icicle

http://icicle.igf.fuw.edu.pl/

a new MPDATA-based solver for systems of transport equations with emphasis on cloud modelling

> Sylwester Arabas Anna Jaruga Hanna Pawłowska

University of Warsaw / Faculty of Physics / Institute of Geophysics

8th International Cloud Modelling Workshop, Warsaw, 26th July 2012

icicle

http://icicle.igf.fuw.edu.pl/

Why develop

a new MPDATA-based solver for systems of transport equations with emphasis on cloud modelling 7

> Sylwester Arabas Anna Jaruga Hanna Pawłowska

University of Warsaw / Faculty of Physics / Institute of Geophysics

8th International Cloud Modelling Workshop, Warsaw, 26th July 2012

- systems of advective transport (continuity) equations pop up frequently in geosciences ~> need for accurate & efficient solvers
 - MPDATA (Smolarkiewicz 1983,) is:
 - method missional and sign innersening by datage
 optionally non-oscillatory (flux-corrected trans
 - iterative ~> readily parallelisable
 - traditionally the solver-of-choice at the University of Warsaw¹
 - is free (as in free speech) and open-source
 - has technical documentation (re-usability)
 - uses object cheeted programming (OOP)

- systems of advective transport (continuity) equations pop up frequently in geosciences ~> need for accurate & efficient solvers
- MPDATA (Smolarkiewicz 1983, ...) is:
 - second-order accurate in space and time
 - optionally non-oscillatory (flux-corrected transpo
 - ullet iterative \leadsto readily parallelisable
 - traditionally the solver-of-choice at the University of Warsaw
 - is free (as in free speech) and open-source
 - has technical documentation (re-usability)
 - uses collect chiested programming (OOP)

- systems of advective transport (continuity) equations pop up frequently in geosciences ~> need for accurate & efficient solvers
- MPDATA (Smolarkiewicz 1983, ...) is:
 - second-order accurate in space and time
 - multidimensional and sign-preserving by design
 - \cdot iterative \rightsquigarrow readily parallelisable
 - traditionally the solver-of-choice at the University of Warsaw
 - is free (as in free speech) and open-source
 - has technical (re-usability)
 uses programming (OOP)

- systems of advective transport (continuity) equations pop up frequently in geosciences → need for accurate & efficient solvers
- MPDATA (Smolarkiewicz 1983, ...) is:
 - second-order accurate in space and time
 - multidimensional and sign-preserving by design
 - optionally non-oscillatory (flux-corrected transport)
 - traditionally the solver-of-choice at the University of Wa
 - is free (as in free speech) and open-source
 - has technical (re-usability)
 uses programming (OOP)

- systems of advective transport (continuity) equations pop up frequently in geosciences → need for accurate & efficient solvers
- MPDATA (Smolarkiewicz 1983, ...) is:
 - second-order accurate in space and time
 - multidimensional and sign-preserving by design
 - optionally non-oscillatory (flux-corrected transport)
 - iterative → readily parallelisable

- is free (as in free speech) and open-source
- has technical documentation (re-usability)
- uses object of entropy programming (OOP)

- systems of advective transport (continuity) equations pop up frequently in geosciences ~> need for accurate & efficient solvers
- MPDATA (Smolarkiewicz 1983, ...) is:
 - second-order accurate in space and time
 - multidimensional and sign-preserving by design
 - optionally non-oscillatory (flux-corrected transport)
 - iterative → readily parallelisable
 - traditionally the solver-of-choice at the University of Warsaw :)
 - is free (as in free speech) and open-source
 - has technical (re-usability)
 uses programming (OOP)

- systems of advective transport (continuity) equations pop up frequently in geosciences → need for accurate & efficient solvers
- MPDATA (Smolarkiewicz 1983, ...) is:
 - second-order accurate in space and time
 - multidimensional and sign-preserving by design
 - optionally non-oscillatory (flux-corrected transport)
 - iterative \rightsquigarrow readily parallelisable
 - traditionally the solver-of-choice at the University of Warsaw :)
- there's no implementation of an MPDATA-based solver that:

- systems of advective transport (continuity) equations pop up frequently in geosciences ~> need for accurate & efficient solvers
- MPDATA (Smolarkiewicz 1983, ...) is:
 - second-order accurate in space and time
 - multidimensional and sign-preserving by design
 - optionally non-oscillatory (flux-corrected transport)
 - iterative \rightsquigarrow readily parallelisable
 - traditionally the solver-of-choice at the University of Warsaw :)
- there's no implementation of an MPDATA-based solver that:
 - is free (as in free speech) and open-source

- systems of advective transport (continuity) equations pop up frequently in geosciences ~> need for accurate & efficient solvers
- MPDATA (Smolarkiewicz 1983, ...) is:
 - second-order accurate in space and time
 - multidimensional and sign-preserving by design
 - optionally non-oscillatory (flux-corrected transport)
 - iterative \rightsquigarrow readily parallelisable
 - traditionally the solver-of-choice at the University of Warsaw :)
- there's no implementation of an MPDATA-based solver that:
 - is free (as in free speech) and open-source
 - has technical documentation (re-usability)

- systems of advective transport (continuity) equations pop up frequently in geosciences ~> need for accurate & efficient solvers
- MPDATA (Smolarkiewicz 1983, ...) is:
 - second-order accurate in space and time
 - multidimensional and sign-preserving by design
 - optionally non-oscillatory (flux-corrected transport)
 - iterative \rightsquigarrow readily parallelisable
 - traditionally the solver-of-choice at the University of Warsaw :)
- there's no implementation of an MPDATA-based solver that:
 - is free (as in free speech) and open-source
 - has technical documentation (re-usability)
 - uses object-oriented programming (OOP)

OOP vs. MPDATA

```
content c
```

```
2D donor-cell in C++/Blitz++
```

```
const vec_t<arr_t> kpsi, const int n,
    const vec_t<arr_t> kpsi, const int n,
    const vec_t<arr_t> kC,
    const rng_t ki, const rng_t kj
) {
    psi[n+1](i,j) = psi[n](i,j)
```

```
- donorcell_1D<pi_ij>(psi[n], C[0], i, j)
```

```
- donorcell_1D<pi_ji>(psi[n], C[1], j, i);
```

 $\psi_{i,j}^{[n+1]} = \psi_{i,j}^{[n]} - \sum_{d=0}^{N-1} \left(F\left[\psi_{\pi_{i,j}^d}^{[n]}, \psi_{\pi_{i+1,j}^d}^{[n]}, C_{\pi_{i+1/2,j}^d}^{[d]}\right] \right)$ $-F\left[\psi_{\pi_{i-1}^d}^{\left[n\right]},\psi_{\pi_{i}^d}^{\left[n\right]},C_{\pi_{i-1}^d}^{\left[d\right]}
ight]
ight)$

Arakawa-C grid with operator overloading

t hlf_t {} h;

nline rng_t operator+(const rng_t &i, const hlf_t & {

_____i + 1;

infine rng_t operator-(
 const rng_t &i, const hlf_t &
) {

return i;

→ BIItz++ o://sf.net/proiects/bli

OOP vs. MPDATA

 $\psi_{i,j}^{[n+1]} = \psi_{i,j}^{[n]} - \sum_{d=0}^{N-1} \left(F\left[\psi_{\pi_{i,j}^d}^{[n]}, \psi_{\pi_{i+1,j}^d}^{[n]}, C_{\pi_{i+1/2,j}^d}^{[d]}\right] \right)$

1D donor-cell in C++/Blitz++

```
template <class pi>
inline auto donorcell_1D(
    const arr_t &psi,
    const arr_t &C,
    const rng_t &i, const rng_t &j
) return_macro(
    F(psi(pi(i, j)), psi(pi(i+1,j)), C(pi(i+h,j)))
    F(psi(pi(i-1,j)), psi(pi(i, j)), C(pi(i-h,j)))
```

```
PD dopor cell in C \pm /Blitz \pm \pm
```

```
inline void donorcell_2D(
    const vec_t<arr_t> &psi, const int n,
    const vec_t<arr_t> &C,
    const rng_t &i, const rng_t &j
    {
    psi[n+1](i,j) = psi[n](i,j)
```

```
- donorcell_1D<pi_ij>(psi[n], C[0], i, j)
```

```
- donorcell_1D<pi_ji>(psi[n], C[1], j, i);
```

