A Scalable Particle-Based Microphysics Model for Atmospheric Flow Simulations

18th U.S. National Congress on Computational Mechanics
Chicago, IL

July 2025

Debojyoti Ghosh, Hassan Beydoun, Dana McGuffin,
David Gardner, and Katie Lundquist

LLNL-CFPRES-2008608
This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC and funded by the LDRD Program at LLNL under project tracking code 24-SI-001.

Lawrence Livermore National Laboratory

Modeling Microphysics

Cloud Microphysics: Dynamics and interactions of aerosol, cloud, and precipitating particles in atmospheric flows

$$\left.\begin{array}{l} q_v\text{: vapour} \\ q_c\text{: cloud} \\ q_r\text{: rain} \end{array}\right\} \text{Microphysics variables (density fractions)}$$

At each time step: $t^n \rightarrow t^{n+1}$

Fluid Solver

- Update the state variables based on the moist Euler/Navier-Stokes equations

$$\begin{bmatrix} \rho \\ \rho\mathbf{u} \\ \rho\theta \end{bmatrix}^n \rightarrow \begin{bmatrix} \rho \\ \rho\mathbf{u} \\ \rho\theta \end{bmatrix}^{n+1}$$

- Advect moisture variables with the current flow velocity ($\mathbf{u}^n$)

Microphysics Model

- Update moisture variables as specified by the model

$$\begin{bmatrix} q_v \\ q_c \\ q_r \\ \vdots \end{bmatrix}^n \rightarrow \begin{bmatrix} q_v \\ q_c \\ q_r \\ \vdots \end{bmatrix}^{n+1}$$

- Update state variables (θ)

Types of Microphysics Models

Computational Expense

Bulk Models:

Evolve averaged moisture quantities (q_v, q_c, …)

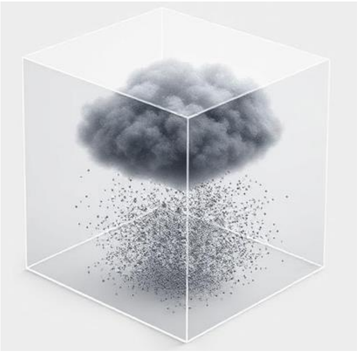
- Computationally cheap – evolve ODEs along with flow equations
- Limited accuracy due to empirical models of droplets dynamics
- Examples: *Kessler, Single-Moment*

$$\frac{d}{dt}\begin{bmatrix} q_v \\ q_c \\ q_r \\ \vdots \end{bmatrix}^n = F\left(q_v, q_c, \cdots\right)$$

Super-Droplets Method (SDM):

Particle-based model for simulating cloud & rain

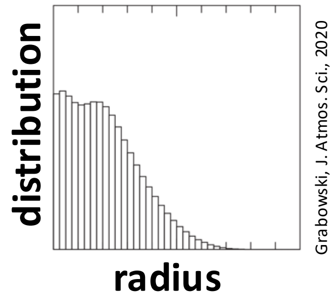
- Include fundamental droplet dynamics
- "Super-particle" approach for acceptable cost
- Examples: *PySDM*, *libcloudph++*, *SCALE-SDM*



Bin Methods:

Evolve droplet density distributions at each grid point based on dynamics

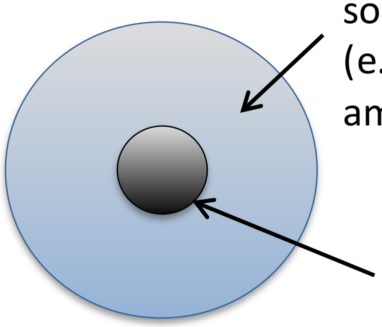
- Evolve the spectral density function discretized in droplet size
- Potentially very accurate since they model droplet dynamics
- Computational expense is prohibitive for practical applications



Grabowski, J. Atmos. Sci., 2020

What is a Super-Droplet?

Droplet



Liquid water with soluble aerosols (e.g., salt, ammonium sulfate)

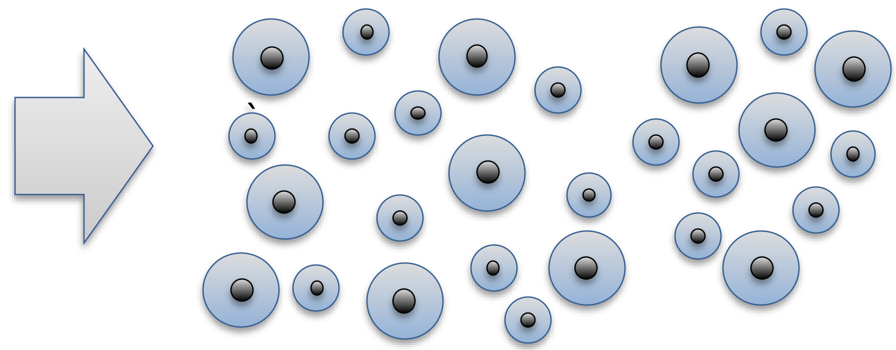
Insoluble aerosols (e.g., soil)

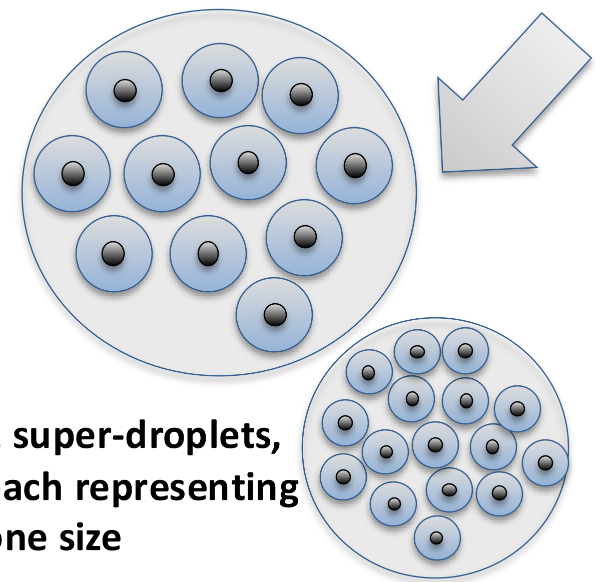
Assumed to be a sphere

Physical Attributes

- Position
- Velocity
- Terminal velocity
- Radius
- Aerosols and their masses

Droplets of 2 different sizes





2 super-droplets, each representing one size

"Super-droplet"

