Elements of modern cloud modelling

PhD thesis defense

candidate:

Sylwester Arabas

supervisor:

Hanna Pawłowska

UNIVERSITAS VARSOVIENSIS

Thesis structure

- **Introductory chapters (or a guide through the papers):**

- **Appendix (the papers):**
  - Arabas & Pawlowska 2011, GMD
    Adaptive method of lines for multi-component aerosol condensational growth and CCN activation
  - Arabas & Shima 2013, JAS
    Large-Eddy Simulations of Trade Wind Cumuli Using Particle-Based Microphysics with Monte Carlo Coalescence
  - Arabas, Jaruga, Pawlowska & Grabowski 2013, arXiv
    libcloudph++ 0.1: single-moment bulk, double-moment bulk, and particle-based warm-rain microphysics library in C++

Thesis structure / plan of the talk

- **Introductory chapters (or a guide through the papers):**
  - Modelled phenomena
  - Model formulation
  - Model implementation
  - Model validation
- **Appendix (the papers):**
  - Arabas & Pawlowska 2011, GMD
    Adaptive method of lines for multi-component aerosol condensational growth and CCN activation
  - Arabas & Shima 2013, JAS
    Large-Eddy Simulations of Trade Wind Cumuli Using Particle-Based Microphysics with Monte Carlo Coalescence
  - Arabas, Jaruga, Pawlowska & Grabowski 2013, arXiv
    libcloudph++ 0.1: single-moment bulk, double-moment bulk, and particle-based warm-rain microphysics library in C++

Thesis structure / plan of the talk

- **Introductory chapters (or a guide through the papers):**
  - Modelled phenomena ⟶ aerosol-cloud-interactions
  - Model formulation
  - Model implementation
  - Model validation
- **Appendix (the papers):**
  - Arabas & Pawlowska 2011, GMD
    Adaptive method of lines for multi-component aerosol condensational growth and CCN activation
  - Arabas & Shima 2013, JAS
    Large-Eddy Simulations of Trade Wind Cumuli Using Particle-Based Microphysics with Monte Carlo Coalescence
  - Arabas, Jaruga, Pawlowska & Grabowski 2013, arXiv
    libcloudph++ 0.1: single-moment bulk, double-moment bulk, and particle-based warm-rain microphysics library in C++

Aerosol-cloud interactions: conceptual picture

Aerosol-cloud interactions: conceptual picture

background image: vitsly.ru / Hokusai

Aerosol-cloud interactions: conceptual picture

background image: `vitsly.ru` / Hokusai

Aerosol-cloud interactions: conceptual picture

- aerosol particles of natural and anthropogenic origin act as condensation nuclei

background image: vitsly.ru / Hokusai

Aerosol-cloud interactions: conceptual picture

- aerosol particles of natural and anthropogenic origin act as condensation nuclei
- cloud droplets grow by water vapour condensation

background image: vitsly.ru / Hokusai

Aerosol-cloud interactions: conceptual picture

- aerosol particles of natural and anthropogenic origin act as condensation nuclei
- cloud droplets grow by water vapour condensation
- rain drops form through collisions of cloud droplets

background image: `vitsly.ru` / Hokusai

Aerosol-cloud interactions: conceptual picture

- aerosol particles of natural and anthropogenic origin act as condensation nuclei
- cloud droplets grow by water vapour condensation
- rain drops form through collisions of cloud droplets
- aqueous chemical reactions irreversibly modify the drop composition

background image: `vitsly.ru` / Hokusai

Aerosol-cloud interactions: conceptual picture

- aerosol particles of natural and anthropogenic origin act as condensation nuclei
- cloud droplets grow by water vapour condensation
- rain drops form through collisions of cloud droplets
- aqueous chemical reactions irreversibly modify the drop composition
- rain drops precipitate washing out aerosol

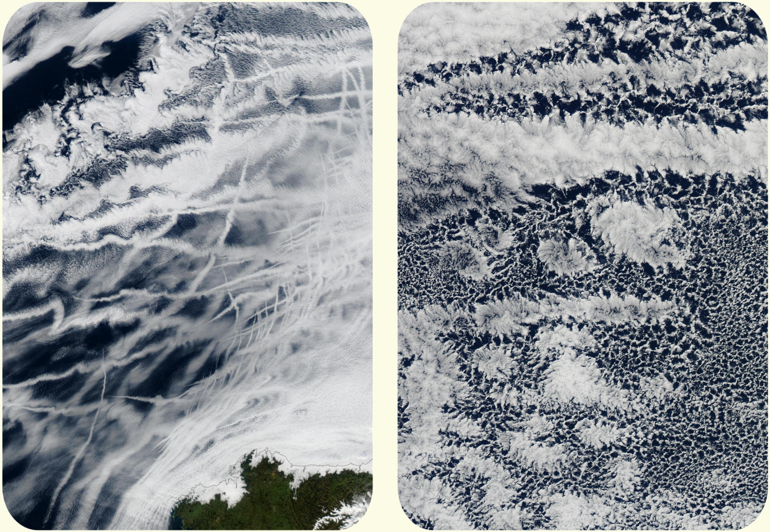
background image: vitsly.ru / Hokusai

Aerosol-cloud interactions: conceptual picture

- aerosol particles of natural and anthropogenic origin act as condensation nuclei
- cloud droplets grow by water vapour condensation
- rain drops form through collisions of cloud droplets
- aqueous chemical reactions irreversibly modify the drop composition
- rain drops precipitate washing out aerosol
- rain drops evaporate into aerosol particles of potentially altered size and/or composition (collisions, chemistry)

background image: vitsly.ru / Hokusai

Aerosol-cloud interactions: as seen from space



NASA/MODIS (27 Jan 2003 – Bay of Biscay; 17 Apr 2010 – off the coast of Peru)
http://visibleearth.nasa.gov/view.php?id=64992
http://earthobservatory.nasa.gov/IOTD/view.php?id=43795

Thesis structure / plan of the talk

- **Introductory chapters (or a guide through the papers):**
  - Modelled phenomena
  - Model formulation
  - Model implementation
  - Model validation
- **Appendix (the papers):**
  - Arabas & Pawlowska 2011, GMD
    Adaptive method of lines for multi-component aerosol condensational growth and CCN activation
  - Arabas & Shima 2013, JAS
    Large-Eddy Simulations of Trade Wind Cumuli Using Particle-Based Microphysics with Monte Carlo Coalescence
  - Arabas, Jaruga, Pawlowska & Grabowski 2013, arXiv
    libcloudph++ 0.1: single-moment bulk, double-moment bulk, and particle-based warm-rain microphysics library in C++

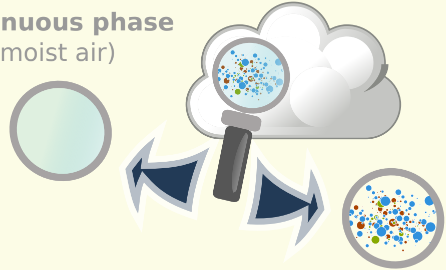
Thesis structure / plan of the talk

- **Introductory chapters (or a guide through the papers):**
  - Modelled phenomena
  - Model formulation ⟶ Lagrangian microphysics
  - Model implementation
  - Model validation
- **Appendix (the papers):**
  - Arabas & Pawlowska 2011, GMD
    Adaptive method of lines for multi-component aerosol condensational growth and CCN activation
  - Arabas & Shima 2013, JAS
    Large-Eddy Simulations of Trade Wind Cumuli Using Particle-Based Microphysics with Monte Carlo Coalescence
  - Arabas, Jaruga, Pawlowska & Grabowski 2013, arXiv
    libcloudph++ 0.1: single-moment bulk, double-moment bulk, and particle-based warm-rain microphysics library in C++

Lagrangian μ-physics

Lagrangian μ-physics

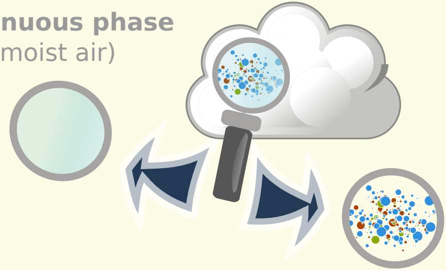
continuous phase
(moist air)



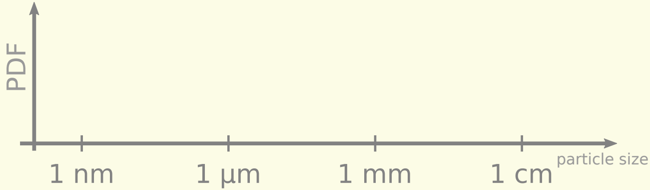
dispersed phase
(aerosol particles, cloud droplets, drizzle, rain, snow, ...)

Lagrangian μ-physics

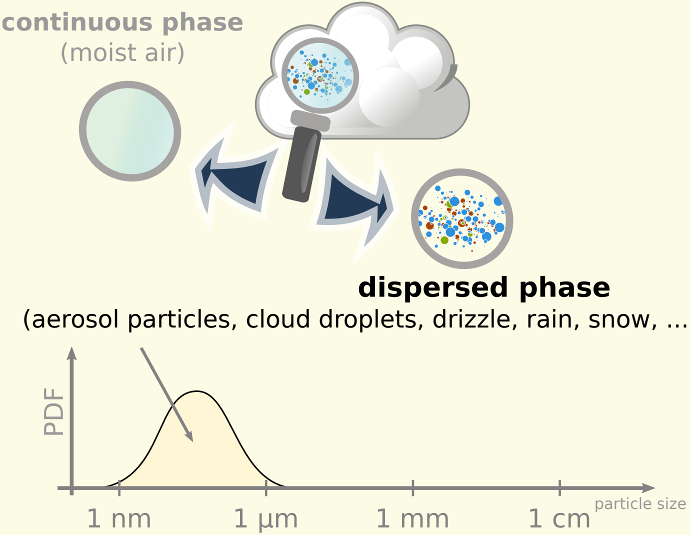
continuous phase
(moist air)



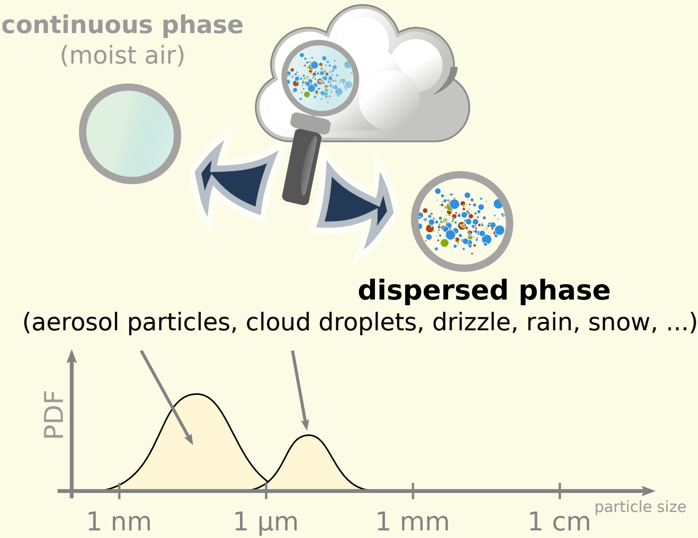
dispersed phase
(aerosol particles, cloud droplets, drizzle, rain, snow, ...)



