Adaptive mesh refinement techniques for well-balanced schemes for shallow water flows

Pep Mulet

Joint work with Rosa Donat and Anna Martínez Gavara
Grup ANIMS, Dpt. Matemàtica Aplicada, Univ. València

NumHyp2011, Roscoff, September 2011

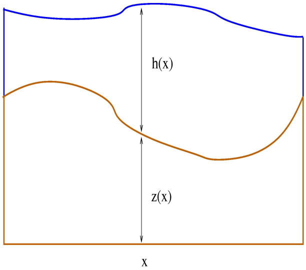
Outline

1. Shock capturing schemes for Shallow water flows
2. Adaptive Mesh Refinement
   - Adaptive schemes
   - Grid hierarchy
3. Well-balanced Adaptive techniques
   - Well-balanced schemes
   - Well-balanced AMR
   - Homogeneous discretization for SWE
   - Well-balanced interpolation
4. Numerical results
   - Numerical results
5. Conclusions

Shallow water flow

- Shallow water equations (SWE) are obtained from incompressible Navier-Stokes equations by depth-averaging and neglecting some terms:

$$h_t + \mathrm{div}(hv) = 0$$
$$(hv)_t + \mathrm{div}(hv \otimes v + \frac{gh^2}{2} I_2) = -gh\nabla z$$



$h \equiv$ water depth,
$v = (v^x, v^y) \equiv$ depth-averaged velocity,
$g \equiv$ gravity acceleration,
$z \equiv$ bottom elevation.

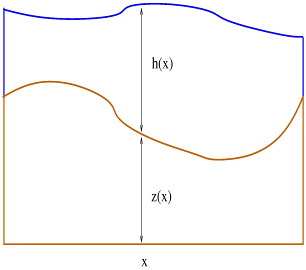
- To simplify, we do the exposition in 1D:

$$h_t + (hv)_x = 0$$
$$(hv)_t + (hv^2 + \frac{gh^2}{2})_x = -ghz_x$$

Shallow water flow

- Shallow water equations (SWE) are obtained from incompressible Navier-Stokes equations by depth-averaging and neglecting some terms:

$$h_t + \mathrm{div}(hv) = 0$$
$$(hv)_t + \mathrm{div}(hv \otimes v + \frac{gh^2}{2} I_2) = -gh\nabla z$$



$h \equiv$ water depth,
$v = (v^x, v^y) \equiv$ depth-averaged velocity,
$g \equiv$ gravity acceleration,
$z \equiv$ bottom elevation.

- To simplify, we do the exposition in 1D:

$$h_t + (hv)_x = 0$$
$$(hv)_t + (hv^2 + \frac{gh^2}{2})_x = -ghz_x$$

Shock capturing schemes

- Use notation:

$$u = \begin{bmatrix} h \\ hv \end{bmatrix}, f(u) = \begin{bmatrix} hv \\ hv^2 + \frac{gh^2}{2} \end{bmatrix}, s(x,u) = \begin{bmatrix} 0 \\ -ghz_x \end{bmatrix}$$

so that SWE system can be written as: $\boxed{u_t + f(u)_x = s(x,u)}$.

- Nonlinear hyperbolic system $\Rightarrow$ solutions can develop discontinuities $\Rightarrow$ use **shock capturing** schemes:

$$u_i^{n+1} = u_i^n - \Delta t\left(\frac{\hat{f}_{i+1/2}^n - \hat{f}_{i-1/2}^n}{\Delta x} - s_i^n\right),$$

where $s_i^n(u(x,t)) \approx s(x_i, u(x_i,t_n))$ and the numerical fluxes $\hat{f}_{i+1/2} = \hat{f}(u_{i-s}, \dots, u_{i+s+1})$ verify

$$\left[\frac{\hat{f}_{i+1/2}^n - \hat{f}_{i-1/2}^n}{\Delta x}\right](u(x,t)) \approx f(u)_x(x_i,t_n), \quad x_i = i\Delta x, t_n = n\Delta t$$

and appropriate stability conditions (through **upwinding** and adding numerical viscosity to comply with entropy conditions).

Shock capturing schemes

- Use notation:

$$u = \begin{bmatrix} h \\ hv \end{bmatrix}, f(u) = \begin{bmatrix} hv \\ hv^2 + \frac{gh^2}{2} \end{bmatrix}, s(x,u) = \begin{bmatrix} 0 \\ -ghz_x \end{bmatrix}$$

  so that SWE system can be written as: $\boxed{u_t + f(u)_x = s(x,u)}$.
- Nonlinear hyperbolic system ⇒ solutions can develop discontinuities ⇒ use **shock capturing** schemes:

$$u_i^{n+1} = u_i^n - \Delta t \left(\frac{\hat{f}^n_{i+1/2} - \hat{f}^n_{i-1/2}}{\Delta x} - s_i^n \right),$$

  where $s_i^n(u(x,t)) \approx s(x_i, u(x_i, t_n))$ and the numerical fluxes $\hat{f}_{i+1/2} = \hat{f}(u_{i-s}, \dots, u_{i+s+1})$ verify

$$\left[\frac{\hat{f}^n_{i+1/2} - \hat{f}^n_{i-1/2}}{\Delta x} \right] (u(x,t)) \approx f(u)_x(x_i, t_n), \quad x_i = i\Delta x, t_n = n\Delta t$$

  and appropriate stability conditions (through **upwinding** and adding numerical viscosity to comply with entropy conditions).

Outline

1. Shock capturing schemes for Shallow water flows

2. Adaptive Mesh Refinement
   - Adaptive schemes
   - Grid hierarchy

3. Well-balanced Adaptive techniques
   - Well-balanced schemes
   - Well-balanced AMR
   - Homogeneous discretization for SWE
   - Well-balanced interpolation

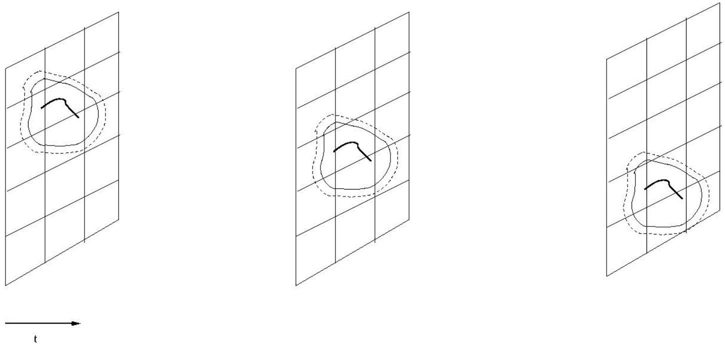
4. Numerical results
   - Numerical results

5. Conclusions

Adaptive schemes

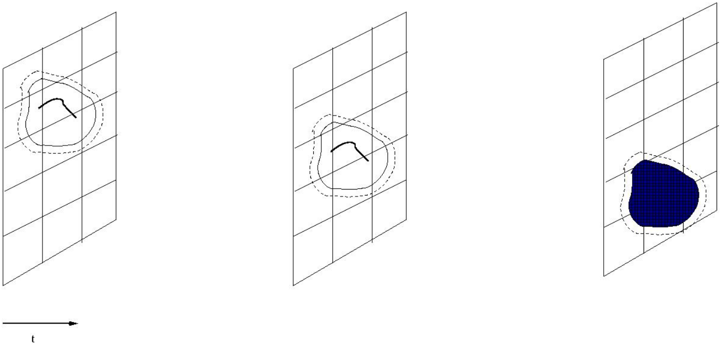
- For $N = 1/\Delta$ and d dimensions, computational cost of scheme is $\mathcal{O}(N^{d+1})$, storage is $\mathcal{O}(N^d)$, huge to get small errors.
- Numerical errors are not uniformly distributed:
  - larger errors at discontinuities
  - smaller errors at smooth regions
- An **Adaptive Scheme**, with a smaller Δ where higher errors occur, would be necessary for $d \geq 2$ and high precision needs.
- Many approaches [Cohen et al., 2003, Müller and Stiriba, 2007] $\cdots$, we briefly review the (**S**tructured) **A**daptive **M**esh **R**efinement algorithm, proposed by [Berger and Oliger, 1984] and extended by many authors (Colella, Quirk, $\cdots$) to FV schemes.

AMR algorithm



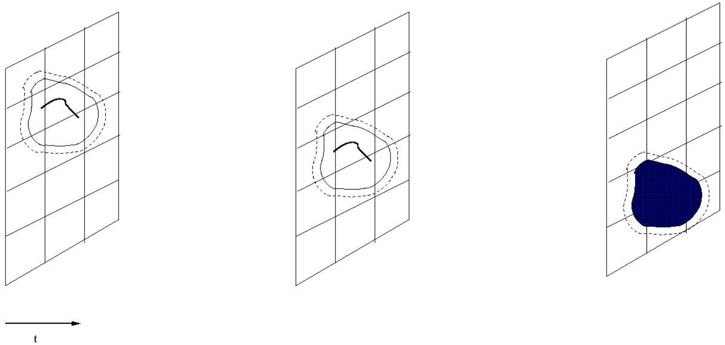
- Time evolution for some grid size $\Delta \equiv \Delta x$ and Δt.