- Represents multiple droplets of the same size
- All physical attributes assumed to be the same
- Computational attribute: ***multiplicity*** (number of physical droplets a super-droplet represents)

Physical Processes

Particle Motion: New position computed with first-order update

$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t \left(\mathbf{u}_p - v_t \hat{\mathbf{k}} \right)$$

Advection: $\mathbf{u}_p$ is computed at particle location from flow velocity

Terminal Velocity: v_t computed from particle size and flow conditions

---

Droplet Growth/Shrinking: Size change due to condensation and evaporation

Stiff ODE solved implicitly (Backward Euler + Newton) for each super-droplet

Saturation ratio $\frac{q_v}{q_{\text{sat}}}$

Curvature effect

Solute effect $b = 4.3 \times 10^{-6} \frac{iM_i}{m_s}$

Solute mass

Solute mol. weight

Radius

$$R_i \frac{dR_i}{dt} = \frac{(S-1) - \frac{a}{R_i T} + \frac{b}{R_i^3}}{F_k + F_d}$$

$$F_k = \left(\frac{L}{R_v T} - 1 \right) \frac{L \rho_l}{KT}$$

$$F_d = \frac{\rho_l R_v T}{D e_s(T)}$$

Coalescence: due to random collisions between particles

<u>Key process for rain formation</u> from cloud particles

→ Computationally efficient **Monte-Carlo algorithm**

*Probability of **collision between two physical droplets***

$$P_{ij} = C\left(r_i, r_j\right) \left| v_i - v_j \right| \frac{\Delta t}{\Delta v}$$

Collision kernel (e.g., Hall, 1980)

Velocity difference

Time interval and volume

Fluid Solver ("DyCore"): ERF

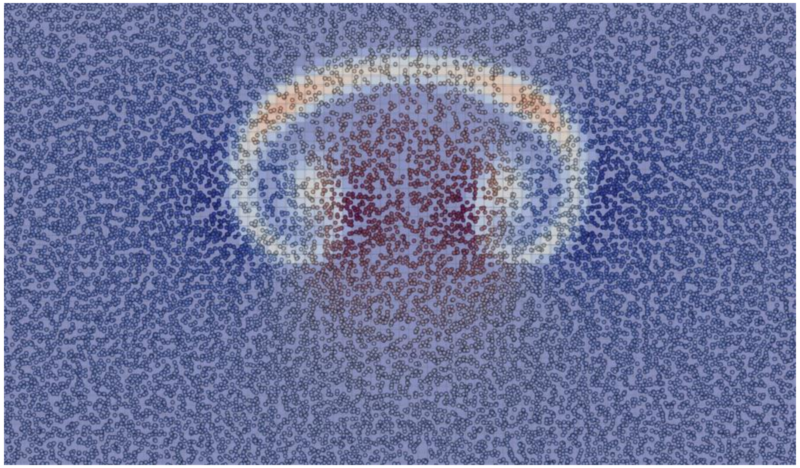
Energy Research & Forecasting (ERF)

- **Nonhydrostatic atmospheric flow simulation** code: Solves the compressible Navier-Stokes equations
- **Built on AMReX** for *scalability* and *portability*: Unified implementation on CPUs and GPUs (NVIDIA, AMD, Intel)
    - C++ with MPI and OpenMP/CUDA/HIP/SYCL
- **Multirate time integration** and **high-order** (2nd to 6th) **spatial discretization**
- Block-structured grids with **AMR**
    - **Terrain-conforming** coordinates
    - **Embedded boundaries** for urban geometries

https://github.com/erf-model/ERF

Super-Droplet Method in ERF

Implementation: Super-droplets using **AMReX Particle** and **ParticleContainer** classes & functions



Time Evolution

Grid

- Update the state variables based on the moist Navier-Stokes equations
- Advect q_v with the current flow velocity ($\mathbf{u}^n$)

Particles

Update super-droplets attributes based on droplet dynamics

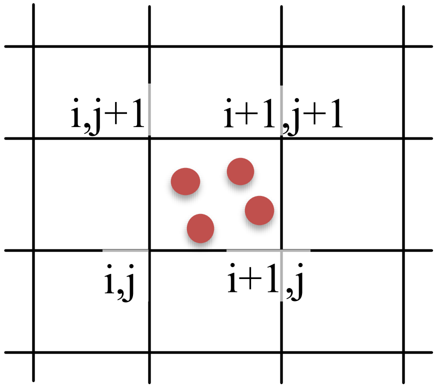
- **Advection** and **terminal velocity** – position update
- **Condensation** and **evaporation** – radius/mass
- **Coalescence** – radius/mass

Particles to Grid

Update moisture variables based on particles

- Compute $q_c(\mathbf{x})$ and $q_r(\mathbf{x})$ from particle positions and masses
- Update by q_v subtracting $q_c + q_r$
- Update θ due to latent heat of vaporization

Initialization of Super-Droplets



Initial position: Super-droplets are placed randomly within each grid cell with **zero initial velocity**

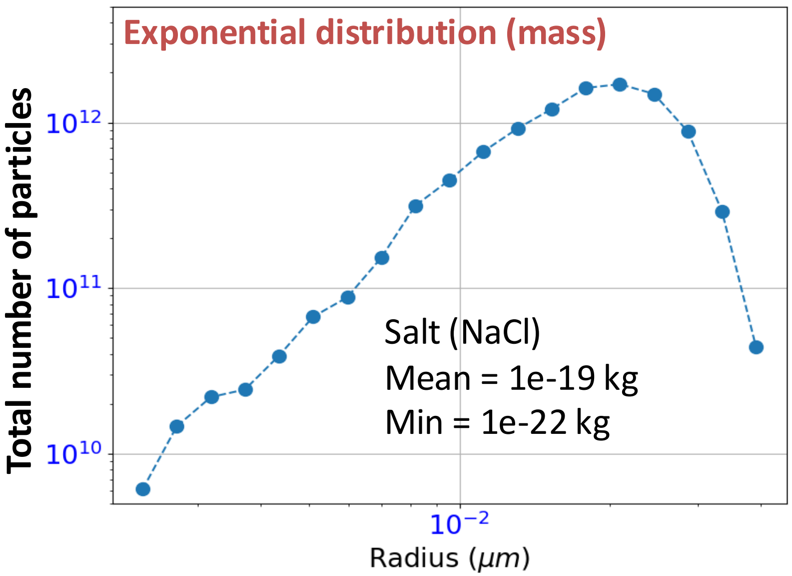
Physical number density (may vary spatially)

