Adaptive mesh refinement method: numerical density of entropy production and automatic thresholding.

Ersoy, M.\textsuperscript{1a} Pons, K.\textsuperscript{1,2b}

\textsuperscript{1}EA 2134, IMATH, Université de Toulon
\textsuperscript{2}Principia S.A.S., 13705 La Ciotat cedex

2017, 17 February, LAMA – Chambéry

\textsuperscript{a} ersoy@univ-tln.fr
\textsuperscript{b} pons@univ-tln.fr
Motivations

- numerical simulations of real-life large scale fluid flows in dimension $d = 1, 2, 3$
  
(a) Monster waves (Nazare)  
(b) Wave-breaking (Nazare)

(c) Tsunami (Japan)  
(d) tsunami (Brazil)
Motivations

- numerical simulations of real-life large scale fluid flows in dimension $d = 1, 2, 3$
- Simulating fluids in large-scale $\rightarrow$ high memory and computer requirements.
  - how to compute fast and accurate: $\rightarrow$ meshing or moving the mesh only in "desired" regions with a suitable mesh refinement criterion
Motivations

- numerical simulations of real-life large scale fluid flows in dimension $d = 1, 2, 3$
- Simulating fluids in large-scale $\rightarrow$ high memory and computer requirements.
- Mathematical motivations: introducing new tool for numerical purpose!
Outline of the talk

1 Adaptive Mesh Refinement method
   - Generality
   - An example

2 Automatic mesh refinement threshold

3 Numerical simulations
1. **Adaptive Mesh Refinement method**
   - Generality
   - An example

2. **Automatic mesh refinement threshold**

3. **Numerical simulations**
1. **Adaptive Mesh Refinement method**
   - Generality
   - An example

2. **Automatic mesh refinement threshold**

3. **Numerical simulations**
We focus on general non linear hyperbolic conservation laws

\[
\begin{cases}
\frac{\partial w}{\partial t} + \frac{\partial f(w)}{\partial x} = 0, \ (t, x) \in \mathbb{R}^+ \times \mathbb{R} \\
w(0, x) = w_0(x), \ x \in \mathbb{R}
\end{cases}
\]

where \( w \in \mathbb{R}^d \) : vector state,
\( f \) : flux governing the physical description of the flow.
We focus on general non linear hyperbolic conservation laws

\[
\begin{align*}
\frac{\partial \mathbf{w}}{\partial t} + \frac{\partial f(\mathbf{w})}{\partial x} &= 0, \quad (t, x) \in \mathbb{R}^+ \times \mathbb{R} \\
\mathbf{w}(0, x) &= \mathbf{w}_0(x), \quad x \in \mathbb{R}
\end{align*}
\]

Weak solutions satisfy

\[
S = \frac{\partial s(\mathbf{w})}{\partial t} + \frac{\partial \psi(\mathbf{w})}{\partial x} \begin{cases} = 0 \quad \text{for smooth solution} \\ = 0 \quad \text{across rarefaction} \\ < 0 \quad \text{across shock} \end{cases}
\]

where \((s, \psi)\) stands for a convex entropy-entropy flux pair:

\[
(\nabla \psi(\mathbf{w}))^T = (\nabla s(\mathbf{w}))^T \quad D_w f(\mathbf{w})
\]
Hyperbolic equations and entropy inequality

We focus on general non linear hyperbolic conservation laws

\[
\begin{aligned}
\frac{\partial w}{\partial t} + \frac{\partial f(w)}{\partial x} &= 0, \quad (t, x) \in \mathbb{R}^+ \times \mathbb{R} \\
w(0, x) &= w_0(x), \quad x \in \mathbb{R}
\end{aligned}
\]

Weak solutions satisfy

\[
S = \frac{\partial s(w)}{\partial t} + \frac{\partial \psi(w)}{\partial x} \begin{cases} = 0 & \text{for smooth solution} \\ = 0 & \text{across rarefaction} \\ < 0 & \text{across shock}
\end{cases}
\]

where \((s, \psi)\) stands for a convex entropy-entropy flux pair:

\[
(\nabla \psi(w))^T = (\nabla s(w))^T D_w f(w)
\]

Entropy inequality \(\simeq\) “smoothness indicator”

