Antidiffusion techniques to refine the numerical solution of the advection equation

Case study: Smolarkiewicz' iterative approach

Gerhard Burger Jelmer Wolterink

Scientific Computing
Department of Mathematics
Utrecht University

Project presentations SOAC, 31 October 2011

Universiteit Utrecht

Outline

1. The advection equation in climate modelling
   - Importance
   - Diffusion
   - Antidiffusion methods
2. Case study: Smolarkiewicz
   - Analyzing the scheme
   - Implementation
   - Numerical Results
3. Current state of research
4. Conclusions

Universiteit Utrecht

Advection

Transport mechanism of a **substance** by a *fluid*, due to the fluid's motion in a particular direction.

Examples in ocean, atmosphere and climate modelling

- Transport of **trace gasses** by *air* due to wind
- Transport of **heat** by *ocean water* due to currents
- Transport of **ice** by *ocean water* due to currents
- Transport of **warm and moist air** over a colder surface by *air* due to wind: advection fog

Advection equation

Continuity equation describing the advection of a nondiffusive quantity in a flow field:

$$\frac{\partial \psi}{\partial t} + \operatorname{div}(V\psi) = 0$$

where $\psi(x, y, z, t)$ is the nondiffusive scalar quantity, $V = (u, v, w)$ is the velocity vector and x, y, z, t are the independent variables of space and time.

Advection simulation

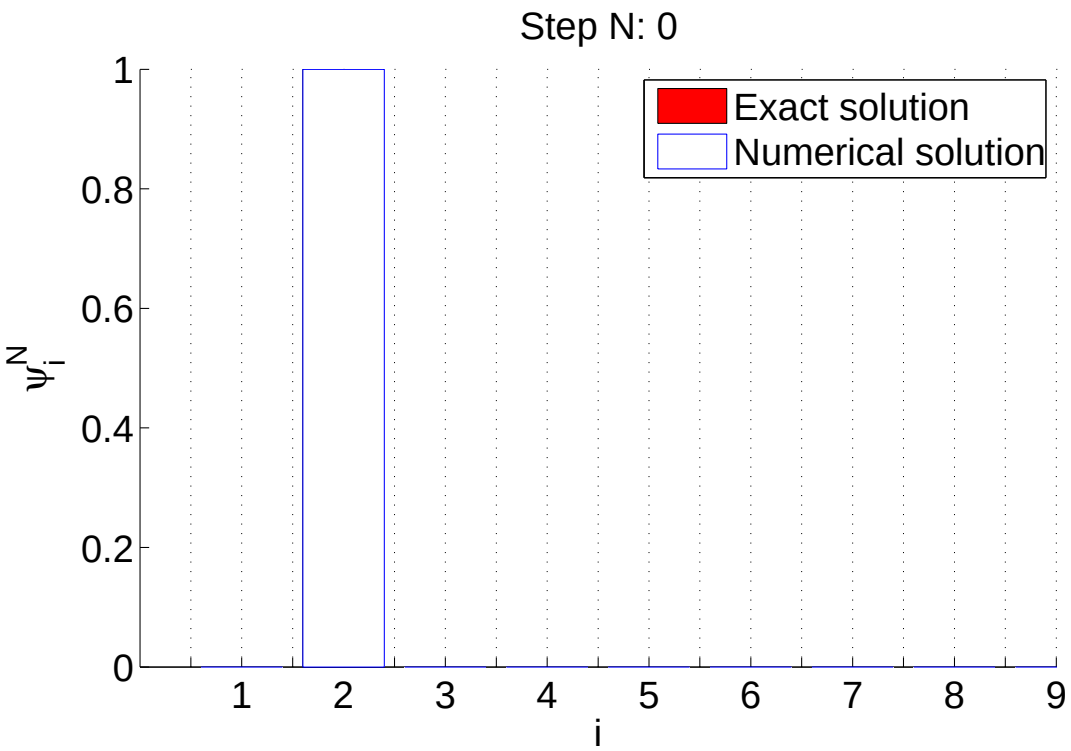
Constraints on advection simulation

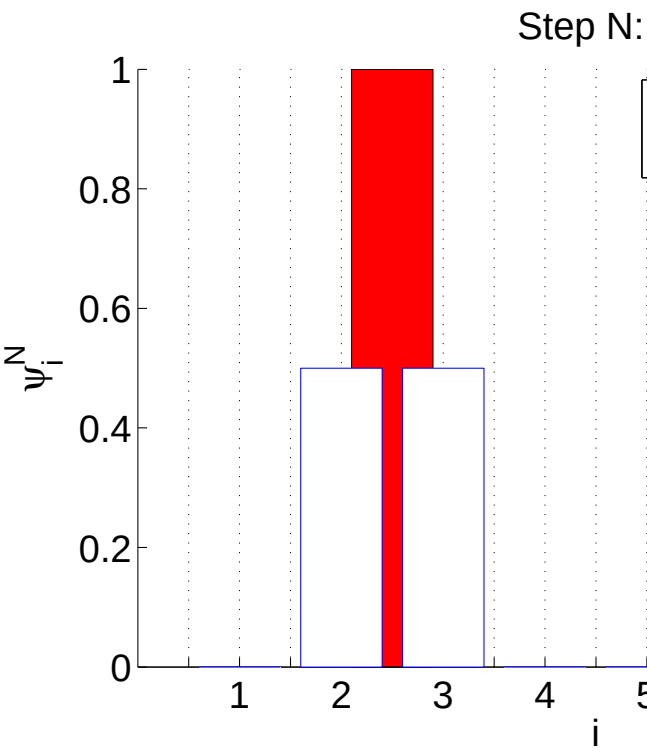
- Solutions should contain no unphysical overshoot or undershoot: positive definite schemes
- Methods should be volume preserving. No loss of matter
- The solutions should be local: the solution at any one point should not be influenced by what is going on far away from that point