Initial number of super-droplets per cell

Initial multiplicity

Aerosol masses and **droplet radius** for each super-droplet are sampled from a specified distribution

- **Aerosol Species:** Salt, Ammonium Sulfate, Soil
- **Exponential distribution** for mass
- **Log-normal distribution** for radius
- Sum of multiple distributions (for example, **bimodal distribution**)

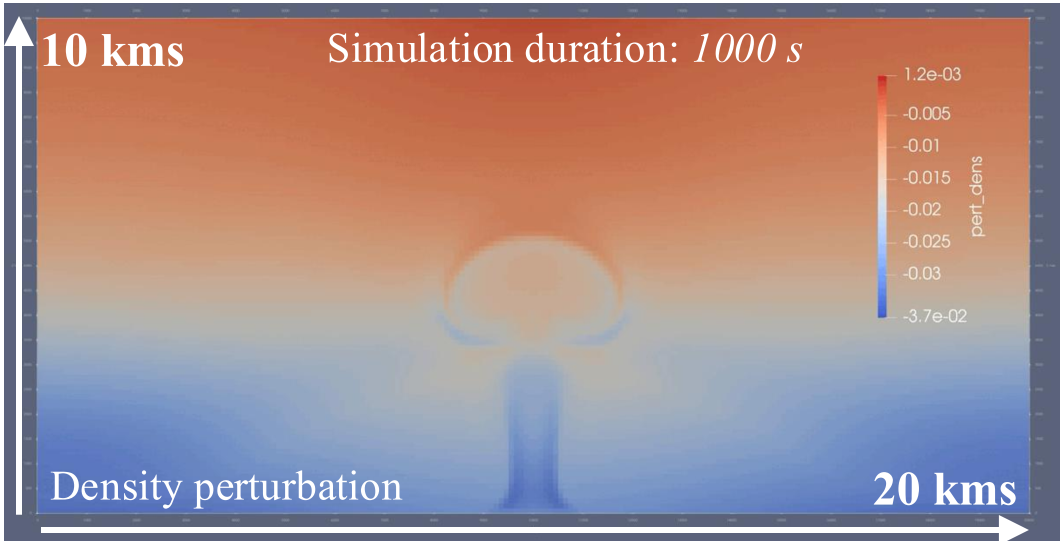


Example: 2D Rising Bubble

2D Rising Bubble in Moist Atmosphere

As the bubble rises, moisture is convected upwards and cools down to form clouds and rain

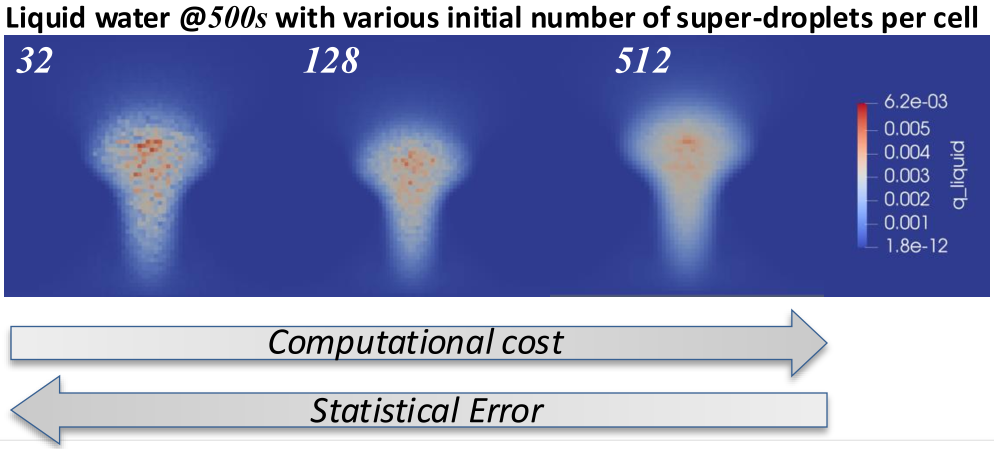
- Domain: *20 km x 10 km*
- “Slip wall” BCs on all sides
- Warm bubble with radius *2 km* initially located at (*10 km, 2 km*)
- Bubble temperature perturbation: *2 K*



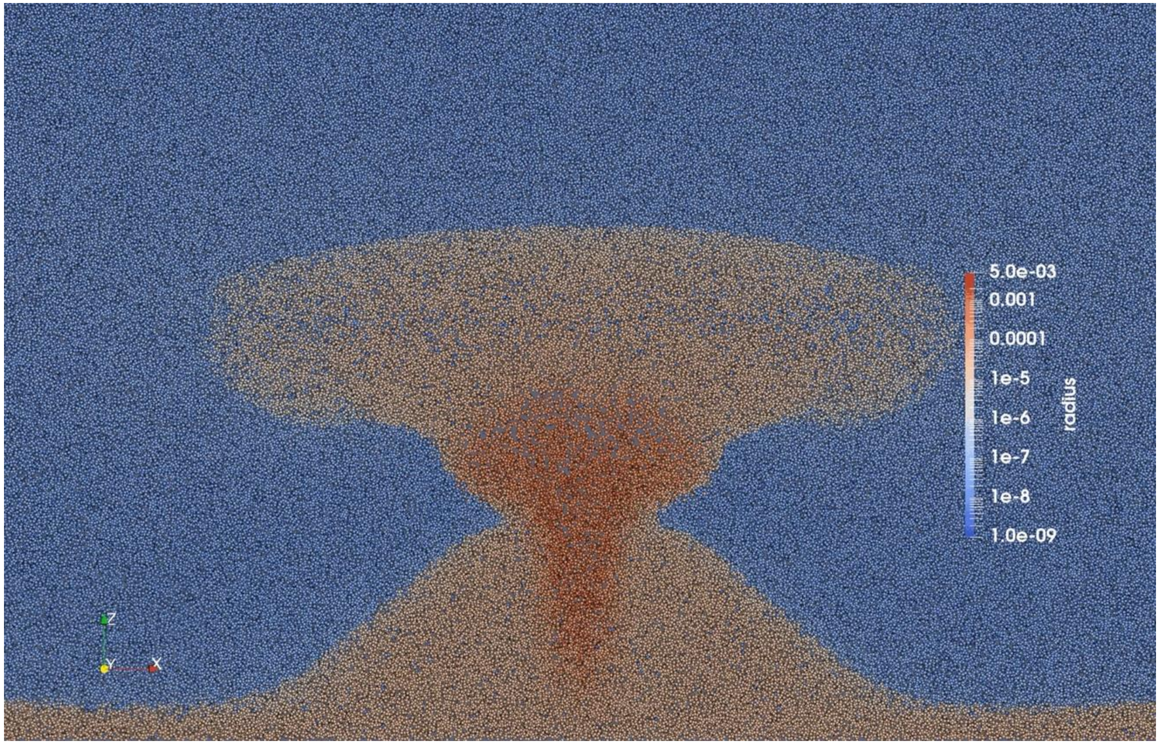
Computational Setup:

