On applications of MPDATA in cloud microphysics and finance

Sylwester Arabas
Jagiellonian University

Jagiellonian University, Kraków, Poland

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

plan of the talk

- MPDATA (Smolarkiewicz '83 ... Smolarkiewicz et al. 20XX)

- MPDATA goes open source: (Arabas et al. '14, Jaruga et al. '15)

- MPDATA meets Black-Scholes (Arabas & Farhat, 2019)

- MPDATA & diffusional growth (with Olesik & Unterstraßer, WIP)

MPDATA

a.k.a. the Smolarkiewicz method

MPDATA in a nutshell (Smolarkiewicz 1983 MWR . . .)

transport PDE: $$\frac{\partial \psi}{\partial t} + \frac{\partial}{\partial x}(v\psi) = 0$$

MPDATA in a nutshell (Smolarkiewicz 1983 MWR . . .)

transport PDE: $\frac{\partial\psi}{\partial t} + \frac{\partial}{\partial x}(v\psi) = 0$

$$\psi_i^{n+1} = \psi_i^n - \left[F(\psi_i^n, \psi_{i+1}^n, \mathcal{C}_{i+1/2}) - F(\psi_{i-1}^n, \psi_i^n, \mathcal{C}_{i-1/2})\right]$$

$$F(\psi_L, \psi_R, \mathcal{C}) = \max(\mathcal{C}, 0)\cdot\psi_L + \min(\mathcal{C}, 0)\cdot\psi_R$$

$$\mathcal{C} = v\Delta t/\Delta x$$

upwind

MPDATA in a nutshell (Smolarkiewicz 1983 MWR ...)

transport PDE: $\frac{\partial \psi}{\partial t} + \frac{\partial}{\partial x}(v\psi) = 0$

$$\psi_i^{n+1} = \psi_i^n - \left[F(\psi_i^n, \psi_{i+1}^n, \mathcal{C}_{i+1/2}) - F(\psi_{i-1}^n, \psi_i^n, \mathcal{C}_{i-1/2})\right]$$
$$F(\psi_L, \psi_R, \mathcal{C}) = \max(\mathcal{C}, 0) \cdot \psi_L + \min(\mathcal{C}, 0) \cdot \psi_R$$
$$\mathcal{C} = v\Delta t/\Delta x$$

upwind

modified eq.: $$\frac{\partial \psi}{\partial t} + \frac{\partial}{\partial x}(v\psi) + \underbrace{K\frac{\partial^2 \psi}{\partial x^2}}_{\text{numerical diffusion}} + \ldots = 0$$

MEA

MPDATA in a nutshell (Smolarkiewicz 1983 MWR . . .)

transport PDE: $\dfrac{\partial \psi}{\partial t} + \dfrac{\partial}{\partial x}(v\psi) = 0$

$$\psi_i^{n+1} = \psi_i^n - \left[F(\psi_i^n, \psi_{i+1}^n, \mathcal{C}_{i+1/2}) - F(\psi_{i-1}^n, \psi_i^n, \mathcal{C}_{i-1/2})\right]$$
$$F(\psi_L, \psi_R, \mathcal{C}) = \max(\mathcal{C}, 0) \cdot \psi_L + \min(\mathcal{C}, 0) \cdot \psi_R$$
$$\mathcal{C} = v\Delta t / \Delta x$$

upwind

modified eq.: $$\frac{\partial \psi}{\partial t} + \frac{\partial}{\partial x}(v\psi) + \underbrace{K\frac{\partial^2 \psi}{\partial x^2}}_{\text{numerical diffusion}} + \ldots = 0$$

MEA

$$\frac{\partial \psi}{\partial t} + \frac{\partial}{\partial x}(v\psi) + \frac{\partial}{\partial x}\underbrace{\left[\left(-\frac{K}{\psi}\frac{\partial \psi}{\partial x}\right)\psi\right]}_{\text{antidiffusive flux}} = 0$$

MPDATA in a nutshell (Smolarkiewicz 1983 MWR . . .)

transport PDE: $\dfrac{\partial \psi}{\partial t} + \dfrac{\partial}{\partial x}(v\psi) = 0$

$$\psi_i^{n+1} = \psi_i^n - \left[F(\psi_i^n, \psi_{i+1}^n, \mathcal{C}_{i+1/2}) - F(\psi_{i-1}^n, \psi_i^n, \mathcal{C}_{i-1/2})\right]$$
$$F(\psi_L, \psi_R, \mathcal{C}) = \max(\mathcal{C}, 0) \cdot \psi_L + \min(\mathcal{C}, 0) \cdot \psi_R$$
$$\mathcal{C} = v\Delta t / \Delta x$$

upwind

modified eq.: $\dfrac{\partial \psi}{\partial t} + \dfrac{\partial}{\partial x}(v\psi) + \underbrace{K\dfrac{\partial^2 \psi}{\partial x^2}}_{\text{numerical diffusion}} + \ldots = 0$

MEA

$$\frac{\partial \psi}{\partial t} + \frac{\partial}{\partial x}(v\psi) + \frac{\partial}{\partial x}\underbrace{\left[\left(-\frac{K}{\psi}\frac{\partial \psi}{\partial x}\right)\psi\right]}_{\text{antidiffusive flux}} = 0$$

$$\mathcal{C}'_{i+1/2} = (|\mathcal{C}_{i+1/2}| - \mathcal{C}^2_{i+1/2})A_{i+1/2}$$
$$A_{i+1/2} = \frac{\psi_{i+1} - \psi_i}{\psi_{i+1} + \psi_i}$$

MPDATA: reverse numerical diffusion by integrating the antidiffusive flux using upwind (in a corrective iteration)

MPDATA: key features (review: e.g. Smolarkiewicz 2006)

Multidimensional **P**ositive **D**efinite Advection Transport Algorithm

MPDATA: key features (review: e.g. Smolarkiewicz 2006)

Multidimensional **P**ositive **D**efinite Advection Transport Algorithm

- **M**ultidimensional:
  antidiffusive fluxes include cross-dimensional terms, as opposed to dimensionally-split schemes

MPDATA: key features (review: e.g. Smolarkiewicz 2006)

Multidimensional **P**ositive **D**efinite Advection Transport Algorithm

- **M**ultidimensional:
  antidiffusive fluxes include cross-dimensional terms, as opposed to dimensionally-split schemes
- **P**ositive **D**efinite:
  sign-preserving + "infinite-gauge formulation for variable-sign fields

MPDATA: key features (review: e.g. Smolarkiewicz 2006)

Multidimensional **P**ositive **D**efinite Advection Transport Algorithm

- **M**ultidimensional:
  antidiffusive fluxes include cross-dimensional terms, as opposed to dimensionally-split schemes
- **P**ositive **D**efinite:
  sign-preserving + “infinite-gauge formulation for variable-sign fields
- Conservative:
  upstream for all iterations ($\rightsquigarrow$ stability cond.)

MPDATA: key features (review: e.g. Smolarkiewicz 2006)

Multidimensional **P**ositive **D**efinite Advection Transport Algorithm

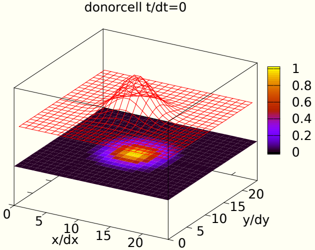
- **M**ultidimensional:
  antidiffusive fluxes include cross-dimensional terms, as opposed to dimensionally-split schemes
- **P**ositive **D**efinite:
  sign-preserving + "infinite-gauge formulation for variable-sign fields
- Conservative:
  upstream for all iterations ($\rightsquigarrow$ stability cond.)
- High-Order Accurate:
  up to 3rd-order in time and space (dep. on options & flow)

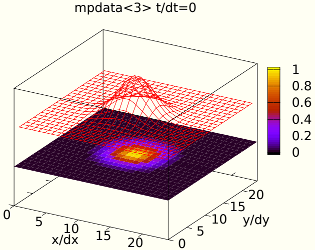
MPDATA: key features (review: e.g. Smolarkiewicz 2006)

Multidimensional **P**ositive **D**efinite Advection Transport Algorithm

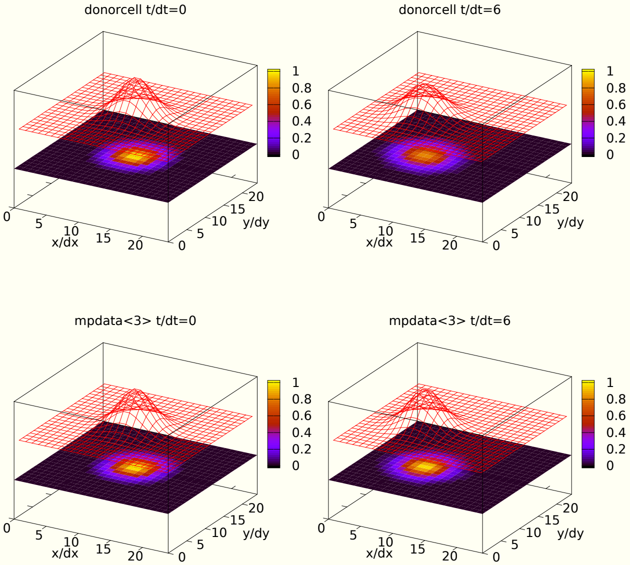
- **M**ultidimensional:
  antidiffusive fluxes include cross-dimensional terms, as opposed to dimensionally-split schemes
- **P**ositive **D**efinite:
  sign-preserving + "infinite-gauge formulation for variable-sign fields
- Conservative:
  upstream for all iterations ($\rightsquigarrow$ stability cond.)
- High-Order Accurate:
  up to 3rd-order in time and space (dep. on options & flow)
- Monotonic:
  with Flux-Corrected Transport option

przykład 2D (Arabas et al. 2014, Sci. Prog.)

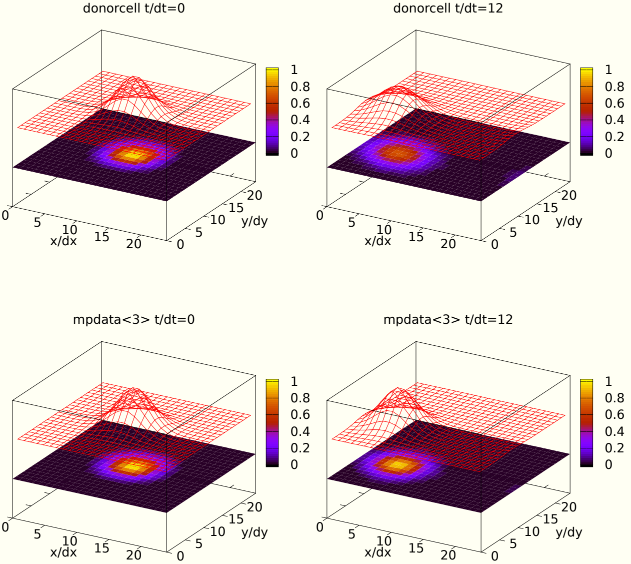




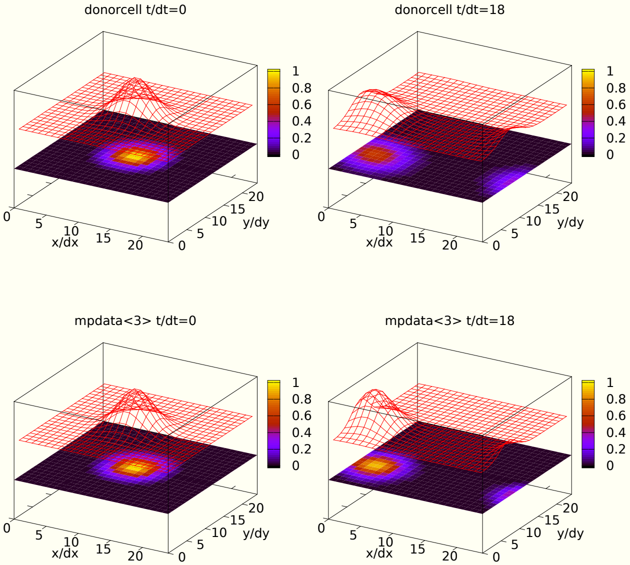
przykład 2D (Arabas et al. 2014, Sci. Prog.)



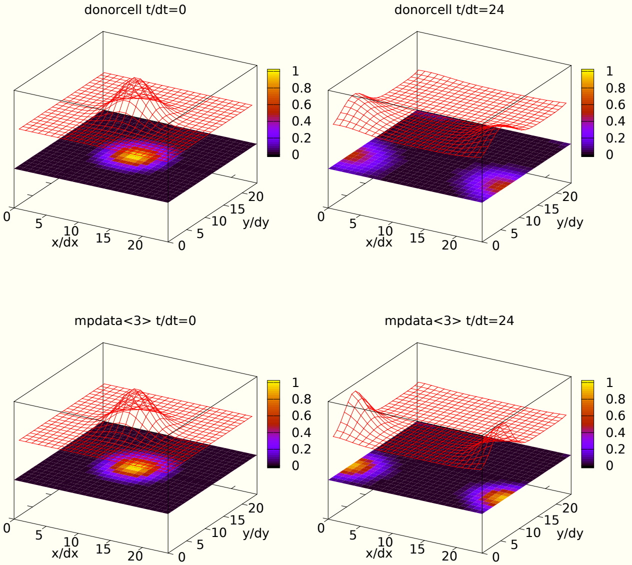
przykład 2D (Arabas et al. 2014, Sci. Prog.)



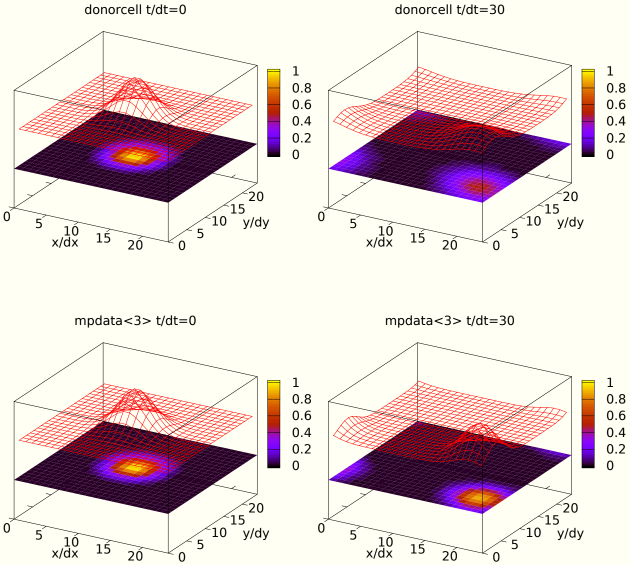
przykład 2D (Arabas et al. 2014, Sci. Prog.)



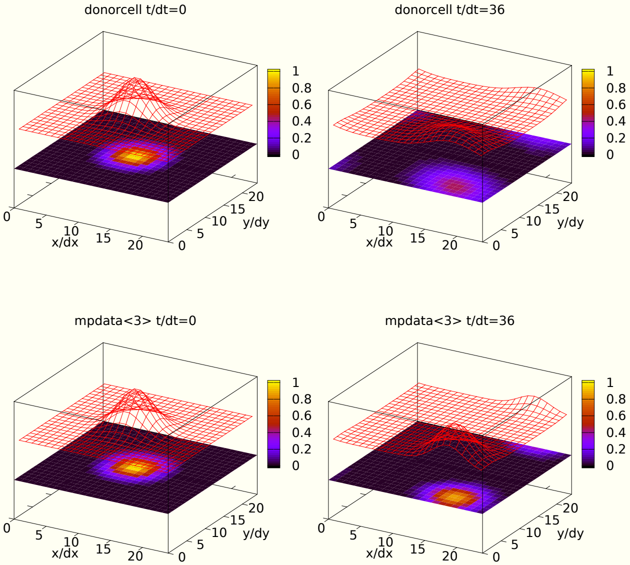
przykład 2D (Arabas et al. 2014, Sci. Prog.)



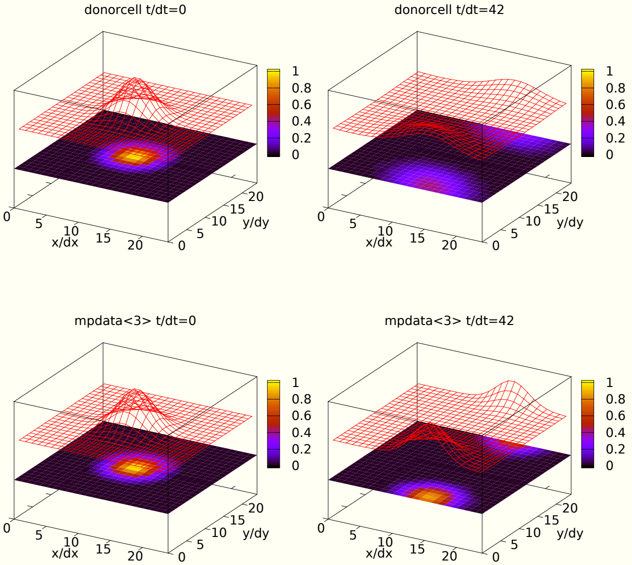
przykład 2D (Arabas et al. 2014, Sci. Prog.)



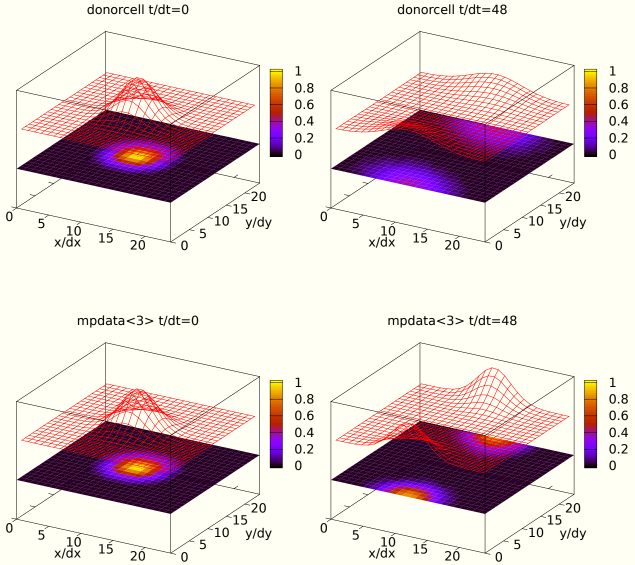
przykład 2D (Arabas et al. 2014, Sci. Prog.)



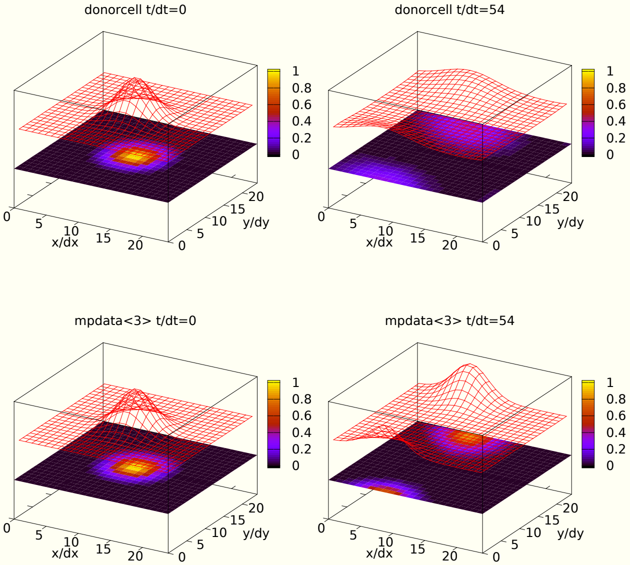
przykład 2D (Arabas et al. 2014, Sci. Prog.)



