

# 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
  - Progress



## Advection

Transport mechanism of a **substance** by a *fluid*, due to the fluids 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 **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:

$$\frac{\partial \psi}{\partial t} + \operatorname{div}(V\psi) = 0 \quad (1)$$

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 space and time independent variables.



# 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
- The numerical solution should not introduce new extrema in the solution, because the continuous form of the solution would not
- The method should be cost effective: memory and computational requirements should be sufficiently small so that practical problems may be solved

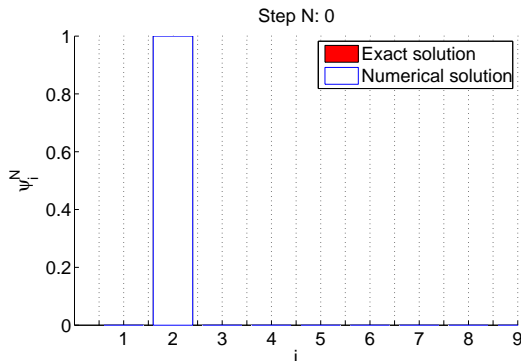


# Problem: diffusion

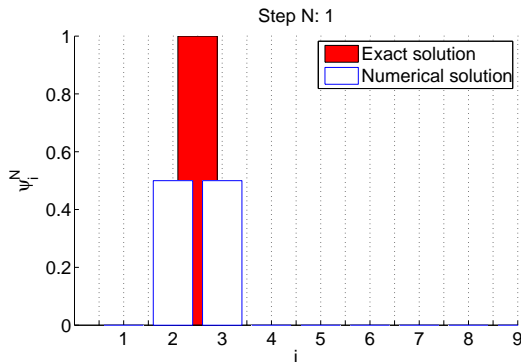
- When simulating an advective process, we need to discretize space. When doing so, we introduce numerical constraints on the calculation and we get diffusion.
- The numerical solution spreads out.
- We get interactions, our simulations become less reliable.
- $\implies$  example



# Example of diffusion with an upstream scheme

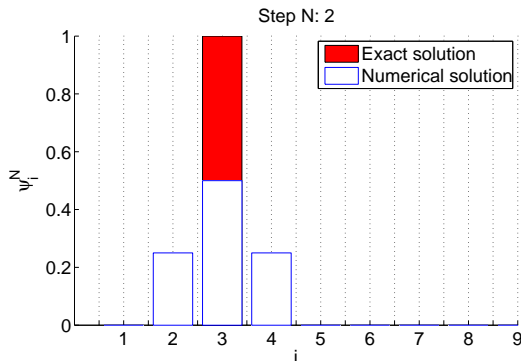


# Example of diffusion with an upstream scheme

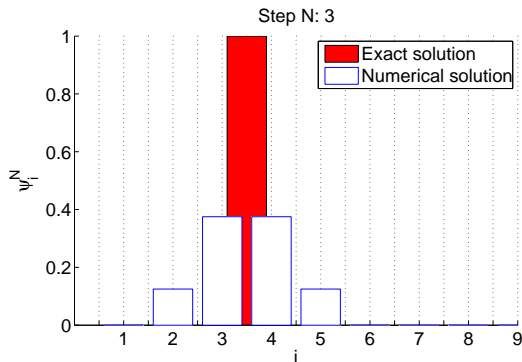




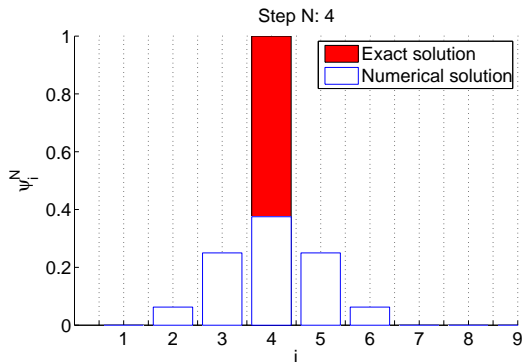
# Example of diffusion with an upstream scheme