We focus on general non linear hyperbolic conservation laws

\[
\begin{aligned}
\frac{\partial w}{\partial t} + \text{div}(f(w)) &= 0, \quad (t, x) \in \mathbb{R}^+ \times \mathbb{R}^d \\
w(0, x) &= w_0(x), \quad x \in \mathbb{R}^d
\end{aligned}
\]

Weak solutions satisfy

\[
S = \frac{\partial s(w)}{\partial t} + \text{div}(\psi(w)) \begin{cases} = 0 & \text{for smooth solution} \\
= 0 & \text{across rarefaction} \\
< 0 & \text{across shock}
\end{cases}
\]

where \((s, \psi)\) stands for a convex entropy-entropy flux pair:

\[
(\nabla \psi_i(w))^T = (\nabla s(w))^T \quad D_w f_i(w), \quad i = 1, \ldots, d
\]

Entropy inequality \(\simeq\) “smoothness indicator”

Finite volume approximation

\[ w_{k}^{n+1} = w_{k}^{n} - \frac{\delta t_{n}}{h_{k}} \left( F_{k+1/2}^{n} - F_{k-1/2}^{n} \right) \]

with

\[ w_{k}^{n} \approx \frac{1}{h_{k}} \int_{C_{k}} w(t_{n}, x) \, dx \quad \text{and} \quad F_{k+1/2}^{n} \approx \frac{1}{\delta t} \int_{C_{k}} f(t, w(t, x_{k+1/2})) \, dx \]
Finite volume approximation:

\[ w^{n+1}_k = w^n_k - \frac{\delta t_n}{h_k} \left( F^n_{k+1/2} - F^n_{k-1/2} \right) \]

with

\[ w^n_k \approx \frac{1}{h_k} \int_{C_k} w(t_n, x) \, dx \quad \text{and} \quad F^n_{k+1/2} \approx \frac{1}{\delta t} \int_{C_k} f(t, w(t, x_{k+1/2})) \, dx \]

The numerical density of entropy production:

\[ S^n_k = \frac{s^{n+1}_k - s^n_k}{\delta t_n} + \frac{\psi^n_{k+1/2} - \psi^n_{k-1/2}}{h_k} \approx 0 \]
Finite volume approximation:

\[ w_{k}^{n+1} = w_{k}^{n} - \delta t_{n} \left( \frac{1}{h_{k}} \sum_{a} F(w_{k}^{n}, w_{a}^{n}; n_{k}/a) \right), \quad h_{k} = \frac{|C_{k}|}{\sum_{a} |\partial C_{k}/a|} \]

with

\[ w_{k}^{n} \approx \frac{1}{h_{k}} \int_{C_{k}} w(t_{n}, x) \, dx, \quad \text{and} \quad F(w_{k}^{n}, w_{a}^{n}; n_{k}/a) \approx \frac{1}{\delta t} \int_{\partial C_{k}} f(t, w) \cdot n_{k}/a \, ds \]
Finite volume approximation:

\[ w_{k}^{n+1} = w_{k}^{n} - \frac{\delta t_n}{h_k} \left( \sum_a F(w_{k}^{n}, w_{a}^{n}; n_{k/a}) \right), \quad h_k = \frac{|C_k|}{\sum_a |\partial C_{k/a}|} \]

with

\[ w_{k}^{n} \approx \frac{1}{h_k} \int_{C_k} w(t_n, x) \, dx, \quad \text{and} \quad F(w_{k}^{n}, w_{a}^{n}; n_{k/a}) \approx \frac{1}{\delta t} \int_{\partial C_k} f(t, w) \cdot n_{k/a} \, ds \]

The numerical density of entropy production:

\[ S_{k}^{n} = \frac{s_{k}^{n+1} - s_{k}^{n}}{\delta t_n} + \sum_a \psi(w_{k}^{n}, w_{a}^{n}; n_{k/a}) \frac{1}{h_k} \approx 0 \]
Mesh refinement indicator: principle & illustration

- Compute $w_k^n$
Mesh refinement indicator: principle & illustration

- Compute $w^n_k$
- Compute $S^n_k : S^n_k \neq 0 \implies$ the cell is refined or coarsened
Mesh refinement indicator : principle & illustration