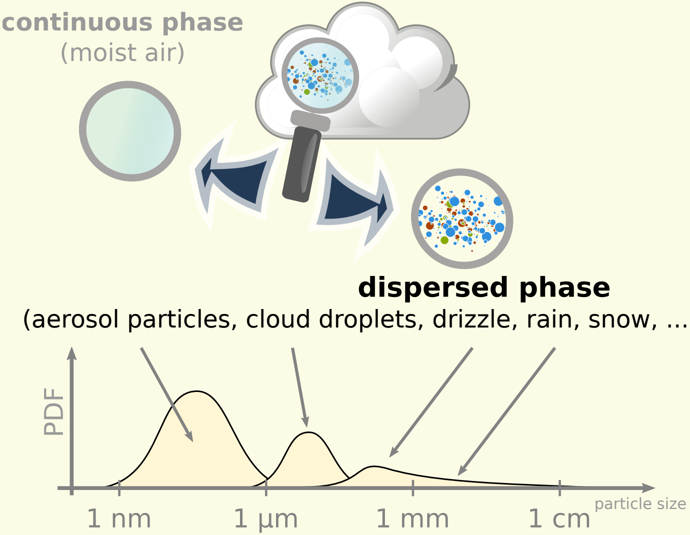
Lagrangian μ-physics



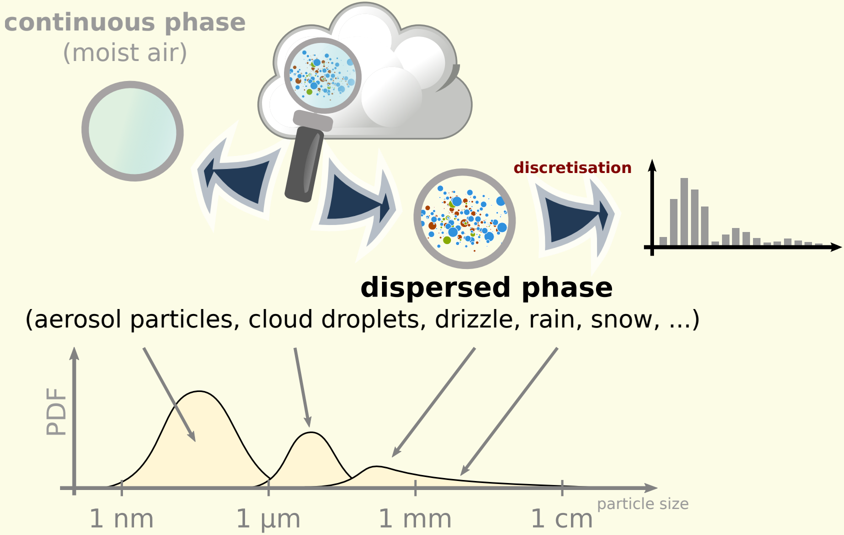
Lagrangian μ-physics



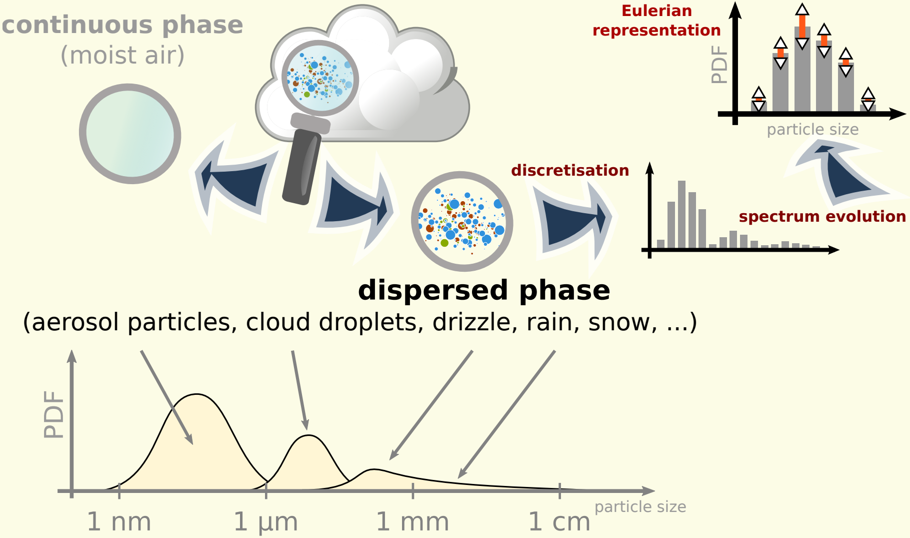
Lagrangian μ-physics



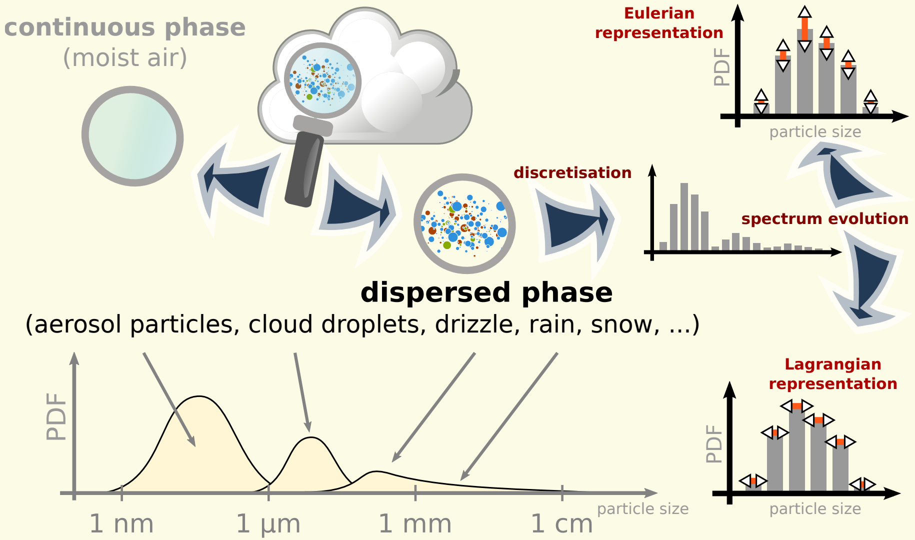
Lagrangian μ-physics



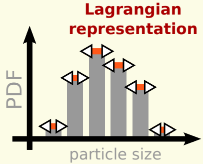
Lagrangian μ-physics



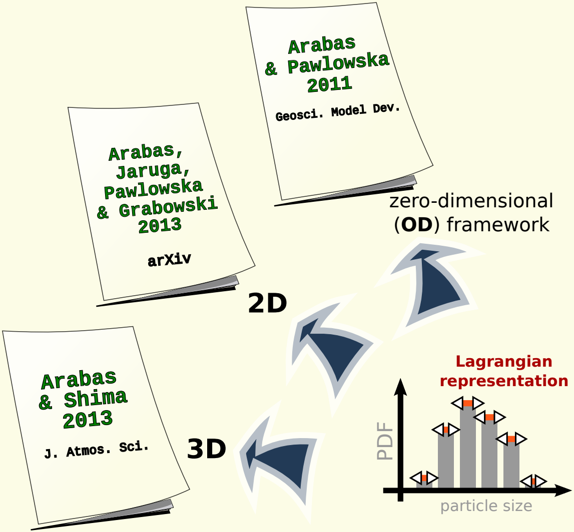
Lagrangian μ-physics



Lagrangian μ-physics in 0D, 2D and 3D

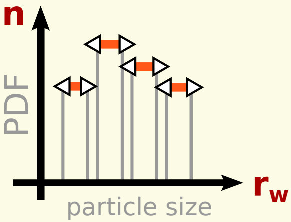


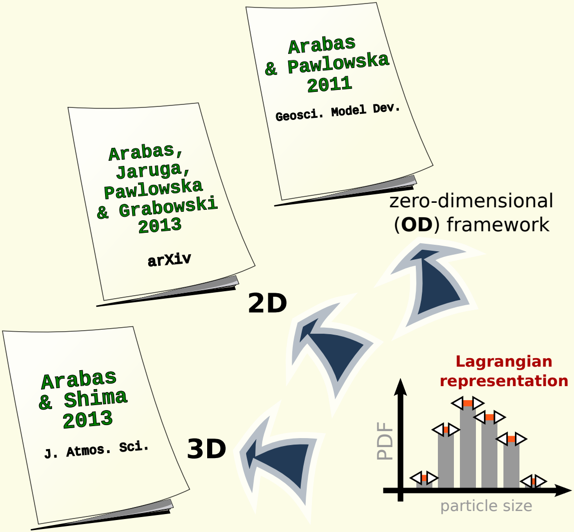
Lagrangian μ-physics in 0D, 2D and 3D



Lagrangian μ-physics in 0D, 2D and 3D

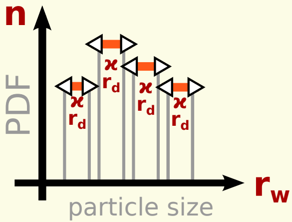
particle attributes:
- multiplicity (**n**),
- "wet" size (**r_w**),

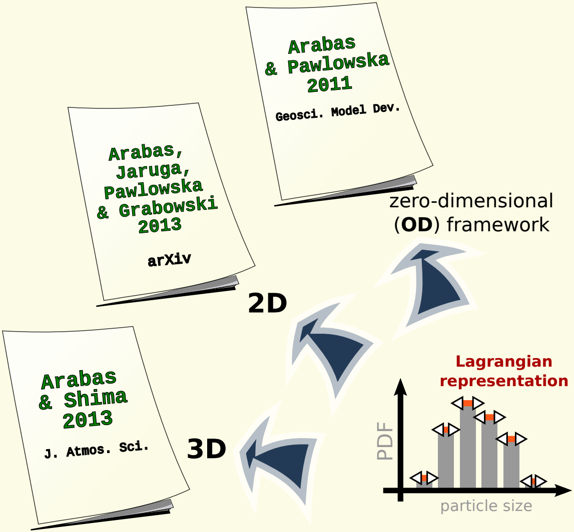




Lagrangian μ-physics in 0D, 2D and 3D

particle attributes:
- multiplicity (**n**),
- "wet" size (**r_w**),
- "dry" size (**r_d**),
- solute hygroscopicity (**$\varkappa$**)

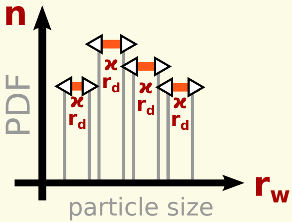


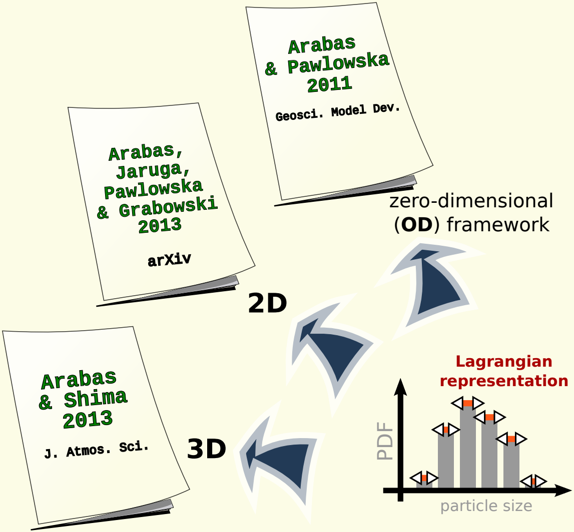


Lagrangian μ-physics in 0D, 2D and 3D

particle attributes:
- multiplicity (**n**),
- "wet" size (**r_w**),
- "dry" size (**r_d**),
- solute hygroscopicity (**$\varkappa$**)

processes:
- condensational growth

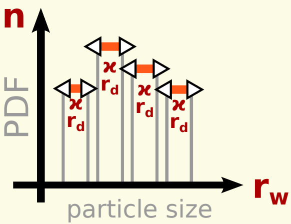


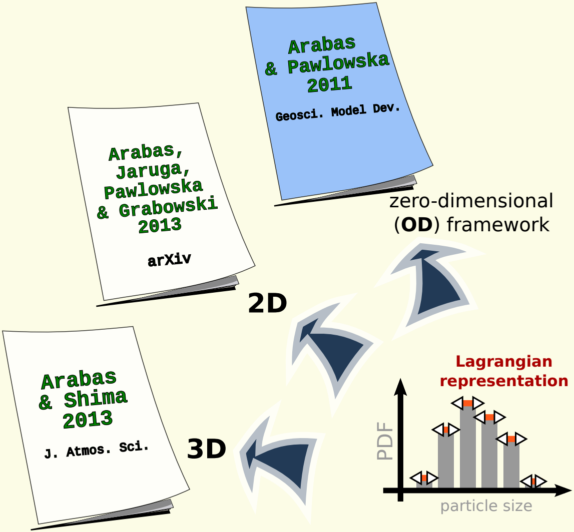


Lagrangian μ-physics in 0D, 2D and 3D

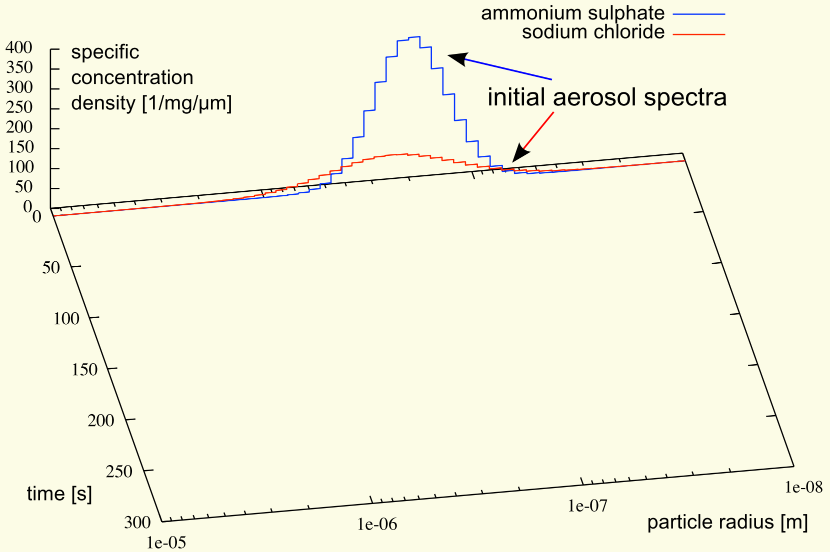
particle attributes:
- multiplicity (**n**),
- "wet" size ($\mathbf{r_w}$),
- "dry" size ($\mathbf{r_d}$),
- solute hygroscopicity (**ϰ**)

processes:
- condensational growth

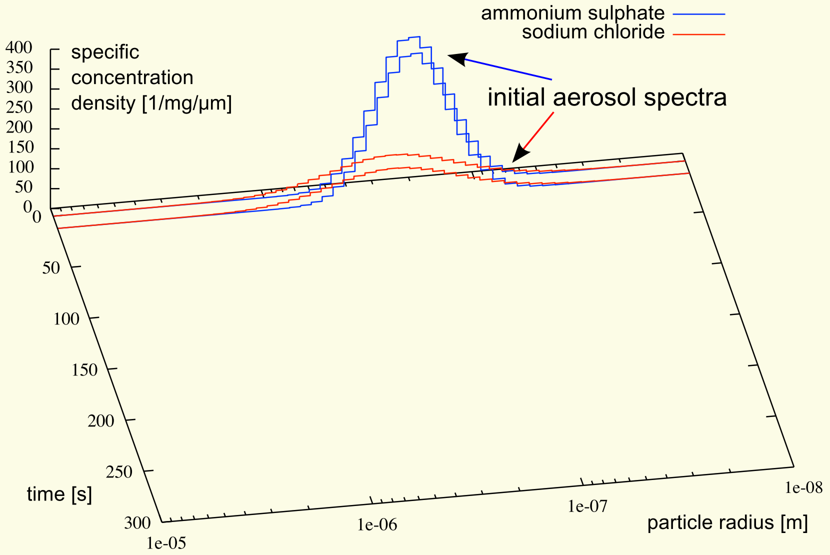




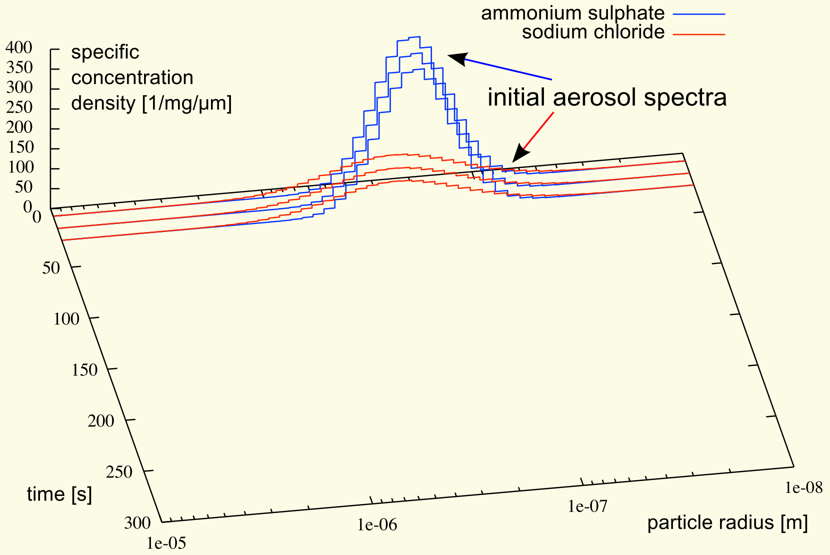
Lagrangian μ-physics in 0D (parcel model)



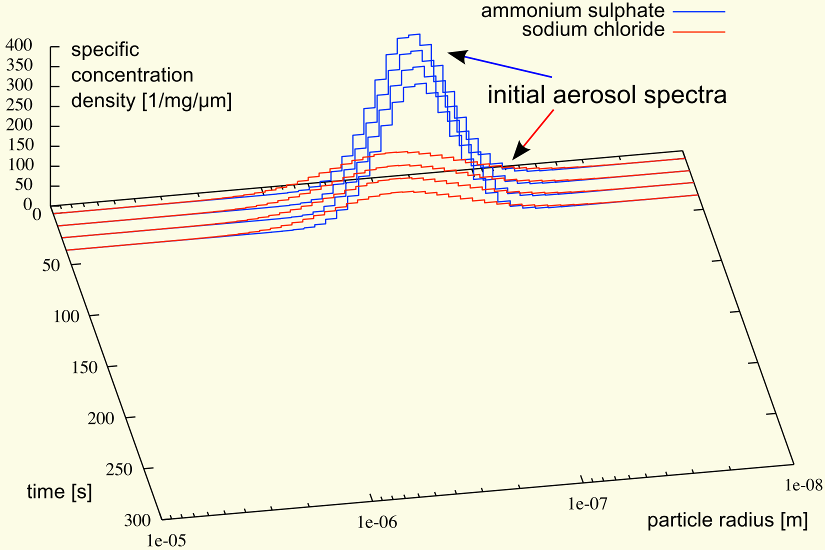
Lagrangian μ-physics in 0D (parcel model)



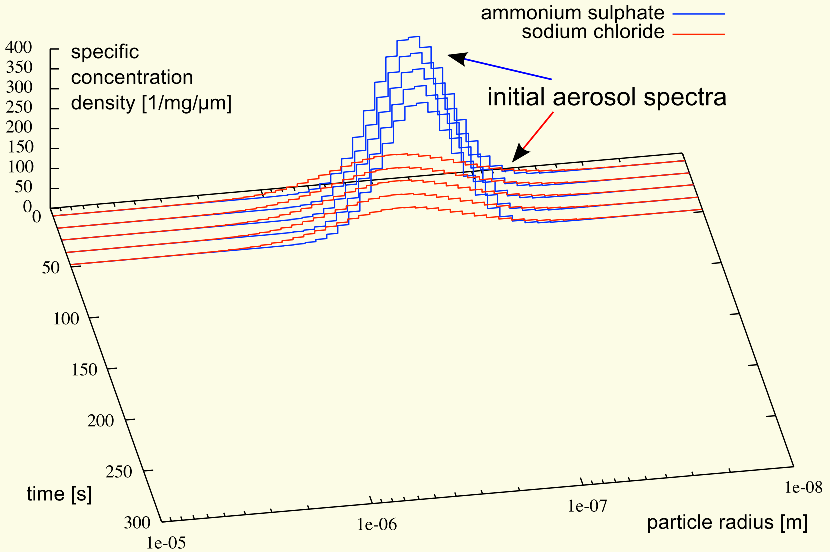
Lagrangian μ-physics in 0D (parcel model)



