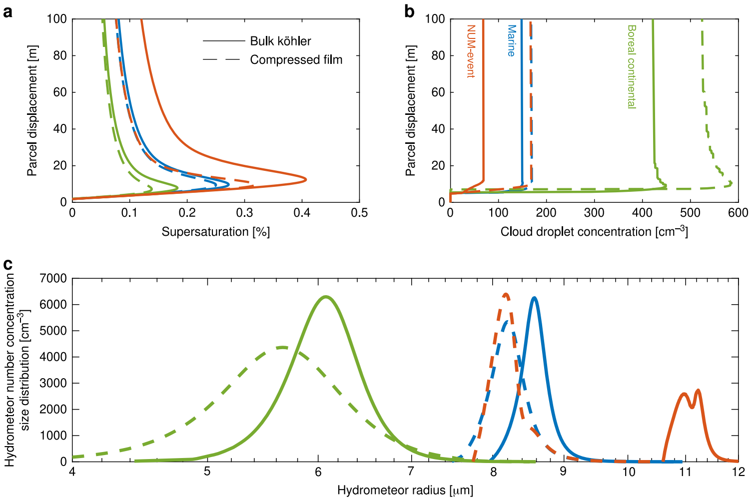


Fig. 2 Simulated microphysics of cloud events on marine (MA, blue), boreal (HYY, green) and NUM-event (NE, orange) aerosol populations. Cloud-formation event simulations using bulk Köhler BK (solid lines) and approximate compressed film CF (dotted lines) models of cloud droplet activation with initial temperature T = 280 K, pressure P = 98,000 Pa, supersaturation s = −0.1% and fixed updraft velocity w = 0.32 ms^{-1}. Simulated (a) ambient parcel supersaturation and (b) cloud droplet number concentration during parcel ascent. c Simulated droplet size distribution at a parcel displacement 200 m above initialisation

PySDM-examples: Lowe et al. 2019

example contributed by Clare Singer et al.
(https://claresinger.github.io/)

File Edit View Run Kernel Tabs Settings Help

Filter files by name
/ PySDM_examples / Lowe_et_al_2019 /

Name	Last Modified
__init__.py	4 hours ago
aerosol.py	4 hours ago
fig_1.ipynb	4 hours ago
fig_2.ipynb	4 hours ago
settings.py	4 hours ago
simulation.py	4 hours ago

Launcher | fig_2.ipynb

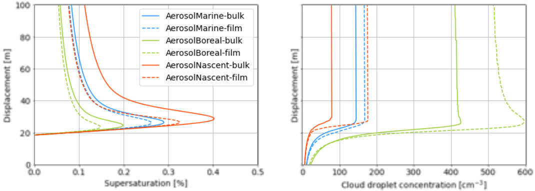
Download GitHub Binder Code

[4]:
```python
figsize = (15, 5)
pylab.rc('font', size=14)
fig, axs = pylab.subplots(1, 2, figsize=figsize, sharey=True)


for idx, var in enumerate(('S_max', 'n_c_cm3')):
    for key in output.keys():
        Y = np.asarray(output[key]['z'])
        axs[idx].plot(output[key][var], Y, label=key,
                      color=output[key]['color'],
                      linestyle='-' if key.endswith('-bulk') else '--'
                     )
    axs[idx].set_ylim(0, 100)

    axs[idx].set_ylabel('Displacement [m]')
    if var == 'S_max':
        axs[idx].set_xlabel('Supersaturation [%]')
        axs[idx].set_xlim(0, 0.5)
    elif var == 'n_c_cm3':
        axs[idx].set_xlabel('Cloud droplet concentration [cm$^{-3}$]')
        axs[idx].set_xlim(0, 600)
    else:
        assert False

for ax in axs:
    ax.grid()
axs[0].legend()
```



Plan of the talk

PySDM: context

PySDM: statement of need & goals

PySDM: tour of the features

PySDM: demo (role play: reviewer)

PySDM: technological stack

PySDM: technological stack

- Python `python.org`
- Numba (JIT, multi-threading) `numba.pydata.org`
- ThrustRTC (GPU-resident backend) `pypi.org/project/ThrustRTC`

PySDM: technological stack

- Python `python.org`
- Numba (JIT, multi-threading) `numba.pydata.org`
- ThrustRTC (GPU-resident backend) `pypi.org/project/ThrustRTC`

- GitHub & GitHub Actions `github.com`
- Codecov `codecov.io`
- AppVeyor `appveyor.com`

PySDM: technological stack

- Python python.org
- Numba (JIT, multi-threading) numba.pydata.org
- ThrustRTC (GPU-resident backend) pypi.org/project/ThrustRTC

- GitHub & GitHub Actions github.com
- Codecov codecov.io
- AppVeyor appveyor.com

- Jupyter jupyter.org
- Binder mybinder.org
- Colab colab.research.google.com

PySDM backend architecture overview

Springer Link

International Conference on Computational Science

ICCS 2021: Computational Science – ICCS 2021 pp 16-30 | Cite as

On the Design of Monte-Carlo Particle Coagulation Solver Interface: A CPU/GPU Super-Droplet Method Case Study with PySDM