- Compute $w_k^n$
- Compute $S_k^n$: $S_k^n \neq 0 \implies$ the cell is refined or coarsened
- More precisely: for a given mesh refinement threshold $\alpha$
  - $S_k^n \geq \alpha \overline{S} \implies$ the cell is refined with, for instance, $\overline{S} = \frac{1}{|\Omega|} \int_\Omega S_k^n$

\[ w(x) \]
\[ S(x) = |w'(x)| \]

\[ x \]

\[ \alpha \overline{S} \text{ with } \alpha = 2 \]
Mesh refinement indicator: principle & illustration

- Compute $w^n_k$
- Compute $S^n_k : S^n_k \neq 0 \implies$ the cell is refined or coarsened
- More precisely: for a given mesh refinement threshold $\alpha$
  
  $\triangleright$ $S^n_k \geq \alpha \overline{S} \implies$ the cell is refined with, for instance, $\overline{S} = \frac{1}{|\Omega|} \int_{\Omega} S^n_k$
  
  $\triangleright$ $S^n_k < \alpha \overline{S} \implies$ the cell is coarsened

\[ S(x) = |w'(x)|, \quad \alpha \overline{S} with \alpha = 2 \]
Mesh refinement indicator: principle & illustration

- Compute $w_n^k$
- Compute $S_n^k : S_n^k \neq 0 \implies$ the cell is refined or coarsened
- More precisely: for a given mesh refinement threshold $\alpha$
  - $S_n^k \geq \alpha S \implies$ the cell is refined with, for instance, $S = \frac{1}{|\Omega|} \int_{\Omega} S_n^k$
  - $S_n^k < \alpha S \implies$ the cell is coarsened
- Dynamic mesh refinement:
  - Dyadic tree (1D)
  - Hierarchical numbering: basis 2
Mesh refinement indicator: principle & illustration

- Compute $w^n_k$
- Compute $S^n_k: S^n_k \neq 0 \implies$ the cell is refined or coarsened
- More precisely: for a given mesh refinement threshold $\alpha$
  - $S^n_k \geq \alpha \overline{S} \implies$ the cell is refined with, for instance, $\overline{S} = \frac{1}{|\Omega|} \int_{\Omega} S^n_k$
  - $S^n_k < \alpha \overline{S} \implies$ the cell is coarsened
- Dynamic mesh refinement:
  - Non-structured grid: macro-cell
  - Dyadic tree (1D), Quadtree (2D)
  - Hierarchical numbering: basis 2,4

\[
\begin{array}{c|cc}
0 & 10 & 11 \\
120 & 121 & 13 \\
122 & 123 & 13 \\
\hline
2 & 3 & \\
\end{array}
\]
Mesh refinement indicator: principle & illustration

- Compute $w^n_k$
- Compute $S^n_k: S^n_k \neq 0 \implies$ the cell is refined or coarsened
- More precisely: for a given mesh refinement threshold $\alpha$
  - $S^n_k \geq \alpha \bar{S} \implies$ the cell is refined with, for instance, $\bar{S} = \frac{1}{|\Omega|} \int_{\Omega} S^n_k$
  - $S^n_k < \alpha \bar{S} \implies$ the cell is coarsened
- Dynamic mesh refinement:
  - Non-structured grid: macro-cell
  - Dyadic tree (1D), Quadtree (2D), Octree (3D)
  - Hierarchical numbering: basis 2, 4, 8
Mesh refinement indicator: principle & illustration

- Compute $w_k^n$
- Compute $S_k^n : S_k^n \neq 0 \implies$ the cell is refined or coarsened
- More precisely: for a given mesh refinement threshold $\alpha$
  - $S_k^n \geq \alpha \overline{S} \implies$ the cell is refined with, for instance, $\overline{S} = \frac{1}{|\Omega|} \int_{\Omega} S_k^n$
  - $S_k^n < \alpha \overline{S} \implies$ the cell is coarsened

\[
\begin{align*}
F_{k_0-1/2} &= F_{k_0+1/2} = F_{k_0+1/2} \\
F_{k_0+1/2} &= F_{k_0-1/2} = f(w_k^n)
\end{align*}
\]
Mesh refinement indicator: principle & illustration

- Compute $w^n_k$
- Compute $S^n_k : S^n_k \neq 0 \implies$ the cell is refined or coarsened
- More precisely: for a given mesh refinement threshold $\alpha$
  - $S^n_k \geq \alpha \bar{S} \implies$ the cell is refined with, for instance, $\bar{S} = \frac{1}{|\Omega|} \int_{\Omega} S^n_k$
  - $S^n_k < \alpha \bar{S} \implies$ the cell is coarsened
Mesh refinement indicator: principle & illustration