Problem: diffusion

- When simulating an advective process, we need to discretize space. When doing so, we introduce numerical constraints on the calculation and we get diffusion.
- The numerical solution spreads out.
- We get interactions, our simulations become less reliable.
- $\Longrightarrow$ example

Example of diffusion with an upstream scheme





Universiteit Utrecht

Antidiffusion methods

- Flux-corrected transport (FCT) method (Boris and Book, 1973),
- self-adjusting hybrid scheme (SAHS) (Harten en Zwas, 1972),

Both can be very accurate but require excessive computing time, better is the

- hybrid-type scheme based on a Crowley advection scheme (Clark and Hall, 1979),

with more diffusion but half the computation time.

Antidiffusion methods

Smolarkiewicz' iterative correction, 1983

- Less time consuming while results are comparable to those of the more complex hybrid schemes
- Positive semidefinite
- Iterative method
- Does not contain strong implicit diffusion

Universiteit Utrecht

Basis: upstream on a staggered grid

1D case: $\frac{\partial \psi}{\partial t} + \frac{\partial}{\partial x}(u\psi) = 0$

We start with the following upstream advection equation on staggered grid:

$$\psi_i^{n+1} = \psi_i^n - \left(F\left(\psi_i^n, \psi_{i+1}^n, u_{i+1/2}^n\right) - F\left(\psi_{i-1}^n, \psi_i^n, u_{i-1/2}^n\right) \right),$$

where

$$F\left(\psi_i^n, \psi_{i+1}^n, u_{i+1/2}^n\right) = \left(\left(u_{i+1/2}^n + \left|u_{i+1/2}^n\right|\right) \psi_i^n + \left(u_{i+1/2}^n - \left|u_{i+1/2}^n\right|\right) \psi_{i+1}^n \right) \frac{\Delta t}{2\Delta x}.$$

Method of lines

Writing the scheme out and collecting terms gives

$$\begin{aligned}\psi_i^{n+1} = &\frac{\Delta t}{2\Delta x}\left(u_{i-1/2}^n + \left|u_{i-1/2}^n\right|\right)\psi_{i-1}^n \\ &+ \left(1 - \frac{\Delta t}{2\Delta x}\left(u_{i+1/2}^n + \left|u_{i+1/2}^n\right| - u_{i-1/2}^n + \left|u_{i-1/2}^n\right|\right)\right)\psi_i^n \\ &- \frac{\Delta t}{2\Delta x}\left(u_{i+1/2}^n - \left|u_{i+1/2}^n\right|\right)\psi_{i+1}^n\end{aligned}$$

Method of lines

This can be rewritten to

$$\psi_i^{n+1} = \alpha_i \psi_{i-1}^n + \beta_i \psi_i^n + \gamma_i \psi_{i+1}^n, \quad \text{for } i = 1, \ldots, M-1,$$

where we have that

$$\alpha_i = \frac{\Delta t}{2\Delta x}\left(u_{i-1/2}^n + \left|u_{i-1/2}^n\right|\right),$$
$$\beta_i = \left(1 - \frac{\Delta t}{2\Delta x}\left(u_{i+1/2}^n + \left|u_{i+1/2}^n\right| - u_{i-1/2}^n + \left|u_{i-1/2}^n\right|\right)\right),$$
$$\gamma_i = -\frac{\Delta t}{2\Delta x}\left(u_{i+1/2}^n - \left|u_{i+1/2}^n\right|\right).$$

Method of lines

We can also write this in matrix form using Direchlet boundary conditions

- $y(0) = 0$ and $y(M) = 0$
- using periodic boundary conditions $y(0) = y(M)$

$$\begin{bmatrix} \psi_1^{n+1} \\ \psi_2^{n+1} \\ \vdots \\ \psi_{M-2}^{n+1} \\ \psi_{M-1}^{n+1} \end{bmatrix} = \begin{bmatrix} \beta_1 & \gamma_1 & & & 0\alpha_1 \\ \alpha_2 & \beta_2 & \gamma_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \alpha_{M-2} & \beta_{M-2} & \gamma_{M-2} \\ 0\gamma_{M-1} & & & \alpha_{M-1} & \beta_{M-1} \end{bmatrix} \begin{bmatrix} \psi_1^{n} \\ \psi_2^{n} \\ \vdots \\ \psi_{M-2}^{n} \\ \psi_{M-1}^{n} \end{bmatrix}$$