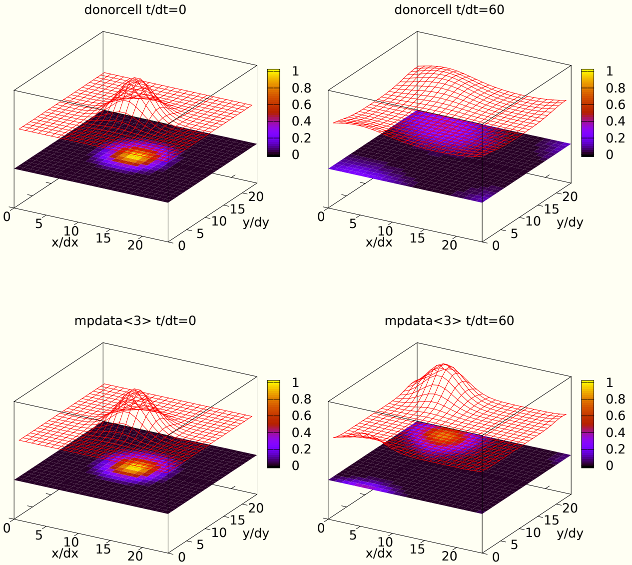
przykład 2D (Arabas et al. 2014, Sci. Prog.)



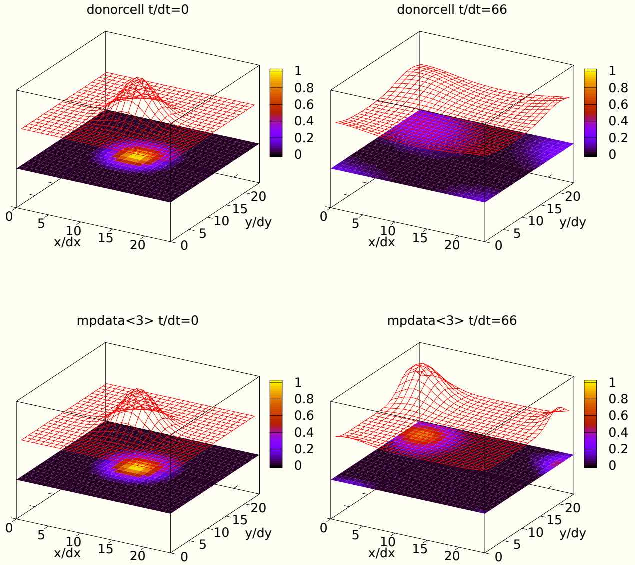
przykład 2D (Arabas et al. 2014, Sci. Prog.)



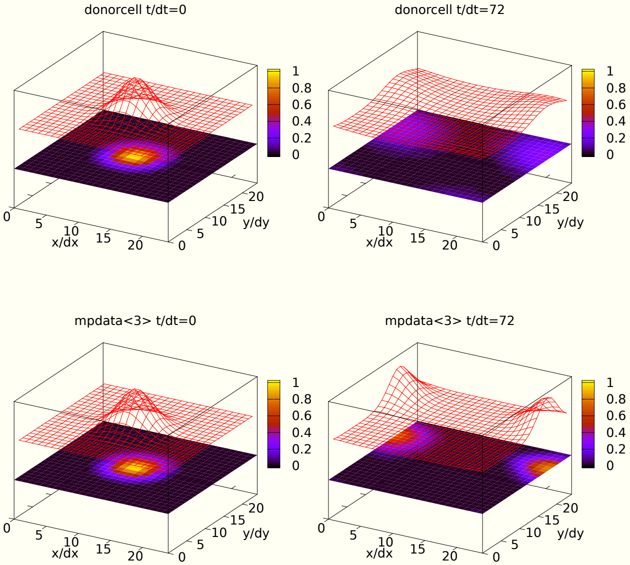
przykład 2D (Arabas et al. 2014, Sci. Prog.)



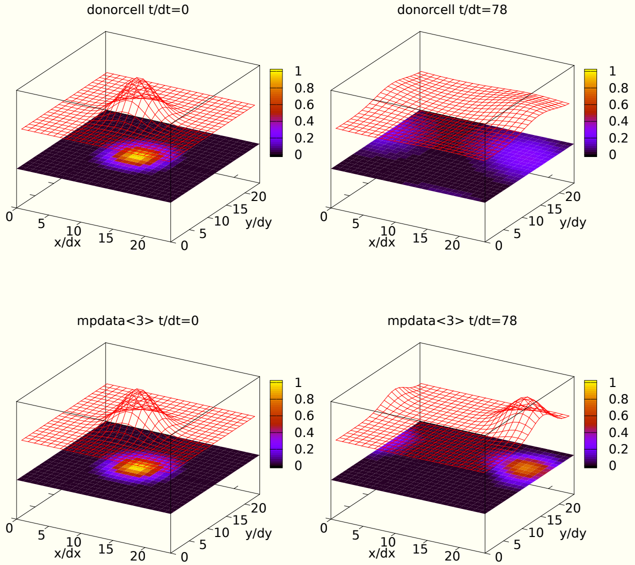
przykład 2D (Arabas et al. 2014, Sci. Prog.)



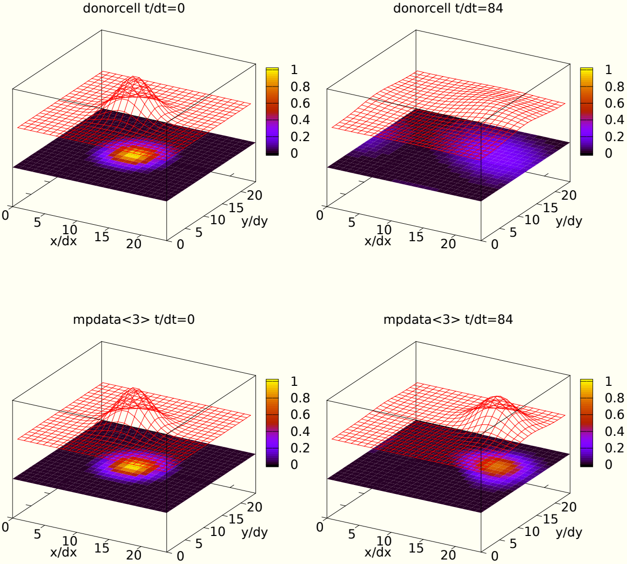
przykład 2D (Arabas et al. 2014, Sci. Prog.)



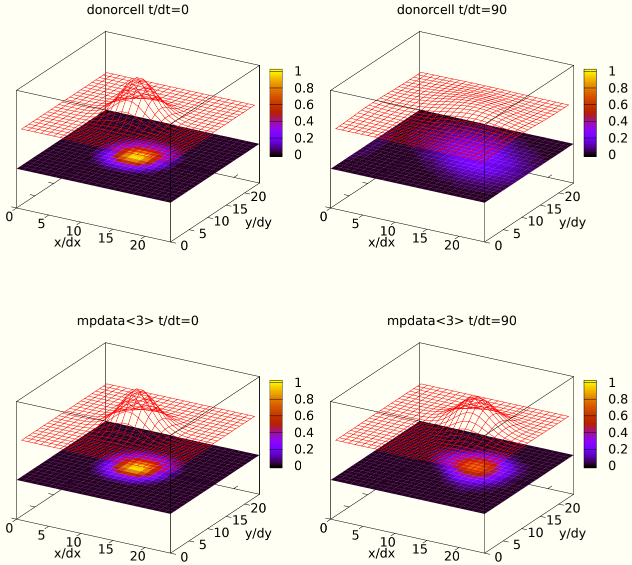
przykład 2D (Arabas et al. 2014, Sci. Prog.)



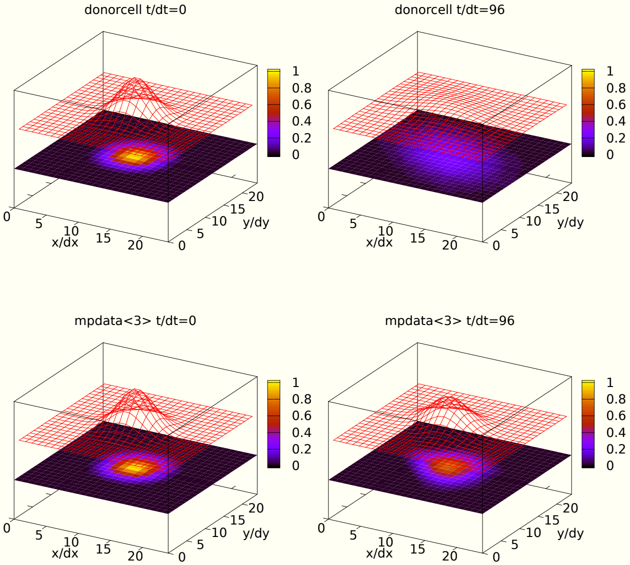
przykład 2D (Arabas et al. 2014, Sci. Prog.)



przykład 2D (Arabas et al. 2014, Sci. Prog.)



przykład 2D (Arabas et al. 2014, Sci. Prog.)



libmpdata++

libmpdata++: new open-source implementation in C++

Jaruga et al. 2015

Geosci. Model Dev., 8, 1005–1032, 2015
www.geosci-model-dev.net/8/1005/2015/
doi:10.5194/gmd-8-1005-2015

Geoscientific Model Development
Open Access

libmpdata++ 1.0: a library of parallel MPDATA solvers for systems of generalised transport equations

A. Jaruga[1], S. Arabas[1], D. Jarecka[1,2], H. Pawlowska[1], P. K. Smolarkiewicz[3], and M. Waruszewski[1]

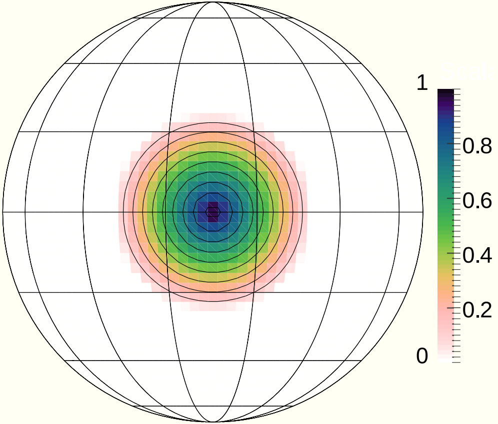
[1]Institute of Geophysics, Faculty of Physics, University of Warsaw, Warsaw, Poland
[2]National Center for Atmospheric Research, Boulder, CO, USA
[3]European Centre for Medium-Range Weather Forecasts, Reading, UK

libmpdata++: generalised transport equation

$$\partial_t(G\psi) + \nabla \cdot (G\vec{u}\psi) = GR$$

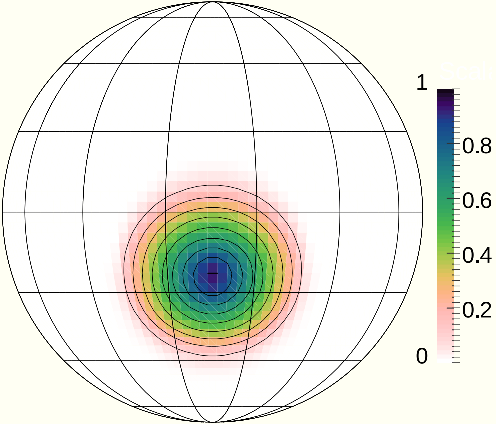
libmpdata++: generalised transport equation

$$\partial_t(G\psi) + \nabla \cdot (G\vec{u}\psi) = GR$$



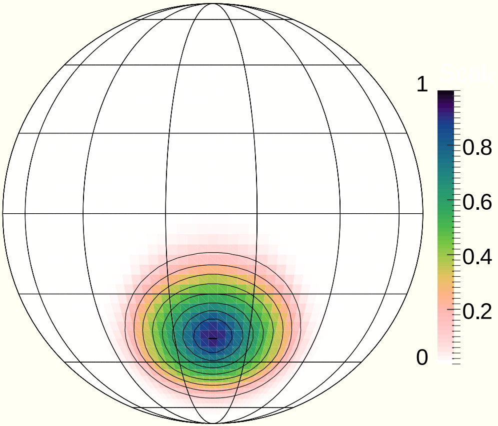
libmpdata++: generalised transport equation

$$\partial_t(G\psi) + \nabla \cdot (G\vec{u}\psi) = GR$$

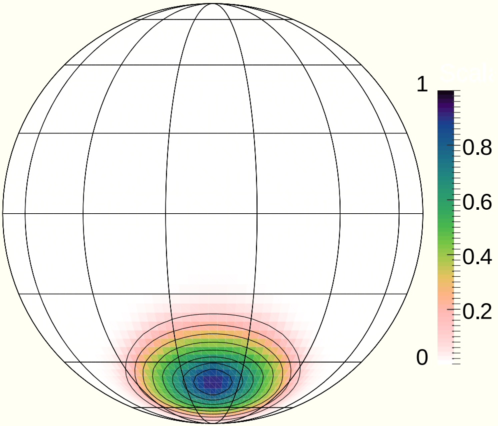


libmpdata++: generalised transport equation

$$\partial_t(G\psi) + \nabla \cdot (G\vec{u}\psi) = GR$$

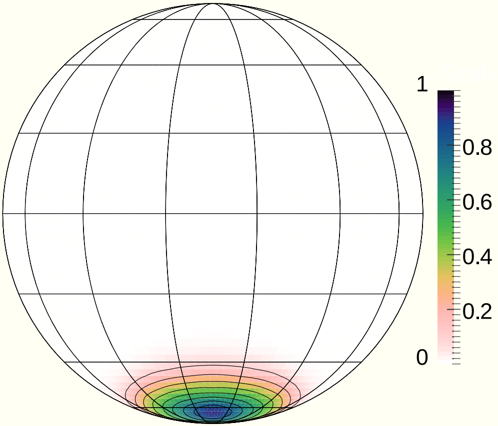


$$\partial_t(G\psi) + \nabla \cdot (G\vec{u}\psi) = GR$$



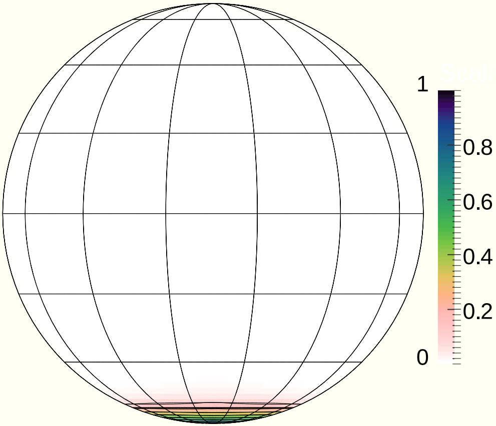
libmpdata++: generalised transport equation

$$\partial_t(G\psi) + \nabla \cdot (G\vec{u}\psi) = GR$$

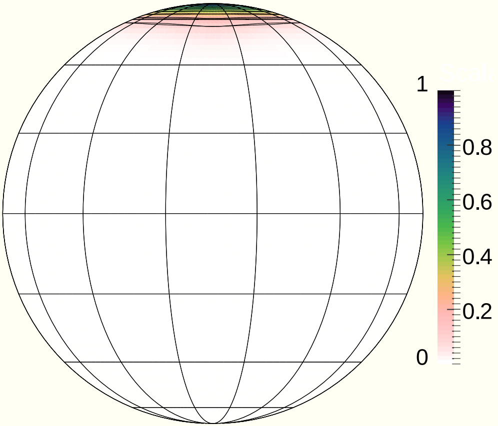


libmpdata++: generalised transport equation

$$\partial_t(G\psi) + \nabla \cdot (G\vec{u}\psi) = GR$$

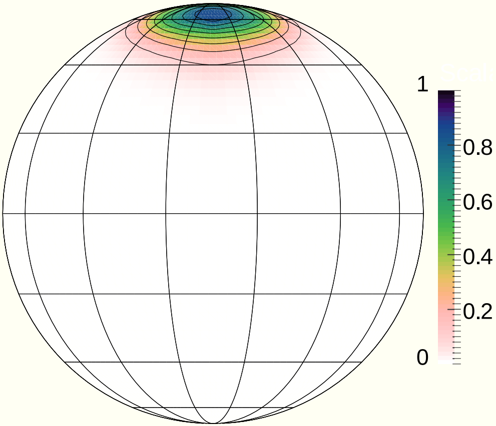


$$\partial_t(G\psi) + \nabla \cdot (G\vec{u}\psi) = GR$$



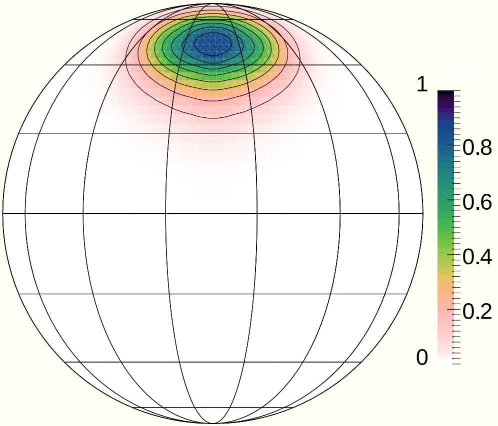
libmpdata++: generalised transport equation

$$\partial_t(G\psi) + \nabla \cdot (G\vec{u}\psi) = GR$$



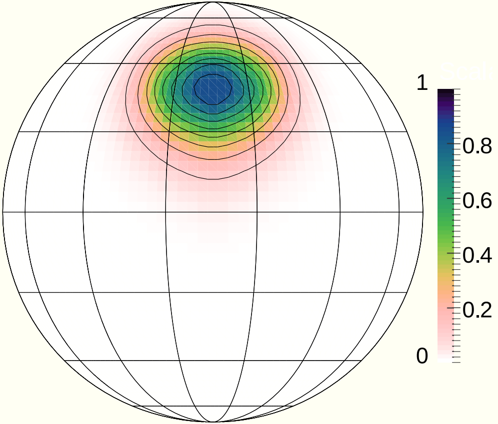
libmpdata++: generalised transport equation

$$\partial_t(G\psi) + \nabla \cdot (G\vec{u}\psi) = GR$$



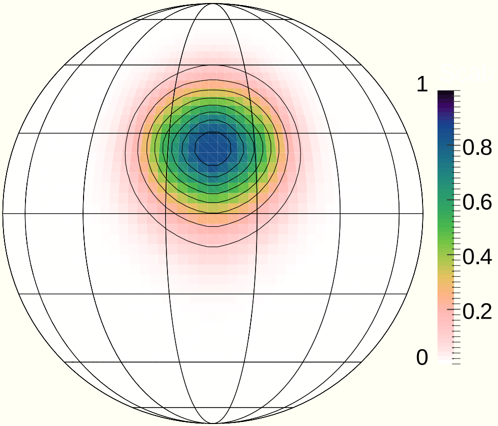
libmpdata++: generalised transport equation

$$\partial_t(G\psi) + \nabla \cdot (G\vec{u}\psi) = GR$$



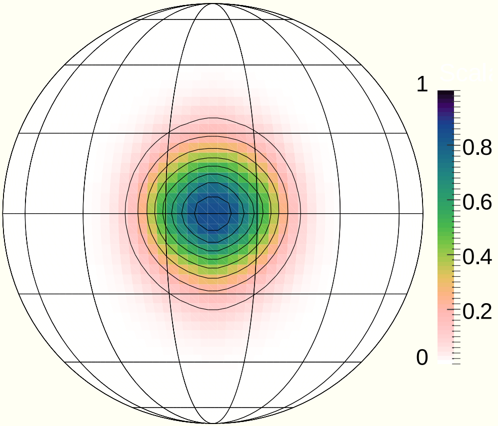
libmpdata++: generalised transport equation

$$\partial_t(G\psi) + \nabla \cdot (G\vec{u}\psi) = GR$$

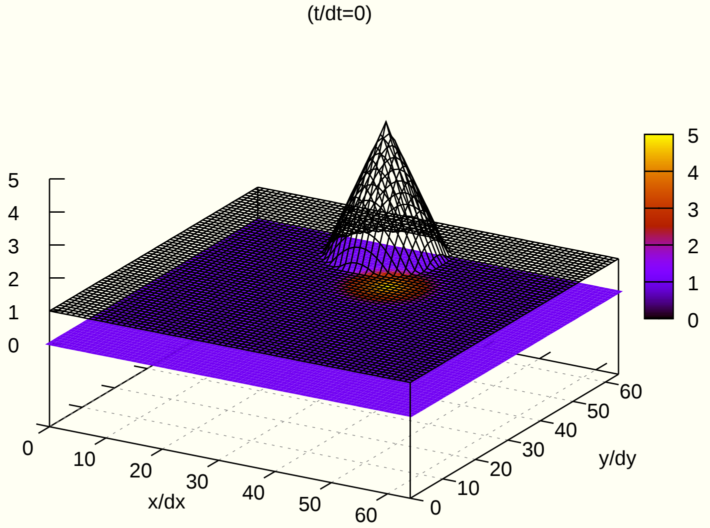


libmpdata++: generalised transport equation

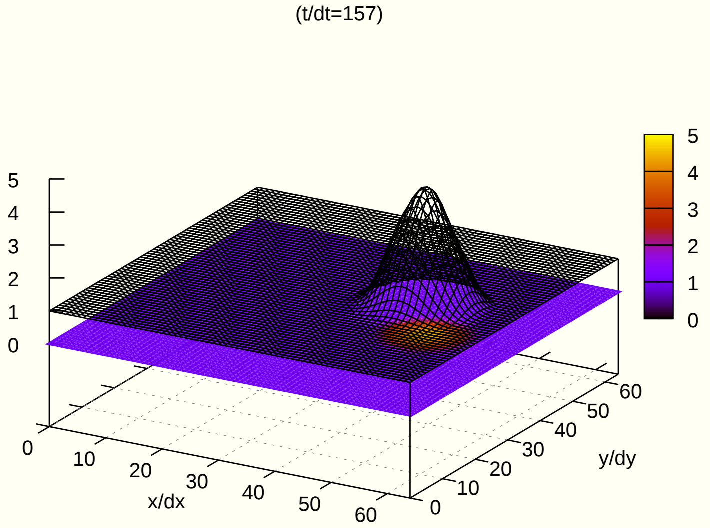
$$\partial_t(G\psi) + \nabla\cdot(G\vec{u}\psi) = GR$$