- Grid: 200 x 4 x 100 (*100 m* resolution)
- Aerosol species: salt (NaCl) - Exponential distribution with mean mass *10^{-19} kg*
- Initial physical concentration: *1e7 m^{-3}*
- Initial number of super-droplets per cell: 256

➔ Approx. **20 million super-droplets** representing 8×10^{17} physical particles



Example: 2D Rising Bubble

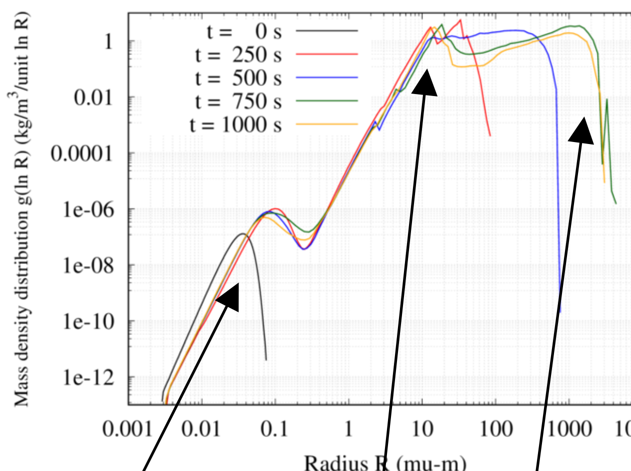


Visualization of the super-droplets (colored by radius)

- Simulated with 4 super-droplets per cell to allow plotting
- Super-droplets convect upwards with the flow and grow due to condensation
- Coalescence causes formation of rain that precipitates

Mass distribution evolves from ***unimodal*** *(dry aerosols)* to ***bimodal*** *(aerosol + cloud)* and ***trimodal*** *(aerosol, cloud, rain)*

Mass distribution evolution

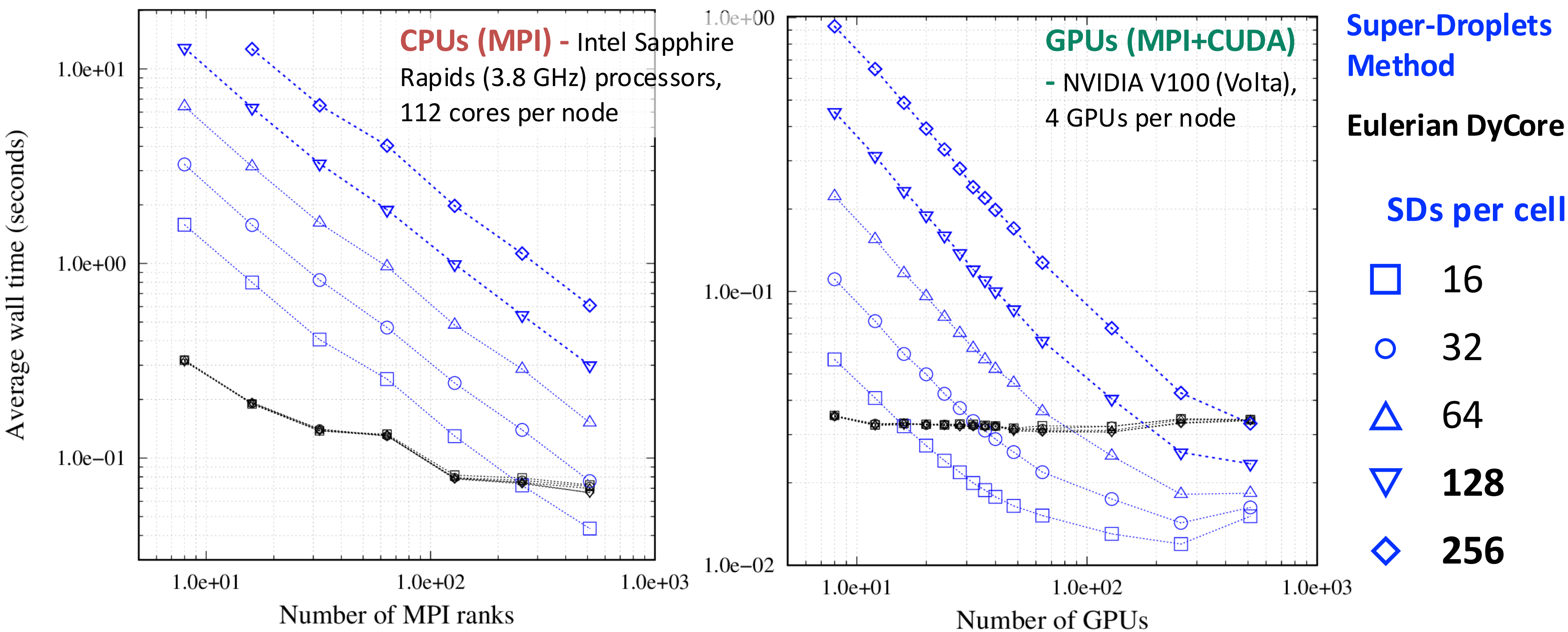


Initial
(dry aerosols)

Cloud
droplets
(~ *1-100 μm*)