So the values of ψ at timestep $N+1$ can be obtained by a sparse matrix-vector multiplication

Universiteit Utrecht

Antidiffusion

To apply antidiffusion we need to introduce an intermediate step

$$\psi_i^* = \psi_i^n - \left(F\left(\psi_i^n, \psi_{i+1}^n, u_{i+1/2}^n\right) - F\left(\psi_{i-1}^n, \psi_i^n, u_{i-1/2}^n\right) \right),$$
$$\psi_i^{n+1} = \psi_i^* - \left(F\left(\psi_i^*, \psi_{i+1}^*, \tilde{u}_{i+1/2}^n\right) - F\left(\psi_*^n, \psi_*^n, \tilde{u}_{i-1/2}^n\right) \right),$$

were the antidiffusion velocity $\tilde{u}_{i+1/2}$ is defined as

$$\tilde{u}_{i+1/2} = \frac{\left(\left|u_{i+1/2}\right| \Delta x - \Delta t u_{i+1/2}^2\right)\left(\psi_{i+1}^* - \psi_i^*\right)}{\left(\psi_i^* + \psi_{i+1}^* + \epsilon\right)\Delta x}$$

Since the second step is similar to the first we can also apply the method of lines to the second step.

Stability

For stability we need that

$$\max_i \left(\frac{\left| u_{i+1/2} \right| \Delta t}{\Delta x} \right) \leq 1$$

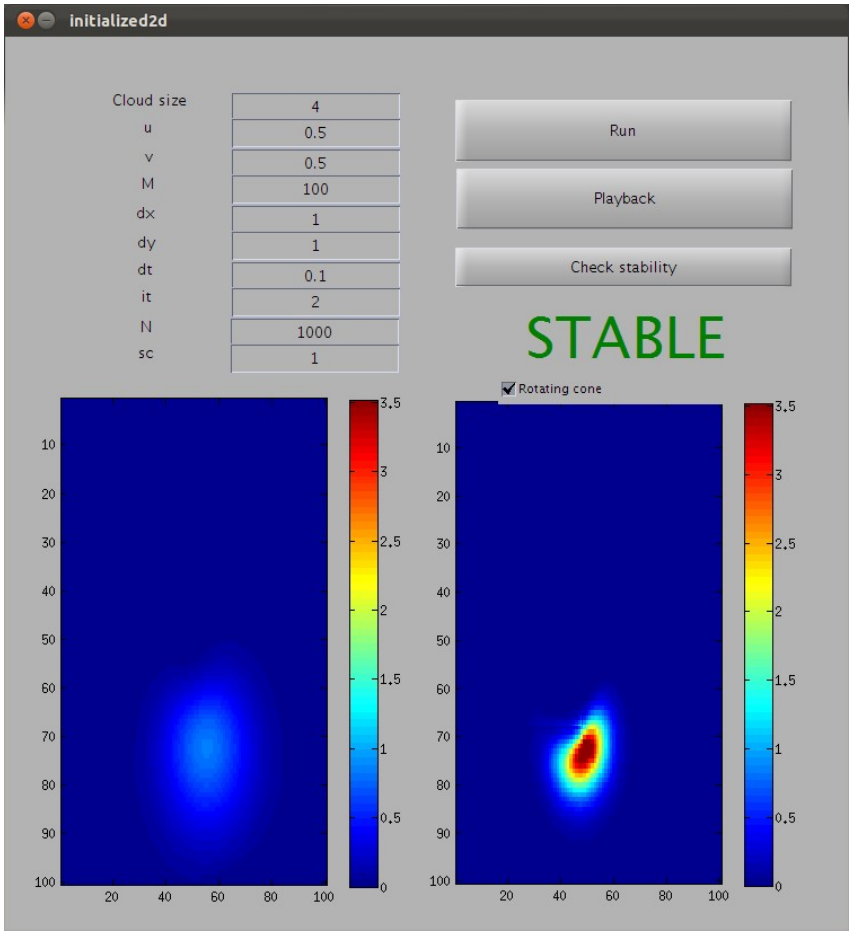
Algorithm

input : Initial state ψ^0, velocities u, iter, tsteps, Δx, Δt
output: Final state ψ^N