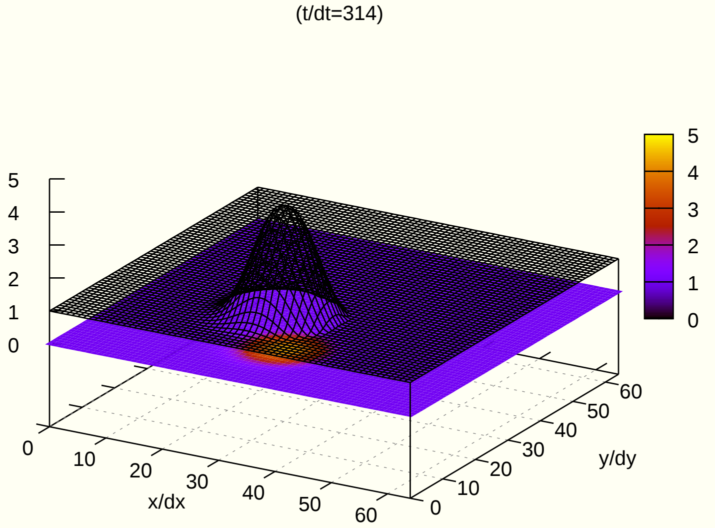
libmpdata++: rotating cone test



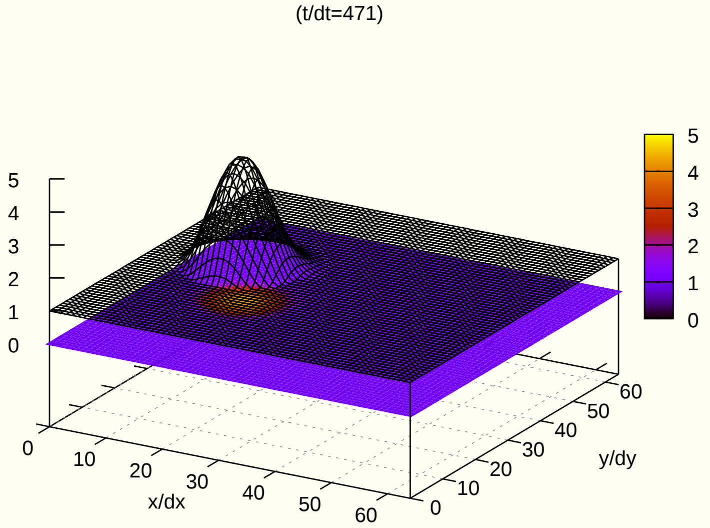
libmpdata++: rotating cone test



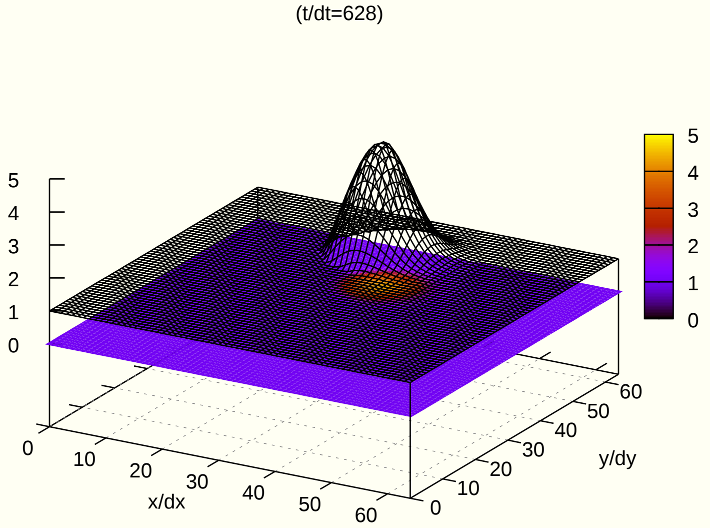
libmpdata++: rotating cone test



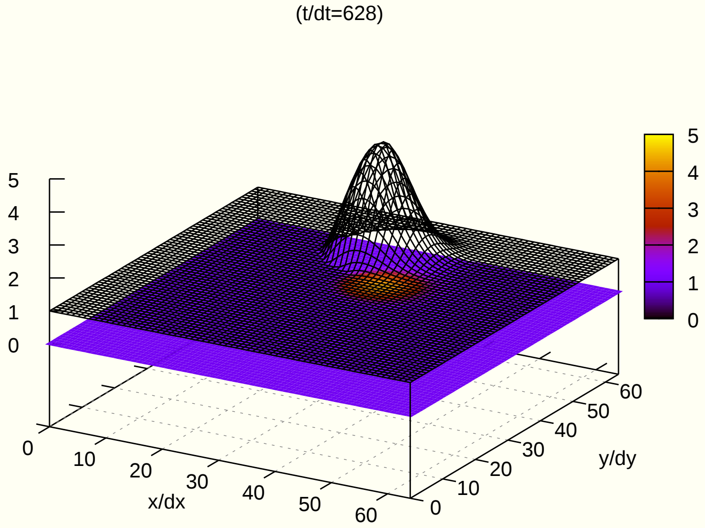
libmpdata++: rotating cone test



libmpdata++: rotating cone test



libmpdata++: rotating cone test



64 LOC using libmpdata++

```cpp
#include <libmpdata++/solvers/mpdata.hpp>
#include <libmpdata++/concurr/serial.hpp>
#include <libmpdata++/output/gnuplot.hpp>

int main()
{
  namespace lmpdt = libmpdataxx;
  const int nx=64, ny=64, nt = 628;

  // compile-time parameters
  struct ct_params_t : lmpdt::ct_params_default_t
  {
    using real_t = double;
    enum { n_dims = 2 };
    enum { n_eqns = 1 };
  };

  // solver choice
  using run_t = lmpdt::output::gnuplot< lmpdt::solvers::mpdata< ct_params_t >>;

  // runtime parameters
  typename run_t::rt_params_t p;
  p.grid_size = {nx+1, ny+1};
  p.outfreq = nt/4;
  p.gnuplot_output = "out_%s_%d.svg";
  p.gnuplot_with = "lines";
  p.gnuplot_cbrange = p.gnuplot_zrange = "[0:5]";

  // sharedmem concurency and boundary condition choice
  lmpdt::concurr::serial<
    run_t,
    lmpdt::bcond::open, lmpdt::bcond::open, // x-left, x-right
    lmpdt::bcond::open, lmpdt::bcond::open  // y-left, y-right
  > run(p);
```

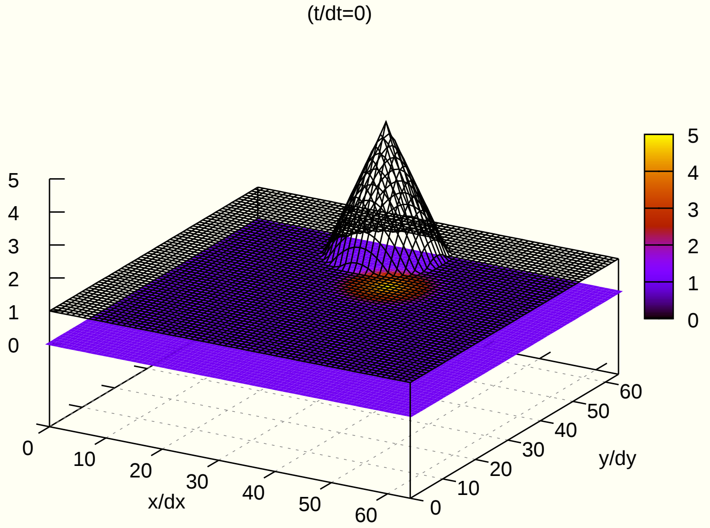
```
  // initial condition
  {
    using namespace blitz::tensor;
    auto psi = run.advectee();

    const double
      dt = .1, dx = 1, dy = 1, omega = .1,
      h = 4., h0 = 1, r = .15 * nx * dx,
      x0 = .5 * nx * dx, y0 = .75 * ny * dy,
      xc = .5 * nx * dx, yc = .50 * ny * dy;

    // cone shape cut at h0
    psi = blitz::pow(i * dx - x0, 2) +
          blitz::pow(j * dy - y0, 2);

    psi = h0 + where(
      psi - pow(r, 2) <= 0,                // if
      h - blitz::sqrt(psi / pow(r/h,2)),   // then
      0.                                   // else
    );

    // constant-angular-velocity rotational field
    run.advector(0) =  omega * (j * dy - yc) * dt/dx;
    run.advector(1) = -omega * (i * dx - xc) * dt/dy;
  }

  // time stepping
  run.advance(nt);
}
```

```

  // initial condition
  {
    using namespace blitz::tensor;
    auto psi = run.advectee();

    const double
      dt = .1, dx = 1, dy = 1, omega = .1,
      h = 4., h0 = 1, r = .15 * nx * dx,
      x0 = .5 * nx * dx, y0 = .75 * ny * dy,
      xc = .5 * nx * dx, yc = .50 * ny * dy;

    // cone shape cut at h0
    psi = blitz::pow(i * dx - x0, 2) +
          blitz::pow(j * dy - y0, 2);

    psi = h0 + where(
      psi - pow(r, 2) <= 0,                // if
      h - blitz::sqrt(psi / pow(r/h,2)),   // then
      0.                                   // else
    );

    // constant-angular-velocity rotational field
    run.advector(0) =  omega * (j * dy - yc) * dt/dx;
    run.advector(1) = -omega * (i * dx - xc) * dt/dy;
  }

  // time stepping
  run.advance(nt);
}
```

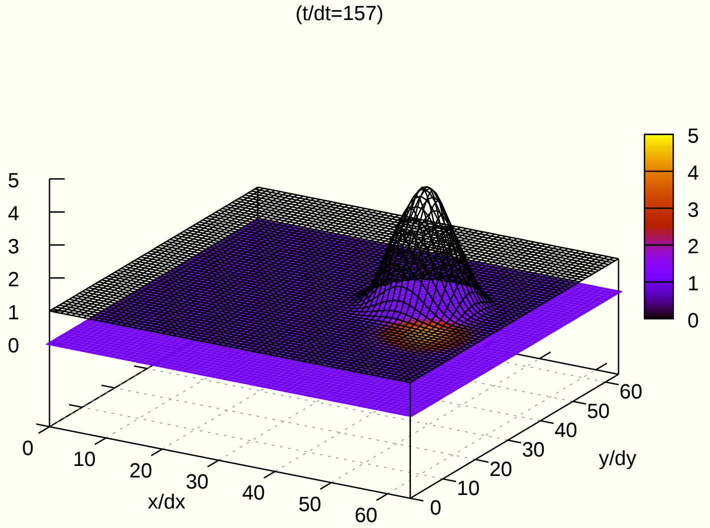
CMakeLists.txt

```
cmake_minimum_required(VERSION 3.0)
project(hello_world CXX)
find_package(libmpdata++)
set(CMAKE_CXX_FLAGS ${libmpdataxx_CXX_FLAGS_RELEASE})
add_executable(hello_world hello_world.cpp)
target_link_libraries(hello_world ${libmpdataxx_LIBRARIES})
```

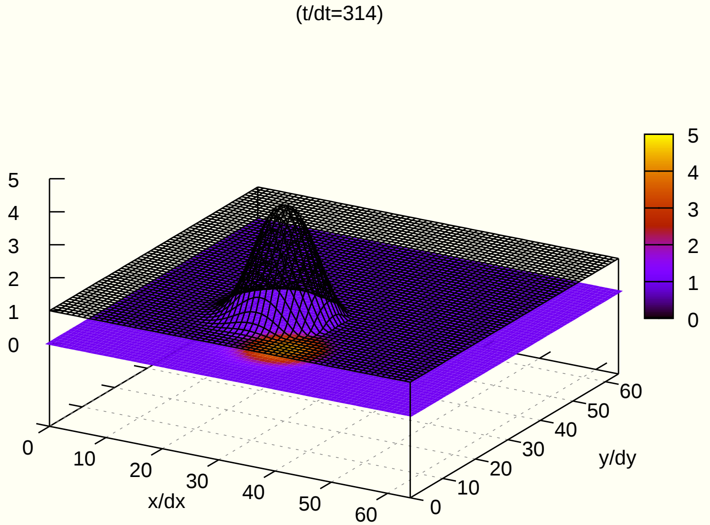
libmpdata++: rotating cone test



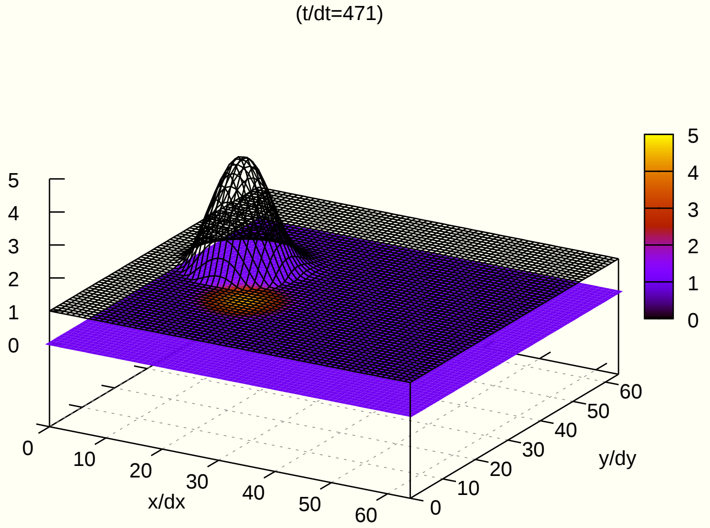
libmpdata++: rotating cone test



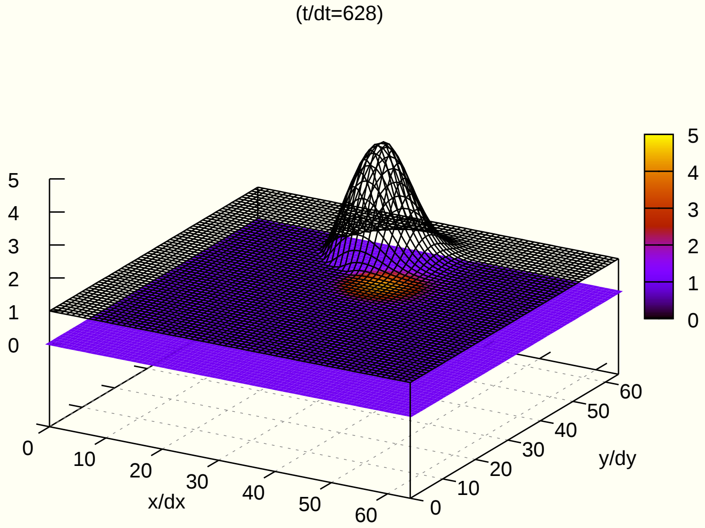
libmpdata++: rotating cone test



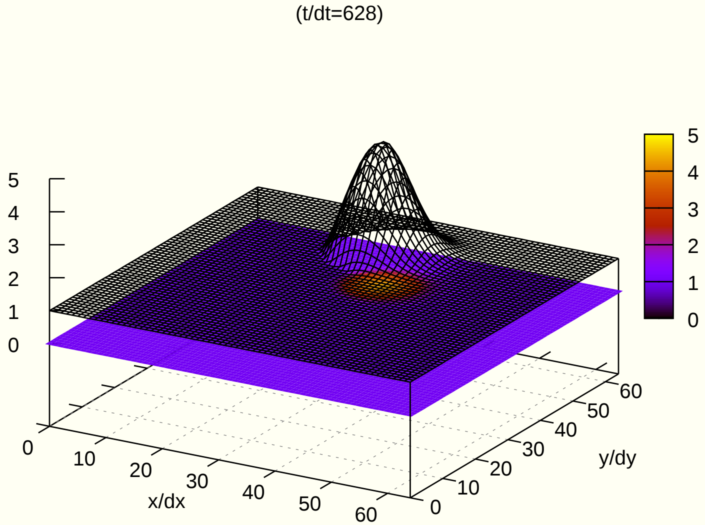
libmpdata++: rotating cone test



libmpdata++: rotating cone test



libmpdata++: rotating cone test



64 LOC using libmpdata++

with multi-threading ⇝ also 64 LOC!