- Compute $w^n_k$
- Compute $S^n_k : S^n_k \neq 0 \implies$ the cell is refined or coarsened
- More precisely: for a given mesh refinement threshold $\alpha$
  - $S^n_k \geq \alpha S \implies$ the cell is refined with, for instance, $S = \frac{1}{|\Omega|} \int_{\Omega} S^n_k$
  - $S^n_k < \alpha S \implies$ the cell is coarsened
  - A simple projection method but the scheme is locally non consistent [S088, TW05]


**Mesh refinement indicator: principle & illustration**

- Compute $w_k^n$
- Compute $S_k^n: S_k^n \neq 0 \implies$ the cell is refined or coarsened
- More precisely: for a given mesh refinement threshold $\alpha$
  - $S_k^n \geq \alpha \bar{S} \implies$ the cell is refined with, for instance, $\bar{S} = \frac{1}{|\Omega|} \int_{\Omega} S_k^n$
  - $S_k^n < \alpha \bar{S} \implies$ the cell is coarsened
  - A simple projection method **but** the scheme is locally non consistent [S088, TW05]
    - less time consuming than other methods

---


Mesh refinement indicator: principle & illustration

- Compute $w_k^n$
- Compute $S_k^n : S_k^n \neq 0 \implies$ the cell is refined or coarsened
- More precisely: for a given mesh refinement threshold $\alpha$
  - $S_k^n \geq \alpha \overline{S} \implies$ the cell is refined with, for instance, $\overline{S} = \frac{1}{|\Omega|} \int_\Omega S_k^n$
  - $S_k^n < \alpha \overline{S} \implies$ the cell is coarsened
  - A simple projection method but the scheme is locally non consistent [S088, TW05]
    - less time consuming than other methods
    - non consistency is almost negligible if the level of adjacent cells is limited by 2


**Mesh refinement indicator: principle & illustration**

- Compute $w_k^n$
- Compute $S_k^n : S_k^n \neq 0 \implies$ the cell is refined or coarsened
- More precisely: for a given mesh refinement threshold $\alpha$
  - $S_k^n \geq \alpha \bar{S} \implies$ the cell is refined with, for instance, $\bar{S} = \frac{1}{|\Omega|} \int_{\Omega} S_k^n$
  - $S_k^n < \alpha \bar{S} \implies$ the cell is coarsened
  - A simple projection method but the scheme is locally non consistent [S088, TW05]
    - less time consuming than other methods
    - non consistency is almost negligible if the level of adjacent cells is limited by 2
  - Improvement (cpu-time): local time stepping [EGY13]  
    ▶ See more details

---


Mesh refinement indicator: principle & illustration

- Compute $w^n_k$
- Compute $S^n_k : S^n_k \neq 0 \implies$ the cell is refined or coarsened
- More precisely: for a given mesh refinement threshold $\alpha$
  - $S^n_k \geq \alpha \bar{S} \implies$ the cell is refined with, for instance, $\bar{S} = \frac{1}{|\Omega|} \int_\Omega S^n_k$
  - $S^n_k < \alpha \bar{S} \implies$ the cell is coarsened

- A simple projection method but the scheme is locally non-consistent [S088, TW05]
  - less time consuming than other methods
  - non-consistency is almost negligible if the level of adjacent cells is limited by 2

- Improvement (cpu-time): local time stepping [EGY13] → See more details
- 2D-3D computer management: BB-AMR [GEYS] → See more details

---


1 Adaptive Mesh Refinement method
   • Generality
   • An example

2 Automatic mesh refinement threshold

3 Numerical simulations
An example: the one-dimensional gas dynamics equations for ideal gas

\[ \frac{\partial \rho}{\partial t} + \frac{\partial (\rho u)}{\partial x} = 0 \]
\[ \frac{\partial \rho u}{\partial t} + \frac{\partial (\rho u^2 + p)}{\partial x} = 0 \]
\[ \frac{\partial \rho E}{\partial t} + \frac{\partial (\rho E + p) u}{\partial x} = 0 \]

\[ p = (\gamma - 1) \rho \varepsilon \]