AMR algorithm



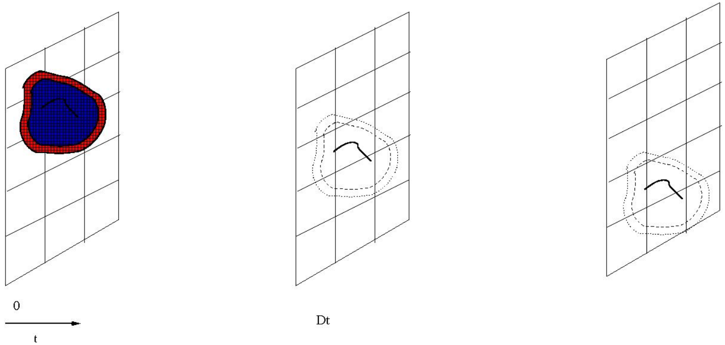
- Want to zoom at **R**egion **O**f **I**nterest, say by using $\Delta/2$.

AMR algorithm



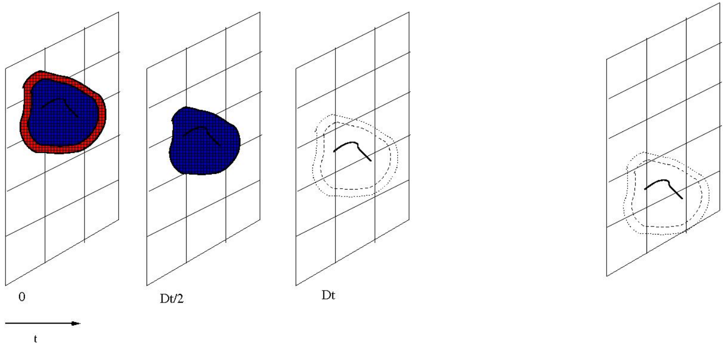
- A: use **interpolation** (zoom), but this causes large errors near shocks.
- B: discard results with Δ, start over with $\Delta/2$.
- C: track region of interest through time evolution.

AMR algorithm



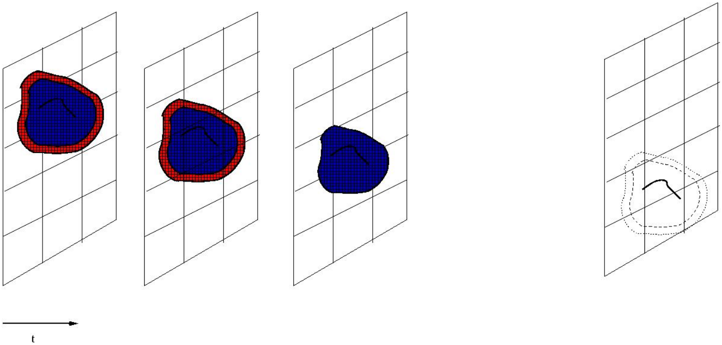
- Before going to B plan, notice that solution on $\Omega \times [0, \Delta t]$ (hopefully) depends on solution at **D**omain of **D**ependence $\widetilde{\Omega} \times \{0\}$ (**by hyperbolicity**).
- Can compute solution at $\Omega \times \{\frac{\Delta t}{2}\}$ (assuming $\Delta/2$ at ROI, same CFL)

AMR algorithm



- How can new DD of region of interest be computed?
- **Zooming by** (x, t)**-interpolation**, OK at (supposedly smooth) surrounding band (**coarse** $\rightarrow$ **fine interpolation**)

AMR algorithm



- Recursion $\Rightarrow$ need **nested Grid Hierarchy** (for interpolation), indexed by level l from $l = 0$ (**coarsest**) to $l = L$ (**finest**).
- Must synchronize data through GH at same (x, t) (**fine $\rightarrow$ coarse project.**)
- More (shorter) time steps at finer resolutions (**local time stepping**).

Outline

1 Shock capturing schemes for Shallow water flows

2 Adaptive Mesh Refinement
- Adaptive schemes
- Grid hierarchy

3 Well-balanced Adaptive techniques
- Well-balanced schemes
- Well-balanced AMR
- Homogeneous discretization for SWE
- Well-balanced interpolation

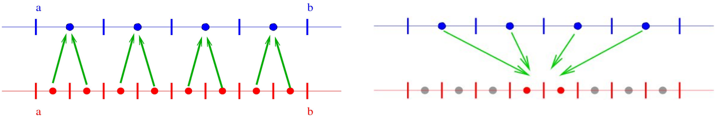
4 Numerical results
- Numerical results

5 Conclusions

Grid hierarchy

- **Based on cell averages:** Points in the grid hierarchy (show 1D, 2D obtained by cartesian product): $x_i^l = (i + \frac{1}{2})\Delta_0/2^l$, $i = 0, \ldots, N_0 2^l - 1$ (**cell centers**).
- Since $\frac{1}{2}(x_{2i}^{l+1} + x_{2i+1}^{l+1}) = x_i^l$, project solution by averaging

$$\text{Proj}_{l+1 \to l}(u^{l+1})_i = \frac{1}{2}(u_{2i}^{l+1} + u_{2i+1}^{l+1}), \quad i = 0, \ldots, N_0 2^l - 1.$$



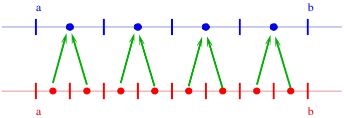
fine → coarse coarse → fine

- Usual hierarchy for finite volume schemes [Berger and Oliger, 1984], can be made conservative.

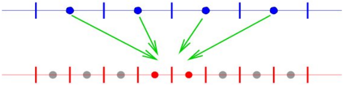
Grid hierarchy

- **Based on cell averages:** Points in the grid hierarchy (show 1D, 2D obtained by cartesian product): $x_i^l = (i + \frac{1}{2})\Delta_0/2^l$, $i = 0, \ldots, N_0 2^l - 1$ (**cell centers**).
- Since $\frac{1}{2}(x_{2i}^{l+1} + x_{2i+1}^{l+1}) = x_i^l$, project solution by averaging

$$\mathrm{Proj}_{l+1\to l}(u^{l+1})_i = \frac{1}{2}(u_{2i}^{l+1} + u_{2i+1}^{l+1}), \quad i = 0, \ldots, N_0 2^l - 1.$$



fine $\to$ coarse



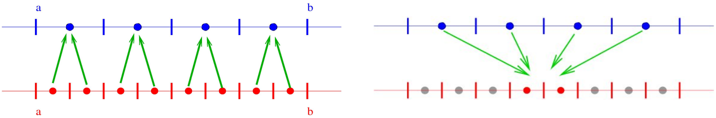
coarse $\to$ fine

- Usual hierarchy for finite volume schemes [Berger and Oliger, 1984], can be made conservative.

Grid hierarchy

- **Based on cell averages:** Points in the grid hierarchy (show 1D, 2D obtained by cartesian product): $x_i^l = (i+\frac{1}{2})\Delta_0/2^l$, $i = 0, \dots, N_0 2^l - 1$ (**cell centers**).
- Since $\frac{1}{2}(x_{2i}^{l+1} + x_{2i+1}^{l+1}) = x_i^l$, project solution by averaging

$$\text{Proj}_{l+1\to l}(u^{l+1})_i = \frac{1}{2}(u_{2i}^{l+1} + u_{2i+1}^{l+1}), \quad i = 0, \dots, N_0 2^l - 1.$$



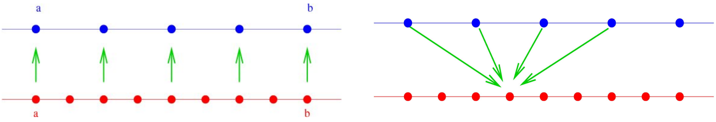
fine $\to$ coarse　　　　coarse $\to$ fine

- Usual hierarchy for finite volume schemes [Berger and Oliger, 1984], can be made conservative.

Grid hierarchy

- **Based on point values:** Points in the grid hierarchy: $x_i^l = i\Delta_0/2^l$, $i = 0, \ldots, N_0 2^l$.
- Since $x_{2i}^{l+1} = x_i^l$ (even indexed points in level $l+1$ are aligned with points in level l), project solution by just copying even indexed values

$$\mathrm{Proj}_{l+1\to l}(u^{l+1})_i = u_{2i}^{l+1}, \quad i = 0, \ldots, N_0 2^l.$$



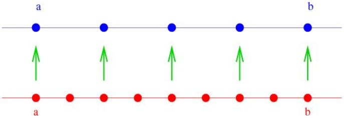
fine $\to$ coarse coarse $\to$ fine

- Loss of information (not conservative) when projecting and refining.

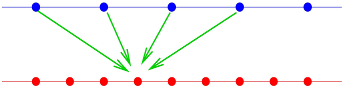
Grid hierarchy

- **Based on point values:** Points in the grid hierarchy: $x_i^l = i\Delta_0/2^l$, $i = 0, \dots, N_0 2^l$.
- Since $x_{2i}^{l+1} = x_i^l$ (even indexed points in level $l+1$ are aligned with points in level l), project solution by just copying even indexed values

$$\mathrm{Proj}_{l+1\to l}(u^{l+1})_i = u_{2i}^{l+1}, \quad i = 0, \dots, N_0 2^l.$$



fine $\to$ coarse



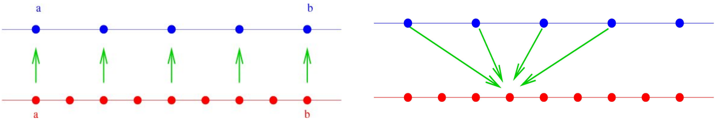
coarse $\to$ fine

- Loss of information (not conservative) when projecting and refining.