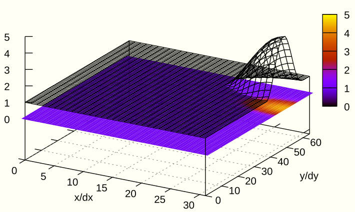
```
2c2
< #include <libmpdata++/concurr/serial.hpp>
---
> #include <libmpdata++/concurr/threads.hpp>
30c30
<   lmpdt::concurr::serial<
---
>   lmpdt::concurr::threads<
```

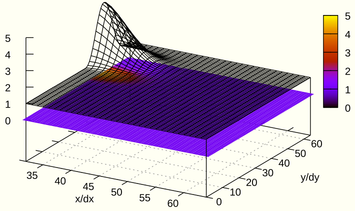
```
$ top
...
  PID USER      PR  NI  S  %CPU %MEM nTH     TIME+ COMMAND
21031 slayoo    20   0  R  73.7  0.1   4   0:01.68 hello_worl   90%
...
```

MPI + threads ⇝ also 64 LOC!!! (recompilation only)

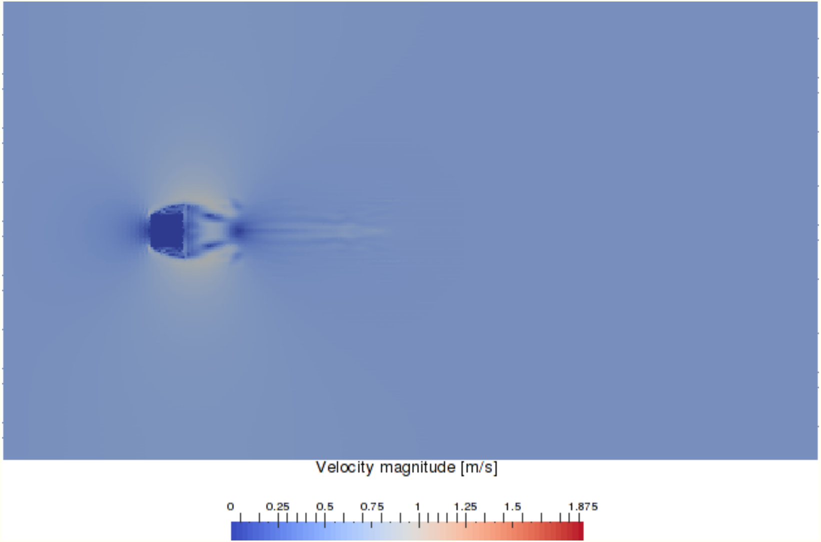
```
$ cmake . -DCMAKE_CXX_COMPILER=mpic++
$ make
$ OMP_NUM_THREADS=2 mpirun -np 2 ./hello_world
```

```
$ top
...
  PID USER      PR  NI  S  %CPU %MEM nTH     TIME+ COMMAND
19640 slayoo    20   0  R  65.5  0.3   2   0:00.92 hello_worl  98%
19641 slayoo    20   0  R  64.0  0.3   2   0:00.91 hello_worl  99%
...
```

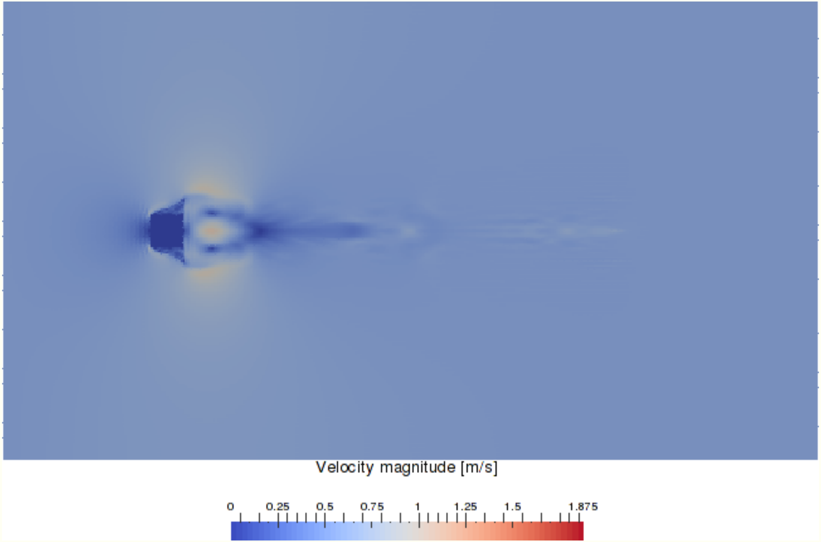




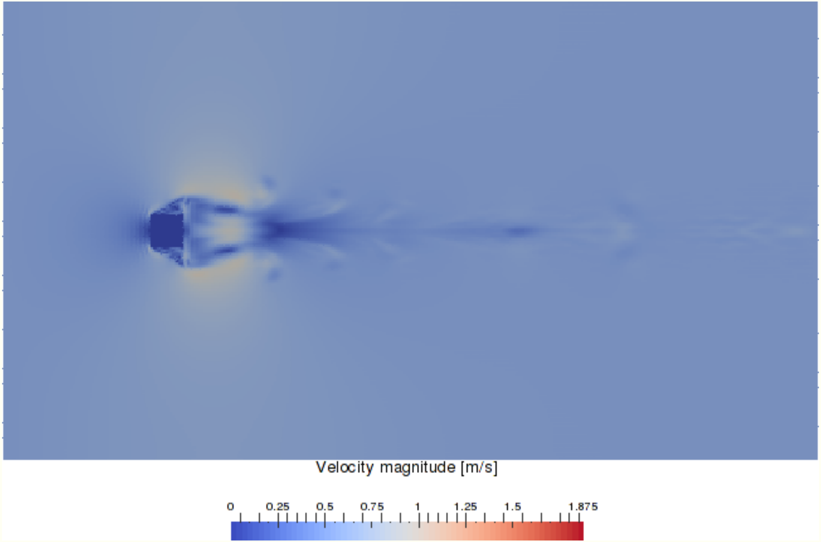
libmpdata++: immersed b.m. (Maciej Waruszewski)



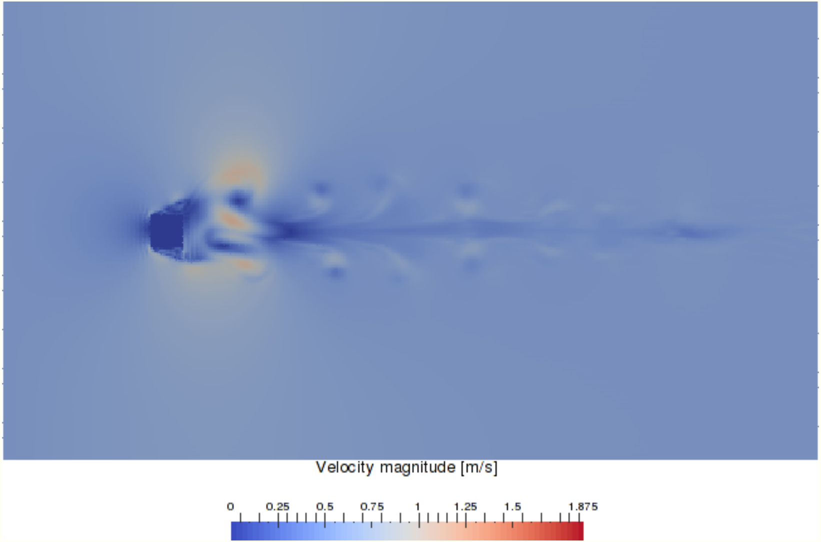
libmpdata++: immersed b.m. (Maciej Waruszewski)



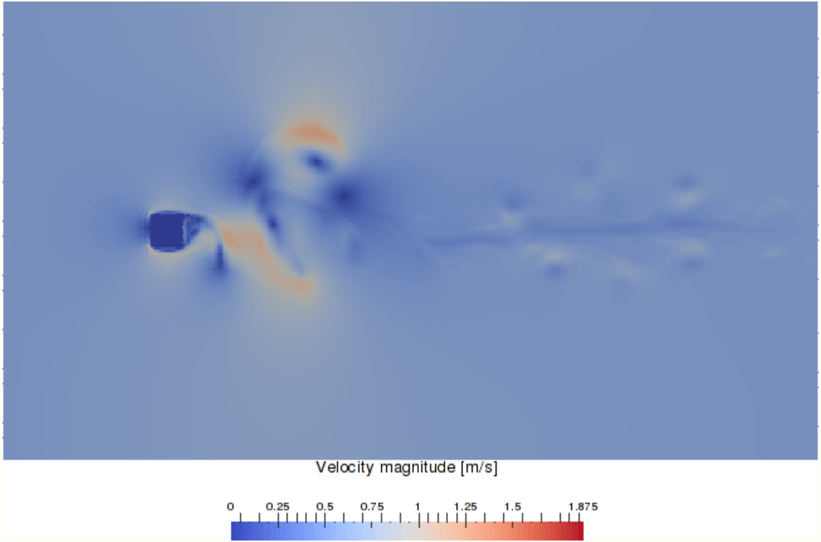
libmpdata++: immersed b.m. (Maciej Waruszewski)



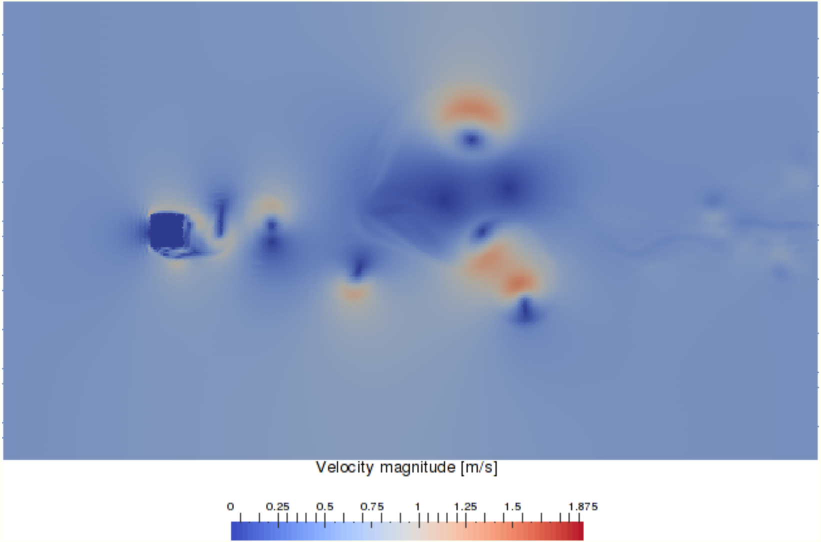
libmpdata++: immersed b.m. (Maciej Waruszewski)



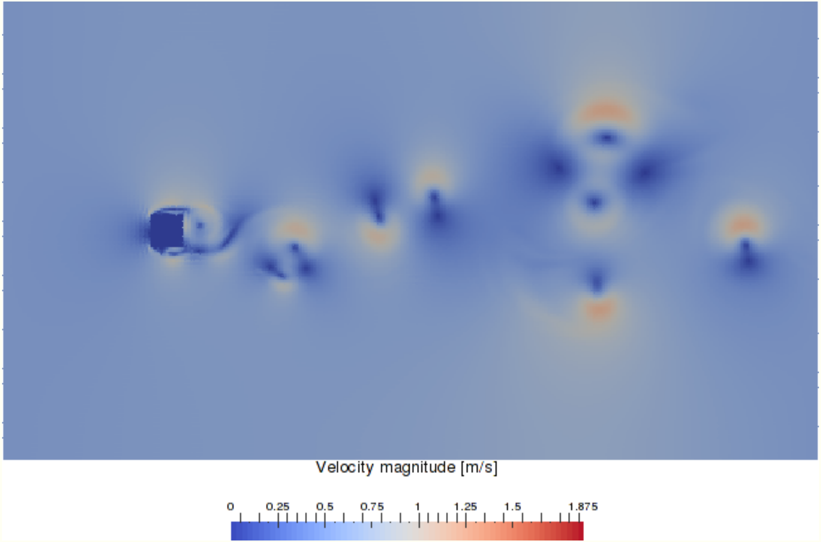
libmpdata++: immersed b.m. (Maciej Waruszewski)



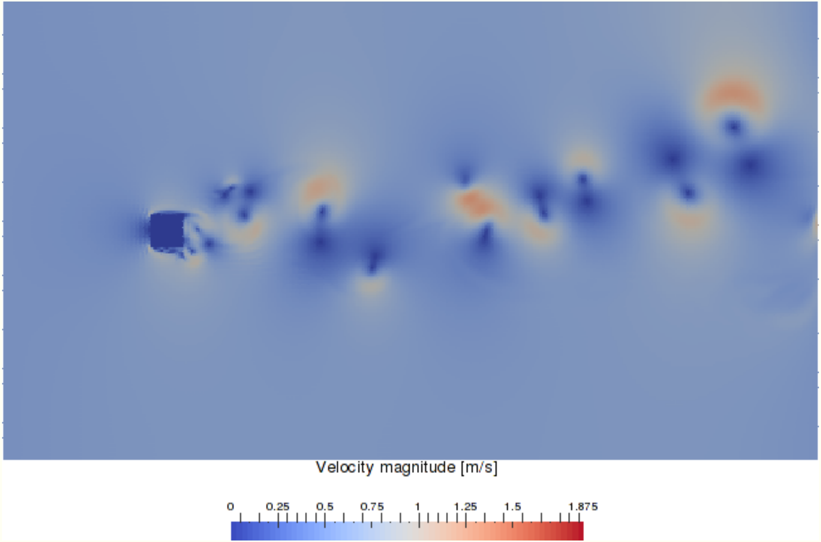
libmpdata++: immersed b.m. (Maciej Waruszewski)



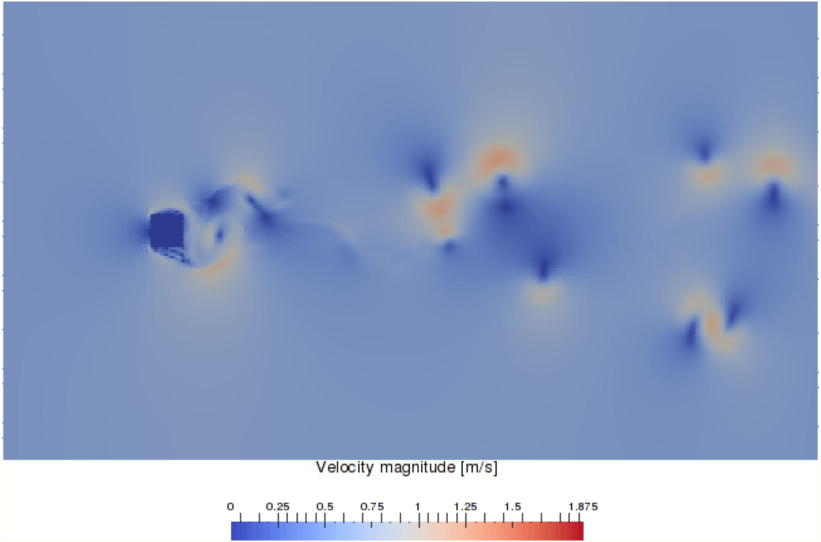
libmpdata++: immersed b.m. (Maciej Waruszewski)



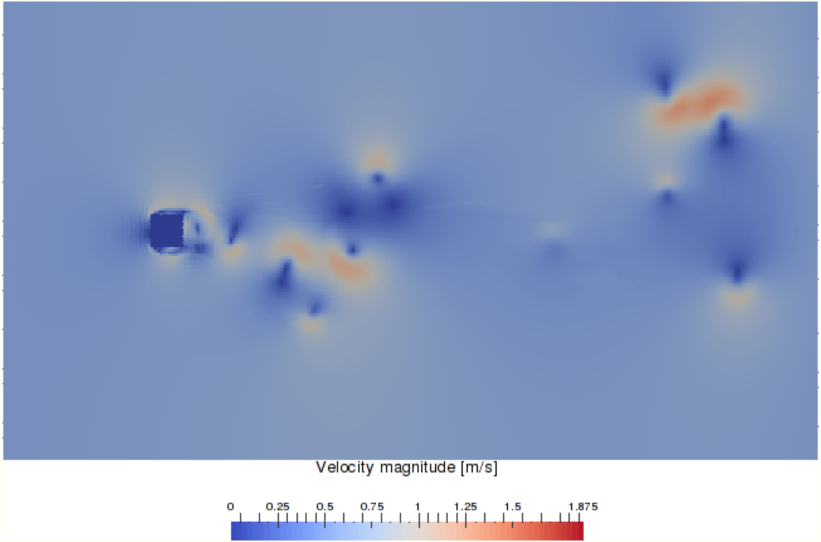
libmpdata++: immersed b.m. (Maciej Waruszewski)



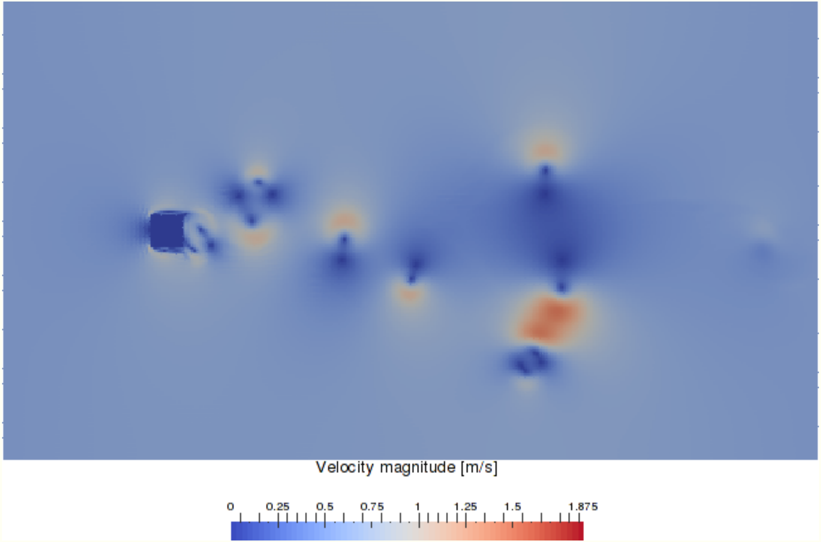
libmpdata++: immersed b.m. (Maciej Waruszewski)



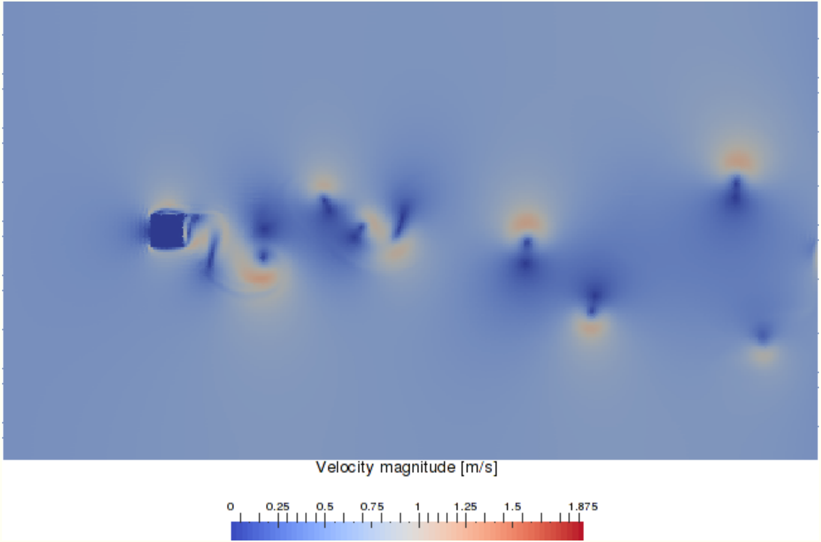
libmpdata++: immersed b.m. (Maciej Waruszewski)