Authors | Authors and affiliations

Piotr Bartman , Sylwester Arabas

Conference paper

First Online: 09 June 2021

Part of the Lecture Notes in Computer Science book series (LNCS, volume 12743)

PySDM: GPU backend internals

```
_kernel = ThrustRTC.For(
    ['perm_cell_start', 'perm_cell_id', 'pair_flag', 'length'], "i", '''
    pair_flag[i] = (
        i < length - 1 &&
        perm_cell_id[i] == perm_cell_id[i+1] &&
        (i - perm_cell_start[perm_cell_id[i]]) % 2 == 0
    );
    ''')

def flag_pairs(pair_flag, cell_start, cell_id, cell_idx):
    perm_cell_id = ThrustRTC.DVPermutation(cell_id.data, cell_id.idx.data)
    perm_cell_start = ThrustRTC.DVPermutation(cell_start.data, cell_idx.data)
    d_length = ThrustRTC.DVInt64(len(cell_id))
    _kernel.launch_n(len(cell_id),
        [perm_cell_start, perm_cell_id, cell_idx.data,
         pair_flag.indicator.data, d_length])
```

PySDM: CPU/multi-threaded backend internals

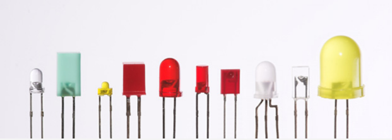
```
@numba.njit(parallel=True, error_model='numpy')
def _update_attributes(length, n, attributes, idx, gamma):
    for i in prange(length//2):
        j = idx[2*i]
        k = idx[2*i + 1]
        if n[j] < n[k]:
            j, k = k, j
        g = min(int(gamma[i]), int(n[j] / n[k]))
        if g == 0:
            continue
        new_n = n[j] - g * n[k]
        if new_n > 0:
            n[j] = new_n
            for attr in range(0, len(attributes)):
                attributes[attr, k] += g * attributes[attr, j]
        else:  # new_n == 0
            n[j] = n[k] // 2
            n[k] = n[k] - n[j]
            for attr in range(0, len(attributes)):
                attributes[attr, j] = attributes[attr, j] * g \
                                    + attributes[attr, k]
                attributes[attr, k] = attributes[attr, j]

def update_attributes(n, intensive, attributes, gamma):
    _update_attributes(len(n.idx),
        n.data, intensive.data, attributes.data,       # in/out
        n.idx.data, gamma.data)                        # in
```

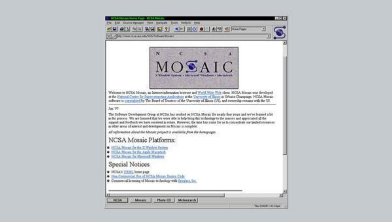
UIUC-originated breakthroughs

MRI

LED



Mosaic

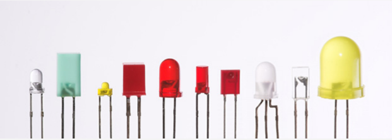


graphics from `https://illinois.edu`, `https://llvm.org`

UIUC-originated breakthroughs

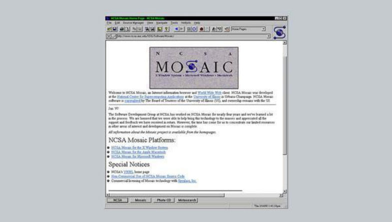
MRI

LED



graphics from `https://illinois.edu`, `https://llvm.org`

Mosaic

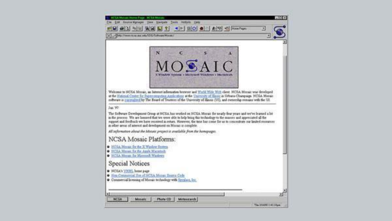


Instantwhip?

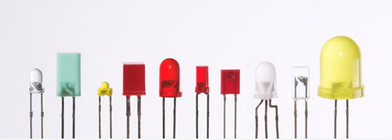
UIUC-originated breakthroughs

MRI

Mosaic



LED



LLVM!

LLVM COMPILER INFRASTRUCTURE

graphics from `https://illinois.edu`, `https://llvm.org`

Acknowledgements

funding

- EU / Foundation for Polish Science
- US / Atmospheric System Research

Acknowledgements

funding

- EU / Foundation for Polish Science
- US / Atmospheric System Research

co-authors & contributors

- @uj.edu.pl: **P. Bartman**, O. Bulenok, G. Łazarski, M. Olesik, ...
- @caltech.edu: A. Jaruga, C. Singer, ...

Acknowledgements

funding

- EU / Foundation for Polish Science
- US / Atmospheric System Research

co-authors & contributors

- @uj.edu.pl: **P. Bartman**, O. Bulenok, G. Łazarski, M. Olesik, ...
- @caltech.edu: A. Jaruga, C. Singer, ...

Thank you for your attention!

https://arxiv.org/abs/2103.17238 (Bartman et al. 2021, arXiv/JOSS)
https://doi.org/10.1007/978-3-030-77964-1_2 (Bartman & Arabas 2021, LNCS)

https://pypi.org/p/PySDM & https://pypi.org/p/PySDM-examples