Grid hierarchy

- **Based on point values:** Points in the grid hierarchy: $x_i^l = i\Delta_0/2^l$, $i = 0, \ldots, N_0 2^l$.
- Since $x_{2i}^{l+1} = x_i^l$ (even indexed points in level $l+1$ are aligned with points in level l), project solution by just copying even indexed values

$$\text{Proj}_{l+1\to l}(u^{l+1})_i = u_{2i}^{l+1}, \quad i = 0, \ldots, N_0 2^l.$$

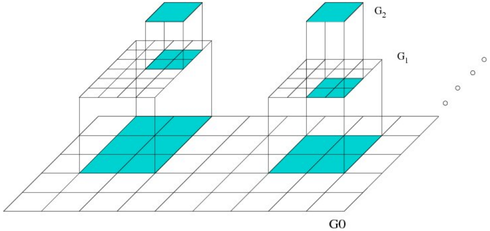


fine → coarse　　coarse → fine

- Loss of information (not conservative) when projecting and refining.

AMR algorithm

- Nested grids as in 2D example with 2 levels. In a time snapshot we have data where marked. All the data is available at level 0.



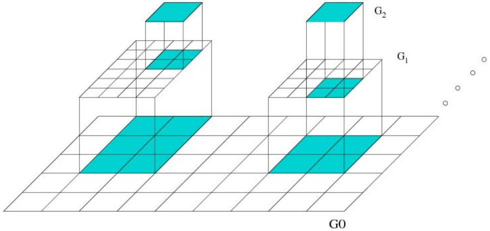
(surrounding band not shown)

- AMR algorithm $\equiv$ **"time evolution" of grid functions** $(u_0^{t_0}, G_0^{t_0}), \dots, (u_L^{t_L}, G_L^{t_L})$ with data $u_l^{t_l} = (u_{l,i}^{t_l} / i \in G_l^{t_l})$ attached to grid points indexed by subsets $G_l^{t_l}$ and associated to times $t_0 \geq t_1 \geq \cdots \geq t_L$ (coarser levels evolve "faster" to provide interpolation data to finer levels)

$$u_{l,i}^{t_l} \approx \begin{cases} u(x_{l,i}, t_l) & \text{point values} \\ \fint_{x_{l,i-\frac{1}{2}}}^{x_{l,i+\frac{1}{2}}} u(x, t_l)dx & \text{cell averages} \end{cases}$$

AMR algorithm

- Nested grids as in 2D example with 2 levels. In a time snapshot we have data where marked. All the data is available at level 0.



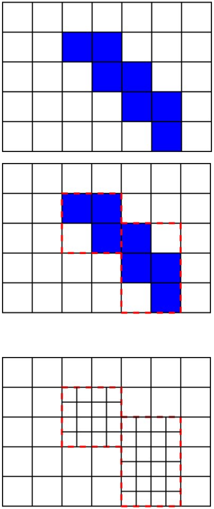
(surrounding band not shown)

- AMR algorithm $\equiv$ **"time evolution"** of **grid functions** $(u_0^{t_0}, G_0^{t_0}), \dots, (u_L^{t_L}, G_L^{t_L})$ with data $u_l^{t_l} = (u_{l,i}^{t_l} / i \in G_l^{t_l})$ attached to grid points indexed by subsets $G_l^{t_l}$ and associated to times $t_0 \geq t_1 \geq \cdots \geq t_L$ (coarser levels evolve "faster" to provide interpolation data to finer levels)

$$u_{l,i}^{t_l} \approx \begin{cases} u(x_{l,i}, t_l) & \text{point values} \\ ⨍_{x_{l,i-\frac{1}{2}}}^{x_{l,i+\frac{1}{2}}} u(x, t_l)dx & \text{cell averages} \end{cases}$$

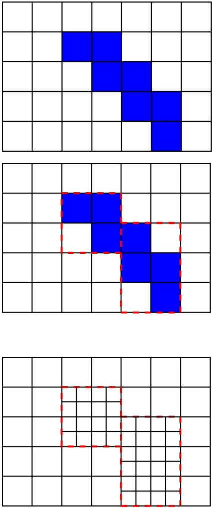
Grid adaption

- Index sets $G_l^{t_l}$ have to evolve in time to track ROI.
- Coarse cells are **marked**, including surrounding band (not shown here), by some **criterion**.
- Marked coarse cells are then **grouped into rectangular patches**, with the **goal** of having (relatively) few large patches for efficiency.
- Coarse cells in rectangular patches are finally **refined**.



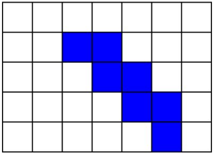
Grid adaption

- Index sets $G_l^{t_l}$ have to evolve in time to track ROI.
- Coarse cells are **marked**, including surrounding band (not shown here), by some **criterion**.
- Marked coarse cells are then **grouped into rectangular patches**, with the **goal** of having (relatively) few large patches for efficiency.
- Coarse cells in rectangular patches are finally **refined**.

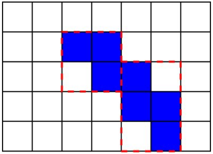


Grid adaption

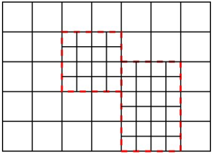
- Index sets $G_l^{t_l}$ have to evolve in time to track ROI.



- Coarse cells are **marked**, including surrounding band (not shown here), by some **criterion**.



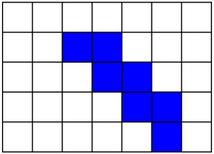
- Marked coarse cells are then **grouped into rectangular patches**, with the **goal** of having (relatively) few large patches for efficiency.



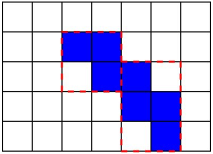
- Coarse cells in rectangular patches are finally **refined**.

Grid adaption

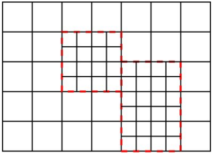
- Index sets $G_l^{t_l}$ have to evolve in time to track ROI.



- Coarse cells are **marked**, including surrounding band (not shown here), by some **criterion**.



- Marked coarse cells are then **grouped into rectangular patches**, with the **goal** of having (relatively) few large patches for efficiency.



- Coarse cells in rectangular patches are finally **refined**.

Criteria for marking for refinement

- Crucial part of algorithm: decide which cells should be refined so as salient flow features are contained in properly refined patches.
- Cells are marked by thresholding based on:
- Large **local truncation errors** [Berger and Oliger, 1984], $\cdots$:
  - Not easy to implement.
- Large **gradients** [Quirk, 1996] $\cdots$
  - Easy, but thresholding is difficult to control (e.g. in rarefactions)
- Large **interpolation errors** (related to wavelet coefficient thresholding [Cohen et al., 2003], refine cells that cannot be accurately predicted)
  - Relatively easy implementation and thresholding.
  - Need improvement: may be combine with large threshold on derivatives of solution, do statistics of interpolation errors for automatic thresholding.

Criteria for marking for refinement

- Crucial part of algorithm: decide which cells should be refined so as salient flow features are contained in properly refined patches.
- Cells are marked by thresholding based on:
- Large **local truncation errors** [Berger and Oliger, 1984], $\cdots$:
  - Not easy to implement.
- Large **gradients** [Quirk, 1996] $\cdots$
  - Easy, but thresholding is difficult to control (e.g. in rarefactions)
- Large **interpolation errors** (related to wavelet coefficient thresholding [Cohen et al., 2003], refine cells that cannot be accurately predicted)
  - Relatively easy implementation and thresholding.
  - Need improvement: may be combine with large threshold on derivatives of solution, do statistics of interpolation errors for automatic thresholding.

Criteria for marking for refinement

- Crucial part of algorithm: decide which cells should be refined so as salient flow features are contained in properly refined patches.
- Cells are marked by thresholding based on:
- Large **local truncation errors** [Berger and Oliger, 1984], $\cdots$:
  - Not easy to implement.
- Large **gradients** [Quirk, 1996] $\cdots$
  - Easy, but thresholding is difficult to control (e.g. in rarefactions)
- Large **interpolation errors** (related to wavelet coefficient thresholding [Cohen et al., 2003], refine cells that cannot be accurately predicted)
  - Relatively easy implementation and thresholding.
  - Need improvement: may be combine with large threshold on derivatives of solution, do statistics of interpolation errors for automatic thresholding, $\cdots$

Criteria for marking for refinement

- Crucial part of algorithm: decide which cells should be refined so as salient flow features are contained in properly refined patches.
- Cells are marked by thresholding based on:
- Large **local truncation errors** [Berger and Oliger, 1984], $\cdots$:
  - Not easy to implement.
- Large **gradients** [Quirk, 1996] $\cdots$
  - Easy, but thresholding is difficult to control (e.g. in rarefactions)
- Large **interpolation errors** (related to wavelet coefficient thresholding [Cohen et al., 2003], refine cells that cannot be accurately predicted)
  - Relatively easy implementation and thresholding.
  - Need improvement: may be combine with large threshold on derivatives of solution, do statistics of interpolation errors for automatic thresholding.

Criteria for marking for refinement

- Crucial part of algorithm: decide which cells should be refined so as salient flow features are contained in properly refined patches.
- Cells are marked by thresholding based on:
- Large **local truncation errors** [Berger and Oliger, 1984], $\cdots$:
  - Not easy to implement.
- Large **gradients** [Quirk, 1996] $\cdots$
  - Easy, but thresholding is difficult to control (e.g. in rarefactions)
- Large **interpolation errors** (related to wavelet coefficient thresholding [Cohen et al., 2003], refine cells that cannot be accurately predicted)
  - Relatively easy implementation and thresholding.
  - Need improvement: may be combine with large threshold on derivatives of solution, do statistics of interpolation errors for automatic thresholding, ...