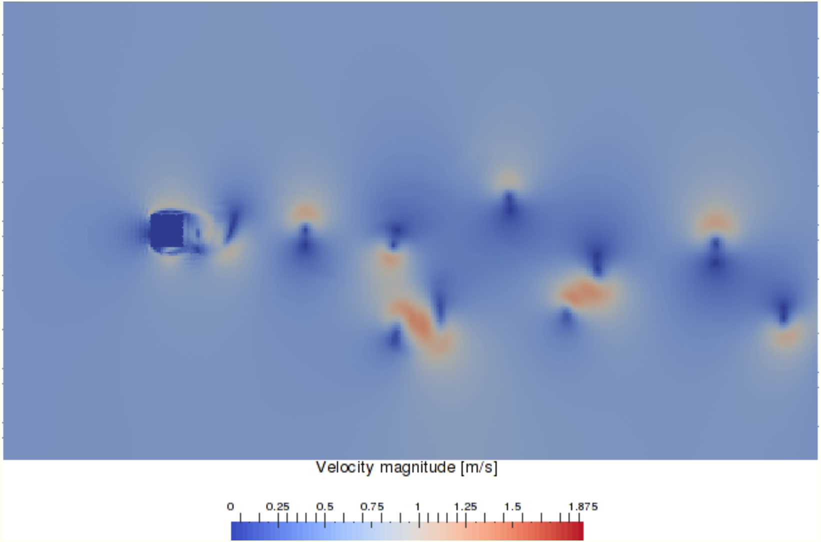
libmpdata++: immersed b.m. (Maciej Waruszewski)



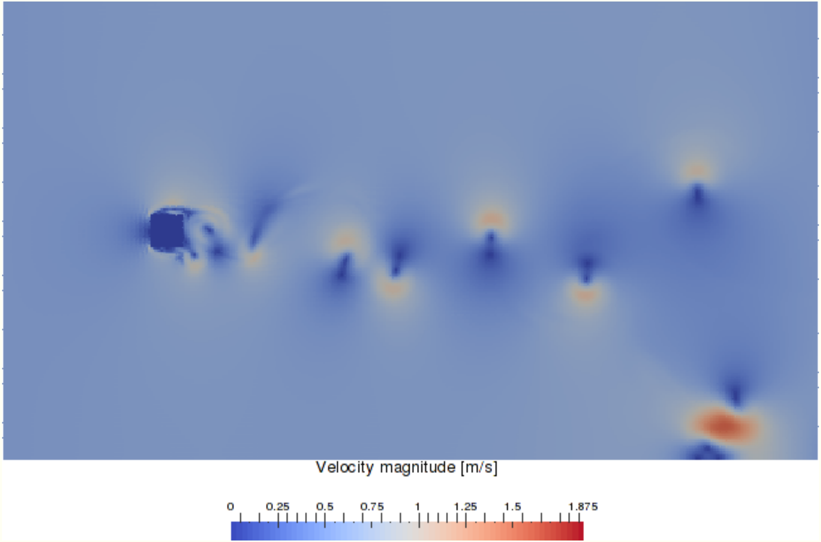
libmpdata++: immersed b.m. (Maciej Waruszewski)



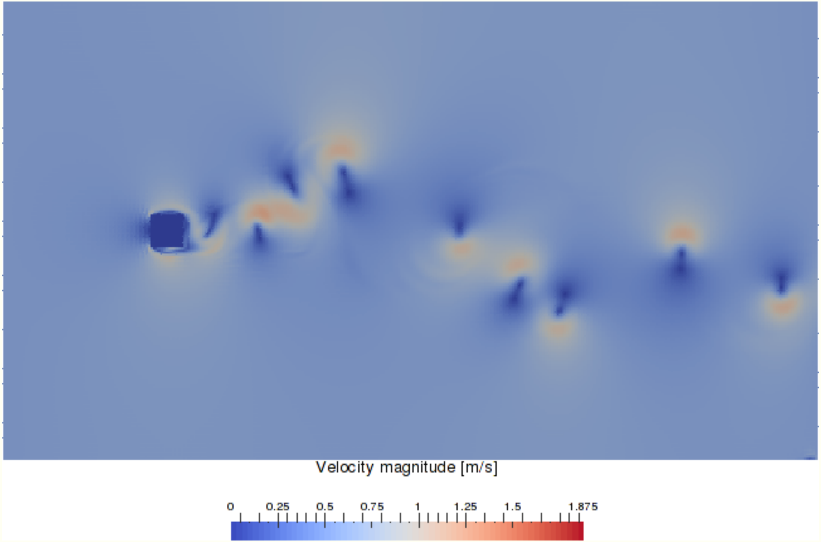
libmpdata++: immersed b.m. (Maciej Waruszewski)



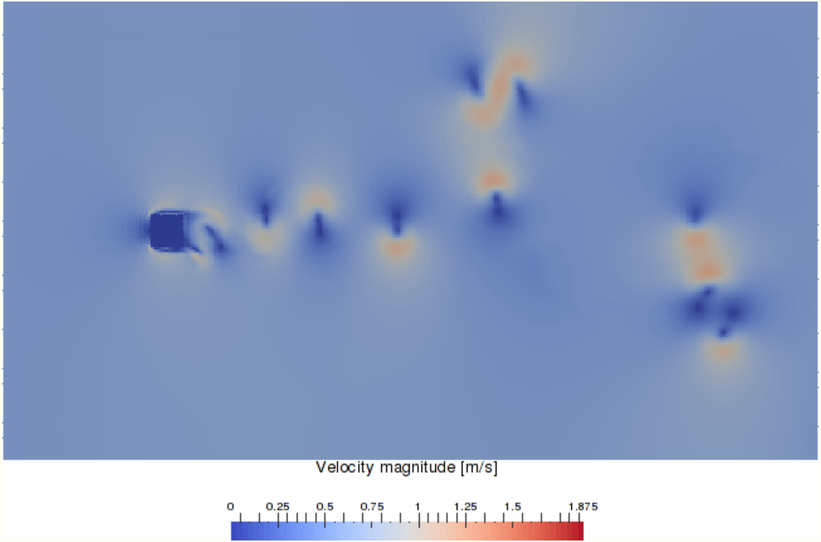
libmpdata++: immersed b.m. (Maciej Waruszewski)



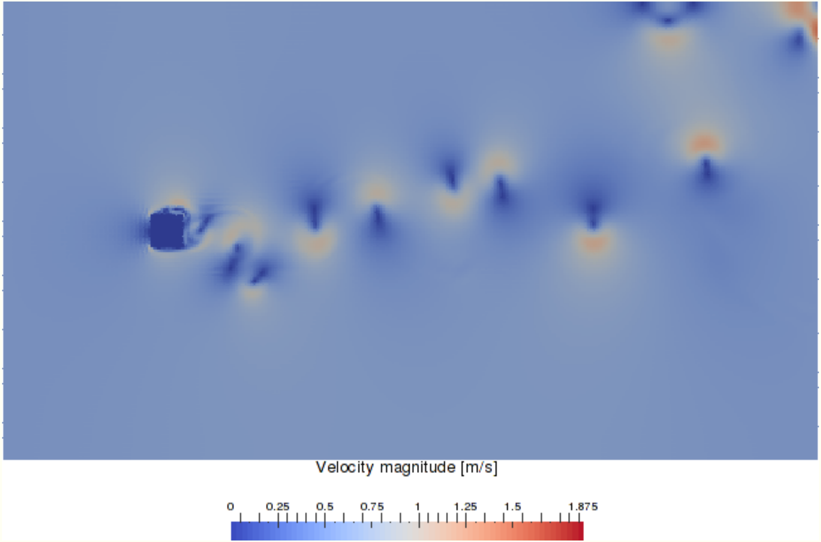
libmpdata++: immersed b.m. (Maciej Waruszewski)



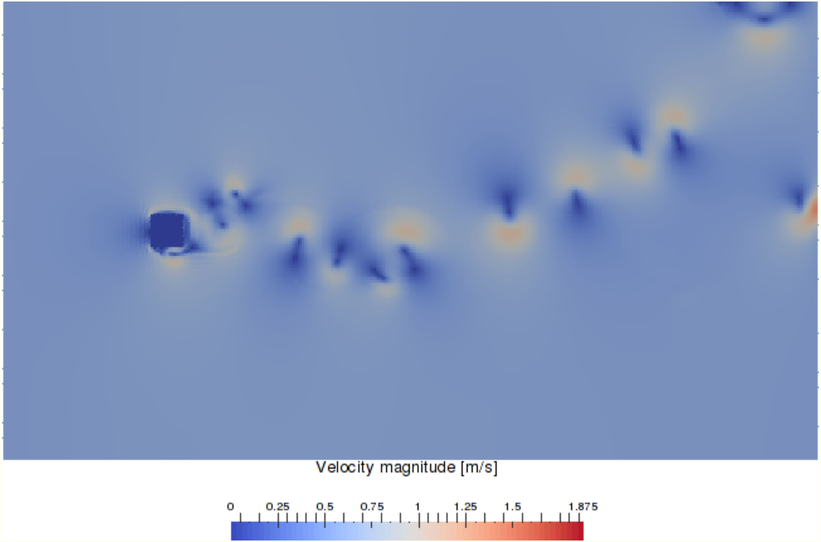
libmpdata++: immersed b.m. (Maciej Waruszewski)



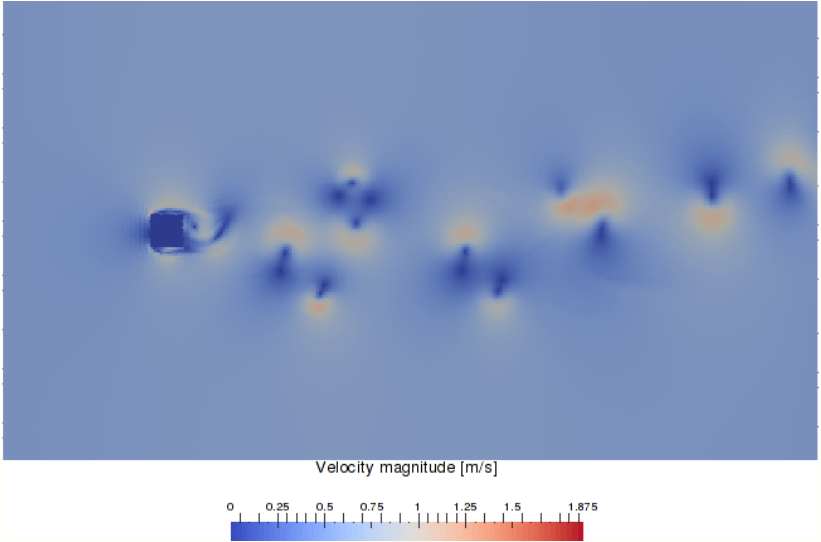
libmpdata++: immersed b.m. (Maciej Waruszewski)



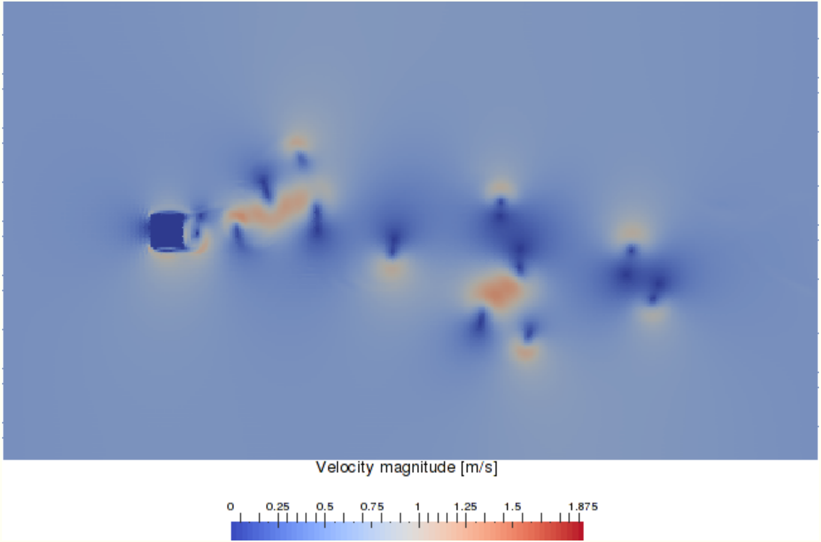
libmpdata++: immersed b.m. (Maciej Waruszewski)



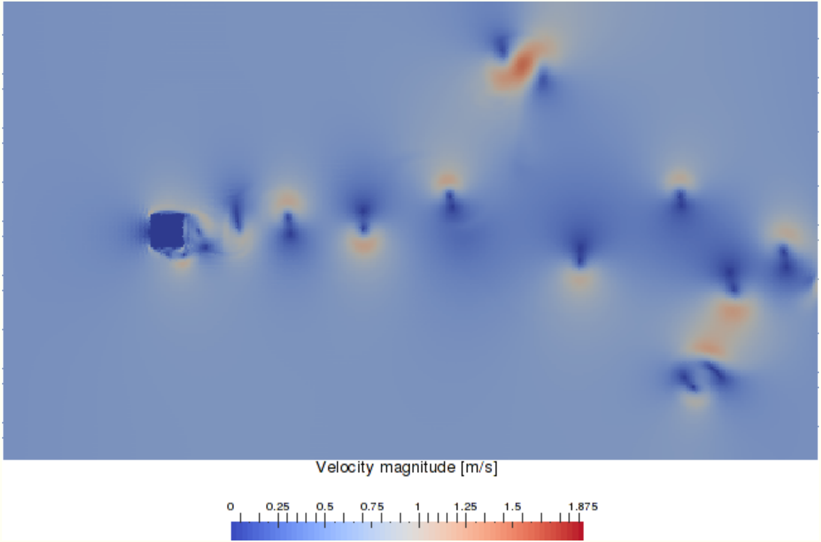
libmpdata++: immersed b.m. (Maciej Waruszewski)



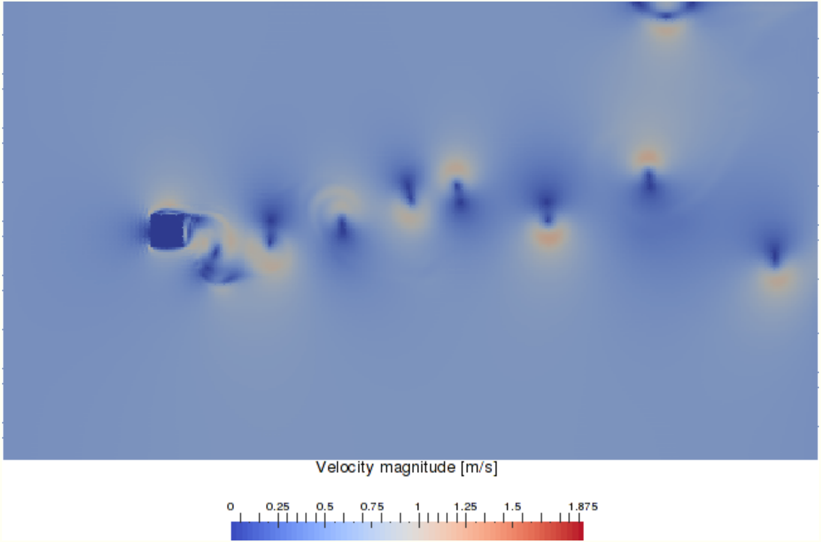
libmpdata++: immersed b.m. (Maciej Waruszewski)



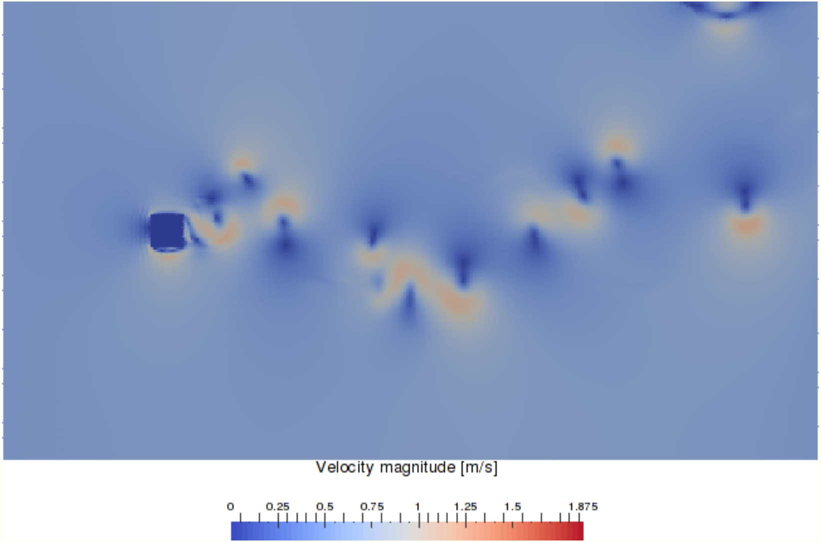
libmpdata++: immersed b.m. (Maciej Waruszewski)



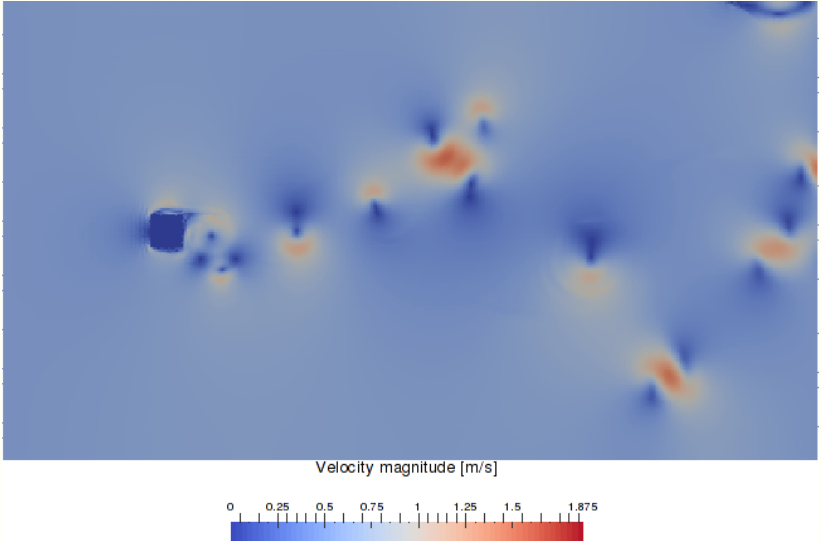
libmpdata++: immersed b.m. (Maciej Waruszewski)

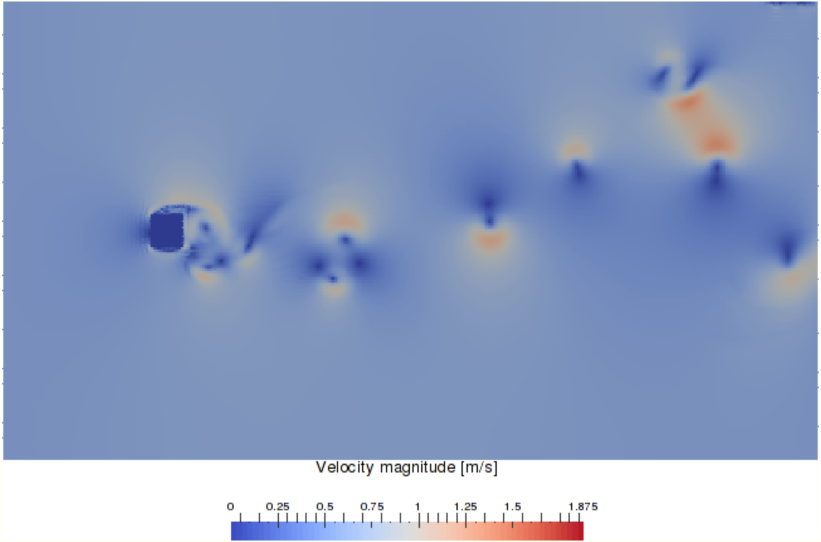


libmpdata++: immersed b.m. (Maciej Waruszewski)