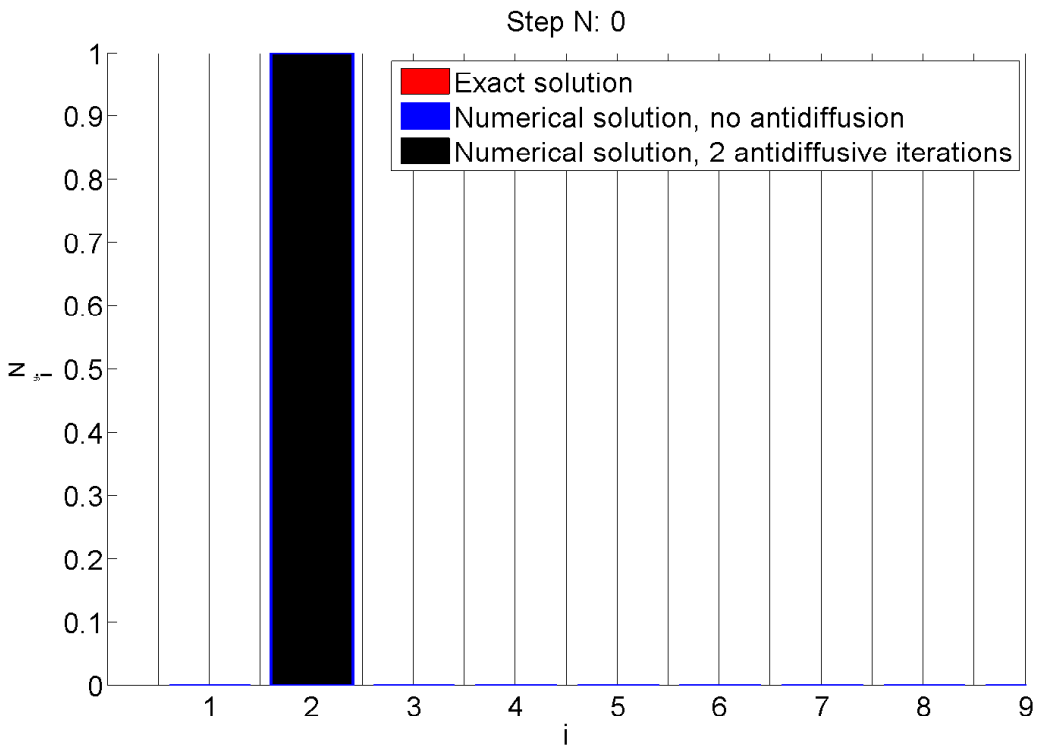
```
ψ ← ψ^0;
mat1 ← ComputeMatrix(u, Δx, Δt);
for i ← 1 to tsteps do
    ψ ← MatrixMultiplication(mat1,ψ);
    for j ← 1 to iter do
        ũ ← ComputeAntidiffusionVelocity(ψ, u);
        mat2 ← ComputeMatrix(ũ, Δx, Δt);
        ψ ← MatrixMultiplication(mat2,ψ);
    end
end
ψ^N ← ψ;
```