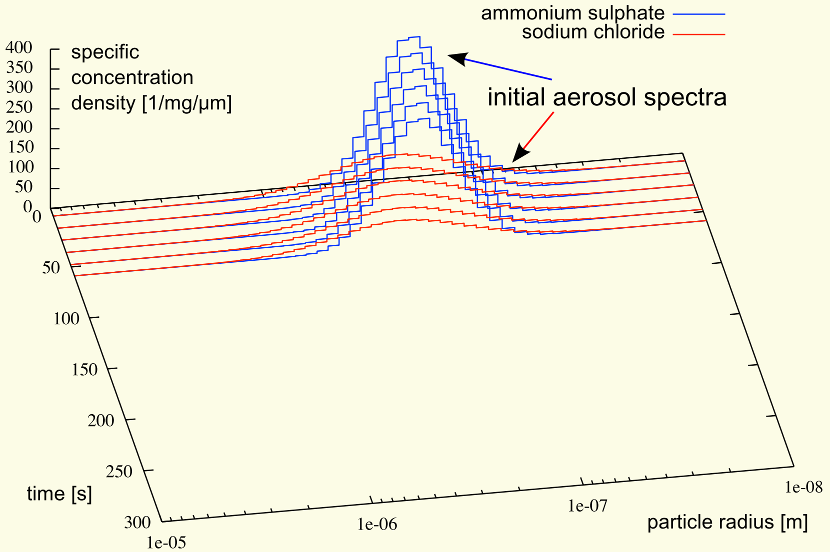
Lagrangian μ-physics in 0D (parcel model)



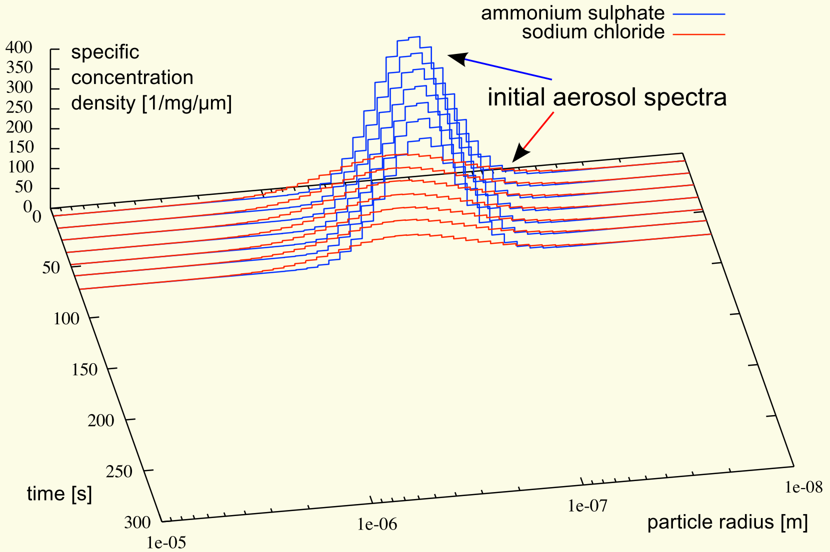
Lagrangian μ-physics in 0D (parcel model)



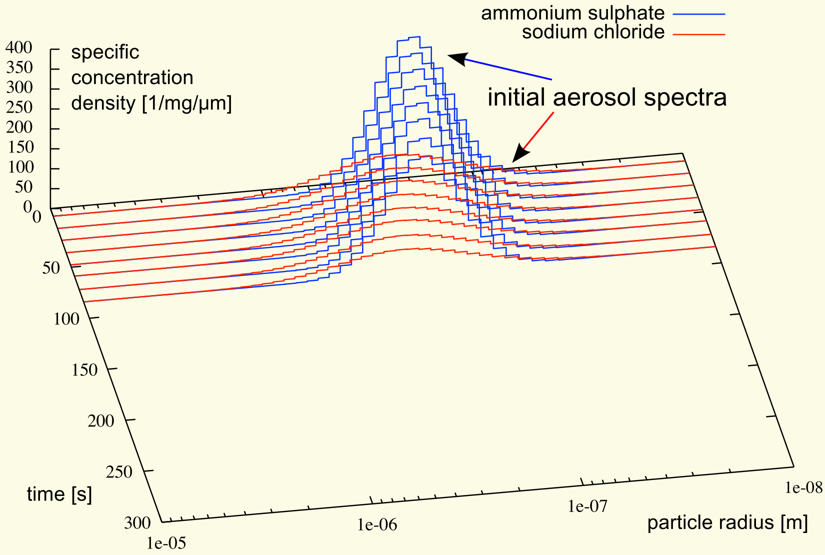
Lagrangian μ-physics in 0D (parcel model)



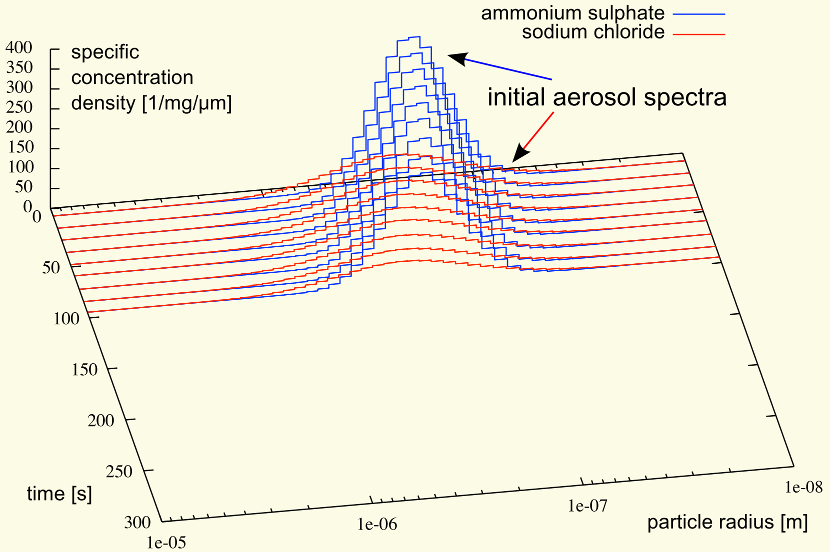
Lagrangian μ-physics in 0D (parcel model)



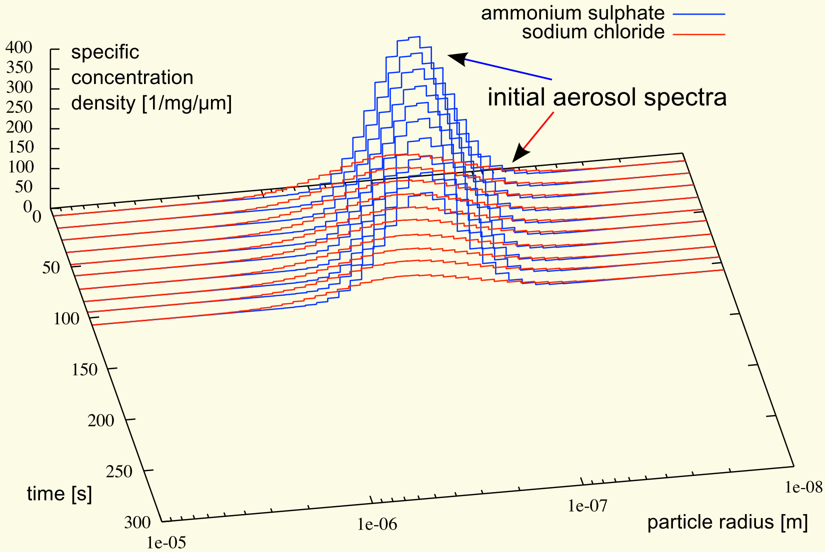
Lagrangian μ-physics in 0D (parcel model)



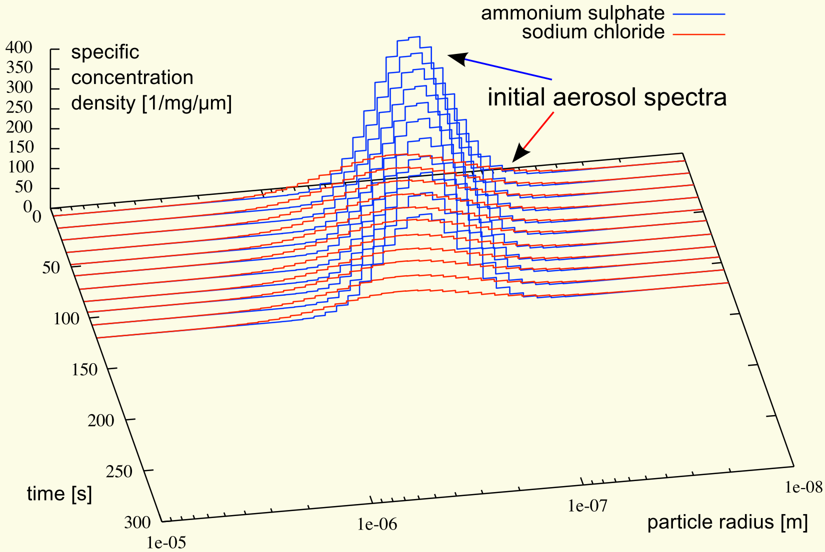
Lagrangian μ-physics in 0D (parcel model)



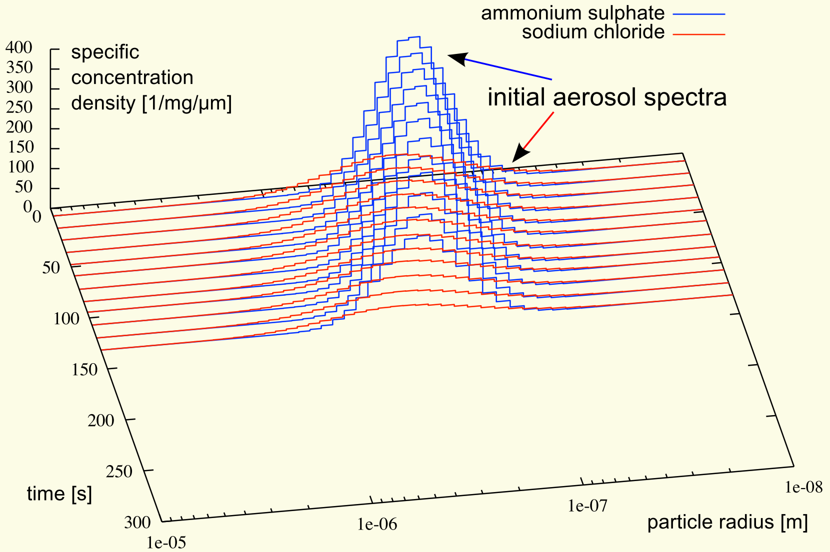
Lagrangian μ-physics in 0D (parcel model)



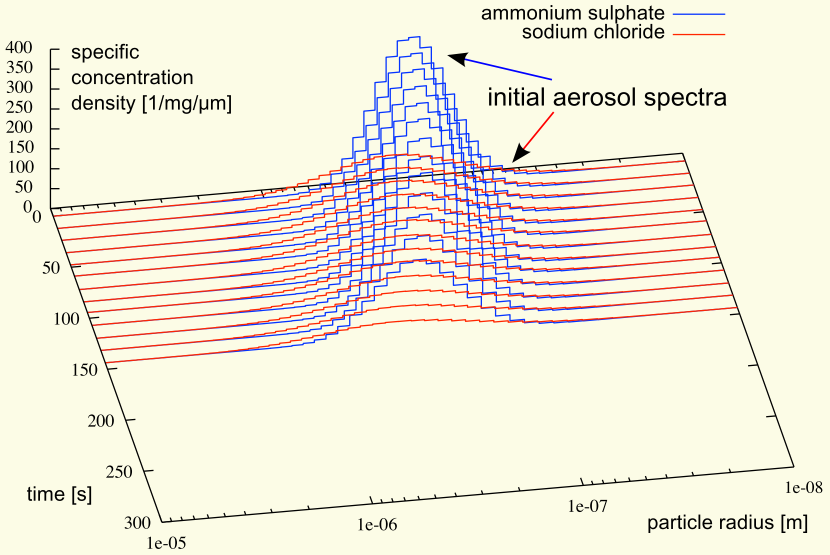
Lagrangian μ-physics in 0D (parcel model)



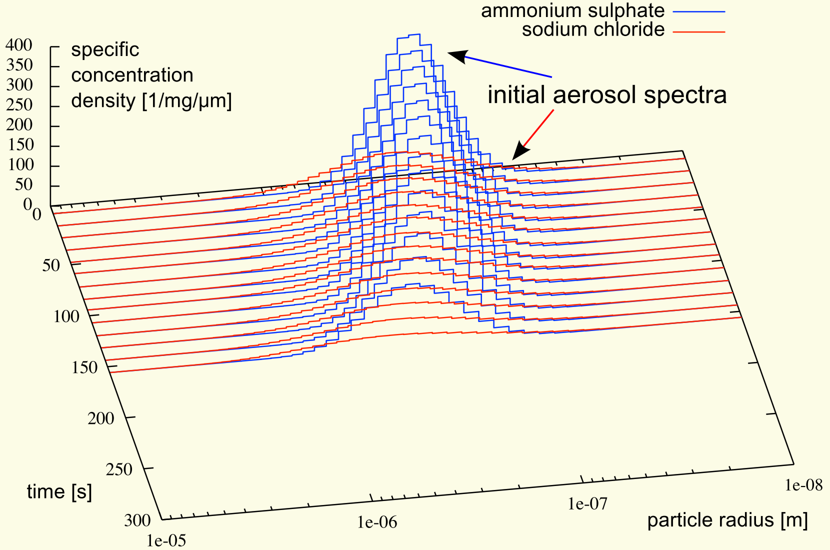
Lagrangian μ-physics in 0D (parcel model)



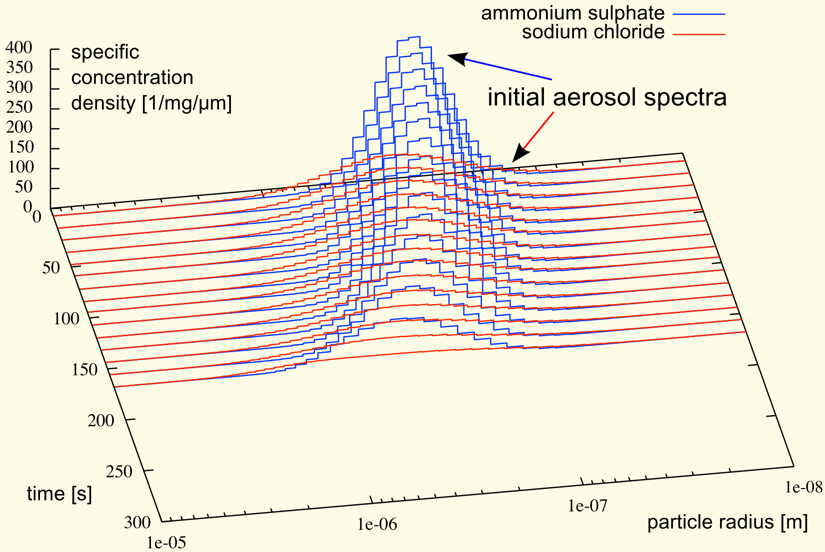
Lagrangian μ-physics in 0D (parcel model)