Criteria for marking for refinement

- Crucial part of algorithm: decide which cells should be refined so as salient flow features are contained in properly refined patches.
- Cells are marked by thresholding based on:
- Large **local truncation errors** [Berger and Oliger, 1984], $\cdots$:
  - Not easy to implement.
- Large **gradients** [Quirk, 1996] $\cdots$
  - Easy, but thresholding is difficult to control (e.g. in rarefactions)
- Large **interpolation errors** (related to wavelet coefficient thresholding [Cohen et al., 2003], refine cells that cannot be accurately predicted)
  - Relatively easy implementation and thresholding.
  - Need improvement: may be combine with large threshold on derivatives of solution, do statistics of interpolation errors for automatic thresholding, . . .

Criteria for marking for refinement

- Crucial part of algorithm: decide which cells should be refined so as salient flow features are contained in properly refined patches.
- Cells are marked by thresholding based on:
- Large **local truncation errors** [Berger and Oliger, 1984], $\cdots$:
  - Not easy to implement.
- Large **gradients** [Quirk, 1996] $\cdots$
  - Easy, but thresholding is difficult to control (e.g. in rarefactions)
- Large **interpolation errors** (related to wavelet coefficient thresholding [Cohen et al., 2003], refine cells that cannot be accurately predicted)
  - Relatively easy implementation and thresholding.
  - Need improvement: may be combine with large threshold on derivatives of solution, do statistics of interpolation errors for automatic thresholding, $\ldots$.

Criteria for marking for refinement

- Crucial part of algorithm: decide which cells should be refined so as salient flow features are contained in properly refined patches.
- Cells are marked by thresholding based on:
- Large **local truncation errors** [Berger and Oliger, 1984], $\cdots$:
  - Not easy to implement.
- Large **gradients** [Quirk, 1996] $\cdots$
  - Easy, but thresholding is difficult to control (e.g. in rarefactions)
- Large **interpolation errors** (related to wavelet coefficient thresholding [Cohen et al., 2003], refine cells that cannot be accurately predicted)
  - Relatively easy implementation and thresholding.
  - Need improvement: may be combine with large threshold on derivatives of solution, do statistics of interpolation errors for automatic thresholding, ...

Criteria for marking for refinement

- Crucial part of algorithm: decide which cells should be refined so as salient flow features are contained in properly refined patches.
- Cells are marked by thresholding based on:
- Large **local truncation errors** [Berger and Oliger, 1984], $\cdots$:
  - Not easy to implement.
- Large **gradients** [Quirk, 1996] $\cdots$
  - Easy, but thresholding is difficult to control (e.g. in rarefactions)
- Large **interpolation errors** (related to wavelet coefficient thresholding [Cohen et al., 2003], refine cells that cannot be accurately predicted)
  - Relatively easy implementation and thresholding.
  - Need improvement: may be combine with large threshold on derivatives of solution, do statistics of interpolation errors for automatic thresholding, ...

Outline

1. Shock capturing schemes for Shallow water flows
2. Adaptive Mesh Refinement
   - Adaptive schemes
   - Grid hierarchy
3. Well-balanced Adaptive techniques
   - Well-balanced schemes
   - Well-balanced AMR
   - Homogeneous discretization for SWE
   - Well-balanced interpolation
4. Numerical results
   - Numerical results
5. Conclusions

Well-balanced schemes

- The convergence of the scheme is usually proved (when possible) through its consistence and stability (this being the harder part).
- When converging to a steady state or dealing with quasi-stationary solutions, the requirement of **preserving steady states** is plausible.
- When the scheme

$$u_i^{n+1} = u_i^n - \Delta t \Big(\frac{\hat{f}_{i+1/2}^n - \hat{f}_{i-1/2}^n}{\Delta x} - s_i^n \Big)$$

does so, that is:

$$f(u(x))_x = s(x, u(x)) \Longrightarrow \left[\frac{\hat{f}_{i+1/2}^n - \hat{f}_{i-1/2}^n}{\Delta x} - s_i^n \right] (u(x)) = 0$$

then the scheme is termed **well-balanced** [Greenberg and Leroux, 1996].
- Special steady state for SWE, **water at rest** ($h + z$ =constant, $v = 0$).
- If a scheme preserves this steady state solution, then the scheme is said to verify the **C-property** [Bermudez and Vazquez, 1994].

Well-balanced schemes

- The convergence of the scheme is usually proved (when possible) through its consistence and stability (this being the harder part).
- When converging to a steady state or dealing with quasi-stationary solutions, the requirement of **preserving steady states** is plausible.
- When the scheme

$$u_i^{n+1} = u_i^n - \Delta t\left(\frac{\hat{f}^n_{i+1/2} - \hat{f}^n_{i-1/2}}{\Delta x} - s_i^n\right)$$

  does so, that is:

$$f(u(x))_x = s(x, u(x)) \Longrightarrow \left[\frac{\hat{f}^n_{i+1/2} - \hat{f}^n_{i-1/2}}{\Delta x} - s_i^n\right](u(x)) = 0$$

  then the scheme is termed **well-balanced** [Greenberg and Leroux, 1996].
- Special steady state for SWE, **water at rest** ($h + z =$constant, $v = 0$).
- If a scheme preserves this steady state solution, then the scheme is said to verify the **C-property** [Bermudez and Vazquez, 1994].

Well-balanced schemes

- The convergence of the scheme is usually proved (when possible) through its consistence and stability (this being the harder part).
- When converging to a steady state or dealing with quasi-stationary solutions, the requirement of **preserving steady states** is plausible.
- When the scheme

$$u_i^{n+1} = u_i^n - \Delta t \left(\frac{\hat{f}_{i+1/2}^n - \hat{f}_{i-1/2}^n}{\Delta x} - s_i^n \right)$$

does so, that is:

$$f(u(x))_x = s(x, u(x)) \Longrightarrow \left[\frac{\hat{f}_{i+1/2}^n - \hat{f}_{i-1/2}^n}{\Delta x} - s_i^n \right] (u(x)) = 0$$

then the scheme is termed **well-balanced** [Greenberg and Leroux, 1996].
- Special steady state for SWE, **water at rest** ($h + z =$constant, $v = 0$).
- If a scheme preserves this steady state solution, then the scheme is said to verify the **C-property** [Bermudez and Vazquez, 1994].

Well-balanced schemes

- The convergence of the scheme is usually proved (when possible) through its consistence and stability (this being the harder part).
- When converging to a steady state or dealing with quasi-stationary solutions, the requirement of **preserving steady states** is plausible.
- When the scheme

$$u_i^{n+1} = u_i^n - \Delta t\left(\frac{\hat{f}^n_{i+1/2} - \hat{f}^n_{i-1/2}}{\Delta x} - s_i^n\right)$$

  does so, that is:

$$f(u(x))_x = s(x, u(x)) \Longrightarrow \left[\frac{\hat{f}^n_{i+1/2} - \hat{f}^n_{i-1/2}}{\Delta x} - s_i^n\right](u(x)) = 0$$

  then the scheme is termed **well-balanced** [Greenberg and Leroux, 1996].
- Special steady state for SWE, **water at rest** ($h + z =$constant, $v = 0$).
- If a scheme preserves this steady state solution, then the scheme is said to verify the **C-property** [Bermudez and Vazquez, 1994].

Well-balanced schemes

- The convergence of the scheme is usually proved (when possible) through its consistence and stability (this being the harder part).
- When converging to a steady state or dealing with quasi-stationary solutions, the requirement of **preserving steady states** is plausible.
- When the scheme

$$u_i^{n+1} = u_i^n - \Delta t \left(\frac{\hat{f}^n_{i+1/2} - \hat{f}^n_{i-1/2}}{\Delta x} - s_i^n \right)$$

  does so, that is:

$$f(u(x))_x = s(x, u(x)) \Longrightarrow \left[\frac{\hat{f}^n_{i+1/2} - \hat{f}^n_{i-1/2}}{\Delta x} - s_i^n \right] (u(x)) = 0$$

  then the scheme is termed **well-balanced** [Greenberg and Leroux, 1996].
- Special steady state for SWE, **water at rest** ($h + z =$constant, $v = 0$).
- If a scheme preserves this steady state solution, then the scheme is said to verify the **C-property** [Bermudez and Vazquez, 1994].

Outline

1. Shock capturing schemes for Shallow water flows
2. Adaptive Mesh Refinement
   - Adaptive schemes
   - Grid hierarchy
3. Well-balanced Adaptive techniques
   - Well-balanced schemes
   - Well-balanced AMR
   - Homogeneous discretization for SWE
   - Well-balanced interpolation
4. Numerical results
   - Numerical results
5. Conclusions

Well-balanced AMR

- AMR with well-balanced solver:
  [Berger-Calhoun-Helzel-LeVeque, 2009, George, 2011].
- Goal: obtain AMR code that preserves steady states (at least water at rest).
- If AMR algorithm should preserve stationary solutions then its ingredients:
  - Single grid solver (basic scheme)
  - Coarse to fine communication (interpolation).
  - Fine to coarse communication (projection).

  should preserve them (mentioned in D. George's talk) $\Rightarrow$
  need well-balanced interpolation ([Bouchut, 2004]) and projection.
- We apply these adaptive techniques to a scheme introduced in [Donat and Martínez-Gavara, 2011] that satisfies the exact C-property. These techniques are applicable to other well-balanced schemes.