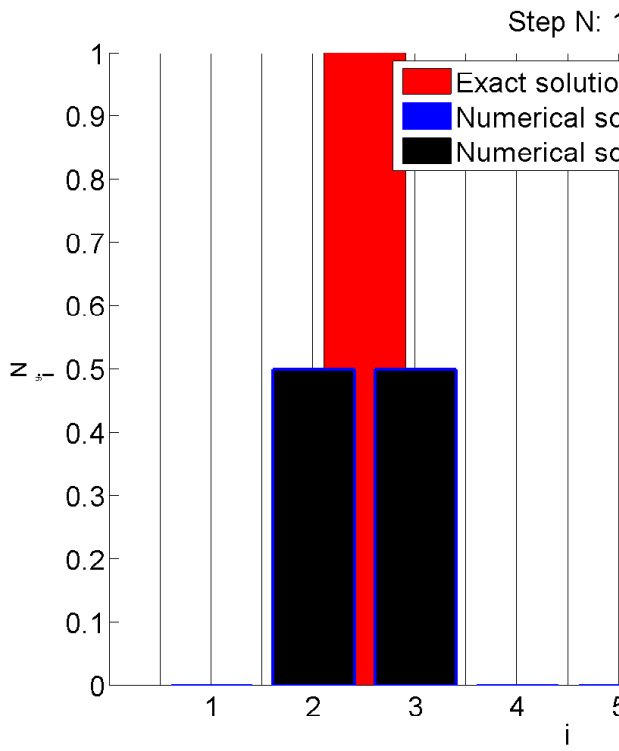
Universiteit Utrecht

Matlab interface



Example of antidiffusion with an upstream scheme

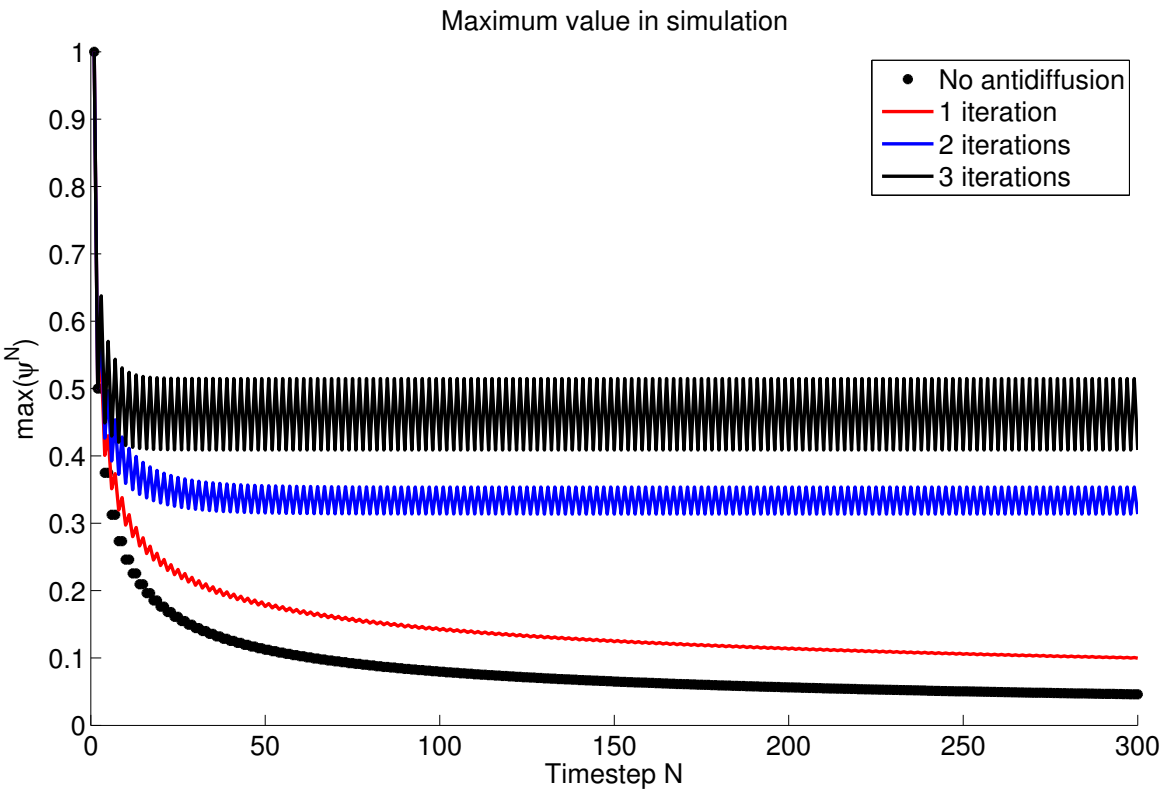




Universiteit Utrecht

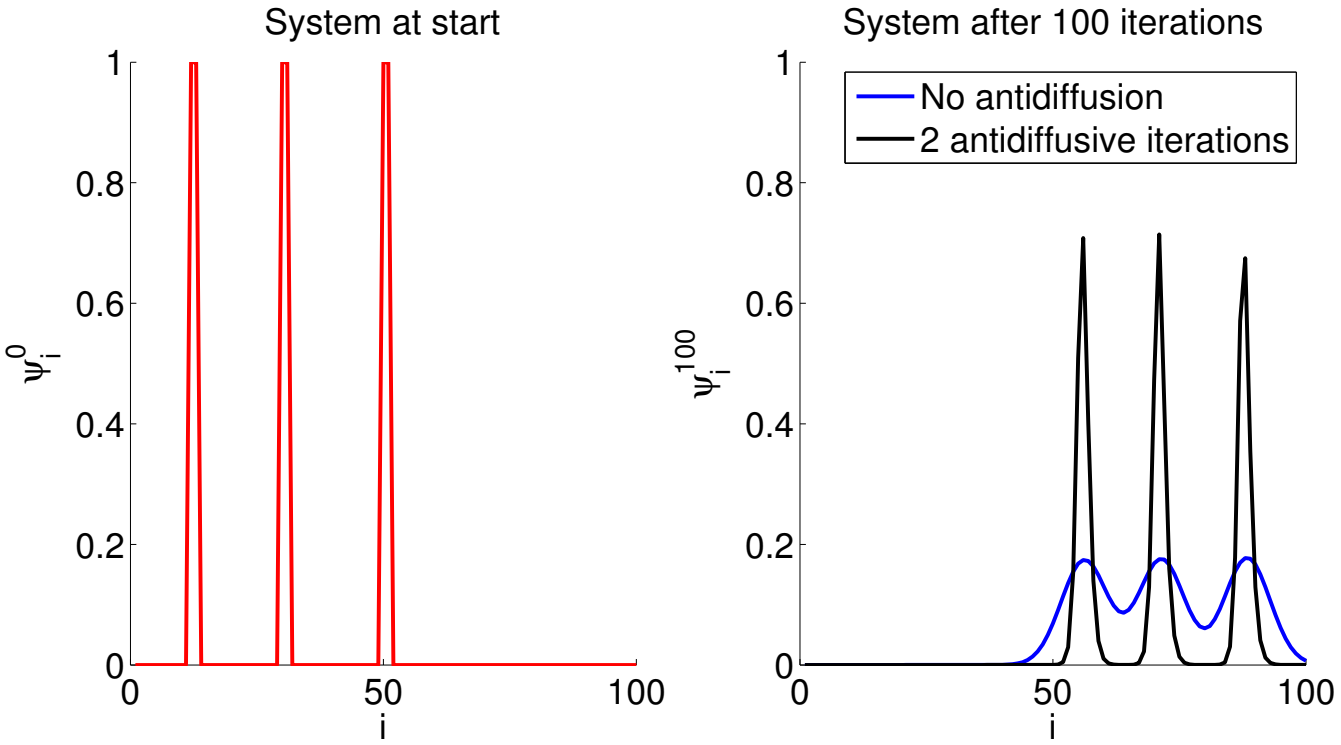
Less diffusion

Iterating with antidiffusion vector reduces diffusion.