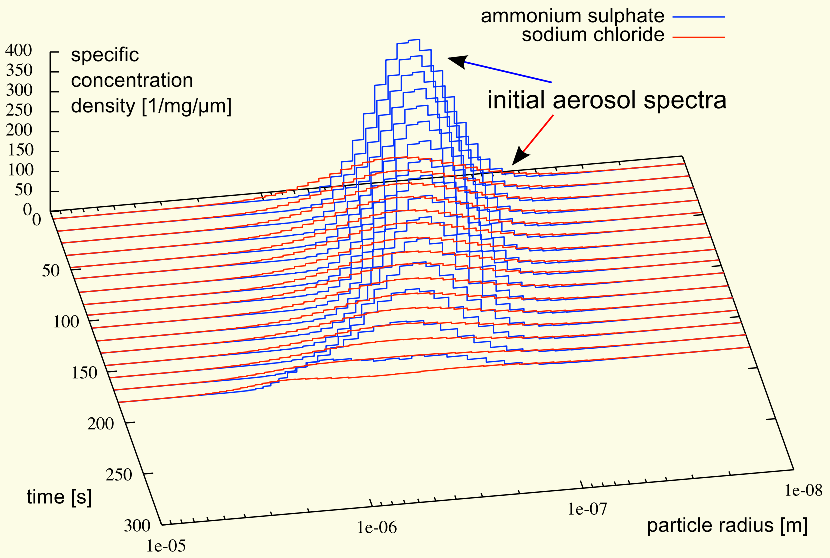
Lagrangian μ-physics in 0D (parcel model)



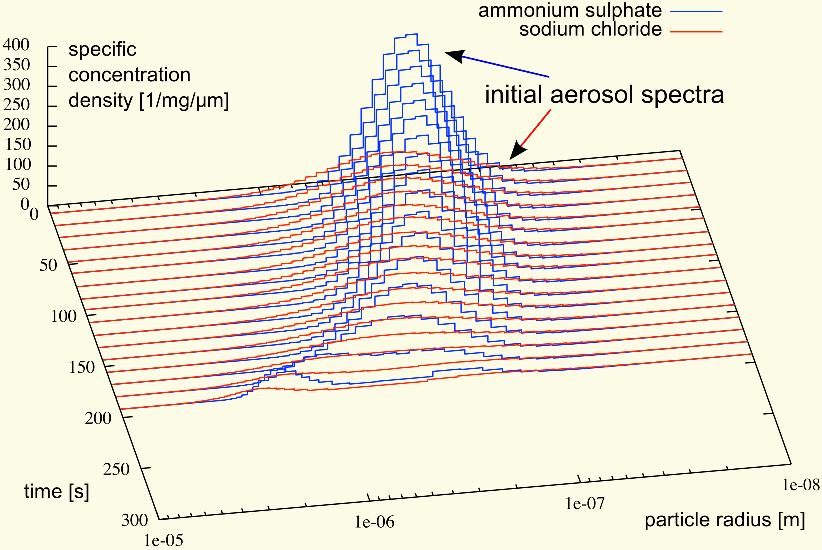
Lagrangian μ-physics in 0D (parcel model)



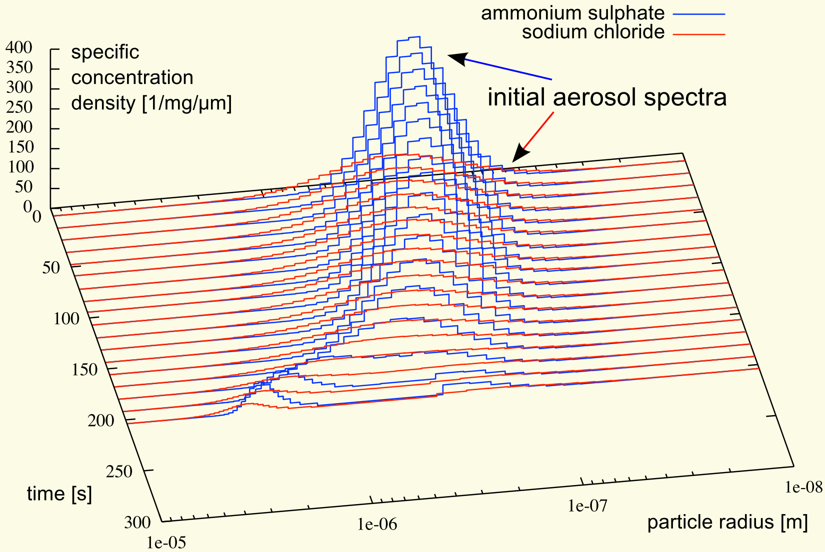
Lagrangian μ-physics in 0D (parcel model)



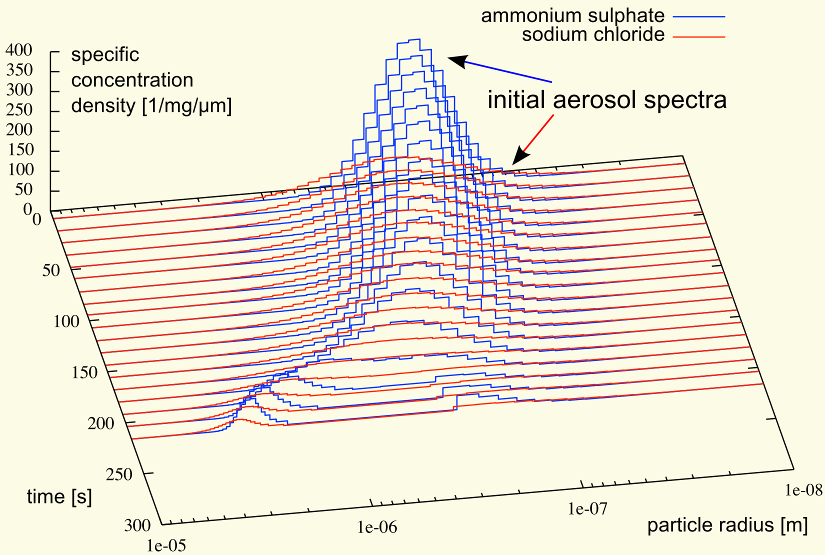
Lagrangian μ-physics in 0D (parcel model)



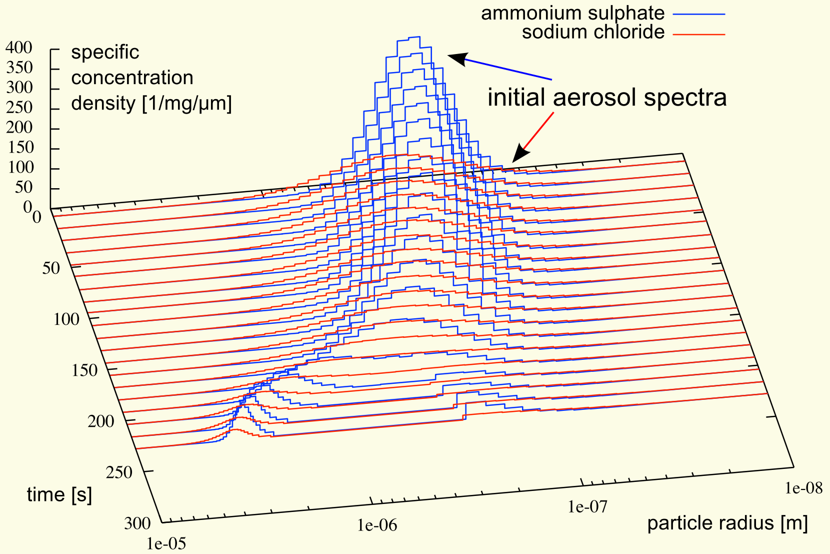
Lagrangian μ-physics in 0D (parcel model)



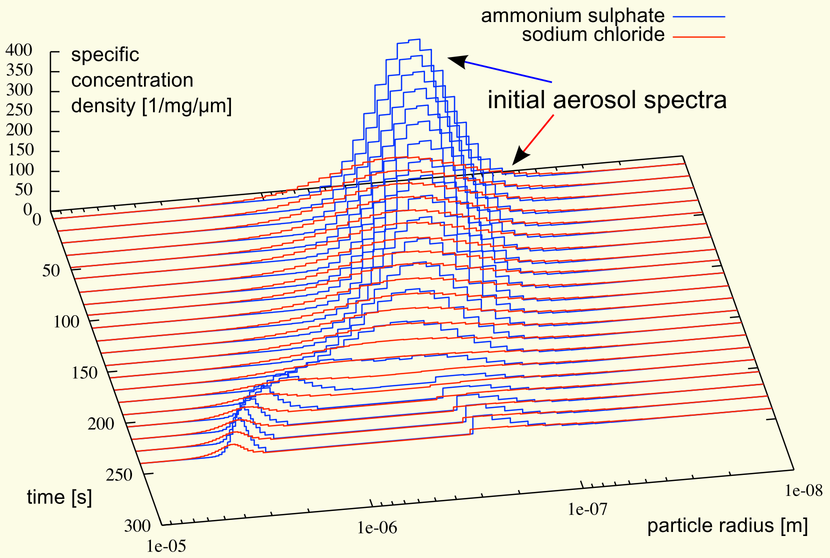
Lagrangian μ-physics in 0D (parcel model)



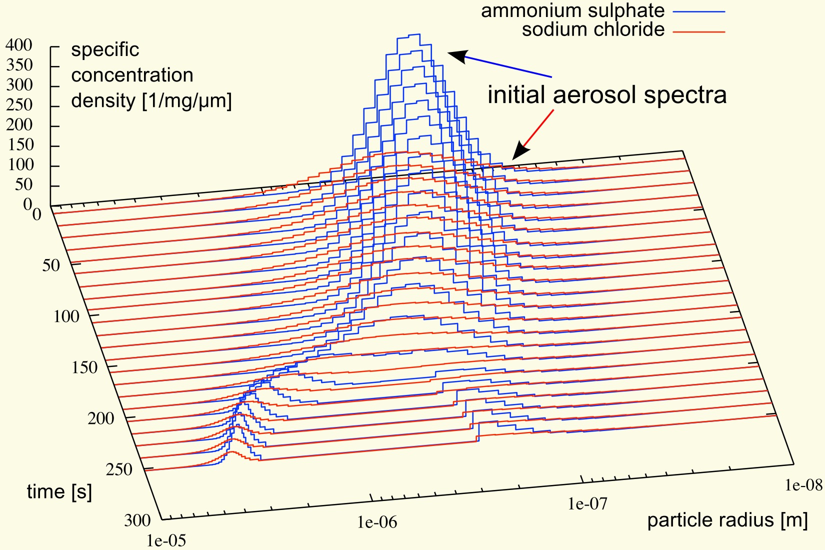
Lagrangian μ-physics in 0D (parcel model)



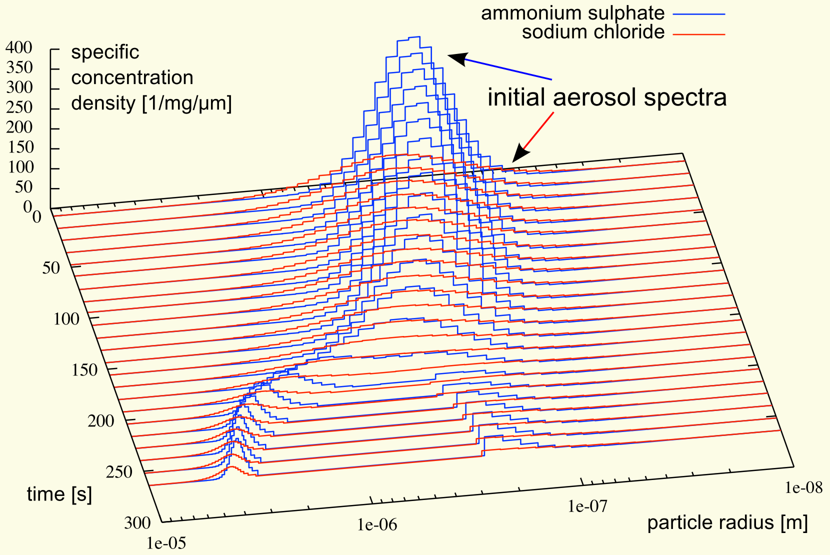
Lagrangian μ-physics in 0D (parcel model)



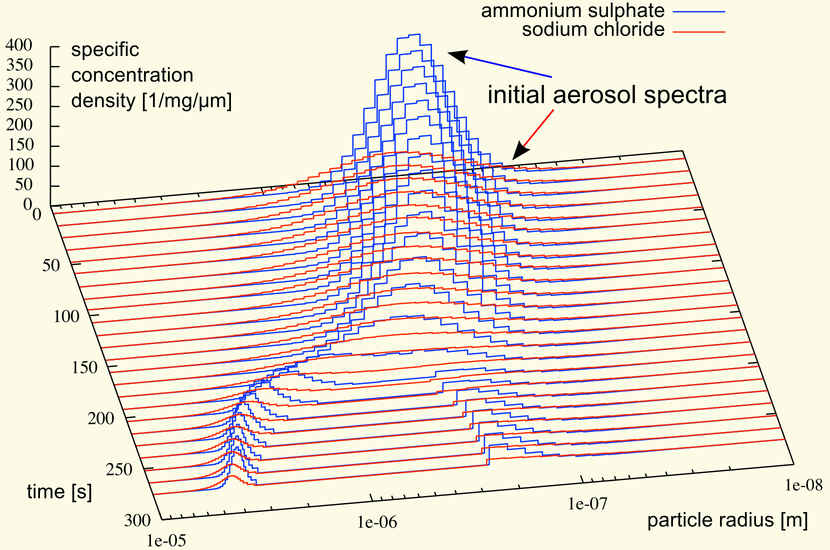
Lagrangian μ-physics in 0D (parcel model)



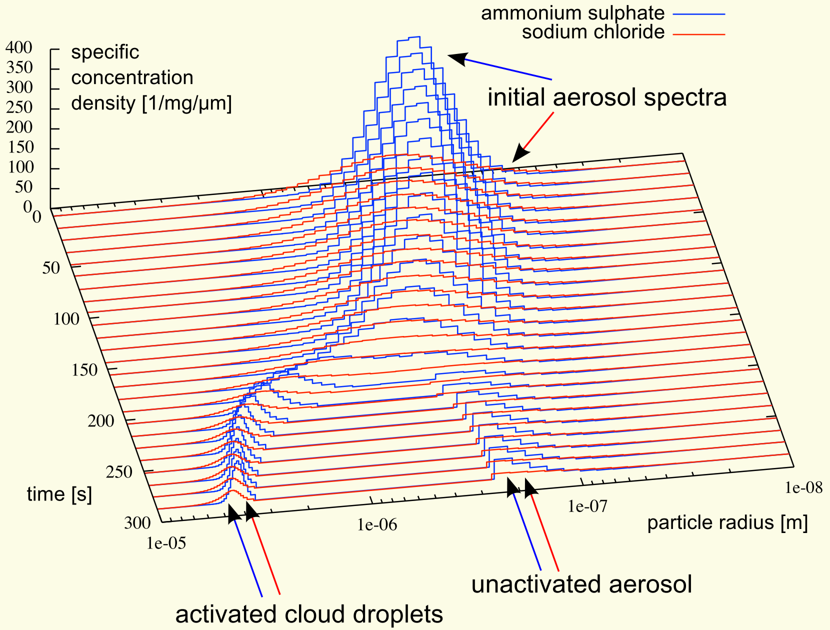
Lagrangian μ-physics in 0D (parcel model)