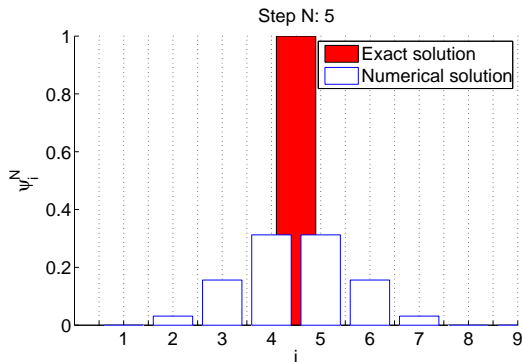
# Example of diffusion with an upstream scheme



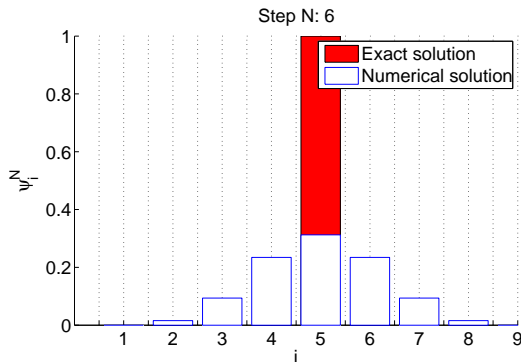
## Example of diffusion with an upstream scheme



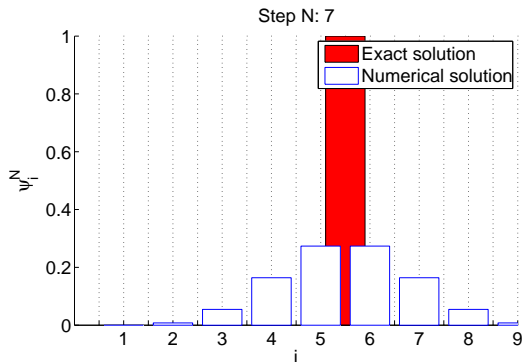
## Example of diffusion with an upstream scheme



## Example of diffusion with an upstream scheme



## Example of diffusion with an upstream scheme



# 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





## Basis: upstream on a staggered grid

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

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

where

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



# Method of lines 1

Writing the scheme out and collecting terms gives

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



## Method of lines 2

This can be rewritten to

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

where we have that

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



## Method of lines 3

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

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

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

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



## Method of lines 3

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

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

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

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



## Method of lines 3

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

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

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

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



# Antidiffusion 1

To apply antidiffusion we need to redefine the scheme into

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

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

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

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



# Algorithm

**input** : Initial state  $\psi^0$ , velocities  $u$ , iter, tsteps,  $\Delta x$ ,  $\Delta t$

**output**: Final state  $\psi^N$

$\psi \leftarrow \psi^0$ ;

mat1  $\leftarrow$  ComputeMatrix( $u, \Delta x, \Delta t$ );

**for**  $i \leftarrow 0$  **to** tsteps **do**

$\psi \leftarrow$  MatrixMultiplication(mat1,  $\psi$ );

**for**  $j \leftarrow 1$  **to** iter **do**

$\tilde{u} \leftarrow$  ComputeAntidiffusionVelocity( $\psi, u$ );

        mat2  $\leftarrow$  ComputeMatrix( $\tilde{u}, \Delta x, \Delta t$ );

$\psi \leftarrow$  MatrixMultiplication(mat2,  $\psi$ );

**end**

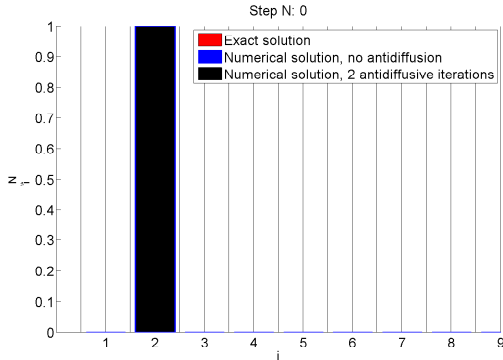
**end**

$\psi^N \leftarrow \psi$ ;