 $-F\left[\psi_{\pi_{i-1}^{d}}^{[n]},\psi_{\pi_{i-i}^{d}}^{[n]},C_{\pi_{i-1/2,i}^{d}}^{[d]}
ight]$

Arakawa-C grid with operator overloading

hlf_t {} h;

lline rng_t operator+(const rng_t &i, const hlf_t & {

```
return i + 1;
```

inline rng_t operator-(
 const rng_t &i, const hlf_t &
) {

return i;

→ Blitz++)://sf.net/projects/bl

OOP vs. MPDATA

psi[n+1](i,j) = psi[n](i,j)

- donorcell_1D<pi_ij>(psi[n], C[0], i, j)
- donorcell_1D<pi_ji>(psi[n], C[1], j, i);



→ Blitz++ ://sf.net/projects/blit



http://sf.net/projects/blitz/



→→ Blitz++ http://sf.net/projects/blitz/

C++ vs. FORTRAN vs. Python: performance



donor-cell times from Jarecka et al. 2012 EGU poster MPDATA paper in preparation

C++ can check units for you (at no runtime cost!)

κ -Köhler parameterisation

```
/// @brief activity of water in solution
/// (eqs. 1,6) in @copydetails Petters_and_Kreidenweis_2007
tamplate <typename real_t>
quantity<si::dimensionless, real_t> a_w(
   quantity<si::volume, real_t> rw3,
   quantity<si::volume, real_t> rd3,
   quantity<si::dimensionless, real_t> kappa
)
{
   return (rw3 - rd3) / (rw3 - rd3 * (real_t(1) - kappa));
};
```

→ Boost.units http://boost.org/doc/libs/release/libs/units/

C++ can check units for you (at no runtime cost!)

κ -Köhler parameterisation

```
/// @brief activity of water in solution
/// (eqs. 1,6) in @copydetails Petters_and_Kreidenweis_2007
template <typename real_t>
quantity<si::dimensionless, real_t> a_w(
   quantity<si::volume, real_t> rw3,
   quantity<si::volume, real_t> rd3,
   quantity<si::dimensionless, real_t> kappa
)
{
   return (rw3 - rd3) / (rw3 - rd3 * (real_t(1) - kappa));
};
```

→ Boost.units http://boost.org/doc/libs/release/libs/units/

the aims of icicle

"[Object oriented programming] has become recognised as the almost unique successful paradigm in the successful paradigm

human-readable + open-source ~> auditable (code review!)

reusable \sim coding time savings in the long run

maintainable --- less bug-prone, easier to co-operate on modular --- full separation of numerics/physics/concurrency/ic

optimisable (by the compiler/library author) ~-> potentially faster

the aims of icicle

"[Object oriented programming] has become recognised as the almost unique successful paradigm for parameter unique argument

- human-readable + open-source ~> auditable (code review!)
- shorter ~> less bug-prone, easier to debug

shareable \rightsquigarrow common libraries instead of copy-paste! maintainable \rightsquigarrow less bug-prone, easier to co-operate on modular \rightsquigarrow full separation of numerics/physics/concurrency/io optimisable (by the compiler/library author) \rightsquigarrow potentially faster

"[Object oriented programming] has become recognised as the almost unique successful paradigm for which any statement of the statement of the

- human-readable + open-source ~> auditable (code review!)
- shorter ~> less bug-prone, easier to debug
- reusable ~> coding time savings in the long run

shareable ---- common libraries instead of copy-past

maintainable ---- less bug-prone, easier to co-operate on modular ---- full separation of numerics/physics/concurrency/io optimisable (by the compiler/library author) ---- potentially faster

"[Object oriented programming] has become recognised as the almost unique successful paradigm in the successful paradigm

- human-readable + open-source ~> auditable (code review!)
- shorter ~> less bug-prone, easier to debug
- reusable → coding time savings in the long run
- shareable → common libraries instead of copy-paste!

maintainable \rightsquigarrow less bug-prone, easier to co-operate on modular \rightsquigarrow full separation of numerics/physics/concurrency/io optimisable (by the compiler/library author) \rightsquigarrow potentially faster

"[Object oriented programming] has become recognised as the almost unique successful paradigm in the second program of the second paradigm

- human-readable + open-source ~> auditable (code review!)
- shorter ~> less bug-prone, easier to debug
- reusable → coding time savings in the long run
- shareable → common libraries instead of copy-paste!