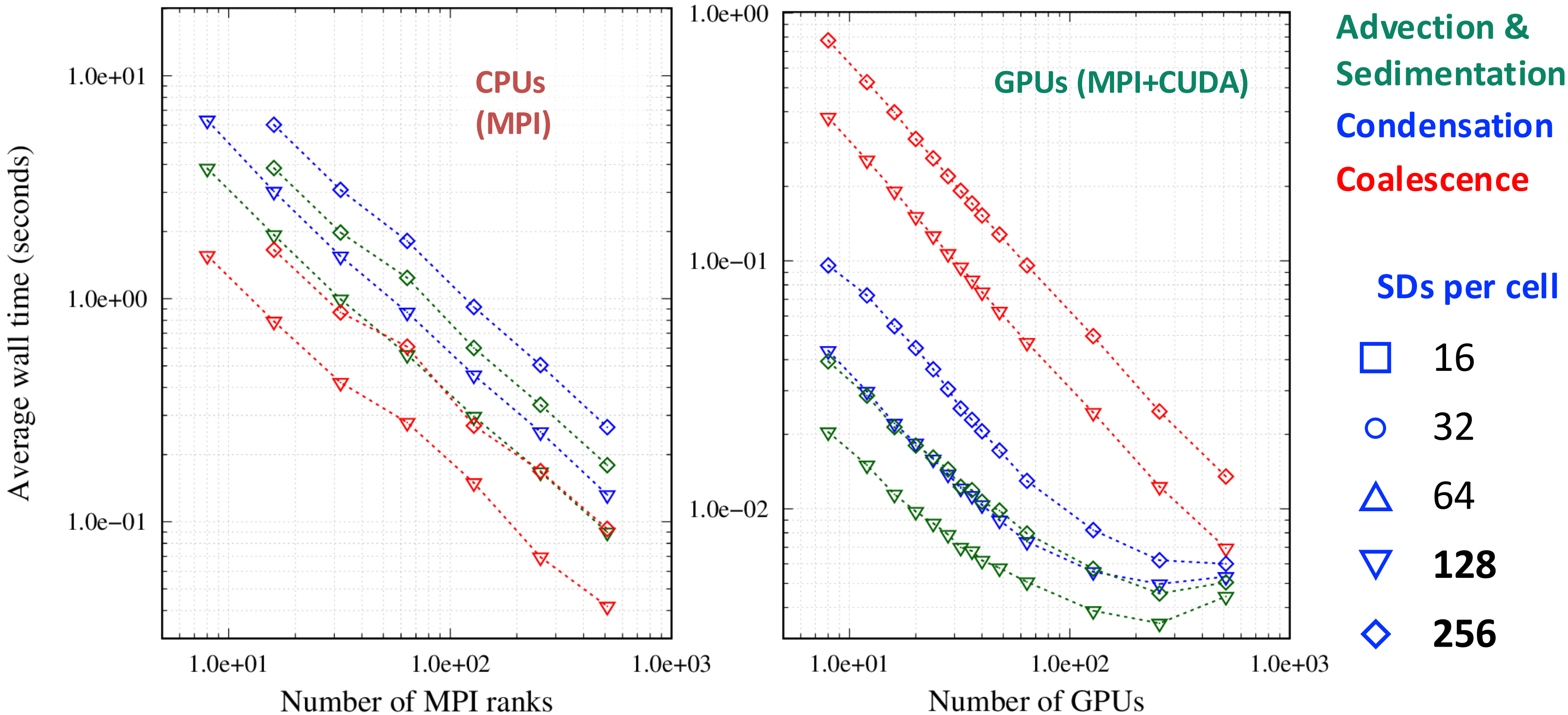
Rain droplets
(~ *1 mm*)

Strong Scaling on CPUs and GPUs



- Grid: **512 x 4 x 512** points (*~17 million to ~2.4 billion particles*)
- Number of CPUs (MPI ranks) or GPUs: **8** to **512**
- **Good strong scaling** observed for SDM on both CPUs and GPUs; note that Eulerian DyCore doesn't scale well in this setup since it is *over-decomposed on CPUs* and *doesn't fill the GPUs*.

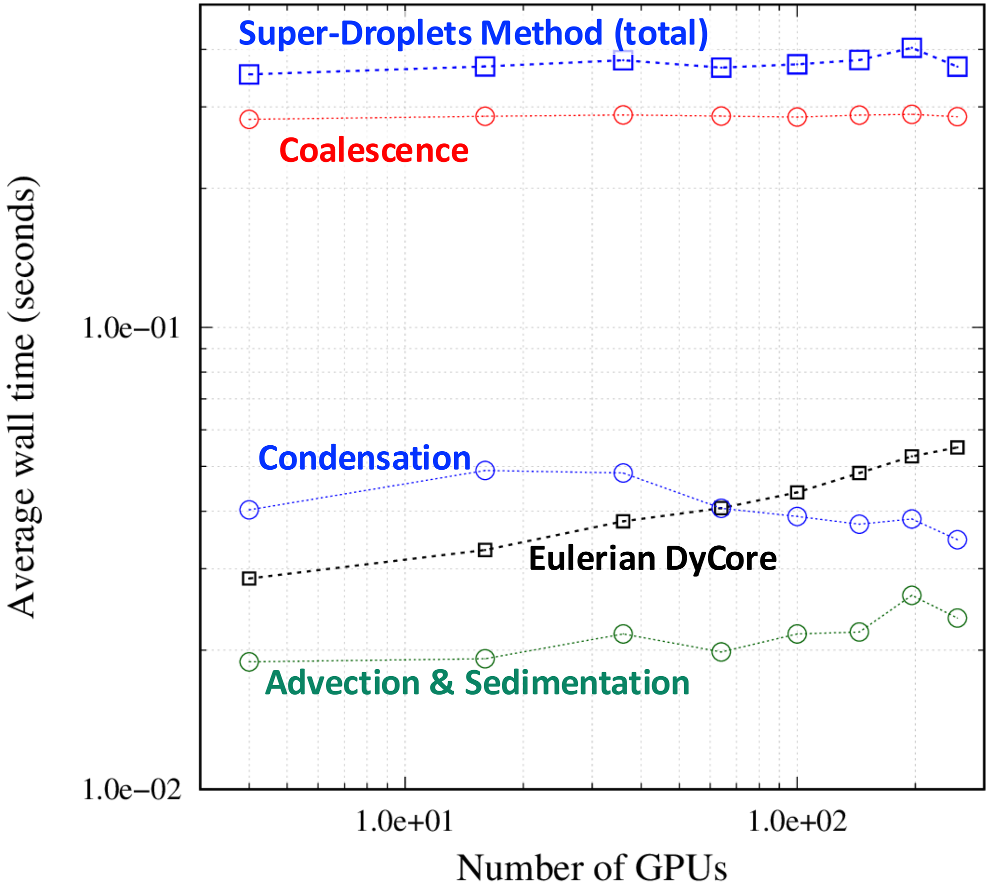
Scaling of SDM Processes



CPU-only: Condensation is the most expensive since it involves implicit solution of ODE for each super-droplet