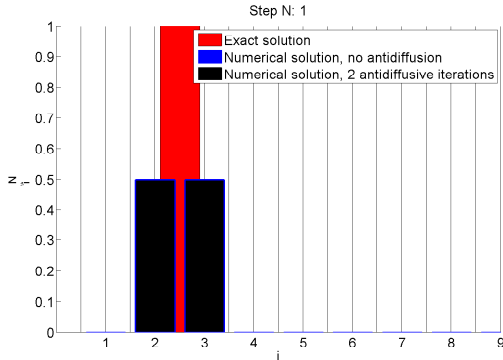




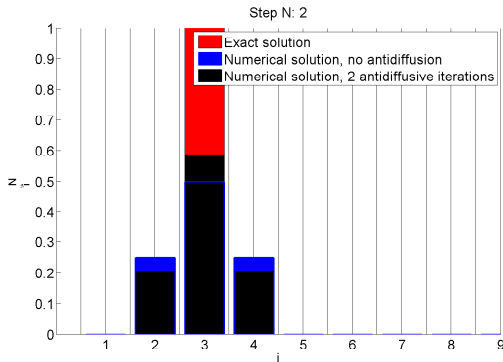
# Example of antidiffusion with an upstream scheme



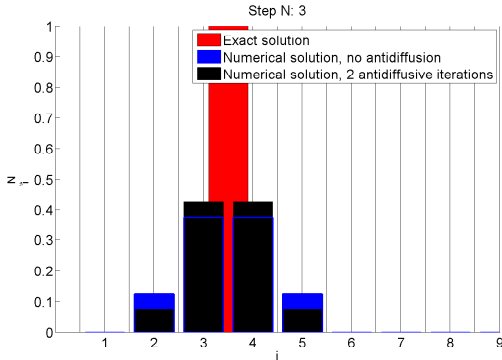
# Example of antidiffusion with an upstream scheme



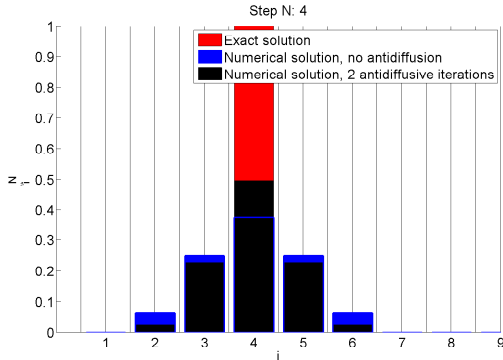
# Example of antidiffusion with an upstream scheme



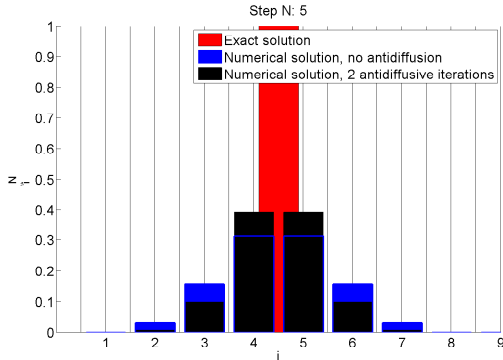
# Example of antidiffusion with an upstream scheme



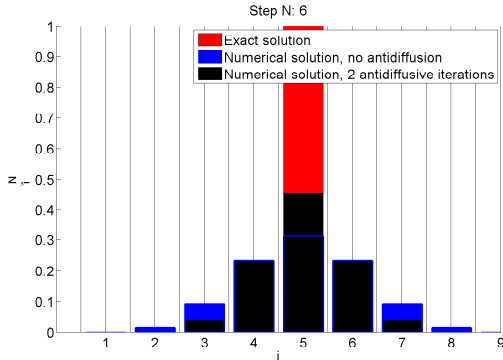
# Example of antidiffusion with an upstream scheme