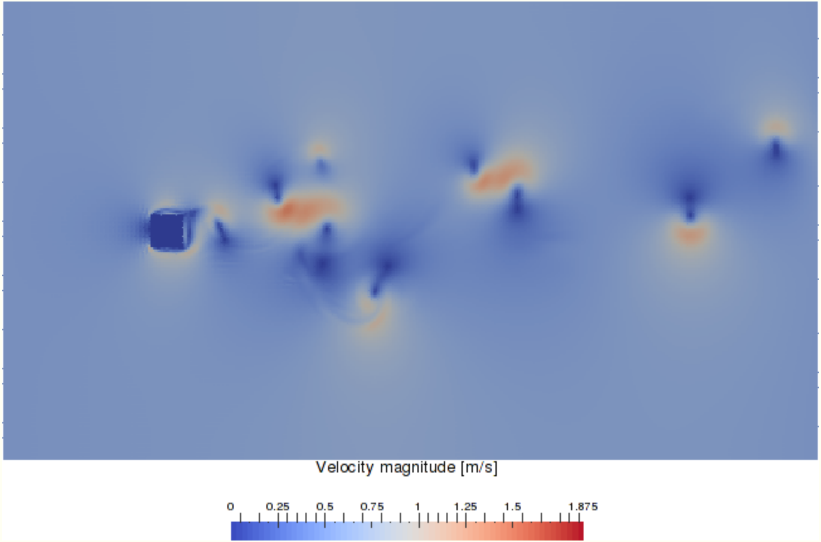


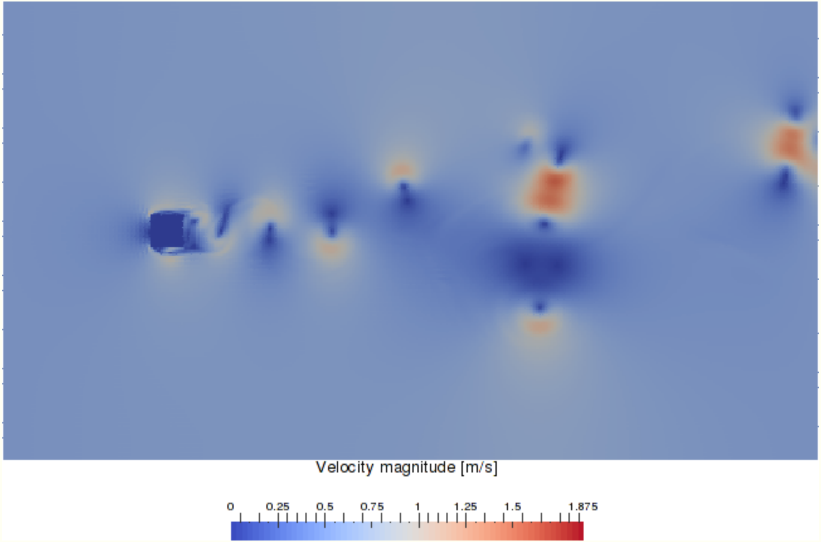
libmpdata++: immersed b.m. (Maciej Waruszewski)



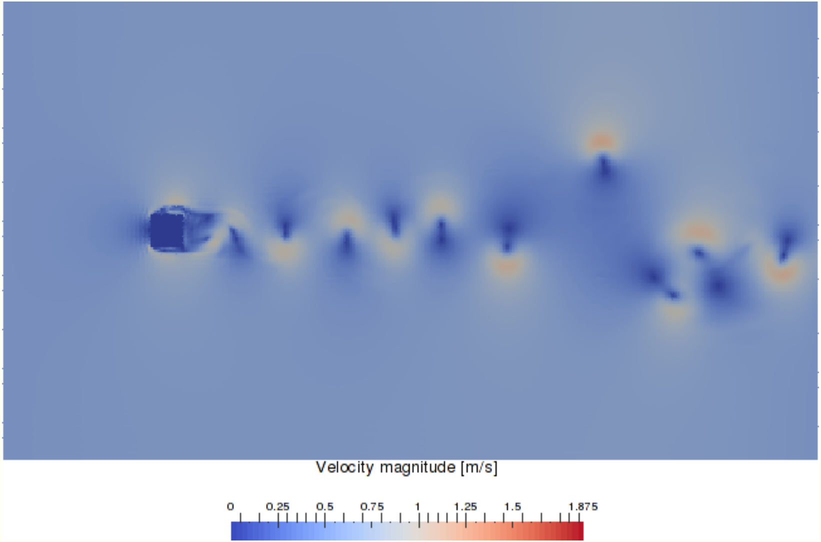

Velocity magnitude [m/s]
0
0.25
0.5
0.75
1
1.25
1.5
1.875

libmpdata++: immersed b.m. (Maciej Waruszewski)

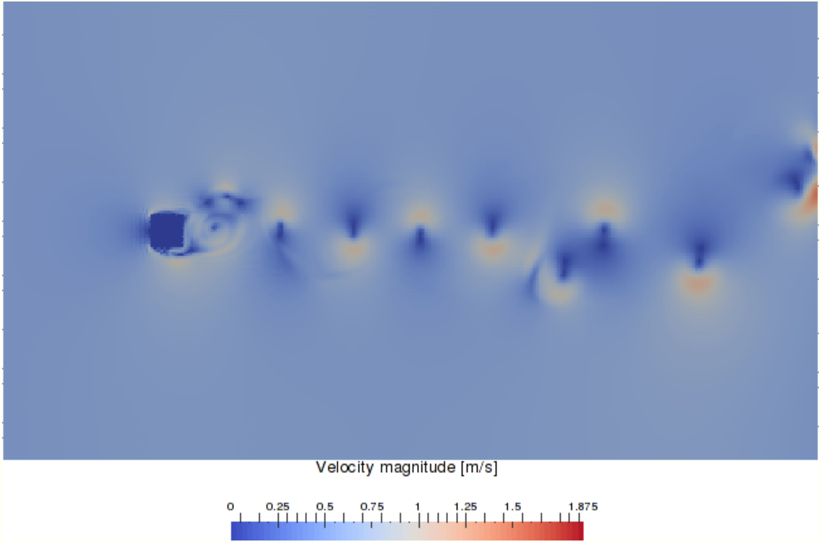



Velocity magnitude [m/s]
0
0.25
0.5
0.75
1
1.25
1.5
1.875

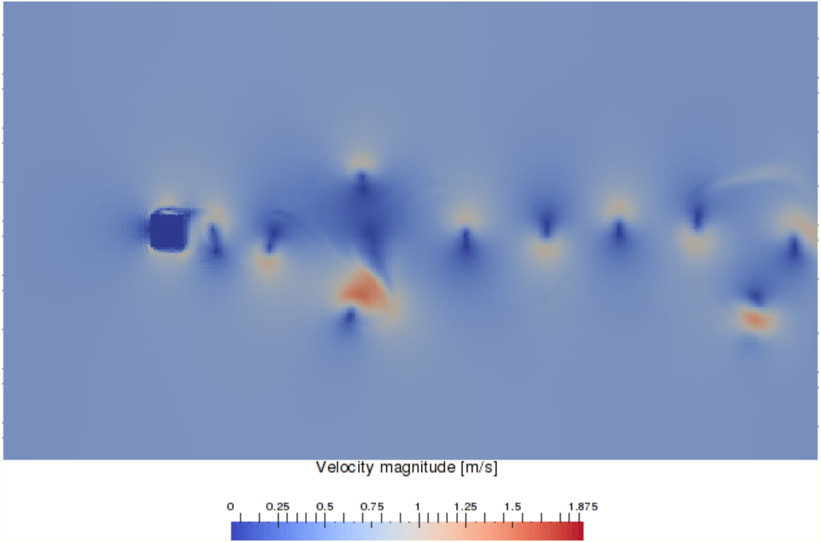
libmpdata++: immersed b.m. (Maciej Waruszewski)



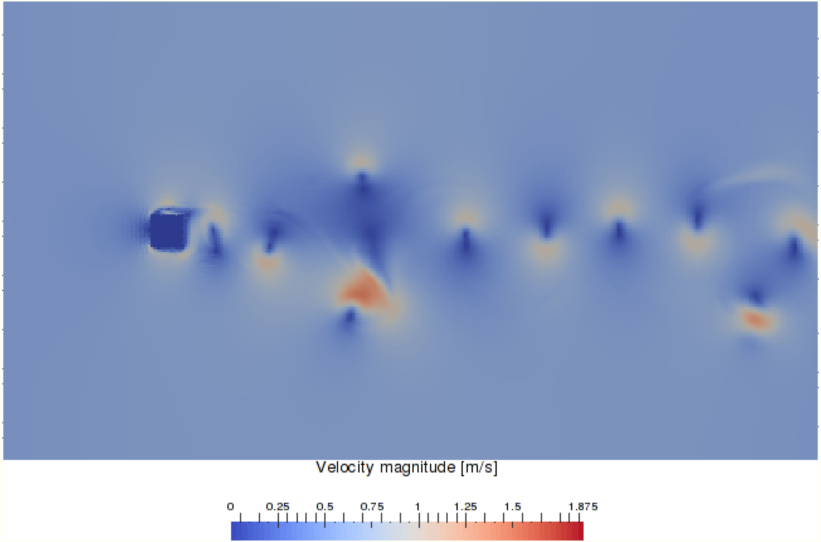
libmpdata++: immersed b.m. (Maciej Waruszewski)



libmpdata++: immersed b.m. (Maciej Waruszewski)



libmpdata++: immersed b.m. (Maciej Waruszewski)



libmpdata++: 3D (I)LES (Dziekan et al. 2019)

Geosci. Model Dev., 12, 2587–2606, 2019
https://doi.org/10.5194/gmd-12-2587-2019

GMD | Articles | Volume 12, issue 6

Article | Assets | Peer review | Metrics | Related articles

Model description paper

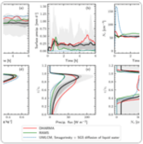
01 Jul 2019

University of Warsaw Lagrangian Cloud Model (UWLCM) 1.0: a modern large-eddy simulation tool for warm cloud modeling with Lagrangian microphysics

Piotr Dziekan, Maciej Waruszewski, and Hanna Pawlowska
Institute of Geophysics, Faculty of Physics, University of Warsaw, Warsaw, Poland

Correspondence: Piotr Dziekan (pdziekan@fuw.edu.pl)

Received: 07 Nov 2018 – Discussion started: 04 Feb 2019 – Revised: 03 Jun 2019 – Accepted: 07 Jun 2019 – Published: 01 Jul 2019



`https://www.youtube.com/watch?v=BEidkhpw-MA`

libmpdata++: summary & some technicalities

- free and open-souce, public repo: github.com/igfuw/libmpdataxx
- automated testsuite, continuous integration (Travis)
- reusable – API documented in the paper; out-of-tree setups
- comprehensive set of MPDATA opts (incl. FCT, infinite-gauge, ...)
- 1D, 2D & 3D integration; optional coordinate transformation
- four types of solvers:
  - adv (homogeneous advection)
  - adv+rhs (+ right-hand-side terms)
  - adv+rhs+vip (+ prognosed velocity)
  - adv+rhs+vip+prs (+ elliptic pressure solver)
- implemented using Blitz++ (no loops, expression templates)
- built-in HDF5/XDMF output
- parallelisation: threads + MPI
- separation of concerns (numerics / boundary cond. / io / concurrency)
- compact C++11 code (O(10) kLOC)

libmpdata++: summary & some technicalities

- free and open-souce, public repo: github.com/igfuw/libmpdataxx
- automated testsuite, continuous integration (Travis)
- reusable – API documented in the paper; out-of-tree setups
- comprehensive set of MPDATA opts (incl. FCT, infinite-gauge, . . .)
- 1D, 2D & 3D integration; optional coordinate transformation
- four types of solvers:
  - adv (homogeneous advection)
  - adv+rhs (+ right-hand-side terms)
  - adv+rhs+vip (+ prognosed velocity)
  - adv+rhs+vip+prs (+ elliptic pressure solver)
- implemented using Blitz++ (no loops, expression templates)
- built-in HDF5/XDMF output
- parallelisation: threads + MPI
- separation of concerns (numerics / boundary cond. / io / concurrency)
- compact C++11 code (O(10) kLOC)

libmpdata++: summary & some technicalities

- free and open-souce, public repo: github.com/igfuw/libmpdataxx
- automated testsuite, continuous integration (Travis)
- reusable – API documented in the paper; out-of-tree setups
- comprehensive set of MPDATA opts (incl. FCT, infinite-gauge, . . .)
- 1D, 2D & 3D integration; optional coordinate transformation
- four types of solvers:
  - adv (homogeneous advection)
  - adv+rhs (+ right-hand-side terms)
  - adv+rhs+vip (+ prognosed velocity)
  - adv+rhs+vip+prs (+ elliptic pressure solver)
- implemented using Blitz++ (no loops, expression templates)
- built-in HDF5/XDMF output
- parallelisation: threads + MPI
- separation of concerns (numerics / boundary cond. / io / concurrency)
- compact C++11 code (O(10) kLOC)

libmpdata++: summary & some technicalities

- free and open-souce, public repo: github.com/igfuw/libmpdataxx
- automated testsuite, continuous integration (Travis)
- reusable – API documented in the paper; out-of-tree setups
- comprehensive set of MPDATA opts (incl. FCT, infinite-gauge, ...)
- 1D, 2D & 3D integration; optional coordinate transformation
- four types of solvers:
  - adv (homogeneous advection)
  - adv+rhs (+ right-hand-side terms)
  - adv+rhs+vip (+ prognosed velocity)
  - adv+rhs+vip+prs (+ elliptic pressure solver)
- implemented using Blitz++ (no loops, expression templates)
- built-in HDF5/XDMF output
- parallelisation: threads + MPI
- separation of concerns (numerics / boundary cond. / io / concurrency)
- compact C++11 code (O(10) kLOC)

libmpdata++: summary & some technicalities

- free and open-souce, public repo: github.com/igfuw/libmpdataxx
- automated testsuite, continuous integration (Travis)
- reusable – API documented in the paper; out-of-tree setups
- comprehensive set of MPDATA opts (incl. FCT, infinite-gauge, ...)
- 1D, 2D & 3D integration; optional coordinate transformation
- four types of solvers:
  - adv (homogeneous advection)
  - adv+rhs (+ right-hand-side terms)
  - adv+rhs+vip (+ prognosed velocity)
  - adv+rhs+vip+prs (+ elliptic pressure solver)
- implemented using Blitz++ (no loops, expression templates)
- built-in HDF5/XDMF output
- parallelisation: threads + MPI
- separation of concerns (numerics / boundary cond. / io / concurrency)
- compact C++11 code (O(10) kLOC)

libmpdata++: summary & some technicalities

- free and open-souce, public repo: github.com/igfuw/libmpdataxx
- automated testsuite, continuous integration (Travis)
- reusable – API documented in the paper; out-of-tree setups
- comprehensive set of MPDATA opts (incl. FCT, infinite-gauge, ...)
- 1D, 2D & 3D integration; optional coordinate transformation
- four types of solvers:
  - adv (homogeneous advection)
  - adv+rhs (+ right-hand-side terms)
  - adv+rhs+vip (+ prognosed velocity)
  - adv+rhs+vip+prs (+ elliptic pressure solver)
- implemented using Blitz++ (no loops, expression templates)
- built-in HDF5/XDMF output
- parallelisation: threads + MPI
- separation of concerns (numerics / boundary cond. / io / concurrency)
- compact C++11 code (O(10) kLOC)

libmpdata++: summary & some technicalities

- free and open-souce, public repo: github.com/igfuw/libmpdataxx
- automated testsuite, continuous integration (Travis)
- reusable – API documented in the paper; out-of-tree setups
- comprehensive set of MPDATA opts (incl. FCT, infinite-gauge, ...)
- 1D, 2D & 3D integration; optional coordinate transformation
- four types of solvers:
  - adv (homogeneous advection)
  - adv+rhs (+ right-hand-side terms)
  - adv+rhs+vip (+ prognosed velocity)
  - adv+rhs+vip+prs (+ elliptic pressure solver)
- implemented using Blitz++ (no loops, expression templates)
- built-in HDF5/XDMF output
- parallelisation: threads + MPI
- separation of concerns (numerics / boundary cond. / io / concurrency)
- compact C++11 code (O(10) kLOC)

libmpdata++: summary & some technicalities

- free and open-souce, public repo: github.com/igfuw/libmpdataxx
- automated testsuite, continuous integration (Travis)
- reusable – API documented in the paper; out-of-tree setups
- comprehensive set of MPDATA opts (incl. FCT, infinite-gauge, ...)
- 1D, 2D & 3D integration; optional coordinate transformation
- four types of solvers:
  - adv (homogeneous advection)
  - adv+rhs (+ right-hand-side terms)
  - adv+rhs+vip (+ prognosed velocity)
  - adv+rhs+vip+prs (+ elliptic pressure solver)
- implemented using Blitz++ (no loops, expression templates)
- built-in HDF5/XDMF output
- parallelisation: threads + MPI
- separation of concerns (numerics / boundary cond. / io / concurrency)
- compact C++11 code (O(10) kLOC)

libmpdata++: summary & some technicalities

- free and open-souce, public repo: github.com/igfuw/libmpdataxx
- automated testsuite, continuous integration (Travis)
- reusable – API documented in the paper; out-of-tree setups
- comprehensive set of MPDATA opts (incl. FCT, infinite-gauge, ...)
- 1D, 2D & 3D integration; optional coordinate transformation
- four types of solvers:
  - adv (homogeneous advection)
  - adv+rhs (+ right-hand-side terms)
  - adv+rhs+vip (+ prognosed velocity)
  - adv+rhs+vip+prs (+ elliptic pressure solver)
- implemented using Blitz++ (no loops, expression templates)
- built-in HDF5/XDMF output
- parallelisation: threads + MPI
- separation of concerns (numerics / boundary cond. / io / concurrency)
- compact C++11 code (O(10) kLOC)

libmpdata++: summary & some technicalities

- free and open-souce, public repo: github.com/igfuw/libmpdataxx
- automated testsuite, continuous integration (Travis)
- reusable – API documented in the paper; out-of-tree setups
- comprehensive set of MPDATA opts (incl. FCT, infinite-gauge, ...)
- 1D, 2D & 3D integration; optional coordinate transformation
- four types of solvers:
  - adv (homogeneous advection)
  - adv+rhs (+ right-hand-side terms)
  - adv+rhs+vip (+ prognosed velocity)
  - adv+rhs+vip+prs (+ elliptic pressure solver)
- implemented using Blitz++ (no loops, expression templates)
- built-in HDF5/XDMF output
- parallelisation: threads + MPI
- separation of concerns (numerics / boundary cond. / io / concurrency)
- compact C++11 code (O(10) kLOC)

libmpdata++: summary & some technicalities

- free and open-souce, public repo: github.com/igfuw/libmpdataxx
- automated testsuite, continuous integration (Travis)
- reusable – API documented in the paper; out-of-tree setups
- comprehensive set of MPDATA opts (incl. FCT, infinite-gauge, ...)
- 1D, 2D & 3D integration; optional coordinate transformation
- four types of solvers:
  - adv (homogeneous advection)
  - adv+rhs (+ right-hand-side terms)
  - adv+rhs+vip (+ prognosed velocity)
  - adv+rhs+vip+prs (+ elliptic pressure solver)
- implemented using Blitz++ (no loops, expression templates)
- built-in HDF5/XDMF output
- parallelisation: threads + MPI
- separation of concerns (numerics / boundary cond. / io / concurrency)
- compact C++11 code (O(10) kLOC)