With GPUs: Coalescence is the most expensive on GPUs since it involves the Monte-Carlo collisions algorithm

Weak Scaling on GPUs (MPI + CUDA)



- **On each GPU:** *128 x 4 x 128 grid* with *256 superdroplets per cell*
  → **~16.8 million particles per GPU**
- **Number of GPUs:** 4 to 256
  → ~67 million to 4.3 billion total particles
- Most MPI communications *in Eulerian DyCore* (filling ghost cells)*; **in SDM, only particle redistribution***
- Excellent weak scaling observed for the SDM and all component processes.

Work-in-Progress…

Implemented a **Lagrangian moisture model in ERF** based on the **super-droplets method**

- Limited to simulation of flows under warm conditions (**no ice/snow**)
- Computationally *more expensive than bulk models*
  - Incorporates higher fidelity droplet dynamics
  - Does not rely on empirical models of phase change
  - With GPUs, computational expense is acceptable
- ***Excellent scalability*** from using **AMReX's particle implementation**
- Currently working on **verifying/validating implementation** for various cases (Congestus clouds, cloud chamber, etc.)

Future plans:

- Implement cold processes (simulate formation of ice/snow/graupel)
- Incorporate terrain into super-droplets dynamics

Thank you. Questions?

Lawrence Livermore
National Laboratory

Implementation and Parallelism

Super-droplets are implemented using the **Particle** and **ParticleContainer** classes and utilities in **AMReX**

Portable and **scalable** on various heterogenous architectures

- MPI is used for domain decomposition over multiple CPUs/nodes
- On-node parallelism using CUDA/HIP on GPUs or OpenMP on CPUs

N_p: number of particles

N_g: number of grid cells

$N_p >> N_g$

Advection

Condensation & Evaporation

Independent for each particle
➔ $O(N_p)$ parallelizable

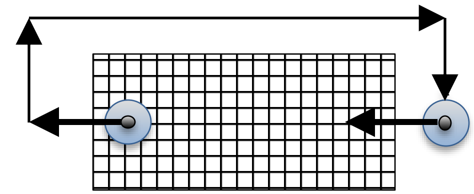
Coalescence (Monte-Carlo Algorithm)

- Shuffling & pairing — Independent for each grid cell ➔ $O(N_g)$ parallelizable
- Attribute update — Independent for each particle ➔ $O(N_p)$ parallelizable

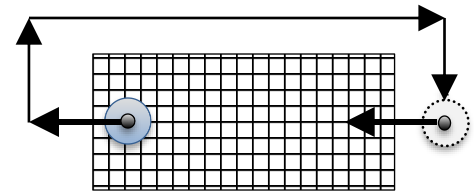
Computing Eulerian moisture variables from particles — Independent for each grid cell ➔ $O(N_g)$ parallelizable

Boundary Treatment

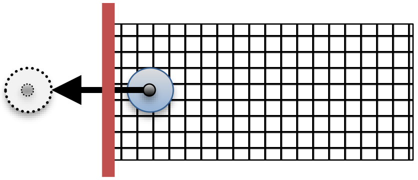
Periodic Boundary: Super-droplet re-enters domain from the other side with attributes preserved



Inflow/Outflow: Super-droplet re-enters domain from the other side ***as dry aerosol***



Side and Top Walls: Super-droplet gets "deactivated" - velocities set to 0, multiplicities set to 0, *does not participate in the simulation anymore*

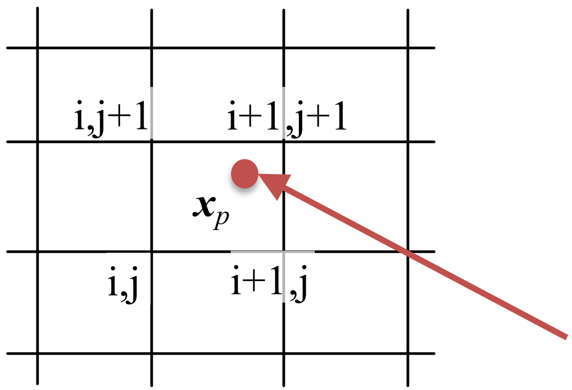


Ground: Same as side/top walls, but *rain accumulation on ground is updated* based on super-droplet mass and multiplicity

Recycling:

Put back deactivated super-droplet as dry aerosol at a random location in domain

Advection & Terminal Velocity



New position computed with first-order update

$$\mathbf{x}_p^{n+1} = \mathbf{x}_p^n + \Delta t \left(\mathbf{u}_p - v_t \hat{\mathbf{k}} \right)$$

Advection: $\mathbf{u}_p$ is computed at particle location from flow velocity

Eulerian flow variables are computed at particle location by **linear interpolation**

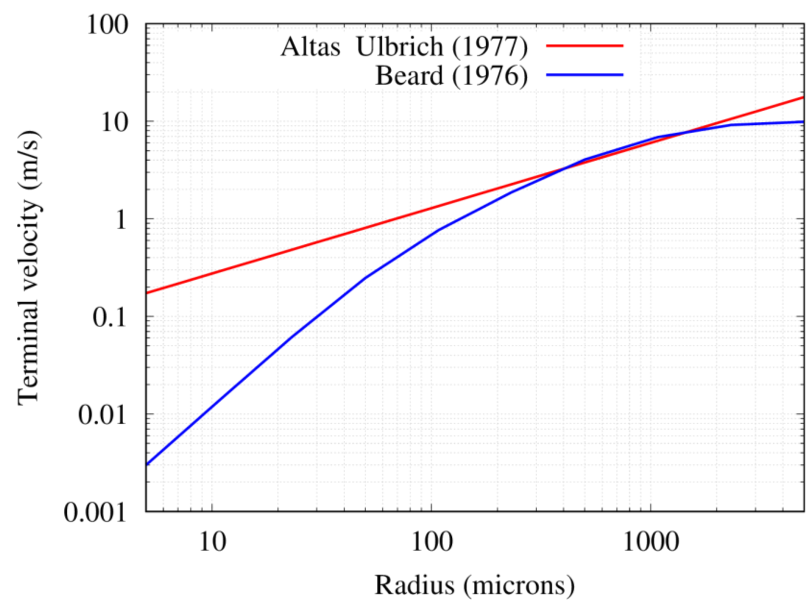
Terminal Velocity Models:

Atlas & Ulbrich (1977)
function of particle size

$$v_t = 3.778 D^{0.67}$$

Beard (1976): Considers three regimes

- **Stoke's:** *diameter less than 20 microns*
- **Transitional:** *20 microns to 1 mm*
- **Newton's:** *larger than 1 mm*



Condensation/Evaporation

Saturation ratio $\frac{q_v}{q_{\text{sat}}}$

Curvature effect

Solute effect $b = 4.3 \times 10^{-6} \frac{iM_i}{m_s}$

Solute mass

Solute molecular weight

Droplet radius

Droplet size equation

$$R_i \frac{dR_i}{dt} = \frac{(S-1) - \frac{a}{R_i T} + \frac{b}{R_i^3}}{F_k + F_d}$$

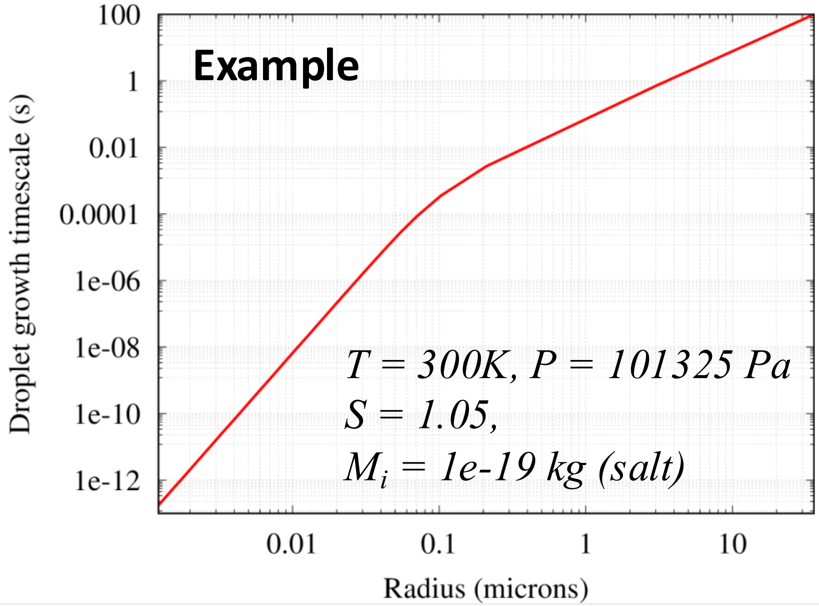
where $F_k = \left(\frac{L}{R_v T} - 1\right) \frac{L \rho_l}{KT}, \; F_d = \frac{\rho_l R_v T}{D e_s(T)}$

Growth timescales are ***much smaller*** *than fluid convection/acoustic timescales*

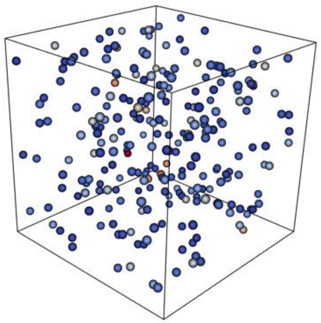
$$\tau^{-1} = \frac{1}{2} \left| \frac{1}{F_k + F_d} \left(-\frac{2a}{R_i^3 T} + \frac{6b}{R_i^5} \right) \right|$$

Sub-stepping within each ERF timestep

- Solve the ODE for each super-droplet independently
- Backward-Euler time integration with CFL 100
- Newton method to solve the nonlinear equation



Stochastic Coalescence



Random collisions of droplets near each other resulting in coalescence

Key process forming rain droplets from cloud

Probability of collision between two physical droplets

$$P_{ij} = C\left(r_i, r_j\right)\left|v_i - v_j\right|\frac{\Delta t}{\Delta v}$$

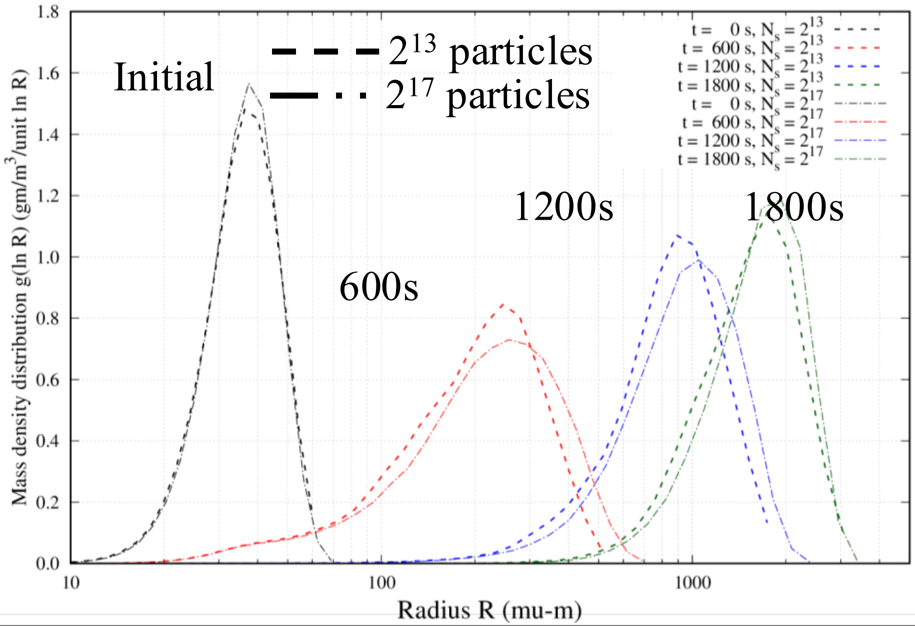
Collision kernel (e.g., Hall, 1980)