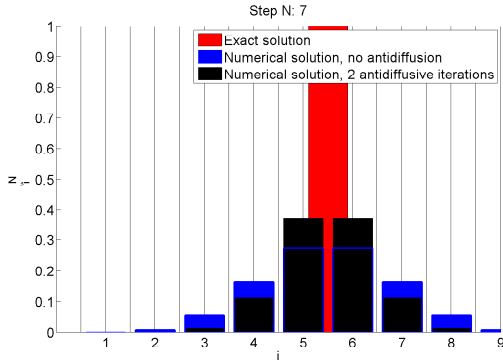
# Example of antidiffusion with an upstream scheme



# Example of antidiffusion with an upstream scheme



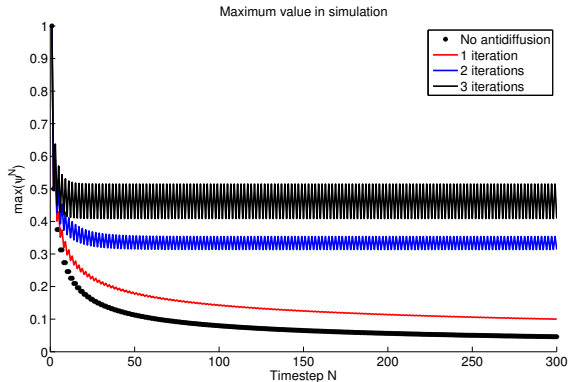
# Example of antidiffusion with an upstream scheme





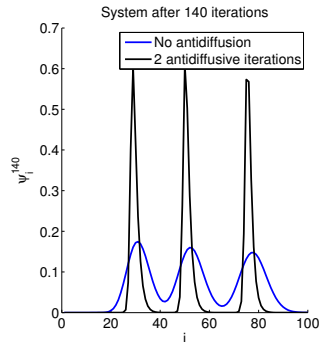
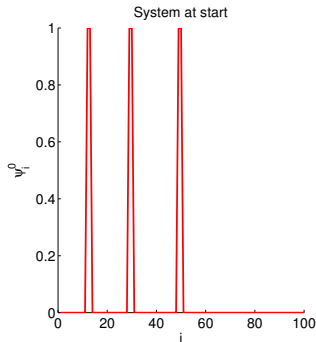
# Less diffusion

Iterating with antidiffusion vector reduces diffusion.



# Less interaction

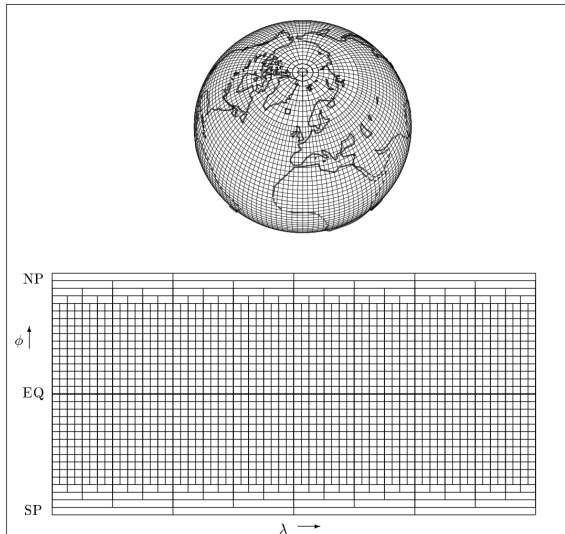
Simulations are more localized, less interaction between neighbouring pockets.



- Since 1983 a lot of progress has been made
- Advection calculations on the sphere pose new problems.



## Reduced grid (Spee, 1991)



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