Lagrangian μ-physics in 0D (parcel model)



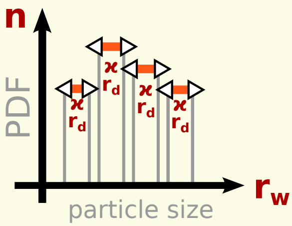
Lagrangian μ-physics in 0D (parcel model)

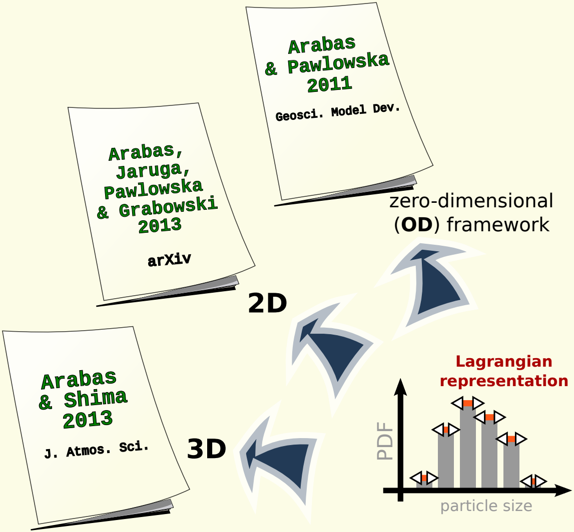


Lagrangian μ-physics in 0D, 2D and 3D

particle attributes:
- multiplicity (**n**),
- "wet" size ($\mathbf{r_w}$),
- "dry" size ($\mathbf{r_d}$),
- solute hygroscopicity ($\boldsymbol{\varkappa}$)

processes:
- condensational growth

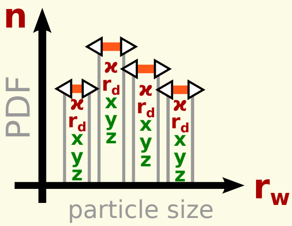


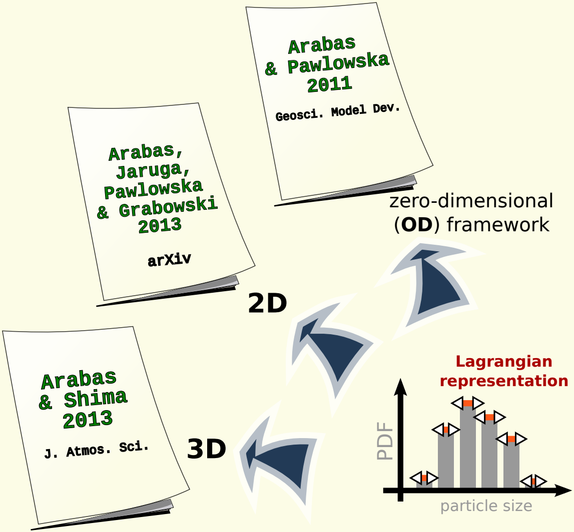


Lagrangian μ-physics in 0D, 2D and 3D

particle attributes:
- multiplicity (**n**),
- "wet" size (r_w),
- "dry" size (r_d),
- solute hygroscopicity ($\varkappa$)
- spatial coordinates (**x, y, z**)

processes:
- condensational growth

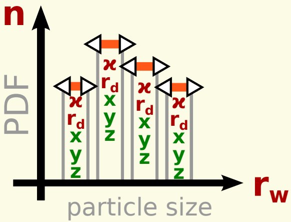


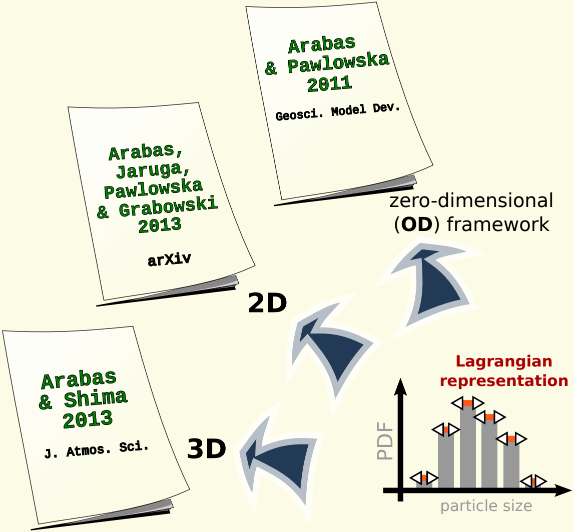


Lagrangian μ-physics in 0D, 2D and 3D

particle attributes:
- multiplicity (**n**),
- "wet" size (**r_w**),
- "dry" size (**r_d**),
- solute hygroscopicity (**κ**)
- spatial coordinates (**x, y, z**)

processes:
- condensational growth
- sedimentation
- collisional growth

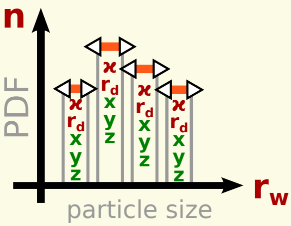


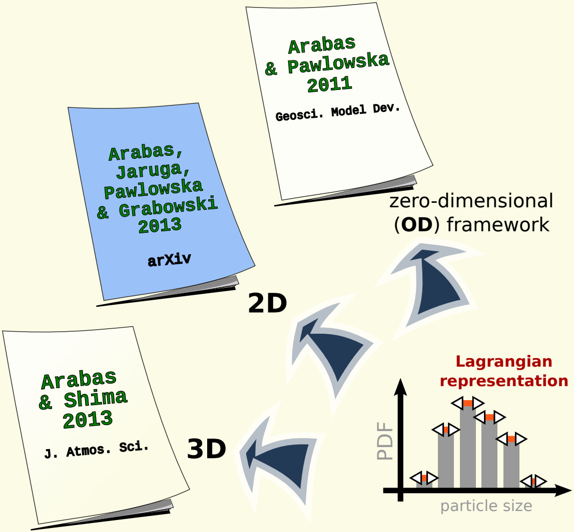


Lagrangian μ-physics in 0D, 2D and 3D

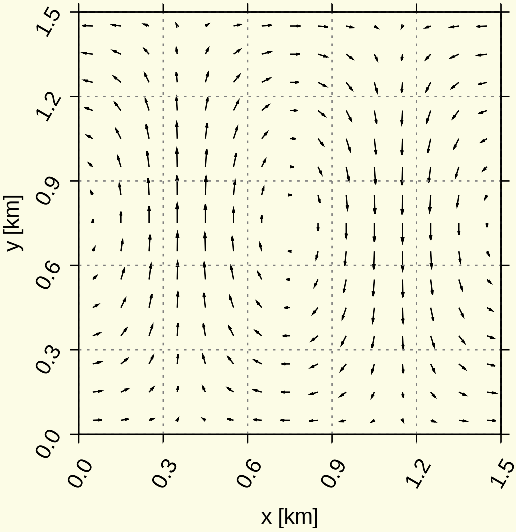
particle attributes:
- multiplicity (**n**),
- "wet" size (r_w),
- "dry" size (r_d),
- solute hygroscopicity ($\varkappa$)
- spatial coordinates (**x, y, z**)

processes:
- condensational growth
- sedimentation
- collisional growth

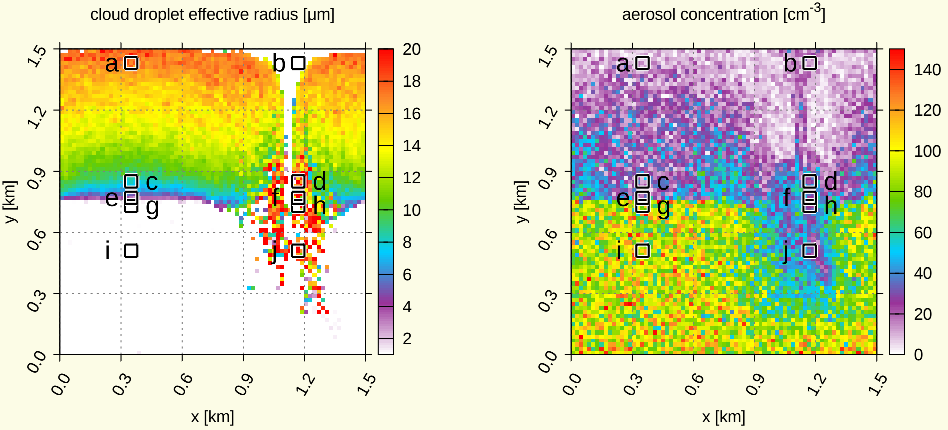




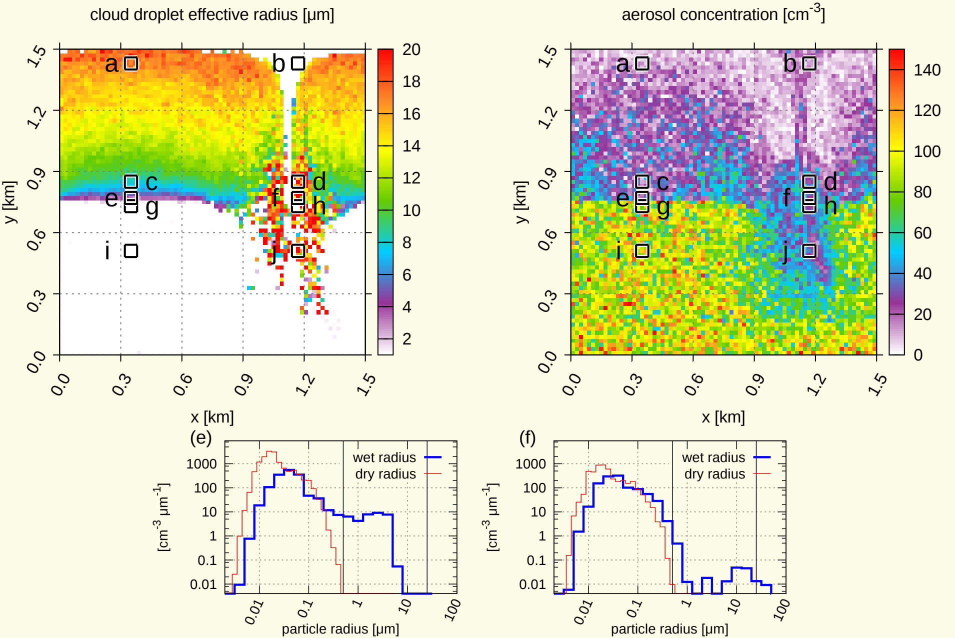
Lagrangian μ-physics in 2D (prescribed-flow model)



Lagrangian μ-physics in 2D (prescribed-flow model)



Lagrangian μ-physics in 2D (prescribed-flow model)



Thesis structure / plan of the talk

- **Introductory chapters (or a guide through the papers):**
  - Modelled phenomena
  - Model formulation
  - Model implementation
  - Model validation
- **Appendix (the papers):**
  - Arabas & Pawlowska 2011, GMD
    Adaptive method of lines for multi-component aerosol condensational growth and CCN activation
  - Arabas & Shima 2013, JAS
    Large-Eddy Simulations of Trade Wind Cumuli Using Particle-Based Microphysics with Monte Carlo Coalescence
  - Arabas, Jaruga, Pawlowska & Grabowski 2013, arXiv
    libcloudph++ 0.1: single-moment bulk, double-moment bulk, and particle-based warm-rain microphysics library in C++

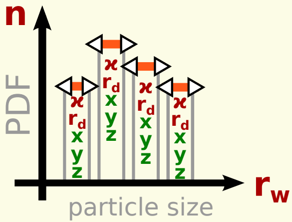
Thesis structure / plan of the talk

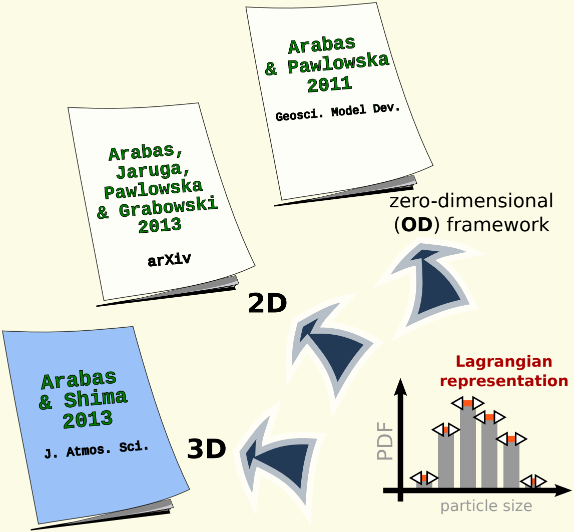
- **Introductory chapters (or a guide through the papers):**
  - Modelled phenomena
  - Model formulation
  - Model implementation
  - Model validation ⟶ comparison with aircraft observations
- **Appendix (the papers):**
  - Arabas & Pawlowska 2011, GMD
    Adaptive method of lines for multi-component aerosol condensational growth and CCN activation
  - Arabas & Shima 2013, JAS
    Large-Eddy Simulations of Trade Wind Cumuli Using Particle-Based Microphysics with Monte Carlo Coalescence
  - Arabas, Jaruga, Pawlowska & Grabowski 2013, arXiv
    libcloudph++ 0.1: single-moment bulk, double-moment bulk, and particle-based warm-rain microphysics library in C++

Lagrangian μ-physics in 0D, 2D and 3D

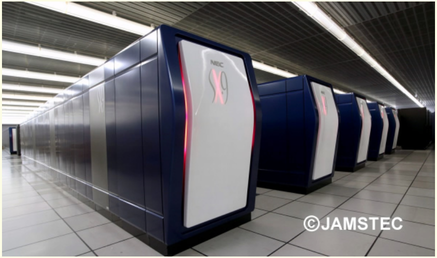
particle attributes:
- multiplicity (**n**),
- "wet" size (**r_w**),
- "dry" size (**r_d**),
- solute hygroscopicity (**$\varkappa$**)
- spatial coordinates (**x, y, z**)

processes:
- condensational growth
- sedimentation
- collisional growth

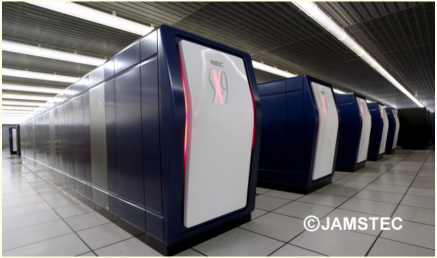




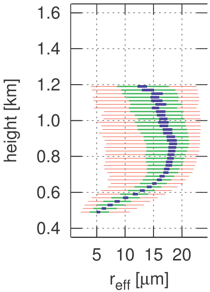
Lagrangian μ-physics in 3D: simulations vs. aircraft data



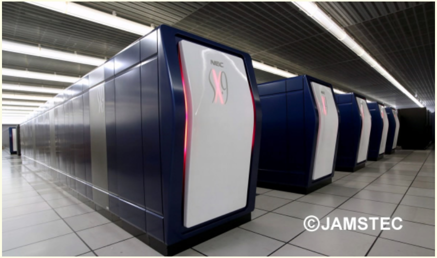
Lagrangian μ-physics in 3D: simulations vs. aircraft data