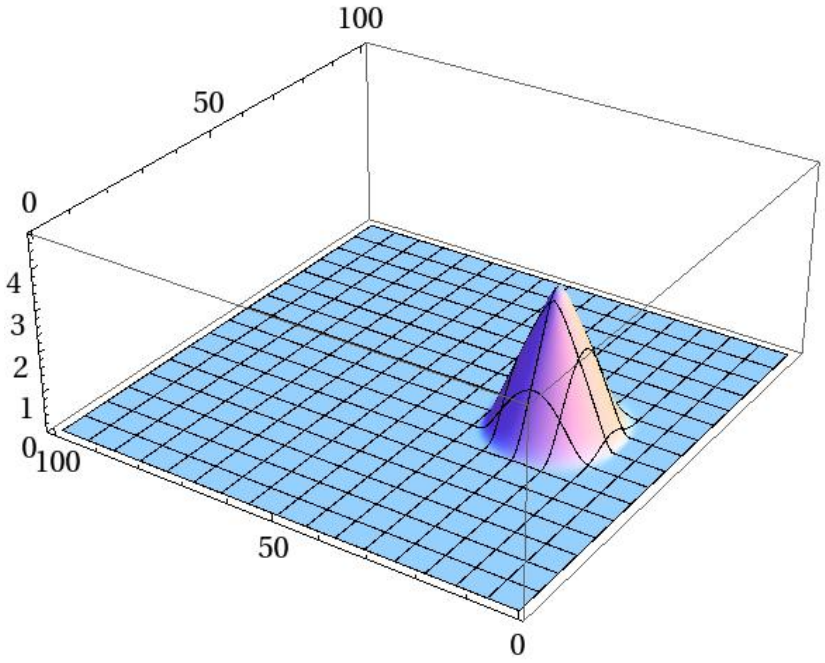
Less interaction

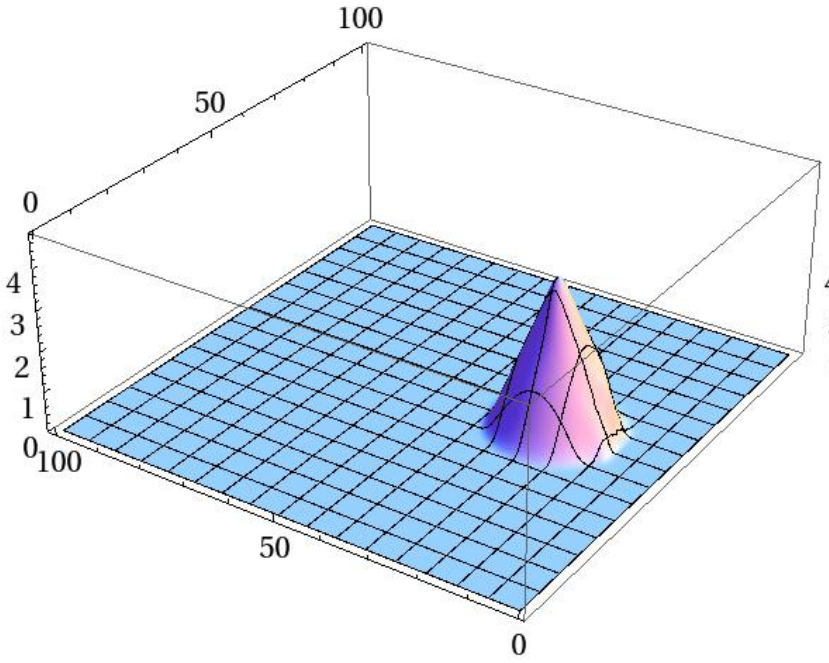
Simulations are more localized, less interaction between neighbouring pockets.