where

\[ \rho(t, x) \quad : \quad \text{density} \]
\[ u(t, x) \quad : \quad \text{velocity} \]
\[ p(t, x) \quad : \quad \text{pressure} \]
\[ \gamma := 1.4 \quad : \quad \text{ratio of the specific heats} \]
\[ E(\varepsilon, u) \quad : \quad \text{total energy} \]
\[ \varepsilon \quad : \quad \text{internal specific energy} \]
\[ E = \varepsilon + \frac{u^2}{2} \]
An example: the one-dimensional gas dynamics equations for ideal gas

\[
\begin{align*}
    \frac{\partial \rho}{\partial t} + \frac{\partial \rho u}{\partial x} &= 0 \\
    \frac{\partial \rho u}{\partial t} + \frac{\partial (\rho u^2 + p)}{\partial x} &= 0 \\
    \frac{\partial \rho E}{\partial t} + \frac{\partial (\rho E + p) u}{\partial x} &= 0
\end{align*}
\]

where

\[
p = (\gamma - 1) \rho \varepsilon
\]

\[
\begin{align*}
\rho(t, x) & : \text{density} \\
u(t, x) & : \text{velocity} \\
p(t, x) & : \text{pressure} \\
\gamma & : \text{ratio of the specific heats} \\
E(\varepsilon, u) & : \text{total energy} \\
\varepsilon & : \text{internal specific energy} \\
E & = \varepsilon + \frac{u^2}{2}
\end{align*}
\]

- Conservative variables

\[
w = (\rho, \rho u, \rho E)^t
\]

- Convex continuous entropy

\[
s(w) = -\rho \ln \left( \frac{p}{\rho \gamma} \right) \text{ of flux } \psi(w) = u s(w).
\]
Sod’s shock tube problem

Mesh coarsening parameter $\alpha$ : 0.001,

Mesh refinement parameter $\bar{S}$ : \( \frac{1}{|\Omega|} \sum_{k_b} S_{kn} \)

CFL : 0.25,

Simulation time (s) : 0.4,

Initial number of cells : 200,

Maximum level of mesh refinement : $L_{\text{max}}$. 

\[ \text{Density} \quad \text{Entropy production} \quad \text{Refinement level & Number of cells} \quad 500 \]
**Accuracy**

(a) Density and numerical density of entropy production.

(b) Mesh refinement level, numerical density of entropy production and local error.

**Figure:** Sod’s shock tube problem: solution at time $t = 0.4$ s using the AB1M scheme on a dynamic grid with $L_{\text{max}} = 5$ and the AB1 scheme on a uniform fixed grid of 681 cells.
Properties of $S$: shock criterion type

**Theorem ([P04])**

Consider a $p^{\text{th}}$ convergent scheme. Let $S^n_k$ be the corresponding numerical density of entropy production and $\Delta t = \lambda h$ be a fixed time step where $h$ stands for the meshsize.

Then

$$\lim_{n \to \infty} S^n_k = \begin{cases} O(\Delta t^p) & \text{if the solution is smooth,} \\ O\left(\frac{1}{\Delta t}\right) & \text{if the solution is discontinuous.} \end{cases}$$

Properties of $S$ : shock criterion type

**Theorem ([P04])**

Consider a $p^{th}$ convergent scheme. Let $S^m_k$ be the corresponding numerical density of entropy production and $\Delta t = \lambda h$ be a fixed time step where $h$ stands for the meshsize.

Then

$$\lim_{n \to \infty} S^m_k = \begin{cases} 
O(\Delta t^p) & \text{if the solution is smooth}, \\
O\left(\frac{1}{\Delta t}\right) & \text{if the solution is discontinuous}.
\end{cases}$$

**Properties**

Consider a monotone scheme. Then, for almost every $k$, every $n$, $S^m_k \leq 0$. 
Properties of $S$ : shock criterion type

**Theorem ([P04])**

Consider a $p^{th}$ convergent scheme. Let $S^n_k$ be the corresponding numerical density of entropy production and $\Delta t = \lambda h$ be a fixed time step where $h$ stands for the meshsize. Then

$$\lim_{n \to \infty} S^n_k = \begin{cases} O(\Delta t^p) & \text{if the solution is smooth}, \\ O\left(\frac{1}{\Delta t}\right) & \text{if the solution is discontinuous}. \end{cases}$$

**Properties**

Consider a monotone scheme. Then, for almost every $k$, every $n$, $S^n_k \leq 0$.