"[Object oriented programming] has become recognised as the almost unique successful paradigm in the almost unique successful paradigm in the successful par

- human-readable + open-source ~> auditable (code review!)
- shorter → less bug-prone, easier to debug
- reusable → coding time savings in the long run
- shareable ~> common libraries instead of copy-paste!
- maintainable ~> less bug-prone, easier to co-operate on
- modular ~> full separation of numerics/physics/concurrency/io

ullet optimisable (by the compiler/library author) \leadsto potentially faste

"[Object oriented programming] has become recognised as the almost unique successful paradigm in unique successful paradigm

- human-readable + open-source ~> auditable (code review!)
- shorter ~> less bug-prone, easier to debug
- reusable → coding time savings in the long run
- shareable ~> common libraries instead of copy-paste!
- maintainable ~> less bug-prone, easier to co-operate on
- modular ~> full separation of numerics/physics/concurrency/io
- optimisable (by the compiler/library author) → potentially faster

[Object oriented programming] has become recognised

- human-readable + open-source ~> auditable (code review!)
- **shorter** \rightsquigarrow less bug-prone, easier to debug
- reusable ~> coding time savings in the long run
- shareable ~> common libraries instead of copy-paste!
- maintainable → less bug-prone, easier to co-operate on
- modular ~> full separation of numerics/physics/concurrency/io
- optimisable (by the compiler/library author) \rightsquigarrow potentially faster

the aims of icicle

"[Object oriented programming] has become recognised as the almost unique successful paradigm

- human-readable + open-source ~> auditable (code review!)
- shorter → less bug-prone, easier to debug
- reusable ~> coding time savings in the long run
- **shareable** \rightsquigarrow common libraries instead of copy-paste!
- maintainable ~> less bug-prone, easier to co-operate on
- modular ~> full separation of numerics/physics/concurrency/io
- optimisable (by the compiler/library author) ~> potentially faster

the aims of icicle

"[Object oriented programming] has become recognised as the almost unique successful paradigm for creating complex software" NUMERICAL RECIPES

THIRD EDITION

William H. Press Saul A. Teukolsky William T. Vetterling Brian P. Flannery

NR: The Art of Scientific Computing (3rd ed., Press et al. 2007)

icicle

http://icicle.igf.fuw.edu.pl/

Why develop

a new MPDATA-based solver for systems of transport equations with emphasis on cloud modelling 7

> Sylwester Arabas Anna Jaruga Hanna Pawłowska

University of Warsaw / Faculty of Physics / Institute of Geophysics

8th International Cloud Modelling Workshop, Warsaw, 26th July 2012

icicle

http://icicle.igf.fuw.edu.pl/

Why develop

a new MPDATA-based solver for systems of transport equations with emphasis on cloud modelling and what one may already do with it?

> Sylwester Arabas Anna Jaruga Hanna Pawłowska

University of Warsaw / Faculty of Physics / Institute of Geophysics

8th International Cloud Modelling Workshop, Warsaw, 26th July 2012

icicle: currently available components

- advection scheme:
 - MPDATA
 - any number of iterations
 - 3rd order accuracy option
 - FCT (aka non-oscillatory) option

command-line string unambiguously defines simulation (in output file

- variable-sign field option
- donor-cell, leapfrog, …

icicle: currently available components

- advection scheme:
 - MPDATA
 - any number of iterations
 - 3rd order accuracy option
 - FCT (aka non-oscillatory) option

command-line string unambiguously defines simulation (in output file

- variable-sign field option
- donor-cell, leapfrog, ...
- equation sets:

icicle: currently available components

- advection scheme:
 - MPDATA
 - any number of iterations
 - 3rd order accuracy option
 - FCT (aka non-oscillatory) option
 - variable-sign field option
 - donor-cell, leapfrog, …
- equation sets:
 - scalar advection (rhs=0 \rightsquigarrow tests)

2D moist kinematic (this workshop's "case

- bulk (Kessler) μ -physics
- particle-based (super-droplet) μ-physics

all selectable at runtime via command-line options and the command-line string unambiguously defines simulation (in output file)
advection scheme:

• MP 1D scalar advection examples



- advection scheme:
 - MPDATA
 - any number of iterations
 - 3rd order accuracy option
 - FCT (aka non-oscillatory) option
 - variable-sign field option
 - donor-cell, leapfrog, …
- equation sets:
 - scalar advection (rhs=0 \rightsquigarrow tests)

2D moist kinematic (this workshop's "case

- bulk (Kessler) μ -physics
- particle-based (super-droplet) μ-physics

all selectable at runtime via command-line options and the command-line string unambiguously defines simulation (in output file)

- advection scheme:
 - MPDATA
 - any number of iterations
 - 3rd order accuracy option
 - FCT (aka non-oscillatory) option
 - variable-sign field option
 - donor-cell, leapfrog, …
- equation sets:
 - scalar advection (rhs=0 \rightsquigarrow tests)
 - 1D/2D shallow water

2D moist kinematic (this workshop's "case

- bulk (Kessler) µ-physics
- particle-based (super-droplet) µ-physics

all selectable at runtime via command-line options command-line string unambiguously defines simulation (in output file)

advection scheme:

• MP 1D shallow water example



- advection scheme:
 - MPDATA
 - any number of iterations
 - 3rd order accuracy option
 - FCT (aka non-oscillatory) option
 - variable-sign field option
 - donor-cell, leapfrog, …
- equation sets:
 - scalar advection (rhs=0 \rightsquigarrow tests)
 - 1D/2D shallow water

2D moist kinematic (this workshop's "case

- bulk (Kessler) µ-physics
- particle-based (super-droplet) µ-physics

all selectable at runtime via command-line options command-line string unambiguously defines simulation (in output file)

- advection scheme:
 - MPDATA
 - any number of iterations
 - 3rd order accuracy option
 - FCT (aka non-oscillatory) option

command-line string unambiguously defines simulation (in output file

- variable-sign field option
- donor-cell, leapfrog, …
- equation sets:
 - scalar advection (rhs=0 \rightsquigarrow tests)
 - 1D/2D shallow water
 - 2D/3D isentropic

advection scheme:

• MP 2D isentropic example



t = 00132 s

- advection scheme:
 - MPDATA
 - any number of iterations
 - 3rd order accuracy option
 - FCT (aka non-oscillatory) option

command-line string unambiguously defines simulation (in output file

- variable-sign field option
- donor-cell, leapfrog, …
- equation sets:
 - scalar advection (rhs=0 \rightsquigarrow tests)
 - 1D/2D shallow water
 - 2D/3D isentropic

- advection scheme:
 - MPDATA
 - any number of iterations
 - 3rd order accuracy option
 - FCT (aka non-oscillatory) option
 - variable-sign field option
 - donor-cell, leapfrog, …
- equation sets:
 - scalar advection (rhs=0 \rightsquigarrow tests)
 - 1D/2D shallow water
 - 2D/3D isentropic
 - 2D moist kinematic (this workshop's "case 1")

command-line string unambiguously defines simulation (in output

- advection scheme:
 - MPDATA
 - any number of iterations
 - 3rd order accuracy option
 - FCT (aka non-oscillatory) option
 - variable-sign field option
 - donor-cell, leapfrog, …
- equation sets:
 - scalar advection (rhs=0 \rightsquigarrow tests)
 - 1D/2D shallow water
 - 2D/3D isentropic
 - 2D moist kinematic (this workshop's "case 1")

command-line string unambiguously defines simulation (in output

• bulk (Kessler) μ -physics

0

0 0.2 0.4 0.6 0.8 1 1.2 1.4

X [km]

• advection scheme:

• MP 2D moist kinematic example (bulk)



0 0.2 0.4 0.6 0.8 1 1.2 1.4 X [km]

• advection scheme:







• advection scheme:

• MP 2D moist kinematic example (bulk)





• don • equation • scal • 1D/ • 2D/ • 2D/

• advection scheme:

• MP 2D moist kinematic example (bulk)





don
equation
scal
1D/
2D/
2D

• advection scheme:

.





• advection scheme:

2D

.





• advection scheme:

• MP 2D moist kinematic example (bulk)





• don • equation • scal • 1D/ • 2D/ • 2D

• advection scheme:

.