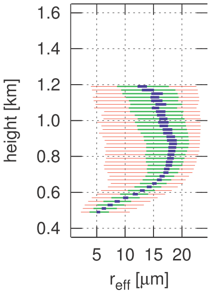
Arabas & Shima 2013, JAS



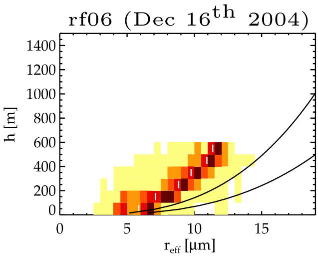
Lagrangian μ-physics in 3D: simulations vs. aircraft data



Arabas & Shima 2013, JAS



Arabas, Pawlowska, Grabowski 2009, GRL



(h = height - 550 m)

Thesis structure / plan of the talk

- **Introductory chapters (or a guide through the papers):**
  - Modelled phenomena
  - Model formulation
  - Model implementation
  - Model validation
- **Appendix (the papers):**
  - Arabas & Pawlowska 2011, GMD
    Adaptive method of lines for multi-component aerosol condensational growth and CCN activation
  - Arabas & Shima 2013, JAS
    Large-Eddy Simulations of Trade Wind Cumuli Using Particle-Based Microphysics with Monte Carlo Coalescence
  - Arabas, Jaruga, Pawlowska & Grabowski 2013, arXiv
    libcloudph++ 0.1: single-moment bulk, double-moment bulk, and particle-based warm-rain microphysics library in C++

Thesis structure / plan of the talk

- **Introductory chapters (or a guide through the papers):**
  - Modelled phenomena
  - Model formulation
  - Model implementation ⟶ software-engineering aspects
  - Model validation
- **Appendix (the papers):**
  - Arabas & Pawlowska 2011, GMD
    Adaptive method of lines for multi-component aerosol condensational growth and CCN activation
  - Arabas & Shima 2013, JAS
    Large-Eddy Simulations of Trade Wind Cumuli Using Particle-Based Microphysics with Monte Carlo Coalescence
  - Arabas, Jaruga, Pawlowska & Grabowski 2013, arXiv
    libcloudph++ 0.1: single-moment bulk, double-moment bulk, and particle-based warm-rain microphysics library in C++

Software development approach embraced at our group

public/social scientific coding

Software development approach embraced at our group

public/social scientific coding

let's let **anyone**:

- reproduce the results (**free**)

Software development approach embraced at our group

public/social scientific coding

let's let **anyone**:

- reproduce the results (**free**)
- look into the code (**open**)

Software development approach embraced at our group

public/social scientific coding

let's let **anyone**:

- reproduce the results (**free**)
- look into the code (**open**)
- understand the code (**succinct, documented**)

Software development approach embraced at our group

public/social scientific coding

let's let **anyone**:

- reproduce the results (**free**)
- look into the code (**open**)
- understand the code (**succinct, documented**)
- reuse the code (**modular, reusable**)

Software development approach embraced at our group

public/social scientific coding

let's let **anyone**:

- reproduce the results (**free**)
- look into the code (**open**)
- understand the code (**succinct, documented**)
- reuse the code (**modular, reusable**)
- continue the work (**extendable, libre**)

Software development approach embraced at our group

Software development approach embraced at our group

- succinct, extendable code $\rightsquigarrow$ object-oriented code in C++, libraries

Software development approach embraced at our group

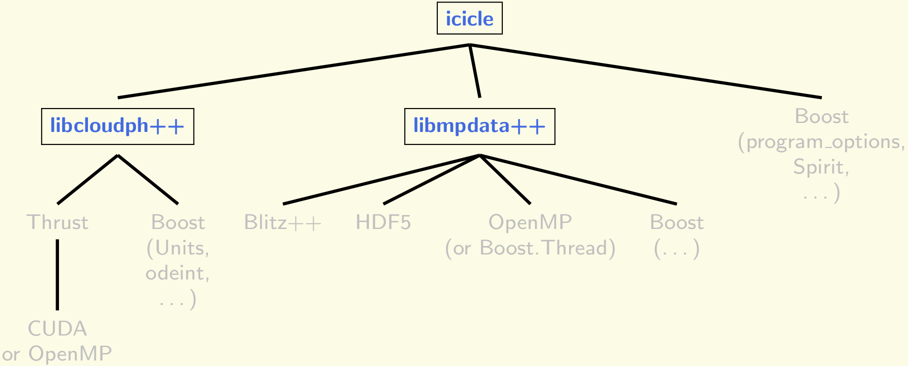
- succinct, extendable code $\rightsquigarrow$ object-oriented code in C++, libraries
- free, open & libre code $\rightsquigarrow$ GPL license, public repository (github)

Software development approach embraced at our group

- succinct, extendable code $\rightsquigarrow$ object-oriented code in C++, libraries
- free, open & libre code $\rightsquigarrow$ GPL license, public repository (github)
- modular, reusable code $\rightsquigarrow$ code itself structured into libraries

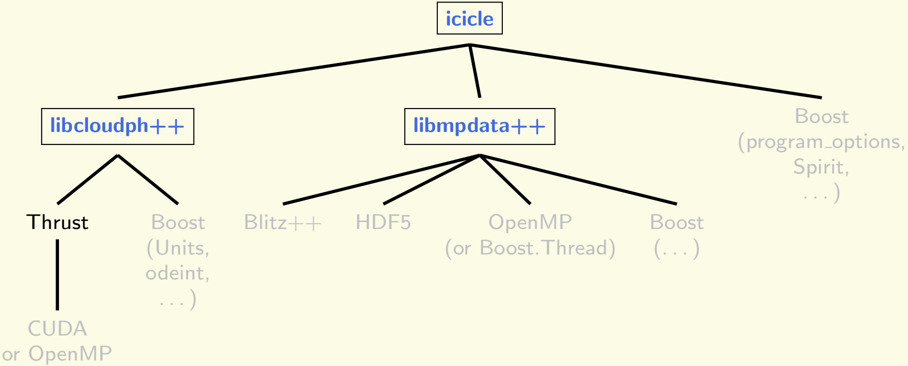
Software development approach embraced at our group

- succinct, extendable code $\rightsquigarrow$ object-oriented code in C++, libraries
- free, open & libre code $\rightsquigarrow$ GPL license, public repository (github)
- modular, reusable code $\rightsquigarrow$ code itself structured into libraries



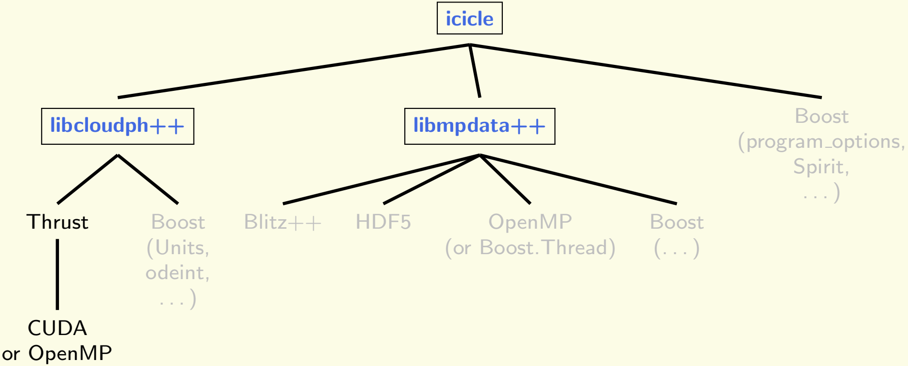
Software development approach embraced at our group

- succinct, extendable code ⇝ object-oriented code in C++, libraries
- free, open & libre code ⇝ GPL license, public repository (github)
- modular, reusable code ⇝ code itself structured into libraries



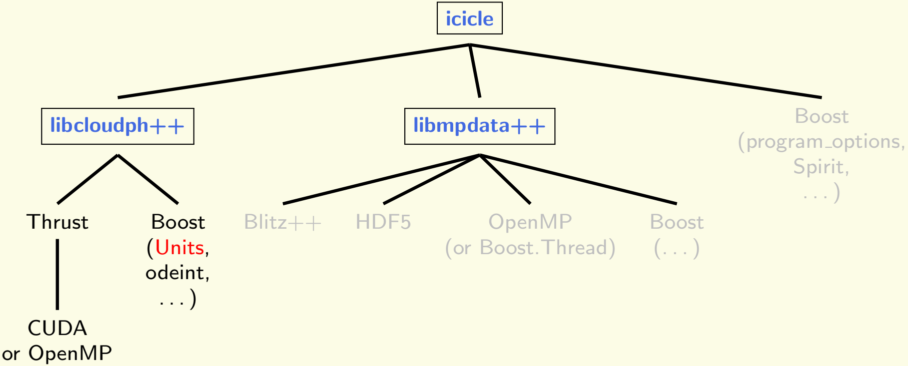
Software development approach embraced at our group

- succinct, extendable code ⇝ object-oriented code in C++, libraries
- free, open & libre code ⇝ GPL license, public repository (github)
- modular, reusable code ⇝ code itself structured into libraries



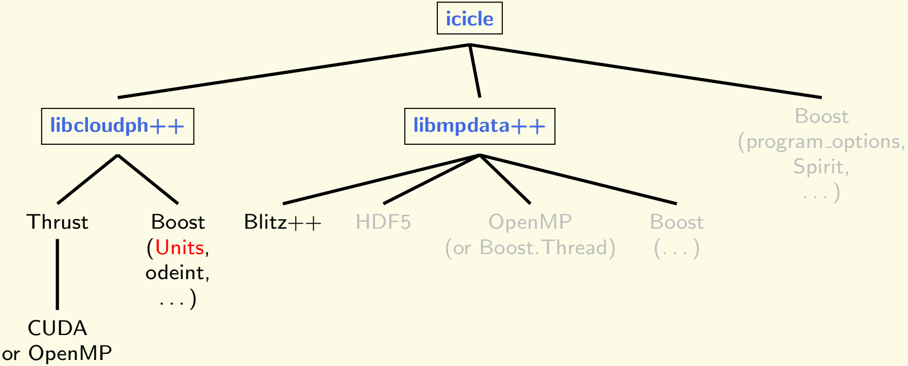
Software development approach embraced at our group

- succinct, extendable code ⇝ object-oriented code in C++, libraries
- free, open & libre code ⇝ GPL license, public repository (github)
- modular, reusable code ⇝ code itself structured into libraries



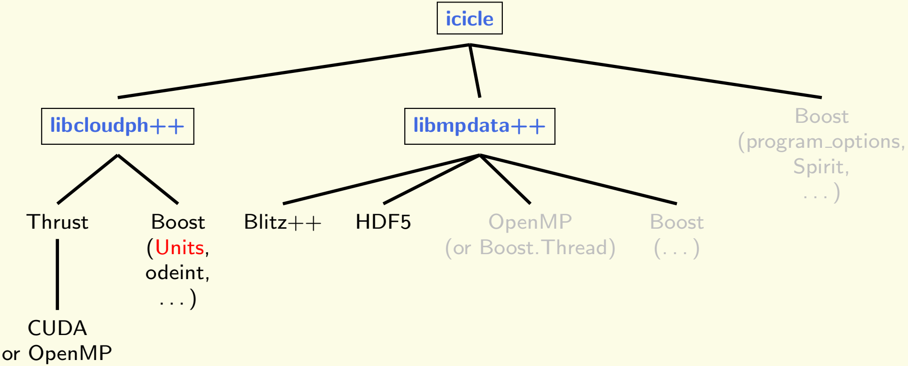
Software development approach embraced at our group

- succinct, extendable code ⇝ object-oriented code in C++, libraries
- free, open & libre code ⇝ GPL license, public repository (github)
- modular, reusable code ⇝ code itself structured into libraries



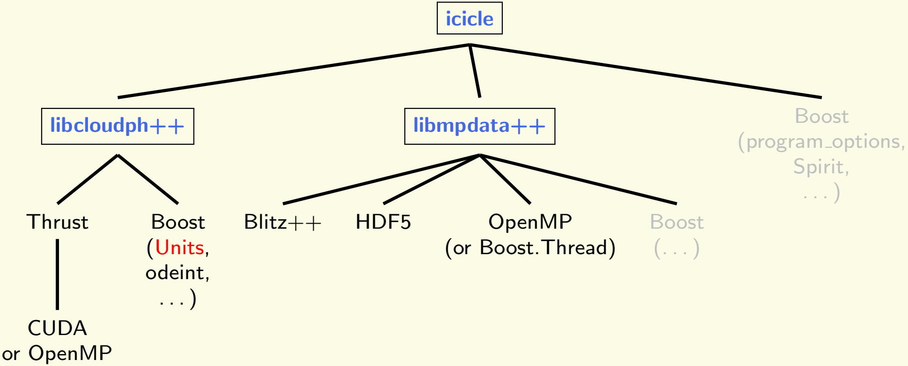
Software development approach embraced at our group

- succinct, extendable code ⇝ object-oriented code in C++, libraries
- free, open & libre code ⇝ GPL license, public repository (github)
- modular, reusable code ⇝ code itself structured into libraries



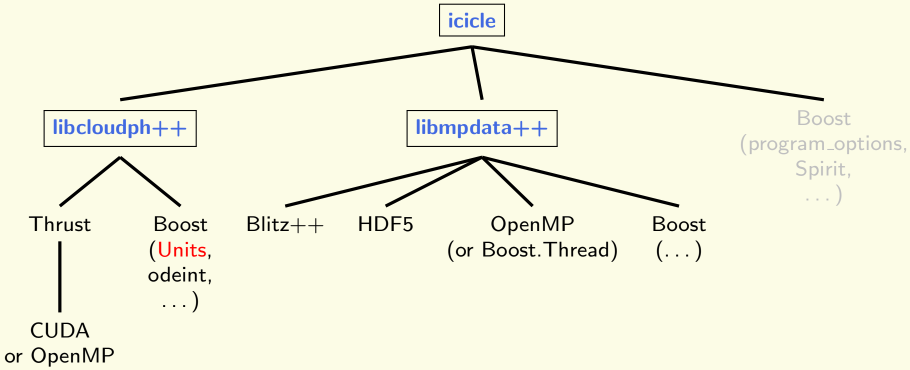
Software development approach embraced at our group

- succinct, extendable code ⇝ object-oriented code in C++, libraries
- free, open & libre code ⇝ GPL license, public repository (github)
- modular, reusable code ⇝ code itself structured into libraries



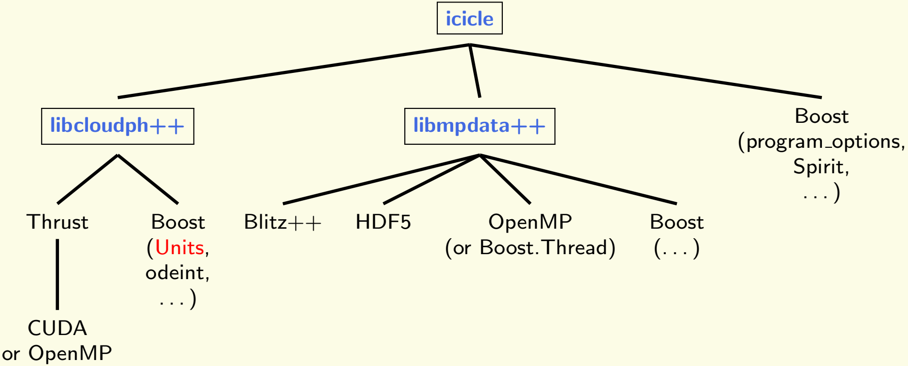
Software development approach embraced at our group

- succinct, extendable code ⇝ object-oriented code in C++, libraries
- free, open & libre code ⇝ GPL license, public repository (github)
- modular, reusable code ⇝ code itself structured into libraries