Thus, even if locally $S^n_k$ can take positive value, one has $S^n_k \leq C \Delta t^q$, $q \geq p$.
1 **Adaptive Mesh Refinement method**
   - Generality
   - An example

2 **Automatic mesh refinement threshold**

3 **Numerical simulations**
Mesh refinement threshold $S^m_{k}$ and the mesh refinement threshold $\alpha$

- $S^m_{k} \geq \alpha \overline{S} \implies$ the cell is refined with $\overline{S} = \frac{1}{|\Omega|} \int_{\Omega} S^m_{k}$
- $S^m_{k} < \alpha \overline{S} \implies$ the cell is coarsened
Mesh refinement threshold

Mesh refinement criterion $S^m_k$ and the mesh refinement threshold $\alpha$

- $S^n_k \geq \alpha \bar{S} \implies$ the cell is refined with $\bar{S} = \frac{1}{|\Omega|} \int_{\Omega} S^n_k$
- $S^n_k < \alpha \bar{S} \implies$ the cell is coarsened

how to set $\alpha$ suitably?
Mesh refinement threshold

Mesh refinement criterion $S^n_k$ and the mesh refinement threshold $\alpha$

- $S^n_k \geq \alpha \overline{S} \implies$ the cell is refined with $\overline{S} = \frac{1}{|\Omega|} \int_{\Omega} S^n_k$

- $S^n_k < \alpha \overline{S} \implies$ the cell is coarsened

- How to set $\alpha$ suitably? **Ideally**
  - automatically
  - accuracy and computational time are “well-balanced”
    - catch the smallest maxima but not the smallest
Mesh refinement threshold

Mesh refinement criterion $S^n_k$ and the mesh refinement threshold $\alpha$

$S^n_k \geq \alpha S \implies$ the cell is refined with $S = \frac{1}{|\Omega|} \int_{\Omega} S^n_k$

$S^n_k < \alpha S \implies$ the cell is coarsened

- how to set $\alpha$ suitably? ideally
  - automatically
  - accuracy and computational time are “well-balanced”
    - catch the smallest maxima but not the smallest
    - simple method (“low-cost”)
Review of classical methods

- Mean method or deviation

\[
S(x, t) > \alpha \frac{1}{|\Omega|} \int_{\Omega} S(x, t) \, dx
\]

where \( \alpha \) is a tunable dimensionless threshold parameter or deviation method

\[
S(x, t) > \alpha \frac{1}{|\Omega|} \int_{\Omega} S(x, t) \, dx + \beta \sigma(x, t)
\]

where \( \sigma \) is the standard deviation with \( \beta \) is a tunable dimensionless threshold parameter


Review of classical methods

- Mean method or deviation → pb: set \( \alpha \) or \( (\alpha, \beta) \)
- Filtering: two-steps

(a) Filtering (mean method): first step

(b) Filtering (mean method): second step


**Review of classical methods**

- Mean method or deviation → pb : set $\alpha$ or $(\alpha, \beta)$
- Filtering : two-steps → pb : two-step ? Is it enough ? still to set $\alpha$
- Filtering : wavelets

![Graphs](image)

**Figure:** Illustration of the wavelet transformation for a given mesh refinement criterion computed with the Daubechies wavelet with four vanishing moments (warm colors correspond to large coefficients and cold colors to small coefficients)

---

Review of classical methods

- Mean method or deviation → pb : set $\alpha$ or $(\alpha, \beta)$
- Filtering : two-steps → pb : two-step? Is it enough? still to set $\alpha$
- Filtering : wavelets→ pb : efficient but the cost! still to set $\alpha$
- Local maxima→ pb : efficient but the cost! still to set $\alpha$

![Graph](image)

Figure: Illustration of the local maxima method for a mesh refinement criterion involving multiple scales
Distribution function ($L$-meas $\{S(x) > \alpha\}$)

- **Assumptions and notations**
  - $S$ is smooth and has $p$ local maxima.
  - $S(0) = S(L) = S'(0) = S'(L) = 0$
  - $0 < S_\infty = \max_{x \in (0,L)} S(x) < \infty$
Distribution function ($L$-meas $\{S(x) > \alpha\}$)

- Assumptions and notations
  - $S$ is smooth and has $p$ local maxima.
  - $S(0) = S(L) = S'(0) = S'(L) = 0$
  - $0 < S_\infty = \max_{x \in (0, L)} S(x) < \infty$