Well-balanced AMR

- AMR with well-balanced solver:
  [Berger-Calhoun-Helzel-LeVeque, 2009, George, 2011].
- Goal: obtain AMR code that preserves steady states (at least water at rest).
- If AMR algorithm should preserve stationary solutions then its ingredients:
  - Single grid solver (basic scheme)
  - Coarse to fine communication (interpolation).
  - Fine to coarse communication (projection).

  should preserve them (mentioned in D. George's talk) $\Rightarrow$
  need well-balanced interpolation ([Bouchut, 2004]) and projection.
- We apply these adaptive techniques to a scheme introduced in [Donat and Martínez-Gavara, 2011] that satisfies the exact C-property. These techniques are applicable to other well-balanced schemes.

Well-balanced AMR

- AMR with well-balanced solver: [Berger-Calhoun-Helzel-LeVeque, 2009, George, 2011].
- Goal: obtain AMR code that preserves steady states (at least water at rest).
- If AMR algorithm should preserve stationary solutions then its ingredients:
  - Single grid solver (basic scheme)
  - Coarse to fine communication (interpolation).
  - Fine to coarse communication (projection).

  should preserve them (mentioned in D. George's talk) $\Rightarrow$
  need well-balanced interpolation ([Bouchut, 2004]) and projection.
- We apply these adaptive techniques to a scheme introduced in [Donat and Martínez-Gavara, 2011] that satisfies the exact C-property. These techniques are applicable to other well-balanced schemes.

Well-balanced AMR

- AMR with well-balanced solver:
  [Berger-Calhoun-Helzel-LeVeque, 2009, George, 2011].
- Goal: obtain AMR code that preserves steady states (at least water at rest).
- If AMR algorithm should preserve stationary solutions then its ingredients:
  - Single grid solver (basic scheme)
  - Coarse to fine communication (interpolation).
  - Fine to coarse communication (projection).

  should preserve them (mentioned in D. George's talk) $\Rightarrow$
  need well-balanced interpolation ([Bouchut, 2004]) and projection.
- We apply these adaptive techniques to a scheme introduced in [Donat and Martínez-Gavara, 2011] that satisfies the exact C-property. These techniques are applicable to other well-balanced schemes.

Outline

1. Shock capturing schemes for Shallow water flows
2. Adaptive Mesh Refinement
   - Adaptive schemes
   - Grid hierarchy
3. Well-balanced Adaptive techniques
   - Well-balanced schemes
   - Well-balanced AMR
   - Homogeneous discretization for SWE
   - Well-balanced interpolation
4. Numerical results
   - Numerical results
5. Conclusions

Homogeneous discretization

- We build on [Gascón and Corberán, 2001, Caselles-Donat-Haro, 2009, Donat and Martínez-Gavara, 2011]: PDE can be rewritten in “homogeneous” form:

$$u_t + f(u)_x = s(x,u) \Leftrightarrow u_t + g[u]_x = 0$$

where the **functional** g (dependent on f and s) acts on $u = u(x,t)$ as:

$$g[u](x,t) = f(u(x,t)) - \int_0^x s(r, u(r,t))\, dr$$

- We can derive upwind numerical methods for **non-homogeneous** conservation law from well established techniques for **homogeneous** conservation laws.

Homogeneous discretization

- We build on [Gascón and Corberán, 2001, Caselles-Donat-Haro, 2009, Donat and Martínez-Gavara, 2011]: PDE can be rewritten in "homogeneous" form:

  $$u_t + f(u)_x = s(x,u) \Leftrightarrow u_t + g[u]_x = 0$$

  where the **functional** g (dependent on f and s) acts on $u = u(x,t)$ as:

  $$g[u](x,t) = f(u(x,t)) - \int_0^x s(r, u(r,t))\, dr$$

- We can derive upwind numerical methods for **non-homogeneous** conservation law from well established techniques for **homogeneous** conservation laws.

Homogeneous discretization

- [Donat and Martínez-Gavara, 2011] propose a **Lax-Wendroff**-type finite differences discretization for $u_t + g[u]_x = 0$, which is hybridized with a first order monotone scheme through **flux-limiting** techniques.
- The scheme applied to exact solution $u(x,t)$ is:

$$u_i^{n+1} = u_i^n - \frac{\Delta t}{\Delta x}(\overbrace{A_i^n \Delta g_{i-\frac{1}{2}}^n + B_i^n \Delta g_{i+\frac{1}{2}}^n}^{G_{i+\frac{1}{2}}^n - G_{i-\frac{1}{2}}^n})$$

where $G_{i+\frac{1}{2}}$ are numerical fluxes for $g[u]$ and:

$$g_i^n = g[u](x_i, t_n) = f(u(x_i,t_n)) - \int_0^{x_i} s(r, u(r,t_n))dr$$

$$\Delta g_{i+\frac{1}{2}}^n = g_{i+1}^n - g_i^n = f(u(x_{i+1},t_n)) - f(u(x_i,t_n)) + b_{i,i+1}^n,$$

where

$$b_{i,i+1}^n = -\int_{x_i}^{x_{i+1}} s(r, u(r,t_n))dr$$

Homogeneous discretization

- [Donat and Martínez-Gavara, 2011] propose a **Lax-Wendroff**-type finite differences discretization for $u_t + g[u]_x = 0$, which is hybridized with a first order monotone scheme through **flux-limiting** techniques.
- The scheme applied to exact solution $u(x,t)$ is:

$$u_i^{n+1} = u_i^n - \frac{\Delta t}{\Delta x}(\overbrace{A_i^n \Delta g_{i-\frac{1}{2}}^n + B_i^n \Delta g_{i+\frac{1}{2}}^n}^{G_{i+\frac{1}{2}}^n - G_{i-\frac{1}{2}}^n})$$

where $G_{i+\frac{1}{2}}$ are numerical fluxes for $g[u]$ and:

$$g_i^n = g[u](x_i, t_n) = f(u(x_i, t_n)) - \int_0^{x_i} s(r, u(r, t_n))dr$$

$$\Delta g_{i+\frac{1}{2}}^n = g_{i+1}^n - g_i^n = f(u(x_{i+1}, t_n)) - f(u(x_i, t_n)) + b_{i,i+1}^n,$$

where

$$b_{i,i+1}^n = -\int_{x_i}^{x_{i+1}} s(r, u(r, t_n))dr$$

Homogeneous discretization

- To get numerical method, need to approximate

$$b_{i,i+1}^n = -\int_{x_i}^{x_{i+1}} s(r, u(r, t_n))dr$$

by some appropriate quadrature rule, $\hat{b}_{i,i+1}^n \approx b_{i,i+1}^n$, so final scheme is

$$u_i^{n+1} = u_i^n - \frac{\Delta t}{\Delta x}(A_i^n \widehat{\Delta g_{i-\frac{1}{2}}^n} + B_i^n \widehat{\Delta g_{i+\frac{1}{2}}^n})$$

$$\Delta g_{i+\frac{1}{2}}^n \approx \widehat{\Delta g_{i+\frac{1}{2}}^n} := f(u_{i+1}^n) - f(u_i^n) + \hat{b}_{i,i+1}^n.$$

- Well balancing is obtained if approximation $\hat{b}_{i,i+1}^n \approx b_{i,i+1}^n$ is exact:

$$f(u(x))_x = s(x, u(x)) \Rightarrow g[u]_x = 0 \Rightarrow g_i^n = g[u](x_i, t_n) = \text{constant} \Rightarrow$$

$$\widehat{\Delta g_{i+\frac{1}{2}}^n} = \Delta g_{i+\frac{1}{2}}^n = g_{i+1}^n - g_i^n = 0, \forall i \Rightarrow u_i^{n+1} = u_i^n, \forall i$$

- For SWE, suitable $\hat{b}_{i,i+1}^n$ can be defined to get **exact C-property** for wet and wet/dry beds. The exactness of $\hat{b}_{i,i+1}^n$ heavily relies on the scheme being based on point-values.

Homogeneous discretization

- To get numerical method, need to approximate

$$b_{i,i+1}^n = -\int_{x_i}^{x_{i+1}} s(r, u(r, t_n))dr$$

by some appropriate quadrature rule, $\hat{b}_{i,i+1}^n \approx b_{i,i+1}^n$, so final scheme is

$$u_i^{n+1} = u_i^n - \frac{\Delta t}{\Delta x}(A_i^n \widehat{\Delta g_{i-\frac{1}{2}}^n} + B_i^n \widehat{\Delta g_{i+\frac{1}{2}}^n})$$

$$\Delta g_{i+\frac{1}{2}}^n \approx \widehat{\Delta g_{i+\frac{1}{2}}^n} := f(u_{i+1}^n) - f(u_i^n) + \hat{b}_{i,i+1}^n.$$

- Well balancing is obtained if approximation $\hat{b}_{i,i+1}^n \approx b_{i,i+1}^n$ is exact:

$$f(u(x))_x = s(x, u(x)) \Rightarrow g[u]_x = 0 \Rightarrow g_i^n = g[u](x_i, t_n) = \text{constant} \Rightarrow$$
$$\widehat{\Delta g_{i+\frac{1}{2}}^n} = \Delta g_{i+\frac{1}{2}}^n = g_{i+1}^n - g_i^n = 0, \forall i \Rightarrow u_i^{n+1} = u_i^n, \forall i$$

- For SWE, suitable $\hat{b}_{i,i+1}^n$ can be defined to get **exact C-property** for wet and wet/dry beds. The exactness of $\hat{b}_{i,i+1}^n$ heavily relies on the scheme being based on point-values.