Velocity difference

Time interval and volume

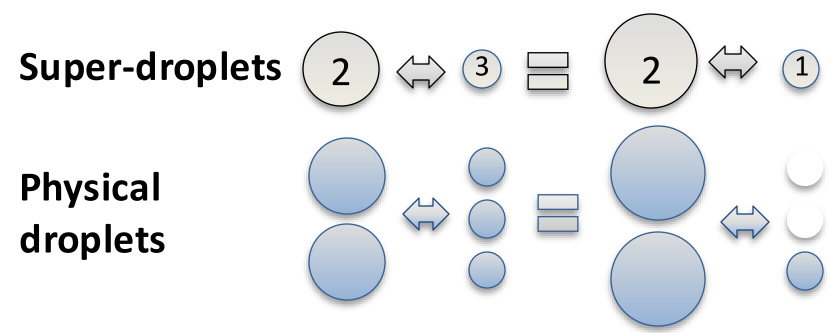
Validation in a box (no flow) – Hall kernel

Good agreement with results in *Shima, et all, 2009*



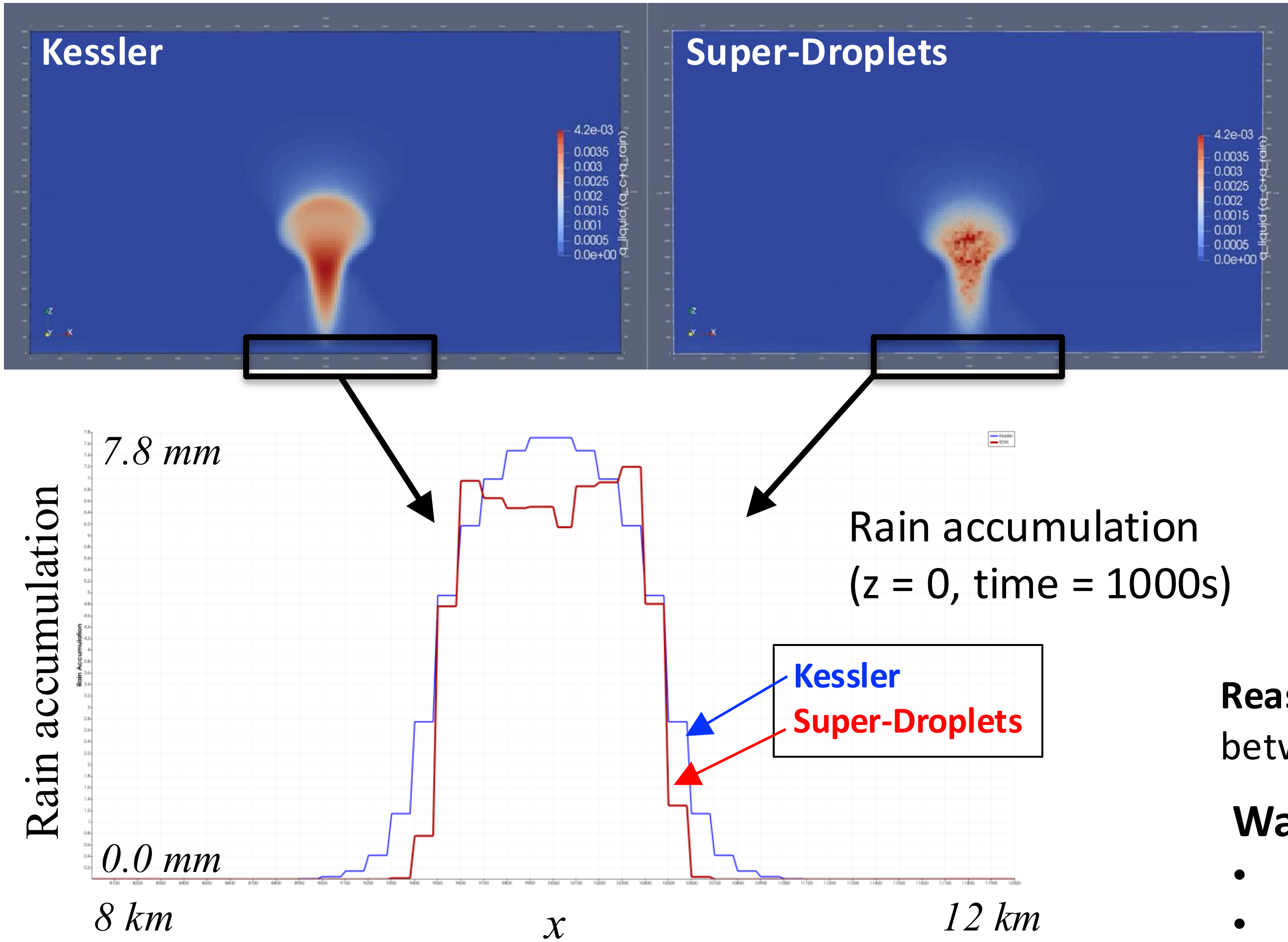
Monte-Carlo algorithm for super-droplets:

- In each grid cell, *shuffle particles*, split into two groups, and *create pairs*
- Compute *probability of collision* for each pair
- If they collide, *update super-droplets attributes*

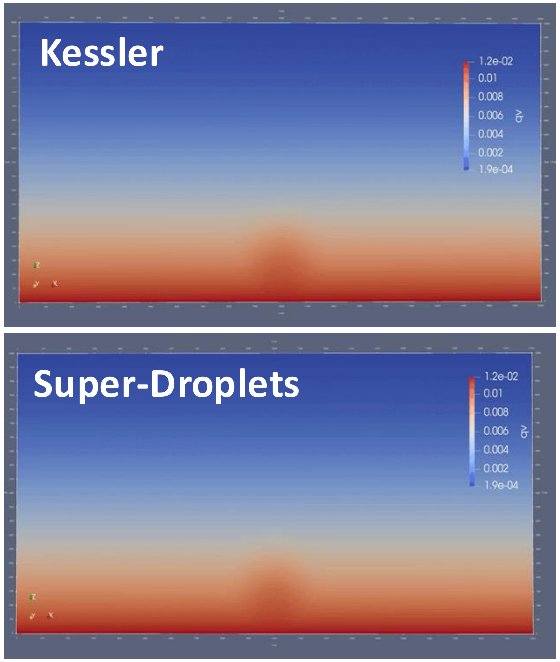


Example: 2D Rising Bubble

Total liquid water fraction q_c+q_r



Vapour fraction q_v



Reasonable agreement observed between Kessler and super-droplets

Wall times (on 4 V100 GPUs):

- Kessler: *63 seconds*
- Super-droplets: *381 seconds*