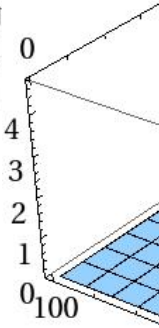


Example of diffusion with an upstream scheme

Solid body rotation





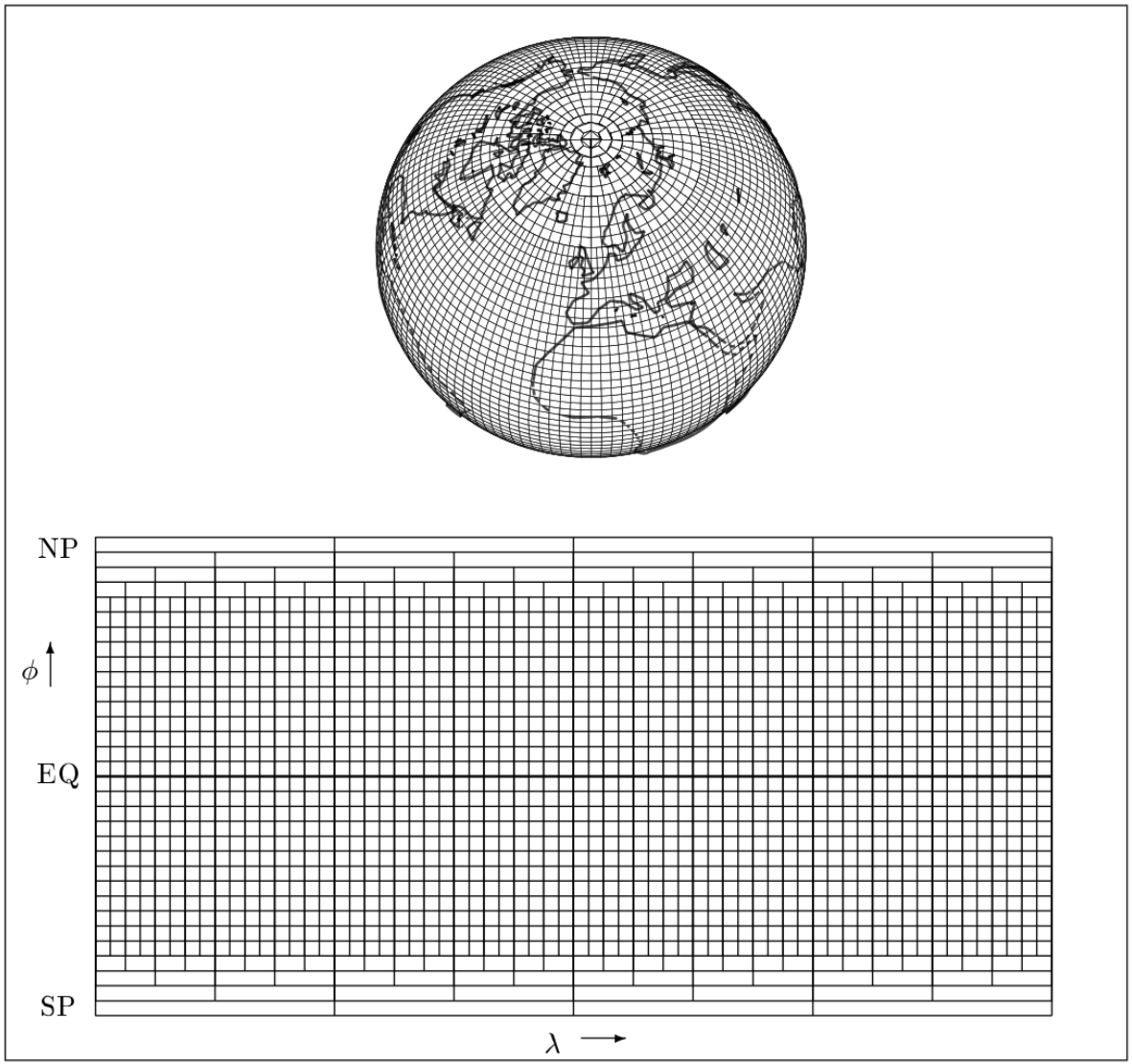


Example of diffusion with an upstream scheme

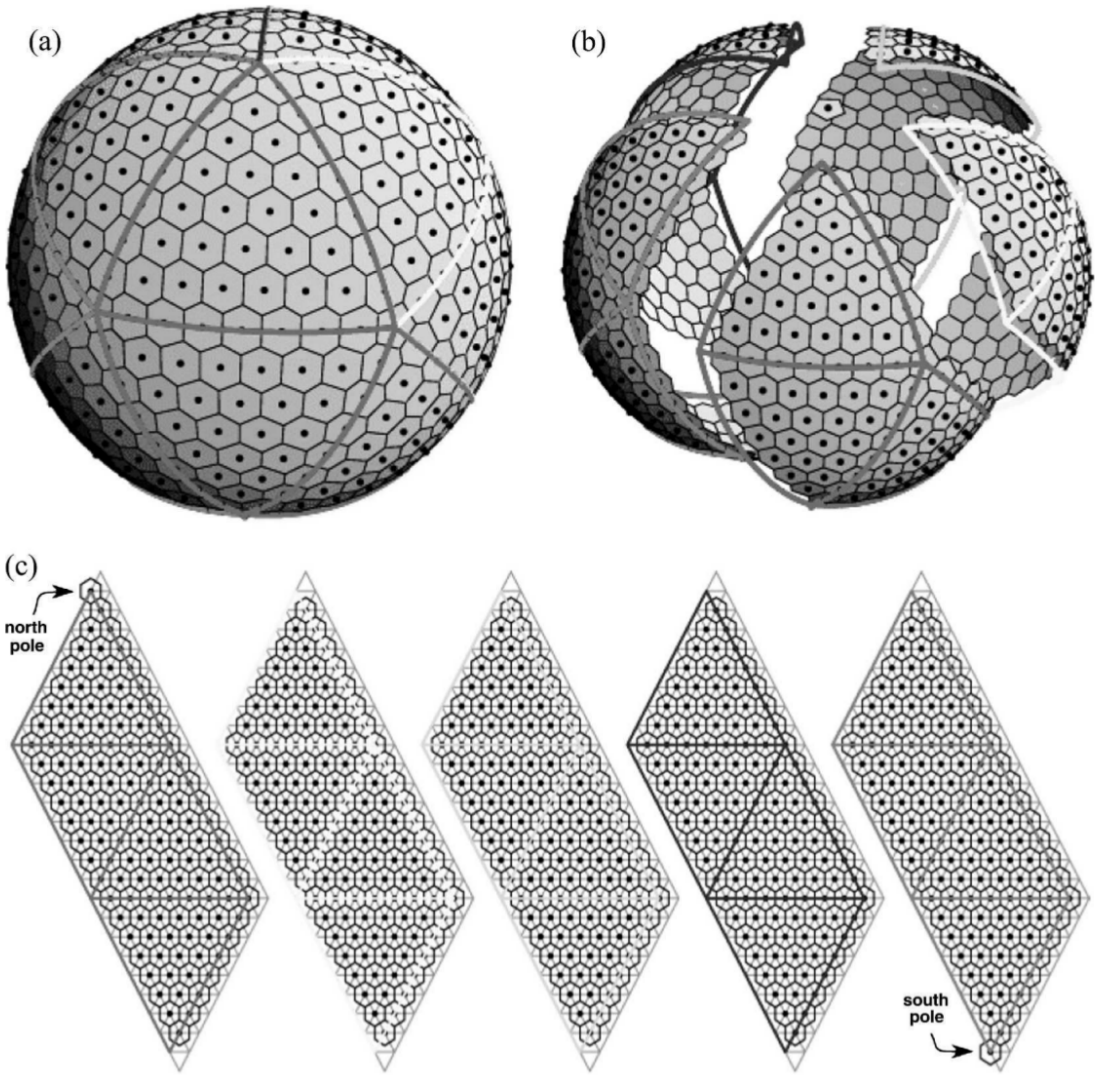
- Without antidiffusion, the body diffuses in one full rotation
- With antidiffusion, the body keeps its shape