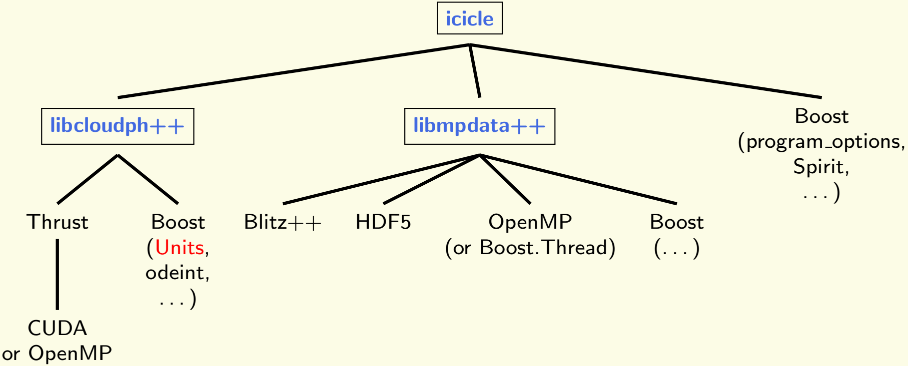
Software development approach embraced at our group

- succinct, extendable code ⇝ object-oriented code in C++, libraries
- free, open & libre code ⇝ GPL license, public repository (github)
- modular, reusable code ⇝ code itself structured into libraries



Software development approach embraced at our group

- succinct, extendable code ⇝ object-oriented code in C++, libraries
- free, open & libre code ⇝ GPL license, public repository (github)
- modular, reusable code ⇝ code itself structured into libraries



Ongoing work (NCN-funded HARMONIA project):
icicles – an LES system based on libmpdata++ and libcloudph++ developed as a tool for studying **aerosol processing by clouds**

code:

- **libcloudph++**: `github.com/slayoo/libcloudphxx`
- **libmpdata++**: `github.com/slayoo/libmpdataxx`
- **icicle**: `github.com/slayoo/icicle`

papers:

- 0D: **Arabas & Pawlowska 2011** doi:10.5194/gmd-4-15-2011
- 3D: **Arabas & Shima 2013** doi:10.1175/JAS-D-12-0295.1
- 2D: **Arabas, Jaruga et al. 2013** arXiv:1310.1905

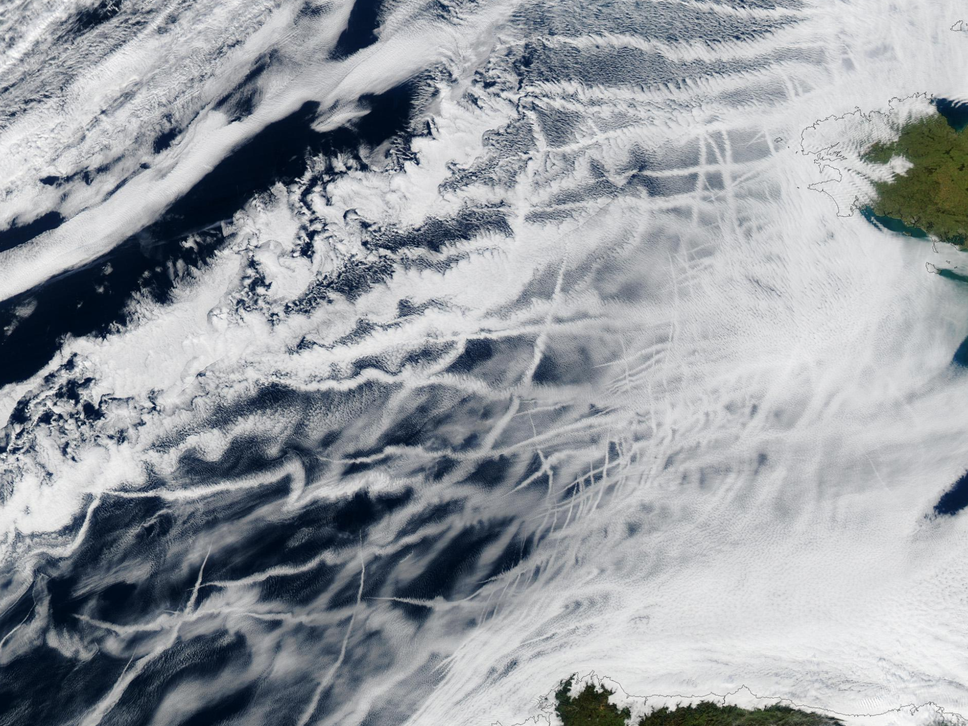
code:

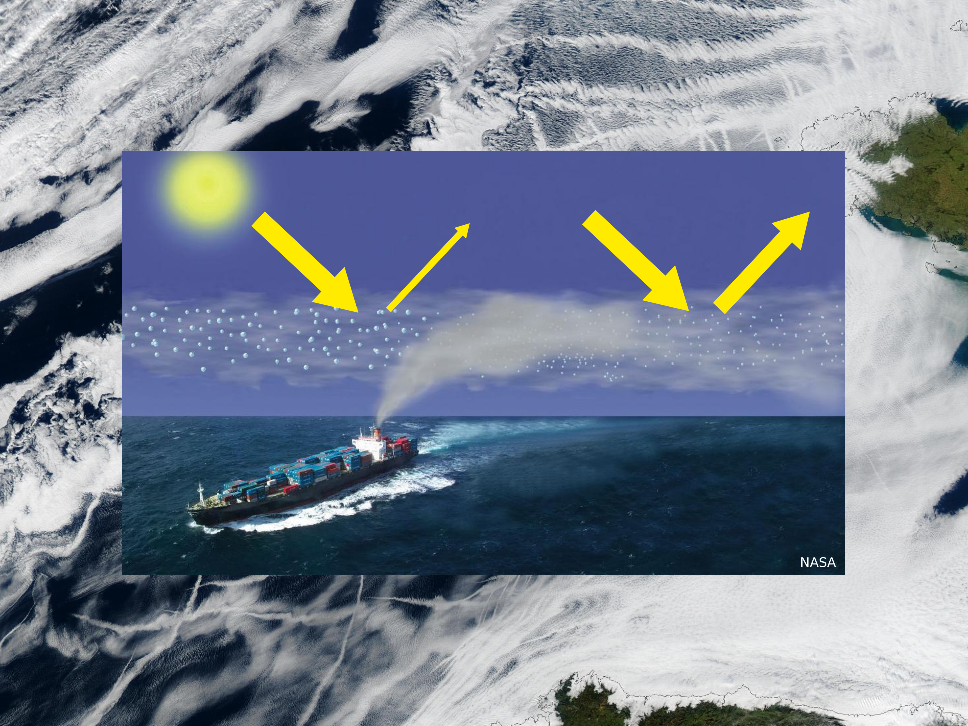
- **libcloudph++**: `github.com/slayoo/libcloudphxx`
- **libmpdata++**: `github.com/slayoo/libmpdataxx`
- **icicle**: `github.com/slayoo/icicle`

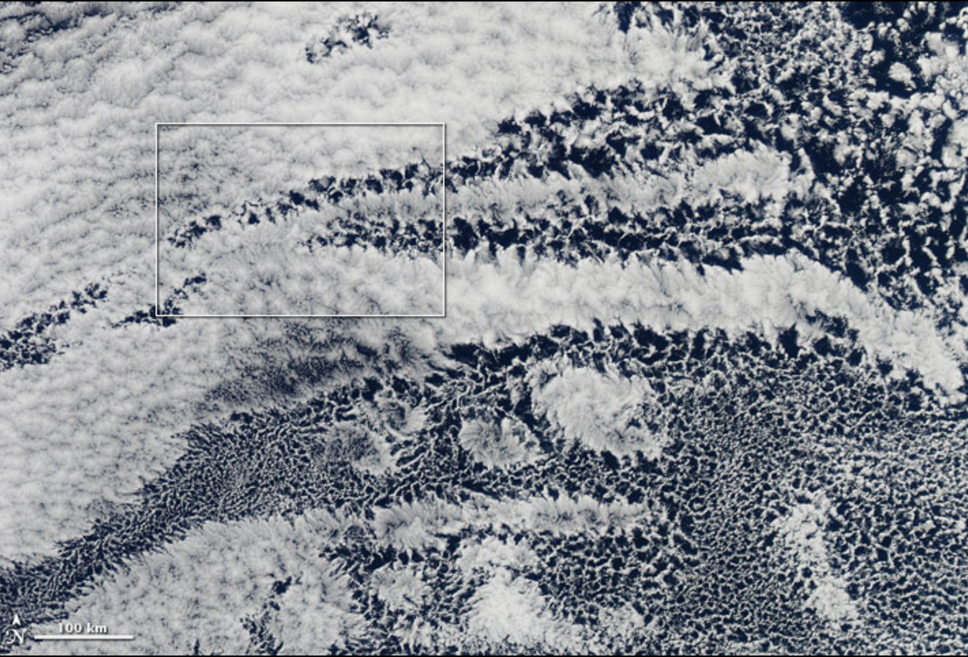
papers:

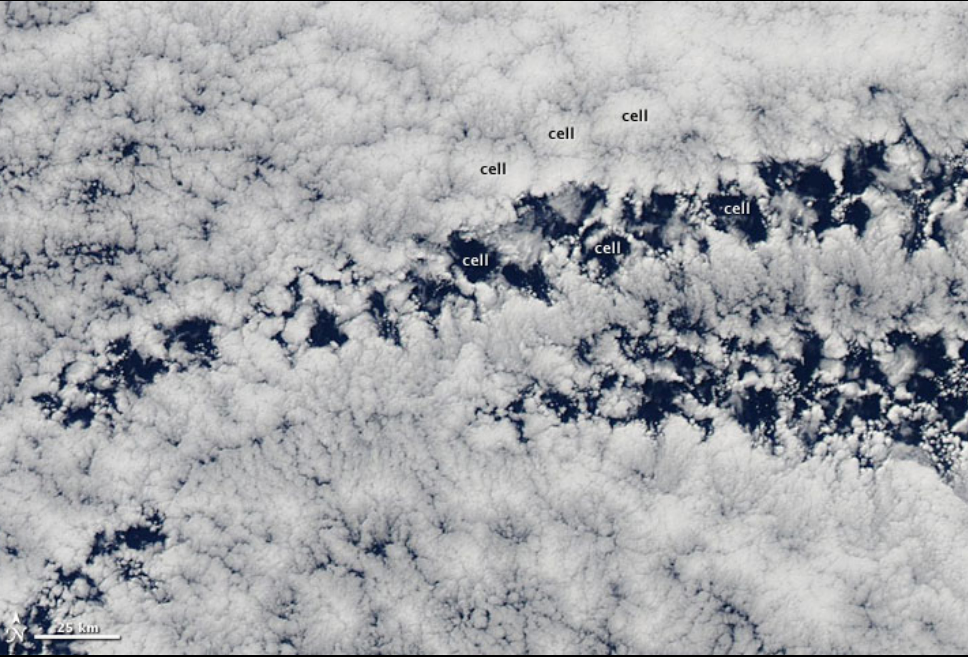
- 0D: **Arabas & Pawlowska 2011** doi:10.5194/gmd-4-15-2011
- 3D: **Arabas & Shima 2013** doi:10.1175/JAS-D-12-0295.1
- 2D: **Arabas, Jaruga et al. 2013** arXiv:1310.1905

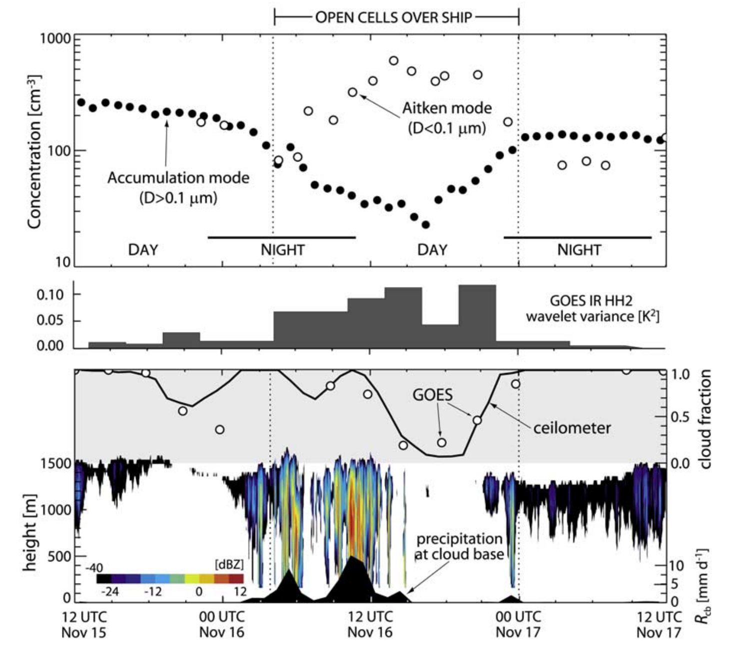
Thank you for your attention




NASA


N
100 km


cell
cell
cell
cell
cell
cell
25 km



Wood et al. doi:10.1029/2007JD009371

Morin et al. 2012 doi:10.1126/science.1218263

„the inability to reproduce many published computational results or to perform credible peer review in the absence of program source code has contributed to a perceived "credibility crisis" for research computation"

Ince et al. 2012 doi:10.1038/nature10836

„anything less than the release of source programs is intolerable for results that depend on computation"

GMD 6. Editorial 2013 doi:10.5194/gmd-6-1233-2013

[all papers] "must be accompanied by the code, or means of accessing the code, for the purpose of peer-review", [while the editors] "strongly encourage referees to compile the code, and run test cases supplied by the authors" arxiv.org/abs/1301.1334

[1301.1334] Object-oriented impl...