Homogeneous discretization

- To get numerical method, need to approximate

$$b^n_{i,i+1} = -\int_{x_i}^{x_{i+1}} s(r, u(r, t_n))dr$$

by some appropriate quadrature rule, $\hat{b}^n_{i,i+1} \approx b^n_{i,i+1}$, so final scheme is

$$u_i^{n+1} = u_i^n - \frac{\Delta t}{\Delta x}\left(A_i^n \widehat{\Delta g^n_{i-\frac{1}{2}}} + B_i^n \widehat{\Delta g^n_{i+\frac{1}{2}}}\right)$$

$$\Delta g^n_{i+\frac{1}{2}} \approx \widehat{\Delta g^n_{i+\frac{1}{2}}} := f(u^n_{i+1}) - f(u^n_i) + \hat{b}^n_{i,i+1}.$$

- Well balancing is obtained if approximation $\hat{b}^n_{i,i+1} \approx b^n_{i,i+1}$ is exact:

$$f(u(x))_x = s(x, u(x)) \Rightarrow g[u]_x = 0 \Rightarrow g^n_i = g[u](x_i, t_n) = \text{constant} \Rightarrow$$

$$\widehat{\Delta g^n_{i+\frac{1}{2}}} = \Delta g^n_{i+\frac{1}{2}} = g^n_{i+1} - g^n_i = 0, \forall i \Rightarrow u_i^{n+1} = u_i^n, \forall i$$

- For SWE, suitable $\hat{b}^n_{i,i+1}$ can be defined to get **exact C-property** for wet and wet/dry beds. The exactness of $\hat{b}^n_{i,i+1}$ heavily relies on the scheme being based on point-values.

Outline

1 Shock capturing schemes for Shallow water flows

2 Adaptive Mesh Refinement
- Adaptive schemes
- Grid hierarchy

3 Well-balanced Adaptive techniques
- Well-balanced schemes
- Well-balanced AMR
- Homogeneous discretization for SWE
- Well-balanced interpolation

4 Numerical results
- Numerical results

5 Conclusions

C-property preserving interpolation: cell-averages

- In cell-based grid hierarchy, projection is given by $h_{i+\frac{1}{2}} = \frac{1}{2}(h_i + h_{i+1})$, where indexes indicate the point the data is attached to.
- If $h_i = h(x_i)$ correspond to a water at rest solution, does $h_{i+\frac{1}{2}} = \frac{1}{2}(h_i + h_{i+1})$ correspond to point values (at $x_{i+\frac{1}{2}}$) of the solution?
- If it were so, from $h(x) = \eta - z(x)$ we get

$$h_{i+\frac{1}{2}} = h(x_{i+\frac{1}{2}}) = \eta - z(x_{i+\frac{1}{2}}),$$

but

$$h_{i+\frac{1}{2}} = \frac{1}{2}\Big(h(x_i) + h(x_{i+1})\Big) = \eta - \frac{1}{2}\Big(z(x_i) + z(x_{i+1})\Big)w$$

so z should verify

$$\frac{z(x_i) + z(x_{i+1})}{2} = z\Big(\frac{x_i + x_{i+1}}{2}\Big), \forall i,$$

which does not hold for general $z \Rightarrow$ Projection not OK for point values
- Projection OK if h_i are cell-averages of stationary solution, but then underlying scheme should preserve them (OK for well-balanced schemes as in Carlos Parés' course, not OK for our scheme).

C-property preserving interpolation: cell-averages

- In cell-based grid hierarchy, projection is given by $h_{i+\frac{1}{2}} = \frac{1}{2}(h_i + h_{i+1})$, where indexes indicate the point the data is attached to.
- If $h_i = h(x_i)$ correspond to a water at rest solution, does $h_{i+\frac{1}{2}} = \frac{1}{2}(h_i + h_{i+1})$ correspond to point values (at $x_{i+\frac{1}{2}}$) of the solution?
- If it were so, from $h(x) = \eta - z(x)$ we get

$$h_{i+\frac{1}{2}} = h(x_{i+\frac{1}{2}}) = \eta - z(x_{i+\frac{1}{2}}),$$

but

$$h_{i+\frac{1}{2}} = \frac{1}{2}\Big(h(x_i) + h(x_{i+1})\Big) = \eta - \frac{1}{2}\Big(z(x_i) + z(x_{i+1})\Big)w$$

so z should verify

$$\frac{z(x_i) + z(x_{i+1})}{2} = z\Big(\frac{x_i + x_{i+1}}{2}\Big), \forall i,$$

which does not hold for general $z \Rightarrow$ Projection not OK for point values
- Projection OK if h_i are cell-averages of stationary solution, but then underlying scheme should preserve them (OK for well-balanced schemes as in Carlos Parés' course, not OK for our scheme).

C-property preserving interpolation: cell-averages

- In cell-based grid hierarchy, projection is given by $h_{i+\frac{1}{2}} = \frac{1}{2}(h_i + h_{i+1})$, where indexes indicate the point the data is attached to.
- If $h_i = h(x_i)$ correspond to a water at rest solution, does $h_{i+\frac{1}{2}} = \frac{1}{2}(h_i + h_{i+1})$ correspond to point values (at $x_{i+\frac{1}{2}}$) of the solution?
- If it were so, from $h(x) = \eta - z(x)$ we get

$$h_{i+\frac{1}{2}} = h(x_{i+\frac{1}{2}}) = \eta - z(x_{i+\frac{1}{2}}),$$

  but

$$h_{i+\frac{1}{2}} = \frac{1}{2}\Big(h(x_i) + h(x_{i+1})\Big) = \eta - \frac{1}{2}\Big(z(x_i) + z(x_{i+1})\Big)w$$

  so z should verify

$$\frac{z(x_i) + z(x_{i+1})}{2} = z\left(\frac{x_i + x_{i+1}}{2}\right), \forall i,$$

  which does not hold for general $z \Rightarrow$ Projection not OK for point values
- Projection OK if h_i are cell-averages of stationary solution, but then underlying scheme should preserve them (OK for well-balanced schemes as in Carlos Parés' course, not OK for our scheme).

C-property preserving interpolation: cell-averages

- In cell-based grid hierarchy, projection is given by $h_{i+\frac{1}{2}} = \frac{1}{2}(h_i + h_{i+1})$, where indexes indicate the point the data is attached to.
- If $h_i = h(x_i)$ correspond to a water at rest solution, does $h_{i+\frac{1}{2}} = \frac{1}{2}(h_i + h_{i+1})$ correspond to point values (at $x_{i+\frac{1}{2}}$) of the solution?
- If it were so, from $h(x) = \eta - z(x)$ we get

$$h_{i+\frac{1}{2}} = h(x_{i+\frac{1}{2}}) = \eta - z(x_{i+\frac{1}{2}}),$$

but

$$h_{i+\frac{1}{2}} = \frac{1}{2}\Big(h(x_i) + h(x_{i+1})\Big) = \eta - \frac{1}{2}\Big(z(x_i) + z(x_{i+1})\Big)w$$

so z should verify

$$\frac{z(x_i) + z(x_{i+1})}{2} = z\Big(\frac{x_i + x_{i+1}}{2}\Big), \forall i,$$

which does not hold for general $z \Rightarrow$ Projection not OK for point values
- Projection OK if h_i are cell-averages of stationary solution, but then underlying scheme should preserve them (OK for well-balanced schemes as in Carlos Parés' course, not OK for our scheme).

C-property preserving interpolation: point-values

- For point value grid hierarchy, the projection from level $l+1$ to level l is given by copying values with even indexes, corresponding to the same point-values, so this projection is automatically well-balanced.
- **Well-balanced** interpolation (related to hydrostatic reconstruction [Audusse-Bouchut-Bristeau-Klein-Perthame, 2004], appears in Carlos Pare's course and Professor Valiani's talk): if we only want to preserve water at rest solutions, given interpolator $I((w_i);x)$ (i.e., $I((w_i);x_j)=w_j$), and

$$V(x,\begin{bmatrix}h\\q\end{bmatrix})=\begin{bmatrix}h+z(x)\\q\end{bmatrix},\quad V(x,\cdot)^{-1}\begin{bmatrix}\eta\\q\end{bmatrix}=\begin{bmatrix}\eta-z(x)\\q\end{bmatrix}$$

  then we can define an interpolator by

$$\tilde{I}((u_i);x)=V(x,\cdot)^{-1}(I((V_i);x)),\quad V_i=V(x_i,u_i)$$

  (i.e.,interpolate total heights, then subtract bottom height).
- I preserves constants $\Rightarrow \tilde{I}$ preserves water at rest.
- Could extend $\tilde{I}$ to cell-averages by changing $z(x)$ by cell-average of z and I by a cell-average interpolator.

C-property preserving interpolation: point-values

- For point value grid hierarchy, the projection from level $l+1$ to level l is given by copying values with even indexes, corresponding to the same point-values, so this projection is automatically well-balanced.
- **Well-balanced** interpolation (related to hydrostatic reconstruction [Audusse-Bouchut-Bristeau-Klein-Perthame, 2004], appears in Carlos Pare's course and Professor Valiani's talk): if we only want to preserve water at rest solutions, given interpolator $I((w_i);x)$ (i.e., $I((w_i);x_j)=w_j$), and

$$V(x, \begin{bmatrix} h \\ q \end{bmatrix}) = \begin{bmatrix} h+z(x) \\ q \end{bmatrix}, \quad V(x,\cdot)^{-1} \begin{bmatrix} \eta \\ q \end{bmatrix} = \begin{bmatrix} \eta - z(x) \\ q \end{bmatrix}$$

  then we can define an interpolator by

$$\tilde{I}((u_i);x) = V(x,\cdot)^{-1}(I((V_i);x)), \quad V_i = V(x_i,u_i)$$

  (i.e., interpolate total heights, then subtract bottom height).
- I preserves constants $\Rightarrow \tilde{I}$ preserves water at rest.
- Could extend $\tilde{I}$ to cell-averages by changing $z(x)$ by cell-average of z and I by a cell-average interpolator.