2D moist kinematic example (bulk) MP





• advection scheme:

2D

.





• advection scheme:

1.4

1.2

1

0.8

0.6

0.4

0.2

0

≺ [km]





• advection scheme:

1.4

1.2

0.8

0.6

0.4

0.2

0

≺ [km]





- advection scheme:
 - MPDATA
 - any number of iterations
 - 3rd order accuracy option
 - FCT (aka non-oscillatory) option
 - variable-sign field option
 - donor-cell, leapfrog, …
- equation sets:
 - scalar advection (rhs=0 \rightsquigarrow tests)
 - 1D/2D shallow water
 - 2D/3D isentropic
 - 2D moist kinematic (this workshop's "case 1")

command-line string unambiguously defines simulation (in output

• bulk (Kessler) μ -physics

- advection scheme:
 - MPDATA
 - any number of iterations
 - 3rd order accuracy option
 - FCT (aka non-oscillatory) option
 - variable-sign field option
 - donor-cell, leapfrog, …
- equation sets:
 - scalar advection (rhs=0 \rightsquigarrow tests)
 - 1D/2D shallow water
 - 2D/3D isentropic
 - 2D moist kinematic (this workshop's "case 1")
 - bulk (Kessler) µ-physics
 - particle-based (super-droplet) μ -physics

command-line string unambiguously defines simulation (in output

• advection scheme:



• advection scheme:

• MP 2D moist kinematic example (super-droplets)



NIW.

• advection scheme:



• advection scheme:



• advection scheme:



• advection scheme:



• advection scheme:



• advection scheme:



• advection scheme:



• advection scheme:



• advection scheme:



• advection scheme:



- advection scheme:
 - MPDATA
 - any number of iterations
 - 3rd order accuracy option
 - FCT (aka non-oscillatory) option
 - variable-sign field option
 - donor-cell, leapfrog, …
- equation sets:
 - scalar advection (rhs=0 \rightsquigarrow tests)
 - 1D/2D shallow water
 - 2D/3D isentropic
 - 2D moist kinematic (this workshop's "case 1")
 - bulk (Kessler) µ-physics
 - particle-based (super-droplet) μ -physics

command-line string unambiguously defines simulation (in output
- advection scheme:
 - MPDATA
 - any number of iterations
 - 3rd order accuracy option
 - FCT (aka non-oscillatory) option
 - variable-sign field option
 - donor-cell, leapfrog, ...
- equation sets:
 - scalar advection (rhs=0 \rightsquigarrow tests)
 - 1D/2D shallow water
 - 2D/3D isentropic
 - 2D moist kinematic (this workshop's "case 1")
 - bulk (Kessler) μ -physics
 - particle-based (super-droplet) μ -physics
- input/output: ASCII, netCDF-4

all selectable at runtime via command-line options command-line string unambiguously defines simulation (in output ~~ easy to call from Python, Octave, GDL, _____, or a web server

- advection scheme:
 - MPDATA
 - any number of iterations
 - 3rd order accuracy option
 - FCT (aka non-oscillatory) option
 - variable-sign field option
 - donor-cell, leapfrog, ...
- equation sets:
 - scalar advection (rhs=0 \rightsquigarrow tests)
 - 1D/2D shallow water
 - 2D/3D isentropic
 - 2D moist kinematic (this workshop's "case 1")
 - bulk (Kessler) µ-physics
 - particle-based (super-droplet) μ -physics

command-line string unambiguously defines simulation (in output

- input/output: ASCII, netCDF-4
- parallelisation: OpenMP, Boost.Threads, Boost.MPI

- advection scheme:
 - MPDATA
 - any number of iterations
 - 3rd order accuracy option
 - FCT (aka non-oscillatory) option
 - variable-sign field option
 - donor-cell, leapfrog, ...
- equation sets:
 - scalar advection (rhs=0 \rightsquigarrow tests)
 - 1D/2D shallow water
 - 2D/3D isentropic
 - 2D moist kinematic (this workshop's "case 1")
 - bulk (Kessler) µ-physics
 - particle-based (super-droplet) μ -physics
- input/output: ASCII, netCDF-4
- parallelisation: OpenMP, Boost.Threads, Boost.MPI