libmpdata++: summary & some technicalities

- free and open-souce, public repo: github.com/igfuw/libmpdataxx
- automated testsuite, continuous integration (Travis)
- reusable – API documented in the paper; out-of-tree setups
- comprehensive set of MPDATA opts (incl. FCT, infinite-gauge, . . .)
- 1D, 2D & 3D integration; optional coordinate transformation
- four types of solvers:
  - adv (homogeneous advection)
  - adv+rhs (+ right-hand-side terms)
  - adv+rhs+vip (+ prognosed velocity)
  - adv+rhs+vip+prs (+ elliptic pressure solver)
- implemented using Blitz++ (no loops, expression templates)
- built-in HDF5/XDMF output
- parallelisation: threads + MPI
- separation of concerns (numerics / boundary cond. / io / concurrency)
- compact C++11 code (O(10) kLOC)

libmpdata++: summary & some technicalities

- free and open-souce, public repo: github.com/igfuw/libmpdataxx
- automated testsuite, continuous integration (Travis)
- reusable – API documented in the paper; out-of-tree setups
- comprehensive set of MPDATA opts (incl. FCT, infinite-gauge, ...)
- 1D, 2D & 3D integration; optional coordinate transformation
- four types of solvers:
  - adv (homogeneous advection)
  - adv+rhs (+ right-hand-side terms)
  - adv+rhs+vip (+ prognosed velocity)
  - adv+rhs+vip+prs (+ elliptic pressure solver)
- implemented using Blitz++ (no loops, expression templates)
- built-in HDF5/XDMF output
- parallelisation: threads + MPI
- separation of concerns (numerics / boundary cond. / io / concurrency)
- compact C++11 code (O(10) kLOC)

libmpdata++: summary & some technicalities

- free and open-souce, public repo: github.com/igfuw/libmpdataxx
- automated testsuite, continuous integration (Travis)
- reusable – API documented in the paper; out-of-tree setups
- comprehensive set of MPDATA opts (incl. FCT, infinite-gauge, ...)
- 1D, 2D & 3D integration; optional coordinate transformation
- four types of solvers:
  - adv (homogeneous advection)
  - adv+rhs (+ right-hand-side terms)
  - adv+rhs+vip (+ prognosed velocity)
  - adv+rhs+vip+prs (+ elliptic pressure solver)
- implemented using Blitz++ (no loops, expression templates)
- built-in HDF5/XDMF output
- parallelisation: threads + MPI
- separation of concerns (numerics / boundary cond. / io / concurrency)
- compact C++11 code (O(10) kLOC)

libmpdata++: summary & some technicalities

- free and open-souce, public repo: github.com/igfuw/libmpdataxx
- automated testsuite, continuous integration (Travis)
- reusable – API documented in the paper; out-of-tree setups
- comprehensive set of MPDATA opts (incl. FCT, infinite-gauge, ...)
- 1D, 2D & 3D integration; optional coordinate transformation
- four types of solvers:
  - adv (homogeneous advection)
  - adv+rhs (+ right-hand-side terms)
  - adv+rhs+vip (+ prognosed velocity)
  - adv+rhs+vip+prs (+ elliptic pressure solver)
- implemented using Blitz++ (no loops, expression templates)
- built-in HDF5/XDMF output
- parallelisation: threads + MPI
- separation of concerns (numerics / boundary cond. / io / concurrency)
- compact C++11 code (O(10) kLOC)

libmpdata++: summary & some technicalities

- free and open-souce, public repo: github.com/igfuw/libmpdataxx
- automated testsuite, continuous integration (Travis)
- reusable – API documented in the paper; out-of-tree setups
- comprehensive set of MPDATA opts (incl. FCT, infinite-gauge, ...)
- 1D, 2D & 3D integration; optional coordinate transformation
- four types of solvers:
  - adv (homogeneous advection)
  - adv+rhs (+ right-hand-side terms)
  - adv+rhs+vip (+ prognosed velocity)
  - adv+rhs+vip+prs (+ elliptic pressure solver)
- implemented using Blitz++ (no loops, expression templates)
- built-in HDF5/XDMF output
- parallelisation: threads + MPI
- separation of concerns (numerics / boundary cond. / io / concurrency)
- compact C++11 code (O(10) kLOC)

libmpdata++: documented applications

libmpdata++: documented applications

- Jarecka et al. 2015 (J. Comp. Phys.):
  shallow water eqs, 3D liquid drop spreading under gravity

libmpdata++: documented applications

- Jarecka et al. 2015 (J. Comp. Phys.):
  shallow water eqs, 3D liquid drop spreading under gravity

- Arabas, Jaruga et al. 2015 (Geosci. Model. Dev.);
  Jaruga & Pawlowska 2018 (″):
  particle-based/Monte-Carlo simulations of clouds

libmpdata++: documented applications

- Jarecka et al. 2015 (J. Comp. Phys.):
  shallow water eqs, 3D liquid drop spreading under gravity

- Arabas, Jaruga et al. 2015 (Geosci. Model. Dev.);
  Jaruga & Pawlowska 2018 (″):
  particle-based/Monte-Carlo simulations of clouds

- Waruszewski et al. 2018 (J. Comp. Phys.):
  MPDATA ext. for 3rd-order accuracy for variable flows

libmpdata++: documented applications

- Jarecka et al. 2015 (J. Comp. Phys.):
  shallow water eqs, 3D liquid drop spreading under gravity
- Arabas, Jaruga et al. 2015 (Geosci. Model. Dev.);
  Jaruga & Pawlowska 2018 (″):
  particle-based/Monte-Carlo simulations of clouds
- Waruszewski et al. 2018 (J. Comp. Phys.):
  MPDATA ext. for 3rd-order accuracy for variable flows
- Dziekan et al. 2019 (Geosci. Model Dev.):
  3D LES for atm. boundary layer simulations

libmpdata++: documented applications

- Jarecka et al. 2015 (J. Comp. Phys.):
  shallow water eqs, 3D liquid drop spreading under gravity
- Arabas, Jaruga et al. 2015 (Geosci. Model. Dev.);
  Jaruga & Pawlowska 2018 (″):
  particle-based/Monte-Carlo simulations of clouds
- Waruszewski et al. 2018 (J. Comp. Phys.):
  MPDATA ext. for 3rd-order accuracy for variable flows
- Dziekan et al. 2019 (Geosci. Model Dev.):
  3D LES for atm. boundary layer simulations
- Arabas & Farhat 2019:
  Derivative pricing as a transport problem

MPDATA meets Black-Scholes

with Ahmad Farhat (HSBC)

Black-Scholes equation and pricing formulæ

Black-Scholes equation and pricing formulæ

- asset price SDE: $$dS = S(\mu dt + \sigma dw)$$

Black-Scholes equation and pricing formulæ

- asset price SDE: $dS = S(\mu dt + \sigma dw)$
- derivative price: $f(S, t)$

Black-Scholes equation and pricing formulæ

- asset price SDE: $dS = S(\mu dt + \sigma dw)$
- derivative price: $f(S, t)$
- riskless portfolio (asset + option): $\Pi = -f + \Delta_t S$

Black-Scholes equation and pricing formulæ

- asset price SDE: $dS = S(\mu dt + \sigma dw)$
- derivative price: $f(S, t)$
- riskless portfolio (asset + option): $\Pi = -f + \Delta_t S$
- no arbitrage (riskless interest rate): $d\Pi = \Pi r dt$

Black-Scholes equation and pricing formulæ

- asset price SDE: $dS = S(\mu dt + \sigma dw)$
- derivative price: $f(S, t)$
- riskless portfolio (asset + option): $\Pi = -f + \Delta_t S$
- no arbitrage (riskless interest rate): $d\Pi = \Pi r dt$
- Itô's lemma: SDE $\rightsquigarrow$ PDE

Black-Scholes equation and pricing formulæ

- asset price SDE: $dS = S(\mu dt + \sigma dw)$
- derivative price: $f(S, t)$
- riskless portfolio (asset + option): $\Pi = -f + \Delta_t S$
- no arbitrage (riskless interest rate): $d\Pi = \Pi r dt$
- Itô's lemma: SDE $\rightsquigarrow$ PDE

$$\frac{\partial f}{\partial t} + rS\frac{\partial f}{\partial S} + \frac{\sigma^2}{2}S^2\frac{\partial^2 f}{\partial S^2} - rf = 0$$

Black-Scholes equation and pricing formulæ

- asset price SDE: $dS = S(\mu dt + \sigma dw)$
- derivative price: $f(S, t)$
- riskless portfolio (asset + option): $\Pi = -f + \Delta_t S$
- no arbitrage (riskless interest rate): $d\Pi = \Pi r dt$
- Itô's lemma: SDE $\rightsquigarrow$ PDE

$$\frac{\partial f}{\partial t} + rS\frac{\partial f}{\partial S} + \frac{\sigma^2}{2}S^2\frac{\partial^2 f}{\partial S^2} - rf = 0$$

- terminal value prob., analytic solutions for vanilla options

Black-Scholes equation and pricing formulæ

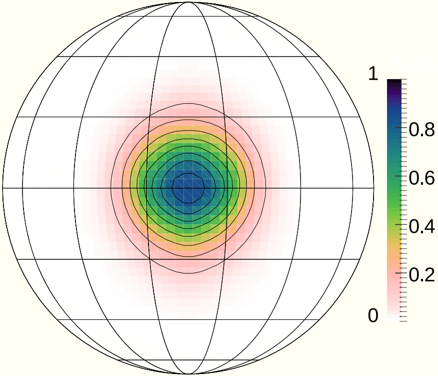
- asset price SDE: $dS = S(\mu dt + \sigma dw)$
- derivative price: $f(S,t)$
- riskless portfolio (asset + option): $\Pi = -f + \Delta_t S$
- no arbitrage (riskless interest rate): $d\Pi = \Pi r dt$
- Itô's lemma: SDE $\rightsquigarrow$ PDE

$$\frac{\partial f}{\partial t} + rS\frac{\partial f}{\partial S} + \frac{\sigma^2}{2}S^2\frac{\partial^2 f}{\partial S^2} - rf = 0$$

- terminal value prob., analytic solutions for vanilla options


SVERIGES RIKSBANK TILL ALFRED NOBELS MINNE 1968

?


1
0.8
0.6
0.4
0.2
0

Black-Scholes $\rightsquigarrow$ ("advection-only") transport problem

$$\frac{\partial f}{\partial t} + rS\frac{\partial f}{\partial S} + \frac{\sigma^2}{2}S^2\frac{\partial^2 f}{\partial S^2} - rf = 0$$

Black-Scholes $\rightsquigarrow$ ("advection-only") transport problem

$$\frac{\partial f}{\partial t} + rS\frac{\partial f}{\partial S} + \frac{\sigma^2}{2}S^2\frac{\partial^2 f}{\partial S^2} - rf = 0$$

$$\xrightarrow{x = \ln S} \frac{\partial f}{\partial t} + \underbrace{\left(r - {\sigma^2}/{2}\right)}_{u}\frac{\partial f}{\partial x} + \underbrace{{\sigma^2}/{2}}_{-\nu}\frac{\partial^2 f}{\partial x^2} - rf = 0$$

Black-Scholes $\rightsquigarrow$ ("advection-only") transport problem

$$\frac{\partial f}{\partial t} + rS\frac{\partial f}{\partial S} + \frac{\sigma^2}{2}S^2\frac{\partial^2 f}{\partial S^2} - rf = 0$$

$$\xrightarrow{x = \ln S} \quad \frac{\partial f}{\partial t} + \underbrace{\left(r - {\sigma^2}/{2}\right)}_{u}\frac{\partial f}{\partial x} + \underbrace{{\sigma^2}/{2}}_{-\nu}\frac{\partial^2 f}{\partial x^2} - rf = 0$$

$$\xrightarrow{\psi = e^{-rt} f} \quad \frac{\partial \psi}{\partial t} + u\frac{\partial \psi}{\partial x} - \nu\frac{\partial^2 \psi}{\partial x^2} = 0$$

Black-Scholes $\rightsquigarrow$ ("advection-only") transport problem

$$\frac{\partial f}{\partial t} + rS\frac{\partial f}{\partial S} + \frac{\sigma^2}{2}S^2\frac{\partial^2 f}{\partial S^2} - rf = 0$$

$$\xrightarrow{x = \ln S} \frac{\partial f}{\partial t} + \underbrace{\left(r - \sigma^2/2\right)}_{u}\frac{\partial f}{\partial x} + \underbrace{\sigma^2/2}_{-\nu}\frac{\partial^2 f}{\partial x^2} - rf = 0$$

$$\xrightarrow{\psi = e^{-rt}f} \frac{\partial \psi}{\partial t} + u\frac{\partial \psi}{\partial x} - \nu\frac{\partial^2 \psi}{\partial x^2} = 0$$

$$\longrightarrow \frac{\partial \psi}{\partial t} + \frac{\partial}{\partial x}\left[\left(u - \frac{\nu}{\psi}\frac{\partial \psi}{\partial x}\right)\psi\right] = 0$$

Black-Scholes $\rightsquigarrow$ ("advection-only") transport problem

$$\frac{\partial f}{\partial t} + rS\frac{\partial f}{\partial S} + \frac{\sigma^2}{2}S^2\frac{\partial^2 f}{\partial S^2} - rf = 0$$

$$\xrightarrow{x = \ln S} \quad \frac{\partial f}{\partial t} + \underbrace{\left(r - \sigma^2/2\right)}_{u}\frac{\partial f}{\partial x} + \underbrace{\sigma^2/2}_{-\nu}\frac{\partial^2 f}{\partial x^2} - rf = 0$$

$$\xrightarrow{\psi = e^{-rt} f} \quad \frac{\partial \psi}{\partial t} + u\frac{\partial \psi}{\partial x} - \nu\frac{\partial^2 \psi}{\partial x^2} = 0$$

$$\longrightarrow \quad \frac{\partial \psi}{\partial t} + \frac{\partial}{\partial x}\left[\left(u - \frac{\nu}{\psi}\frac{\partial \psi}{\partial x}\right)\psi\right] = 0$$

re last step: Smolarkiewicz and Clark (1986, JCP), Sousa (2009, IJNMF), Smolarkiewicz and Szmelter (2005, JCP), Cristiani (2015, JCSMD)

same trick!

MPDATA in a nutshell (Smolarkiewicz 1983, 1984, ...)

transport PDE: $\frac{\partial \psi}{\partial t} + \frac{\partial}{\partial x}(v\psi) = 0$

$$\psi_i^{n+1} = \psi_i^n - \left[F(\psi_i^n, \psi_{i+1}^n, \mathcal{C}_{i+1/2}) - F(\psi_{i-1}^n, \psi_i^n, \mathcal{C}_{i-1/2})\right]$$
$$F(\psi_L, \psi_R, \mathcal{C}) = \max(\mathcal{C}, 0)\cdot\psi_L + \min(\mathcal{C}, 0)\cdot\psi_R$$
$$\mathcal{C} = v\Delta t/\Delta x$$

upwind

modified eq.: $\frac{\partial \psi}{\partial t} + \frac{\partial}{\partial x}(v\psi) + \underbrace{K\frac{\partial^2\psi}{\partial x^2}}_{\text{numerical diffusion}} + \ldots = 0$ MEA

$$\frac{\partial \psi}{\partial t} + \frac{\partial}{\partial x}(v\psi) + \frac{\partial}{\partial x}\underbrace{\left[\left(-\frac{K}{\psi}\frac{\partial\psi}{\partial x}\right)\psi\right]}_{\text{antidiffusive flux}} = 0$$

Black-Scholes ⇝ ("advection-only") transport problem

$$\frac{\partial f}{\partial t} + rS\frac{\partial f}{\partial S} + \frac{\sigma^2}{2}S^2\frac{\partial^2 f}{\partial S^2} - rf = 0$$

$$\xrightarrow{x=\ln S}\ \frac{\partial f}{\partial t} + \underbrace{\left(r - \sigma^2/2\right)}_{u}\frac{\partial f}{\partial x} + \underbrace{\sigma^2/2}_{-\nu}\frac{\partial^2 f}{\partial x^2} - rf = 0$$

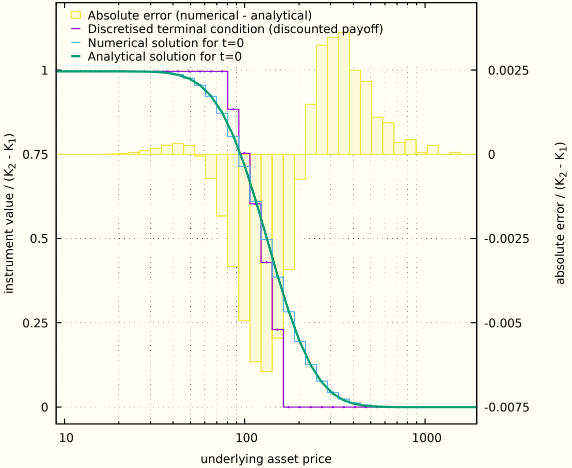
$$\xrightarrow{\psi = e^{-rt}f}\ \frac{\partial \psi}{\partial t} + u\frac{\partial \psi}{\partial x} - \nu\frac{\partial^2\psi}{\partial x^2} = 0$$

$$\frac{\partial \psi}{\partial t} + \frac{\partial}{\partial x}\left[\left(u - \frac{\nu}{\psi}\frac{\partial\psi}{\partial x}\right)\psi\right] = 0$$

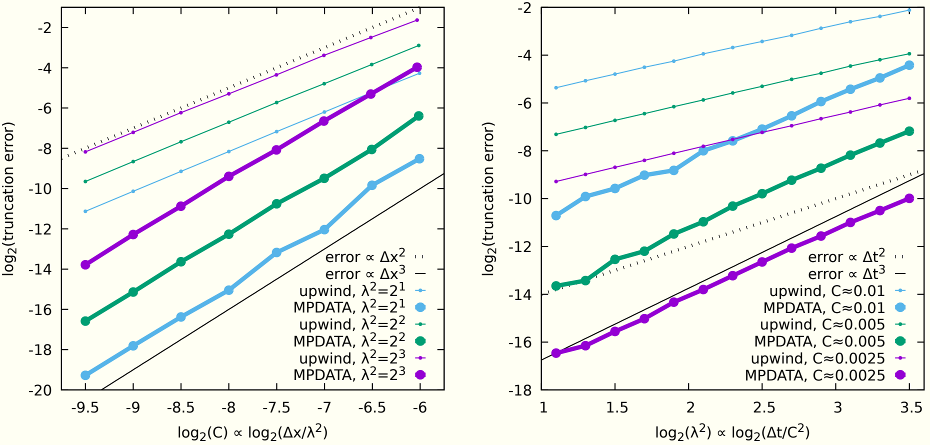
MPDATA meets Black-Scholes: test case

- terminal value problem
- payoff function: corridor
- truncation error est. (ψ_a: B-S formula):

$$E = \sqrt{\sum_{i=1}^{n_x} \left[\psi_n(x_i) - \psi_a(x_i)\right]^2 / (n_x \cdot n_t)}\Bigg|_{t=0}$$

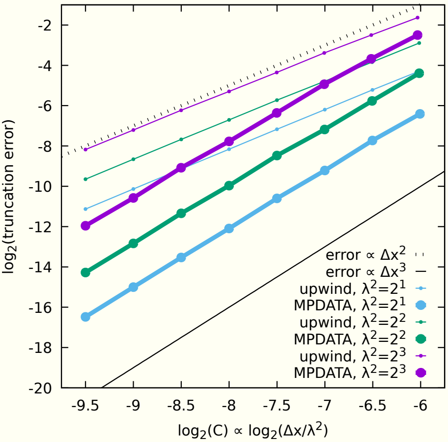


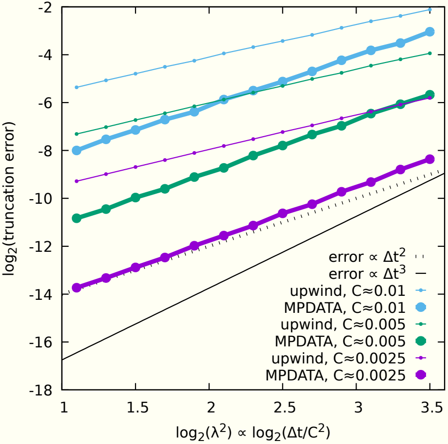
MPDATA meets Black-Scholes: convergence analysis



MPDATA variant: 2 iterations
+ infinite gauge + FCT + divergent flow + third-order terms

MPDATA meets Black-Scholes: convergence analysis





MPDATA variant: 2 iterations

MPDATA & diffusional growth

with Michael Olesik (Jagiellonian) and Simon Unterstraßer (DLR)

what triggered the study

Morrison et al. 2018 (JAS)

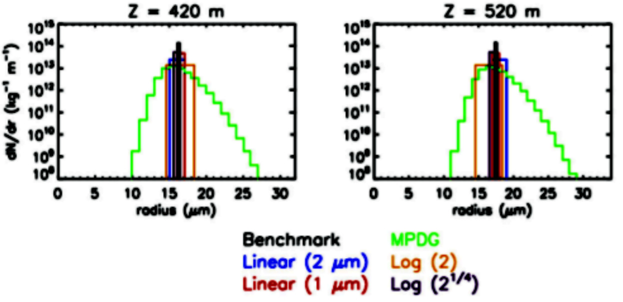


FIG. 7. Drop size distributions at various heights z from the Lagrangian microphysical benchmark (black) and the bin model simulations (colored lines) for the parcel test with a bulk drop number mixing ratio of 50 mg^{-1}. Different colored lines illustrate results using different bin mass grid configurations and growth methods, as listed in Table 1.