C-property preserving interpolation: point-values

- For point value grid hierarchy, the projection from level $l+1$ to level l is given by copying values with even indexes, corresponding to the same point-values, so this projection is automatically well-balanced.
- **Well-balanced** interpolation (related to hydrostatic reconstruction [Audusse-Bouchut-Bristeau-Klein-Perthame, 2004], appears in Carlos Pare's course and Professor Valiani's talk): if we only want to preserve water at rest solutions, given interpolator $I((w_i);x)$ (i.e., $I((w_i);x_j)=w_j$), and

$$V(x, \begin{bmatrix} h \\ q \end{bmatrix}) = \begin{bmatrix} h+z(x) \\ q \end{bmatrix}, \quad V(x,\cdot)^{-1} \begin{bmatrix} \eta \\ q \end{bmatrix} = \begin{bmatrix} \eta - z(x) \\ q \end{bmatrix}$$

  then we can define an interpolator by

$$\tilde{I}((u_i);x) = V(x,\cdot)^{-1}(I((V_i);x)), \quad V_i = V(x_i, u_i)$$

  (i.e.,interpolate total heights, then subtract bottom height).
- I preserves constants $\Rightarrow \tilde{I}$ preserves water at rest.
- Could extend $\tilde{I}$ to cell-averages by changing $z(x)$ by cell-average of z and I by a cell-average interpolator.

C-property preserving interpolation: point-values

- For point value grid hierarchy, the projection from level $l+1$ to level l is given by copying values with even indexes, corresponding to the same point-values, so this projection is automatically well-balanced.
- **Well-balanced** interpolation (related to hydrostatic reconstruction [Audusse-Bouchut-Bristeau-Klein-Perthame, 2004], appears in Carlos Pare's course and Professor Valiani's talk): if we only want to preserve water at rest solutions, given interpolator $I((w_i);x)$ (i.e., $I((w_i);x_j)=w_j$), and

  $$V(x,\begin{bmatrix}h\\q\end{bmatrix})=\begin{bmatrix}h+z(x)\\q\end{bmatrix},\quad V(x,\cdot)^{-1}\begin{bmatrix}\eta\\q\end{bmatrix}=\begin{bmatrix}\eta-z(x)\\q\end{bmatrix}$$

  then we can define an interpolator by

  $$\tilde{I}((u_i);x)=V(x,\cdot)^{-1}(I((V_i);x)),\quad V_i=V(x_i,u_i)$$

  (i.e.,interpolate total heights, then subtract bottom height).
- I preserves constants $\Rightarrow$ $\tilde{I}$ preserves water at rest.
- Could extend $\tilde{I}$ to cell-averages by changing $z(x)$ by cell-average of z and I by a cell-average interpolator.

General well-balanced interpolation

- If we can re-write $f(u)_x = s(x,u)$ as $V(x,u)_x = 0$, then $u(x)$ is solution of PDE $\Leftrightarrow V(x,u(x))$ is constant at regions of smoothness + jump conditions .
- $V(x,u) \equiv$ **equilibrium variables**, which are for SWE:

$$V(x, \begin{bmatrix} h \\ hv \end{bmatrix}) = \begin{bmatrix} \frac{v^2}{2} + g(h+z(x)) \\ hv \end{bmatrix}$$

- If $V(x,\cdot)$ is bijective onto some relevant range then we can define an interpolator that preserves equilibrium variables by:

$$\tilde{I}((u_i);x) = V(x,\cdot)^{-1}(I((V_i);x)), \quad V_i = V(x_i,u_i)$$

- For SWE, $V(x,\cdot)$ is not injective, but could select, as in [Bouchut and Morales de Luna, 2010], appropriate branch of inverse (helped here by the fact that interpolation takes place at smooth regions).
- Could get well-balanced interpolation in the cell-average sense by using techniques that Carlos Parés showed in his course.

General well-balanced interpolation

- If we can re-write $f(u)_x = s(x,u)$ as $V(x,u)_x = 0$, then $u(x)$ is solution of PDE $\Leftrightarrow$ $V(x,u(x))$ is constant at regions of smoothness + jump conditions .
- $V(x,u) \equiv$ **equilibrium variables**, which are for SWE:

$$V(x, \begin{bmatrix} h \\ hv \end{bmatrix}) = \begin{bmatrix} \frac{v^2}{2} + g(h + z(x)) \\ hv \end{bmatrix}$$

- If $V(x,\cdot)$ is bijective onto some relevant range then we can define an interpolator that preserves equilibrium variables by:

$$\tilde{I}((u_i);x) = V(x,\cdot)^{-1}(I((V_i);x)), \quad V_i = V(x_i,u_i)$$

- For SWE, $V(x,\cdot)$ is not injective, but could select, as in [Bouchut and Morales de Luna, 2010], appropriate branch of inverse (helped here by the fact that interpolation takes place at smooth regions).
- Could get well-balanced interpolation in the cell-average sense by using techniques that Carlos Parés showed in his course.

General well-balanced interpolation

- If we can re-write $f(u)_x = s(x,u)$ as $V(x,u)_x = 0$, then $u(x)$ is solution of PDE $\Leftrightarrow V(x,u(x))$ is constant at regions of smoothness + jump conditions .
- $V(x,u) \equiv$ **equilibrium variables**, which are for SWE:

$$V(x, \begin{bmatrix} h \\ hv \end{bmatrix}) = \begin{bmatrix} \frac{v^2}{2} + g(h+z(x)) \\ hv \end{bmatrix}$$

- If $V(x,\cdot)$ is bijective onto some relevant range then we can define an interpolator that preserves equilibrium variables by:

$$\tilde{I}((u_i);x) = V(x,\cdot)^{-1}(I((V_i);x)), \quad V_i = V(x_i,u_i)$$

- For SWE, $V(x,\cdot)$ is not injective, but could select, as in [Bouchut and Morales de Luna, 2010], appropriate branch of inverse (helped here by the fact that interpolation takes place at smooth regions).
- Could get well-balanced interpolation in the cell-average sense by using techniques that Carlos Parés showed in his course.

General well-balanced interpolation

- If we can re-write $f(u)_x = s(x,u)$ as $V(x,u)_x = 0$, then
  $u(x)$ is solution of PDE $\Leftrightarrow V(x,u(x))$ is constant at regions of smoothness + jump conditions .
- $V(x,u) \equiv$ **equilibrium variables**, which are for SWE:

$$V(x, \begin{bmatrix} h \\ hv \end{bmatrix}) = \begin{bmatrix} \frac{v^2}{2} + g(h+z(x)) \\ hv \end{bmatrix}$$

- If $V(x,\cdot)$ is bijective onto some relevant range then we can define an interpolator that preserves equilibrium variables by:

$$\tilde{I}((u_i);x) = V(x,\cdot)^{-1}(I((V_i);x)), \quad V_i = V(x_i,u_i)$$

- For SWE, $V(x,\cdot)$ is not injective, but could select, as in [Bouchut and Morales de Luna, 2010], appropriate branch of inverse (helped here by the fact that interpolation takes place at smooth regions).
- Could get well-balanced interpolation in the cell-average sense by using techniques that Carlos Parés showed in his course.

General well-balanced interpolation

- If we can re-write $f(u)_x = s(x,u)$ as $V(x,u)_x = 0$, then $u(x)$ is solution of PDE $\Leftrightarrow V(x,u(x))$ is constant at regions of smoothness + jump conditions .
- $V(x,u) \equiv$ **equilibrium variables**, which are for SWE:

$$V(x, \begin{bmatrix} h \\ hv \end{bmatrix}) = \begin{bmatrix} \frac{v^2}{2} + g(h + z(x)) \\ hv \end{bmatrix}$$

- If $V(x,\cdot)$ is bijective onto some relevant range then we can define an interpolator that preserves equilibrium variables by:

$$\tilde{I}((u_i); x) = V(x,\cdot)^{-1}(I((V_i); x)), \quad V_i = V(x_i, u_i)$$

- For SWE, $V(x,\cdot)$ is not injective, but could select, as in [Bouchut and Morales de Luna, 2010], appropriate branch of inverse (helped here by the fact that interpolation takes place at smooth regions).
- Could get well-balanced interpolation in the cell-average sense by using techniques that Carlos Parés showed in his course.

Outline

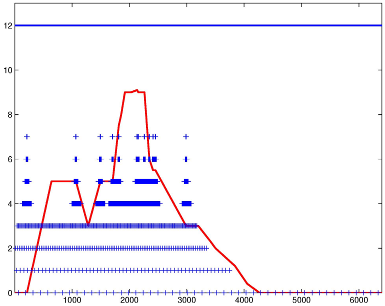
1. Shock capturing schemes for Shallow water flows
2. Adaptive Mesh Refinement
   - Adaptive schemes
   - Grid hierarchy
3. Well-balanced Adaptive techniques
   - Well-balanced schemes
   - Well-balanced AMR
   - Homogeneous discretization for SWE
   - Well-balanced interpolation
4. Numerical results
   - Numerical results
5. Conclusions

Tests setup

- Based on code developed by A. Baeza for cell-based AMR.
- We use point-value-based grid hierarchy, with well-balanced interpolation based on linear interpolation.
- Refinement criterion: mark cells to refine when interpolation error exceeds some relative error **rtol** with respect to the maximal interpolation error at each level.