all selectable at runtime via command-line options

- advection scheme:
 - MPDATA
 - any number of iterations
 - 3rd order accuracy option
 - FCT (aka non-oscillatory) option
 - variable-sign field option
 - donor-cell, leapfrog, ...
- equation sets:
 - scalar advection (rhs=0 \rightsquigarrow tests)
 - 1D/2D shallow water
 - 2D/3D isentropic
 - 2D moist kinematic (this workshop's "case 1")
 - bulk (Kessler) µ-physics
 - particle-based (super-droplet) μ -physics
- input/output: ASCII, netCDF-4
- parallelisation: OpenMP, Boost.Threads, Boost.MPI

all selectable at runtime via command-line options

 \rightsquigarrow command-line string unambiguously defines simulation (in output file)

- advection scheme:
 - MPDATA
 - any number of iterations
 - 3rd order accuracy option
 - FCT (aka non-oscillatory) option
 - variable-sign field option
 - donor-cell, leapfrog, …
- equation sets:
 - scalar advection (rhs=0 \rightsquigarrow tests)
 - 1D/2D shallow water
 - 2D/3D isentropic
 - 2D moist kinematic (this workshop's "case 1")
 - bulk (Kessler) µ-physics
 - particle-based (super-droplet) μ -physics
- input/output: ASCII, netCDF-4
- parallelisation: OpenMP, Boost.Threads, Boost.MPI

all selectable at runtime via command-line options

→ command-line string unambiguously defines simulation (in output file)
 → easy to call from Python, Octave, GDL, ... or a web server

• tutoring:

- Piotr Smolarkiewicz / NCAR
- Wojciech Grabowski / NCAR

free/libre/open-source software used in icicle:
 libs: Blitz++, Thrust, Boost.[Units,Thread,MPI,...], odeint, netCDF
 tools: GNU gcc, Apple/LLVM clang, Doxygen, CMake, ...

Polish National Science Centre

Anna Jaruga Hanna Pawłows

- tutoring:
 - Piotr Smolarkiewicz / NCAR
 - Wojciech Grabowski / NCAR
- free/libre/open-source software used in icicle:
 - libs: Blitz++, Thrust, Boost.[Units,Thread,MPI,...], odeint, netCDF
 - tools: GNU gcc, Apple/LLVM clang, Doxygen, CMake, ...

Polish National Science Centre (project no. DEC-2011/01/N/ST10/01483

Anna Jaruga Hanna Pawłows

- tutoring:
 - Piotr Smolarkiewicz / NCAR
 - Wojciech Grabowski / NCAR
- free/libre/open-source software used in icicle:
 - libs: Blitz++, Thrust, Boost.[Units,Thread,MPI,...], odeint, netCDF
 - tools: GNU gcc, Apple/LLVM clang, Doxygen, CMake, ...
- funding:
 - Polish National Science Centre (project no. DEC-2011/01/N/ST10/01483)

tutoring:

- Piotr Smolarkiewicz / NCAR
- Wojciech Grabowski / NCAR

• free/libre/open-source software used in icicle:

- libs: Blitz++, Thrust, Boost.[Units,Thread,MPI,...], odeint, netCDF
- tools: GNU gcc, Apple/LLVM clang, Doxygen, CMake, ...

• funding:

 Polish National Science Centre (project no. DEC-2011/01/N/ST10/01483)

co-authors:

- Anna Jaruga
- Hanna Pawłowska

icicle code repository (pre-alpha, GPL): http://icicle.igf.fuw.edu.pl/

Thanks for your attention!

icicle code repository (pre-alpha, GPL): http://icicle.igf.fuw.edu.pl/

beta planned for 2013 Q2 (code & docs)

Thanks for your attention!

icicle code repository (pre-alpha, GPL): http://icicle.igf.fuw.edu.pl/

beta planned for 2013 Q2 (code & docs)

Thanks for your attention!

Merali 2010 (Nature, vol. 467, p. 775-777)

C:\lab> f77 -o data.exe



...why scientific programming does not compute

BY ZEEYA MERALI

http://www.nature.com/news/2010/101013/full/467775a.html

return_macro() definition (preprocessor)

define return_macro(expr) \
-> decltype(safeToReturn(expr)) \
{ return safeToReturn(expr); }