Cornell University Library

arXiv.org > physics > arXiv:1301.1334

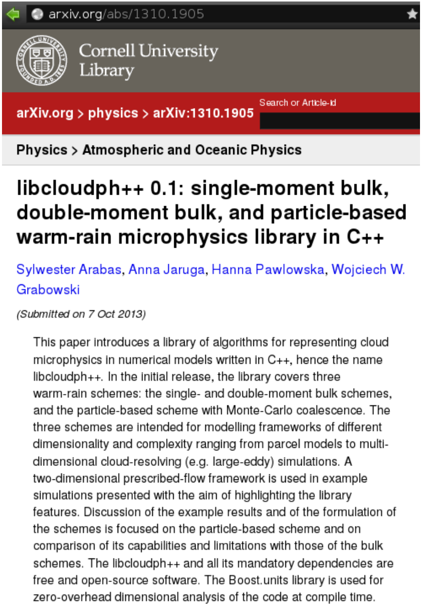
Physics > Computational Physics

Object-oriented implementations of the MPDATA advection equation solver in C++, Python and Fortran

Sylwester Arabas, Dorota Jarecka, Anna Jaruga, Maciej Fijałkowski

(Submitted on 7 Jan 2013 (v1), last revised 19 Mar 2013 (this version, v2))

libcloudph++ API (part of the paper)



arxiv.org/abs/1310.1905

Cornell University Library

arXiv.org > physics > arXiv:1310.1905

Search or Article-id

Physics > Atmospheric and Oceanic Physics

libcloudph++ 0.1: single-moment bulk, double-moment bulk, and particle-based warm-rain microphysics library in C++

Sylwester Arabas, Anna Jaruga, Hanna Pawlowska, Wojciech W. Grabowski

(Submitted on 7 Oct 2013)

This paper introduces a library of algorithms for representing cloud microphysics in numerical models written in C++, hence the name libcloudph++. In the initial release, the library covers three warm-rain schemes: the single- and double-moment bulk schemes, and the particle-based scheme with Monte-Carlo coalescence. The three schemes are intended for modelling frameworks of different dimensionality and complexity ranging from parcel models to multi-dimensional cloud-resolving (e.g. large-eddy) simulations. A two-dimensional prescribed-flow framework is used in example simulations presented with the aim of highlighting the library features. Discussion of the example results and of the formulation of the schemes is focused on the particle-based scheme and on comparison of its capabilities and limitations with those of the bulk schemes. The libcloudph++ and all its mandatory dependencies are free and open-source software. The Boost.units library is used for zero-overhead dimensional analysis of the code at compile time.

libcloudph++ API (part of the paper)

arxiv.org/abs/1310.1905

Cornell University Library

arXiv.org > physics > arXiv:1310.1905

Search or Article-Id

Physics > Atmospheric and Oceanic Physics

libcloudph++ 0.1: single-moment bulk, double-moment bulk, and particle-based warm-rain microphysics library in C++

Sylwester Arabas, Anna Jaruga, Hanna Pawlowska, Wojciech W. Grabowski

(Submitted on 7 Oct 2013)

This paper introduces a library of algorithms for representing cloud microphysics in numerical models written in C++, hence the name libcloudph++. In the initial release, the library covers three warm-rain schemes: the single- and double-moment bulk schemes, and the particle-based scheme with Monte-Carlo coalescence. The three schemes are intended for modelling frameworks of different dimensionality and complexity ranging from parcel models to multi-dimensional cloud-resolving (e.g. large-eddy) simulations. A two-dimensional prescribed-flow framework is used in example simulations presented with the aim of highlighting the library features. Discussion of the example results and of the formulation of the schemes is focused on the particle-based scheme and on comparison of its capabilities and limitations with those of the bulk schemes. The libcloudph++ and all its mandatory dependencies are free and open-source software. The Boost.units library is used for zero-overhead dimensional analysis of the code at compile time.

tion, and are grouped into a structure named **lgrngn::opts_init_t** (Listing 5.2). The initial

```
template<typename real_t>
struct opts_init_t
{
  // initial dry sizes of aerosol
  typedef boost::ptr_unordered_map<
    real_t,                   // kappa
    unary_function<real_t> // n(ln(rd)) @ STP
  > dry_distros_t;
  dry_distros_t dry_distros;

  // Eulerian component parameters
  int nx, ny, nz;
  real_t dx, dy, dz, dt;

  // mean no. of super-droplets per cell
  real_t sd_conc_mean;

  // coalescence Kernel type
  kernel_t kernel;

  // ctor with defaults (C++03 compliant) ...
```

Listing 5.2: **lgrngn::opts_init_t** structure definition

dry size spectrum of aerosol is represented with a map associating values of the solubility parameter κ with pointers to functors returning con-

Questions and comments sent by Graham Feingold

Some broad questions and some that might be worth clarifying in unpublished work:

1) Is the particle-based method (Arabas and Shima 2012) moment conserving? If not, how does moment conservation change with the number of superdroplets
2) How would you design a more rigorous comparison of model output with observations?
3) Why C++ when modern fortran compilers are able to use GPUs? Is it mostly the stability of the code (maintainability, compatibility)? C++ code has a reputation of requiring an extremely careful programming style. Might this outweigh the advantages when the new modeling framework proposed here becomes more of a community model?
4) In many aspects of cloud modeling, we lack a basic understanding of the processes themselves (e.g., collection kernels). How do you view the balance in effort expended on modeling methods as opposed to laboratory and theoretical descriptions of the physics?
5) Have you used the particle-based approach (Appendices A.2 and A.3) to investigate where the raindrop embryos first form?
6) How is supersaturation calculated in the particle-based method (Appendices A.2 and A.3)? Thi was not clear from these papers. Is it the semi-analytical method of Clark (1973)?
7) Where do you view the bin microphysical schemes in terms of their future application in the proposed modeling framework?

Q: Is the particle-based method **moment conserving**? If not, how does moment conservation change with the number of superdroplets?

- initialisation: no (the more super droplets, the better)

[a] e.g.: Tzivion, Reisin, Levin 1999, JCP

[b] e.g.: Alfonso, Raga & Baumgardner 2008, 2010, 2013, ACP

Q: Is the particle-based method **moment conserving**? If not, how does moment conservation change with the number of superdroplets?

- initialisation: no (the more super droplets, the better)
- advection: all moments conserved for both dry and wet spectra

[a] e.g.: Tzivion, Reisin, Levin 1999, JCP
[b] e.g.: Alfonso, Raga & Baumgardner 2008, 2010, 2013, ACP

Q: Is the particle-based method **moment conserving**? If not, how does moment conservation change with the number of superdroplets?

- initialisation: no (the more super droplets, the better)
- advection: all moments conserved for both dry and wet spectra
- sedimentation: no (as in reality)

[a] e.g.: Tzivion, Reisin, Levin 1999, JCP
[b] e.g.: Alfonso, Raga & Baumgardner 2008, 2010, 2013, ACP

Q: Is the particle-based method **moment conserving**? If not, how does moment conservation change with the number of superdroplets?

- initialisation: no (the more super droplets, the better)
- advection: all moments conserved for both dry and wet spectra
- sedimentation: no (as in reality)
- condensation: particle number conserved (0^{th} moment)

[a] e.g.: Tzivion, Reisin, Levin 1999, JCP
[b] e.g.: Alfonso, Raga & Baumgardner 2008, 2010, 2013, ACP

Q: Is the particle-based method **moment conserving**? If not, how does moment conservation change with the number of superdroplets?

- initialisation: no (the more super droplets, the better)
- advection: all moments conserved for both dry and wet spectra
- sedimentation: no (as in reality)
- condensation: particle number conserved (0^{th} moment)
- collisions: mass conserved (3^{rd} moment of dry and wet spectra)
  note: this may not always be the case for Smoluchowski coagulation equation-based methods either due to discretisation issues[a] or due to gelation[b]

[a] e.g.: Tzivion, Reisin, Levin 1999, JCP
[b] e.g.: Alfonso, Raga & Baumgardner 2008, 2010, 2013, ACP

Q: How would you design a more rigorous **comparison of model output with observations**?

- a wider synergy among analyses of macro- & micro-physics

Q: How would you design a more rigorous **comparison of model output with observations**?

- a wider synergy among analyses of macro- & micro-physics
- **more error bars**:

Q: How would you design a more rigorous **comparison of model output with observations**?

- a wider synergy among analyses of macro- & micro-physics
- **more error bars**:
  - **simulations**: more realisations (different grids, timesteps, random seeds, ensembles of initial parameters, ensembles of tuning parameters, different parameterisations)

Q: How would you design a more rigorous **comparison of model output with observations**?

- a wider synergy among analyses of macro- & micro-physics
- **more error bars**:
  - **simulations**: more realisations (different grids, timesteps, random seeds, ensembles of initial parameters, ensembles of tuning parameters, different parameterisations)
  - **observations**: propagation of instrumental error throughout the data analysis procedures

Q: **Why C++** when modern fortran compilers are able to use GPUs? Is it mostly the stability of the code (maintainability, compatibility)? C++ has a reputation of requiring an extremely careful programming style. Might this outweigh the advantages when the new modeling framework proposed here becomes more of a community model?

Q: **Why C++** when modern fortran compilers are able to use GPUs?
Is it mostly the stability of the code (maintainability, compatibility)?
C++ has a reputation of requiring an extremely careful programming style.
Might this outweigh the advantages when the new modeling framework proposed here becomes more of a community model?

- **re: C++ requiring extremely careful programming style:**

  It's an exciting challenge!
  It pays off!

Q: **Why C++** when modern fortran compilers are able to use GPUs?
Is it mostly the stability of the code (maintainability, compatibility)?
C++ has a reputation of requiring an extremely careful programming style.
Might this outweigh the advantages when the new modeling framework proposed here becomes more of a community model?

- **re: C++ requiring extremely careful programming style:**

  It's an exciting challenge!
  It pays off!

  - hardly any University teaches Fortran
    (not mentioning software engineering or OOP with Fortran)
    C++ $\rightsquigarrow$ easier access to trained personnel

Q: **Why C++** when modern fortran compilers are able to use GPUs?
Is it mostly the stability of the code (maintainability, compatibility)?
C++ has a reputation of requiring an extremely careful programming style.
Might this outweigh the advantages when the new modeling framework proposed here becomes more of a community model?

- **re: C++ requiring extremely careful programming style:**

  It's an exciting challenge!
  It pays off!

  - hardly any University teaches Fortran
    (not mentioning software engineering or OOP with Fortran)
    C++ $\rightsquigarrow$ easier access to trained personnel
  - Fortran is domain-specific language $\rightsquigarrow$ no cross-domain benefits
    (C++: gaming, banking, defense, CAD/CAM, telecom, ...)
    C++ $\rightsquigarrow$ easier access to reusable code, information resources

Q: **Why C++** when modern fortran compilers are able to use GPUs?
Is it mostly the stability of the code (maintainability, compatibility)?
C++ has a reputation of requiring an extremely careful programming style.
Might this outweigh the advantages when the new modeling framework
proposed here becomes more of a community model?

- **re: C++ vs. Fortran in context of GPU programming:**

Q: **Why C++** when modern fortran compilers are able to use GPUs?
Is it mostly the stability of the code (maintainability, compatibility)?
C++ has a reputation of requiring an extremely careful programming style.
Might this outweigh the advantages when the new modeling framework proposed here becomes more of a community model?

- **re: C++ vs. Fortran in context of GPU programming:**
  - OpenCL (multi-vendor, open standard, based on C11/C++11)
    "FortranCL is an independent effort and is not endorsed in any way by the Khronos group or other institution related to OpenCL or Fortran."

Q: **Why C++** when modern fortran compilers are able to use GPUs? Is it mostly the stability of the code (maintainability, compatibility)? C++ has a reputation of requiring an extremely careful programming style. Might this outweigh the advantages when the new modeling framework proposed here becomes more of a community model?

- **re: C++ vs. Fortran in context of GPU programming:**
  - OpenCL (multi-vendor, open standard, based on C11/C++11)
    "FortranCL is an independent effort and is not endorsed in any way by the Khronos group or other institution related to OpenCL or Fortran."
  - CUDA (vendor-specific, neither open nor libre)
    C/C++ CUDA compiler is free
    Fortran CUDA compiler can be purchased from Portland group only

Q: **Why C++** when modern fortran compilers are able to use GPUs? Is it mostly the stability of the code (maintainability, compatibility)? C++ has a reputation of requiring an extremely careful programming style. Might this outweigh the advantages when the new modeling framework proposed here becomes more of a community model?

- **re: C++ vs. Fortran in context of GPU programming:**
  - OpenCL (multi-vendor, open standard, based on C11/C++11)
    "FortranCL is an independent effort and is not endorsed in any way by the Khronos group or other institution related to OpenCL or Fortran."
  - CUDA (vendor-specific, neither open nor libre)
    C/C++ CUDA compiler is free
    Fortran CUDA compiler can be purchased from Portland group only
  - High-level libs (where the user don't have to know a single bit of OpenCL/CUDA, and that allow to run the programs with no GPU):
    Boost.compute, VexCL, ViennaCL, nVidia's Thrust, AMD's Bolt, Microsoft's AMP – all in C++

Q: **Why C++** when modern fortran compilers are able to use GPUs? Is it mostly the stability of the code (maintainability, compatibility)? C++ has a reputation of requiring an extremely careful programming style. Might this outweigh the advantages when the new modeling framework proposed here becomes more of a community model?

- **re: Why C++ when modern Fortran . . . (more technical)**

Q: **Why C++** when modern fortran compilers are able to use GPUs? Is it mostly the stability of the code (maintainability, compatibility)? C++ has a reputation of requiring an extremely careful programming style. Might this outweigh the advantages when the new modeling framework proposed here becomes more of a community model?

- **re: Why C++ when modern Fortran . . . (more technical)**
  - Fortran lacks error handling facility (exceptions in C++)
    $\rightsquigarrow$an issue when using multiple libraries

Q: **Why C++** when modern fortran compilers are able to use GPUs? Is it mostly the stability of the code (maintainability, compatibility)? C++ has a reputation of requiring an extremely careful programming style. Might this outweigh the advantages when the new modeling framework proposed here becomes more of a community model?

- **re: Why C++ when modern Fortran . . . (more technical)**
  - Fortran lacks error handling facility (exceptions in C++)
    ⇝an issue when using multiple libraries
  - Fortran lacks compile-time programming facility (templates in C++)
    ⇝no zero-runtime-overhead mechanisms, e.g. units checking

Q: In many aspects of cloud modeling, we lack a basic understanding of the processes themselves (e.g., collection kernels). How do you view the balance in effort expended on modelling methods as opposed to laboratory and theoretical descriptions of the physics?

Q: In many aspects of cloud modeling, we lack a basic understanding of the processes themselves (e.g., collection kernels). How do you view the balance in effort expended on modelling methods as opposed to laboratory and theoretical descriptions of the physics?

- **re: balance in effort on modelling, experiment and theory**
  - disproportionate efforts between software and instrument engineering?
    (would aerosol/cloud models pass airworthiness or spaceworthines tests of aerosol/cloud instruments?)

Q: In many aspects of cloud modeling, we lack a basic understanding of the processes themselves (e.g., collection kernels). How do you view the balance in effort expended on modelling methods as opposed to laboratory and theoretical descriptions of the physics?

- **re: balance in effort on modelling, experiment and theory**
  - disproportionate efforts between software and instrument engineering?
    (would aerosol/cloud models pass airworthiness or spaceworthines tests of aerosol/cloud instruments?)
- **a related remark**
  - open data needed as much as open code
    (to foster transfer of knowledge between modellers and observationalists)

Q: Have you used the particle-based approach to investigate where the raindrop embryos first form?

- no, thanks for suggestion

Q: How is supersaturation calculated in the particle-based method? This was not clear from these papers. Is it the semi-analytical method of Clark (1973)?

- explicitly, from dynamical tendencies alone

[a]

Q: How is supersaturation calculated in the particle-based method?
This was not clear from these papers.
Is it the semi-analytical method of Clark (1973)?

- explicitly, from dynamical tendencies alone
- but $\leqslant$ 1s timesteps and marine aerosol were used

[a]

Q: How is supersaturation calculated in the particle-based method?
This was not clear from these papers.
Is it the semi-analytical method of Clark (1973)?

- explicitly, from dynamical tendencies alone
- but $\leqslant$ 1s timesteps and marine aerosol were used
- again: thanks for suggestion, will look into it
  (starting off by implementing the scheme of Thouron et. al. 2013[a])

[a] O. Thouron, J.-L. Brenguier, and F. Burnet: Supersaturation calculation in large eddy simulation models for prediction of the droplet number concentration Geosci. Model Dev., 5, 761-772, 2012

Q: Where do you view the bin microphysical schemes in terms of their future application in the proposed modeling framework

Q: Where do you view the bin microphysical schemes in terms of their future application in the proposed modeling framework

- no experience yet with bin schemes

Q: Where do you view the bin microphysical schemes in terms of their future application in the proposed modeling framework

- no experience yet with bin schemes
- reproduction of previously-published bin results with the Lagrangian scheme ($\rightsquigarrow$ validation of the implementation)

Q: Where do you view the bin microphysical schemes in terms of their future application in the proposed modeling framework

- no experience yet with bin schemes
- reproduction of previously-published bin results with the Lagrangian scheme ($\rightsquigarrow$ validation of the implementation)
- quantification of some of the limitations of the Lagrangian method (e.g. importance of regions that become void of particles)