Test for stationary 1D solutions

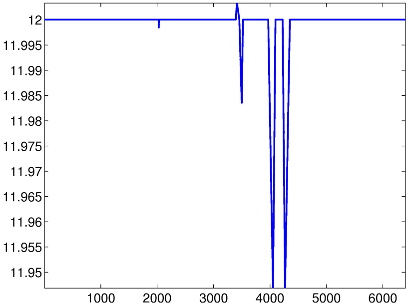
- Water at rest solution of total height=12, bottom topography below. Solution at $T = 200$.
- Have used rtol=10^{-1}, $N_0 = 50$, and eight levels ($L = 7$, $N_7 = 6400$) to obtain:



  with a CPU speedup ≈ 11.5.
- Scheme gives approximated solution such that $||h + z - 12||_\infty = 1.06 \cdot 10^{-14}$ and $||v||_\infty = 3.36 \cdot 10^{-14} \Rightarrow$ **C-property OK to double precision**.

Test for stationary 1D solutions

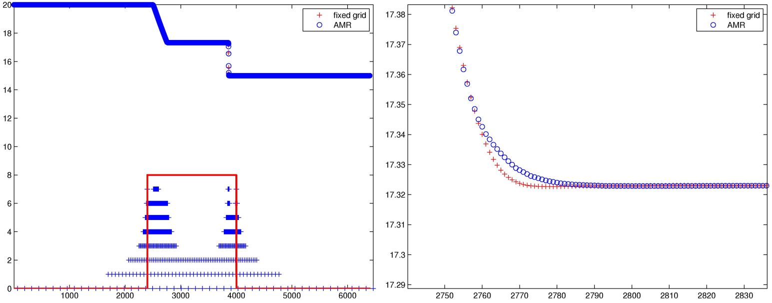
- Same setup, but without well balanced interpolation:



- Scheme gives approximated solution such that $||h+z-12||_\infty = 5.31 \cdot 10^{-2}$ and $||v||_\infty = 2.16 \cdot 10^{-14} \Rightarrow$ **loss of exact C-property.**

Test for non stationary 1D solutions

- Dam break problem with square bump bottom topography.
- Solution at $T = 15$. Have used rtol=10^{-3}, $N_0 = 50$, and eight levels ($L = 7$, $N_7 = 6400$) to obtain:

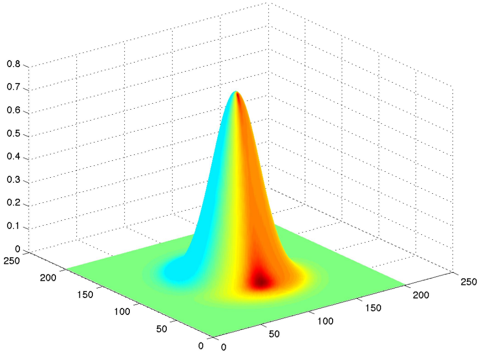


with CPU speedup ≈ 14.04.

- Scheme gives approximated solution such that $||h_{AMR} - h_{fixed}||_1 = 1.44 \cdot 10^{-4}$, $||v_{AMR} - v_{fixed}||_1 = 1.47 \cdot 10^{-4}$

Test for stationary 2D solutions

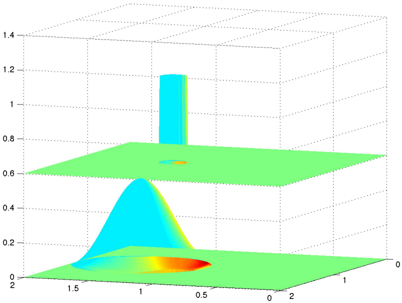
- ([LeVeque, 1998]) Water at rest, total height$= 1$ and bottom:



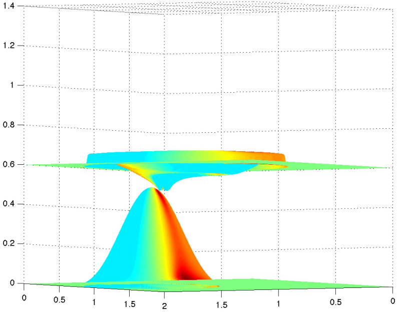
- Have used rtol=10^{-1}, $N_0 = 25$, and 4 levels ($L = 3$, $N_3 = 200$), $T = 0.1$ to obtain: $||h + z - 1||_\infty = 1.11 \cdot 10^{-15}$, $||v^x||_\infty = 3.52 \cdot 10^{-15}$, $||v^y||_\infty = 3.88 \cdot 10^{-15} \Rightarrow$ **C-property OK to double precision.**
- CPU speedup=3.96

Test for non stationary 2D solutions

- Circular dam break problem ([Castro-Fernández-Nieto-Ferreiro-García-Rodríguez-Parés, 2009]). Have used rtol=10^{-1}, $N_0 = 100$, and 5 levels ($L = 4$, $N_4 = 1600$), $T = 0.25$



$T = 0$



$T = 0.25$

- CPU speedup=5.22
- $\|h_{AMR} - h_{fixed}\|_1 = 8.33 \cdot 10^{-4}$, $\|v^x_{AMR} - v^x_{fixed}\|_1 = 1.5 \cdot 10^{-3}$, $\|v^y_{AMR} - v^y_{fixed}\|_1 = 1.4 \cdot 10^{-3}$, difference of mass$\approx 7 \cdot 10^{-4}$.

Test for non stationary 2D solutions

water and grids

In grid hierarchy, ligther color means finer resolution.

Conclusions and future research

Conclusions

- We have presented a technique for obtaining well-balanced point-value-based adaptive mesh refinement schemes for shallow water equations.
- We have seen some of the difficulties for getting well-balanced adaptive mesh refinement schemes for SWE based on cell-averages.
- We have tested the scheme with Donat&Martinez-Gavara homogenized SWE solver and we have obtained an adaptive scheme with the exact C-property.

Future research

- We are working on its parallelization and extension to deal with dry zones.
- Possibility of getting an adaptive scheme that preserves more stationary solutions if underlying scheme does so.
- Comparison of present code with AMR without well-balanced interpolation
- Comparison of present code with AMR with cell-average-based AMR.

Audusse-Bouchut-Bristeau-Klein-Perthame (2004).
A fast and stable well-balanced scheme with hydrostatic reconstruction for shallow water flows.
SIAM J. Sci. Comp, 25:2050–2065.

Baeza, A. and Mulet, P. (2006).
Adaptive mesh refinement techniques for high-order shock capturing schemes for multi-dimensional hydrodynamic simulations.
Internat. J. Numer. Methods Fluids, 52(4):455–471.

Berger, M. J. and Oliger, J. (1984).
Adaptive mesh refinement for hyperbolic partial differential equations.
J. Comput. Phys., 53(3):484–512.

Berger-Calhoun-Helzel-LeVeque (2009).
Logically rectangular finite volume methods with adaptive refinement on the sphere.
Phil. Trans. R. Soc. A, 367:4483–4496.

Bermudez, A. and Vazquez, M. E. (1994).
Upwind methods for hyperbolic conservation laws with source terms.
Comput. & Fluids, 23(8):1049–1071.

Bouchut, F. (2004).

Nonlinear stability of finite volume methods for hyperbolic conservation laws and well-balanced schemes for sources.

Frontiers in Mathematics. Birkhäuser Verlag, Basel.

Bouchut, F. and Morales de Luna, T. (2010).

A subsonic-well-balanced reconstruction scheme for shallow water flows.

SIAM J. Numer. Anal., 48(5):1733–1758.

Caselles-Donat-Haro (2009).

Flux-gradient and source-term balancing for certain high resolution shock-capturing schemes.

Comput. & Fluids, 38(1):16–36.

Castro-Fernández-Nieto-Ferreiro-García-Rodríguez-Parés (2009).

High order extensions of roe schemes for two-dimensional nonconservative hyperbolic systems.

J. Sci. Comput., 39:67–114.

Cohen, A., Kaber, S. M., Müller, S., and Postel, M. (2003).

Fully adaptive multiresolution finite volume schemes for conservation laws.

Math. Comp., 72(241):183–225 (electronic).

Donat, R. and Martínez-Gavara, A. (2011).

A hybrid second order scheme for shallow water flows.
to appear in APNUM.

Gascón, L. and Corberán, J. M. (2001).
Construction of second-order TVD schemes for nonhomogeneous hyperbolic conservation laws.
J. Comput. Phys., 172(1):261–297.

George, D. L. (2011).
Adaptive finite volume methods with well-balanced Riemann solvers for modeling floods in rugged terrain: Application to the Malpasset dam-break flood (France, 1959).
INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS, 66(8):1000–1018.

Greenberg, J. M. and Leroux, A. Y. (1996).
A well-balanced scheme for the numerical processing of source terms in hyperbolic equations.
SIAM J. Numer. Anal., 33(1):1–16.

LeVeque, R. J. (1998).
Balancing source terms and flux gradients in high-resolution godunov methods: the quasi-steady wave-propagation algorithm.

J. Comput. Phys., 146:346–365.

Müller, S. and Stiriba, Y. (2007).
Fully adaptive multiscale schemes for conservation laws employing locally varying time stepping.
J. Sci. Comput., 30(3):493–531.

Quirk, J. (1996).
A parallel adaptive grid algorithm for computational shock hydrodynamics.
APPLIED NUMERICAL MATHEMATICS, 20(4):427–453.