"... MPDG growth produces significant numerical diffusion and DSD broadening relative to the Lagrangian benchmark and all of the TH-MOM configurations"

more on MPDATA for condensational growth

Smolarkiewicz 1984 (sec. 5.1 "Divergent Flow Field")

"On the other hand when the velocity is strongly convergent, application of Eq. (38) to the problem of the evolution of the droplet size distribution due to the evaporation-condensation process improves the results (William Hall, personal communication)"

Tsang & Korgaonkar 1987

"novel numerical scheme is devised for the solution of evaporation of aerosol clouds. This scheme combines the salient features of the Galerkin Finite Element Method and the positive definite method of Smolarkiewicz"

more on MPDATA for condensational growth

Tsang and Rao 1988

"Smolarkiewicz method provides a much narrower size distribution than upwind differencing and the sectional method, its prediction of mass concentration is worse than upwind differencing and the sectional method"

Williams & Loyalka 1991

"Smolarkiewicz studied the problem of advection in fluid flows but his method applies directly to the problem of aerosol growth"

Kostoglou and Karabelas 1995

"A finite difference type of technique proposed by Smolarkiewicz (1983) for fluid flows is not compared with other methods here, even though it appears to reduce errors in size computations"

test case: East 1957, Fig. 3

66 T. W. R. EAST

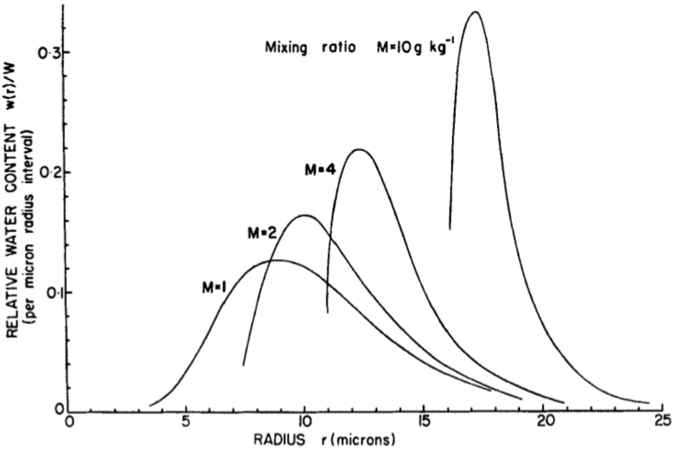


Figure 3. Modification of water-content distribution by condensation. The distribution at $M = 1$ is assumed to be the same as in fair-weather cloud : the other curves show the distribution after water is condensed on to it rapidly. All are normalised to have equal area : the peak water content $w(r)_{max}$ actually increased 26 times from $M = 1$ to 10 g/kg.

test case: setup & analytic solution

test case: setup & analytic solution

initial spectrum (East & Marshall 1954)

$n_0(r) = \mathrm{lognormal}(r)/r$

test case: setup & analytic solution

initial spectrum (East & Marshall 1954)

$n_0(r) = \text{lognormal}(r)/r$

drop growth (i.e., velocity field)

$dr/dt = \xi(S-1)/r \qquad \rightsquigarrow$ divergent

test case: setup & analytic solution

initial spectrum (East & Marshall 1954)

$n_0(r) = \text{lognormal}(r)/r$

drop growth (i.e., velocity field)

$dr/dt = \xi(S-1)/r \qquad \rightsquigarrow$ divergent

analytic solution (Rogers & Yau)

$r'(r,t) = \sqrt{r^2 - 2\xi(S-1)t}$
$n(r,t) = n_0(r') \cdot r/r'$

test case: setup & analytic solution

initial spectrum (East & Marshall 1954)

$n_0(r) = \text{lognormal}(r)/r$

drop growth (i.e., velocity field)

$dr/dt = \xi(S-1)/r \qquad \rightsquigarrow$ divergent

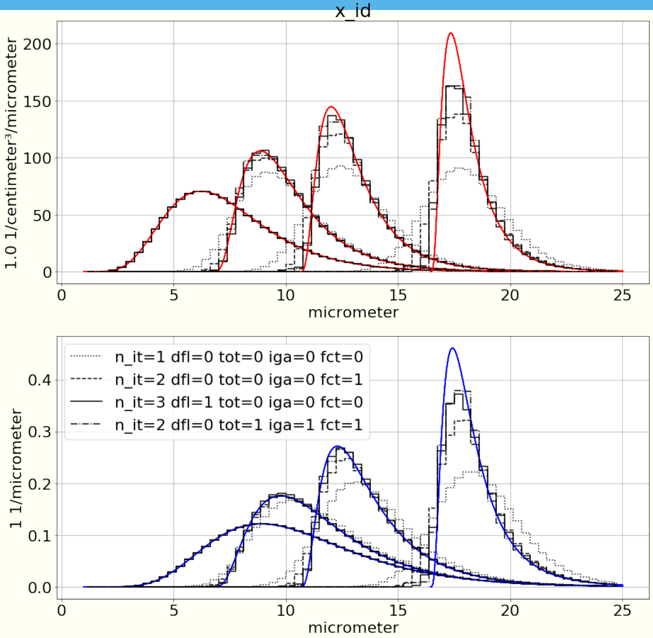
analytic solution (Rogers & Yau)

$r'(r,t) = \sqrt{r^2 - 2\xi(S-1)t}$
$n(r,t) = n_0(r') \cdot r/r'$

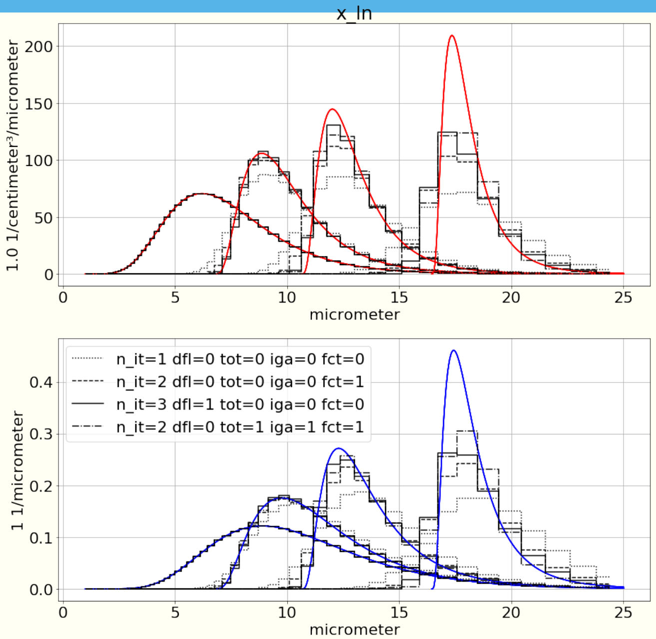
integration parameters

$\Delta t = 0.5s$
$r \in (1...25)\mu m$
$nx = 64$ (linear, log-linear or r^2-linear)
nt: two-, four- & tenfold increase in water content

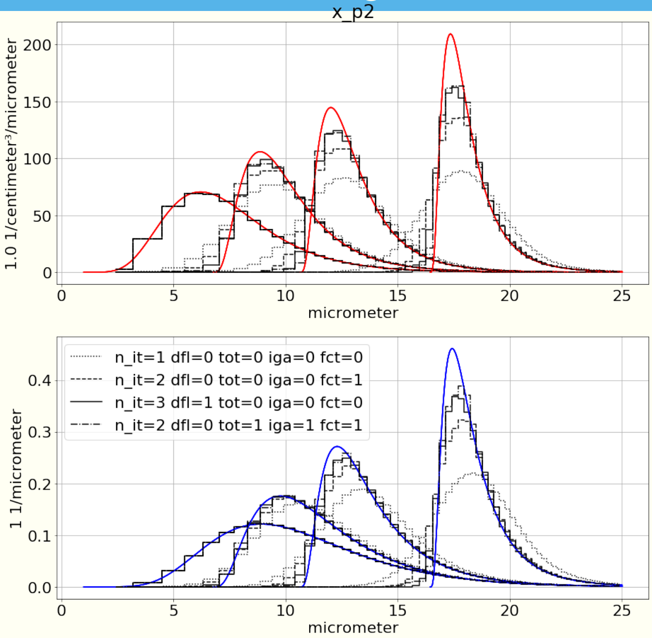
test case: results with linear grid



test case: results with log-linear grid



test case: results with r^2-linear grid



MPDATA variants (structured grid, homogeneous prob.)

- basic (+iterations): Smolarkiewicz 1983

MPDATA variants (structured grid, homogeneous prob.)

- basic (+iterations): Smolarkiewicz 1983
- coordinate transformation: Smolarkiewicz and Clark 1986, Smolarkiewicz and Margolin 1993

MPDATA variants (structured grid, homogeneous prob.)

- basic (+iterations): Smolarkiewicz 1983
- coordinate transformation: Smolarkiewicz and Clark 1986, Smolarkiewicz and Margolin 1993
- divergent flow corrections: Smolarkiewicz 1984

MPDATA variants (structured grid, homogeneous prob.)

- basic (+iterations): Smolarkiewicz 1983
- coordinate transformation: Smolarkiewicz and Clark 1986, Smolarkiewicz and Margolin 1993
- divergent flow corrections: Smolarkiewicz 1984
- infinite-gauge variant: Smolarkiewicz 2006

MPDATA variants (structured grid, homogeneous prob.)

- basic (+iterations): Smolarkiewicz 1983
- coordinate transformation: Smolarkiewicz and Clark 1986, Smolarkiewicz and Margolin 1993
- divergent flow corrections: Smolarkiewicz 1984
- infinite-gauge variant: Smolarkiewicz 2006
- flux-corrected transport: Smolarkiewicz and Grabowski 1990

MPDATA variants (structured grid, homogeneous prob.)

- basic (+iterations): Smolarkiewicz 1983
- coordinate transformation: Smolarkiewicz and Clark 1986, Smolarkiewicz and Margolin 1993
- divergent flow corrections: Smolarkiewicz 1984
- infinite-gauge variant: Smolarkiewicz 2006
- flux-corrected transport: Smolarkiewicz and Grabowski 1990
- third-order terms: Smolarkiewicz and Margolin 1998

MPDATA variants (structured grid, homogeneous prob.)

- basic (+iterations): Smolarkiewicz 1983
- coordinate transformation: Smolarkiewicz and Clark 1986, Smolarkiewicz and Margolin 1993
- divergent flow corrections: Smolarkiewicz 1984
- infinite-gauge variant: Smolarkiewicz 2006
- flux-corrected transport: Smolarkiewicz and Grabowski 1990
- third-order terms: Smolarkiewicz and Margolin 1998
- ...

MPDATA variants (structured grid, homogeneous prob.)

- basic (+iterations): Smolarkiewicz 1983
- coordinate transformation: Smolarkiewicz and Clark 1986, Smolarkiewicz and Margolin 1993
- divergent flow corrections: Smolarkiewicz 1984
- infinite-gauge variant: Smolarkiewicz 2006
- flux-corrected transport: Smolarkiewicz and Grabowski 1990
- third-order terms: Smolarkiewicz and Margolin 1998
- ...
- fully third-order variant: Waruszewski et al. 2018

demo

detour: new 2019 GMD journal policy

doi:10.5194/gmd-12-2215-2019

detour: new 2019 GMD journal policy

doi:10.5194/gmd-12-2215-2019

- *„everything required to run the experiment must be provided, apart from the model itself"*

detour: new 2019 GMD journal policy

doi:10.5194/gmd-12-2215-2019

- *„everything required to run the experiment must be provided, apart from the model itself"*
- *„ensure that there is no manual processing of the data: models are run by a script, and all pre- and post-processing is scripted"*

detour: new 2019 GMD journal policy

doi:10.5194/gmd-12-2215-2019

- *„everything required to run the experiment must be provided, apart from the model itself"*
- *„ensure that there is no manual processing of the data: models are run by a script, and all pre- and post-processing is scripted"*
- *„All figures and tables must be scientifically reproducible from the scripts"*

detour: new 2019 GMD journal policy

doi:10.5194/gmd-12-2215-2019

- *„everything required to run the experiment must be provided, apart from the model itself"*
- *„ensure that there is no manual processing of the data: models are run by a script, and all pre- and post-processing is scripted"*
- *„All figures and tables must be scientifically reproducible from the scripts"*
- *„It is the opinion of the GMD editors that if the code is not ready, then neither is the manuscript"*

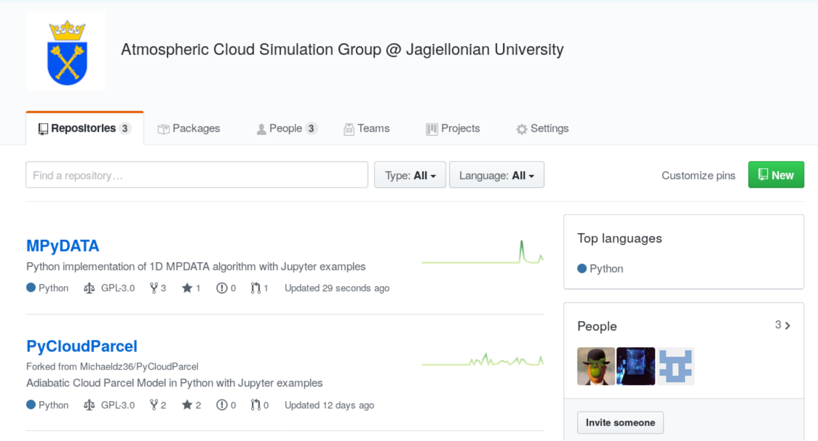
detour: new 2019 GMD journal policy

doi:10.5194/gmd-12-2215-2019

- *„everything required to run the experiment must be provided, apart from the model itself"*
- *„ensure that there is no manual processing of the data: models are run by a script, and all pre- and post-processing is scripted"*
- *„All figures and tables must be scientifically reproducible from the scripts"*
- *„It is the opinion of the GMD editors that if the code is not ready, then neither is the manuscript"*
- *„During the review process, the ease of model download, compilation, and running of test cases may be assessed"*

demo

github.com/atmos-cloud-sim-uj



demo

github.com/atmos-cloud-sim-uj/MPyDATA

README.md

MPyDATA

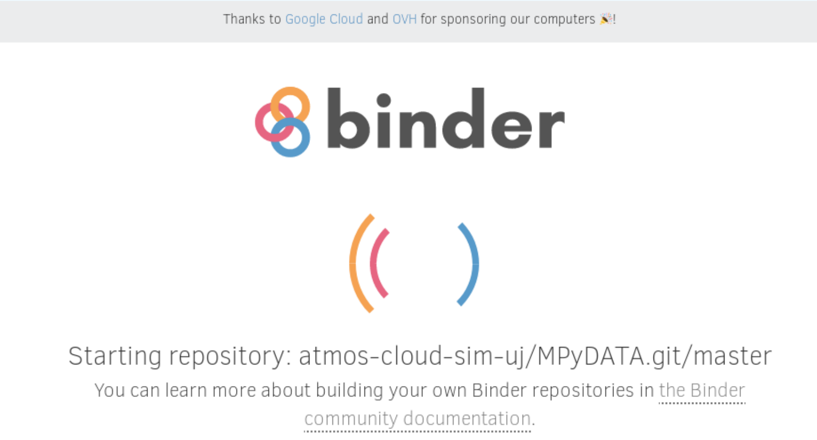
code quality B | build passing | coverage 19%

Examples:

- Smolarkiewicz 2006 Figs 3,4,10,11,12: launch binder | render nbviewer
- East 1957 Fig 3: launch binder | render nbviewer

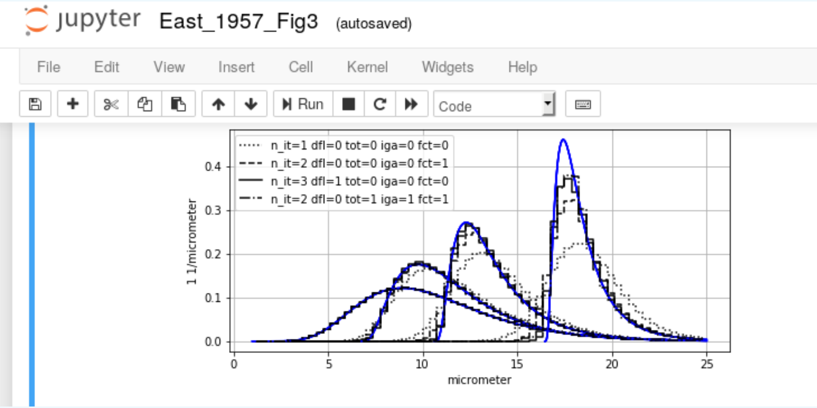
demo

mybinder.org/...



demo

mybinder.org/...



- Ahmad Farhat (HSBC)
- Michael Olesik (Jagiellonian)

- Hanna Pawłowska & libmpdata++ team (Univ. Warsaw)
- Piotr Smolarkiewicz (NCAR)

- Poland's National Science Centre (ncn.gov.pl)
- Foundation for Polish Science (fnp.org.pl)

- Ahmad Farhat (HSBC)
- Michael Olesik (Jagiellonian)

- Hanna Pawłowska & libmpdata++ team (Univ. Warsaw)
- Piotr Smolarkiewicz (NCAR)

- Poland's National Science Centre (ncn.gov.pl)
- Foundation for Polish Science (fnp.org.pl)

Thank you for your attention!