- Since 1983 a lot of progress has been made
- Advection calculations on the sphere pose new problems/possibilities
- Finite difference as well as finite volume methods
- Faster machines, more complex schemes
- Much higher accuracy
- Improvements by Smolarkiewicz and others

Reduced grid (Spee, 1991)



Spherical hexagonal-pentagonal grid (Lipscomb and Ringler, 2005)



MPDATA

MPDATA: Multidimensional Positive Definite Advection Transport Algorithm

- Multidimensional generalization of Smolarkiewicz' original scheme
- Widely used in finite difference form
- Recently a transition to finite volume has been made

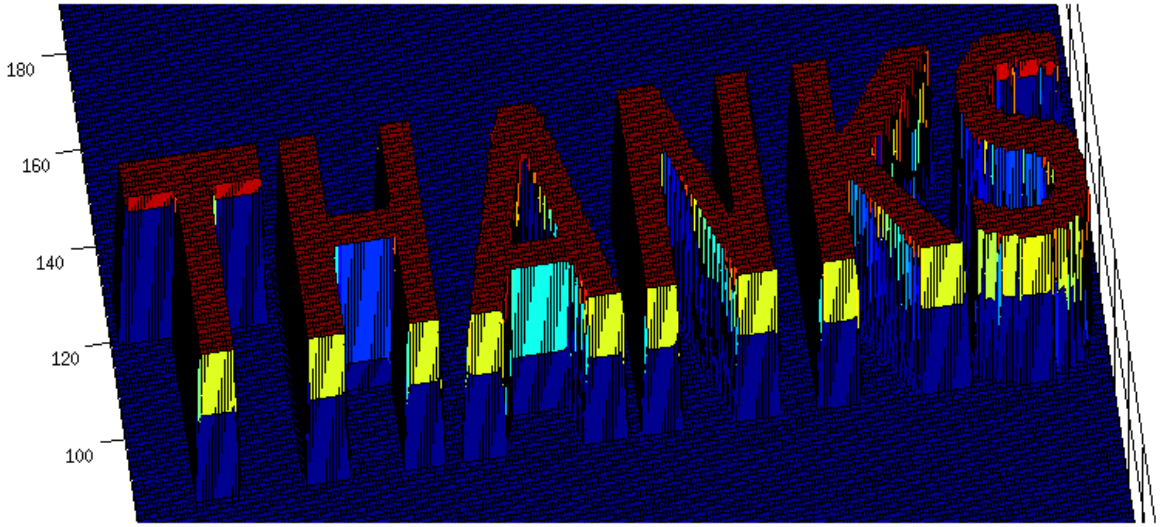
> *Over the two decades, MPDATA evolved from an advection scheme into a class of generalized transport algorithms that expand beyond advective transport to alternate PDEs and complete fluid models with a wide range of underlying governing equations.*

Smolarkiewicz, 2005

Universiteit Utrecht

Conclusions

- Diffusion is a threat for reliable advection simulations
- Antidiffusive advection schemes reduce this threat and are thus important in SOAC
- Smolarkiewicz' relatively simple scheme already significantly improves results



Universiteit Utrecht