- Then
  - $Z_\alpha = \{x \in (0, L); \varphi_\alpha(x) = S(x) - \alpha = 0 \text{ and } S'(x) \neq 0\} \neq \emptyset$
  - $= \{x_0(\alpha) < x_1(\alpha) < \cdots < x_{2p_\alpha - 2}(\alpha) < x_{2p_\alpha - 1}(\alpha)\}$
  - $\#Z_\alpha = 2p_\alpha$
Distribution function (L-meas \( \{ S(x) > \alpha \} \))

- **Assumptions and notations**
  - \( S \) is smooth and has \( p \) local maxima.
  - \( S(0) = S(L) = S'(0) = S'(L) = 0 \)
  - \( 0 < S_\infty = \max_{x \in (0,L)} S(x) < \infty \)

- **Then**
  - \( Z_\alpha = \{ x \in (0, L); \phi_\alpha(x) = S(x) - \alpha = 0 \text{ and } S'(x) \neq 0 \} \neq \emptyset \)
  - \( = \{ x_0(\alpha) < x_1(\alpha) < \cdots < x_{2p_\alpha-2}(\alpha) < x_{2p_\alpha-1}(\alpha) \} \)
  - \( \# Z_\alpha = 2p_\alpha \)
  - there exist sequence \( (x^*_k)_{1 \leq k \leq p} \)
Distribution function ($L$-meas $\{S(x) > \alpha\}$)

- **Assumptions and notations**
  - $S$ is smooth and has $p$ local maxima.
  - $S(0) = S(L) = S'(0) = S'(L) = 0$
  - $0 < S_\infty = \max_{x \in (0, L)} S(x) < \infty$

- **Then**
  - $Z_\alpha = \{x \in (0, L); \ \varphi_\alpha(x) = S(x) - \alpha = 0 \text{ and } S''(x) \neq 0\} \neq \emptyset$
  - $\#Z_\alpha = 2p_\alpha$
  - There exist sequences $(x_k^*)_{1 \leq k \leq p}$ and $(\alpha_k^*)_{1 \leq k \leq p}$ such that
    - $\forall k = 1, \ldots, p$
    - $S(x_k^*) = \alpha_k^*$ and $S'(x_k^*) = 0$, $S''(x_k^*) < 0$
    - $x_k^* \notin Z_\alpha$ and $\alpha_p^* = S_\infty$
Distribution function (L-meas \( \{S(x) > \alpha\} \))

- **Assumptions and notations**
  - \( S \) is smooth and has \( p \) local maxima.
  - \( S(0) = S(L) = S'(0) = S'(L) = 0 \)
  - \( 0 < S_\infty = \max_{x \in (0,L)} S(x) < \infty \)

- With these settings, we define the distribution \( d \)

\[
\alpha \in [0, S_\infty] \mapsto d(\alpha) := \begin{cases} 
1 & \text{if } \alpha = 0, \\
\frac{1}{L} \sum_{k=1}^{p_\alpha} x_{2k+1}(\alpha) - x_{2k}(\alpha) & \text{if } 0 < \alpha < S_\infty, \\
0 & \text{if } \alpha = S_\infty.
\end{cases}
\]
Properties of $d$ and $\alpha_{PE}$ threshold

**Properties**

1. $\int_0^{S_\infty} d(\alpha) \, d\alpha = \int_0^L S(x) \, dx$.

2. $d \in C^0([0, S_\infty], \mathbb{R}^+) \text{ and } d' \text{ satisfies}$
   - $\forall \alpha \in [0, S_\infty], \ d'(\alpha) < 0$
   - $\forall k \in [0, p], \ \lim_{\alpha \to \alpha_k^*} d'(\alpha) = -\infty$ with the convention $\alpha_0^* := 0$

3. $d \in C^l \left( D_*, \mathbb{R}^+ \right)$ on the set $D^* := \bigcup_{k=0}^{p-1} (\alpha_k^*, \alpha_{k+1}^*)$. 

![Graphs showing properties of d and alpha_P(E) threshold](image-url)
Automatic thresholding: distribution function ($L$-meas $\{S(x) > \alpha\}$)

- Hard to define $\alpha$ using only $d$
Automatic thresholding: distribution function \( (L\text{-meas } \{S(x) > \alpha\}) \)

- Hard to define \( \alpha \) using only \( d \)
- However, one can set \( \alpha = \alpha_D \) or
  - either \( \alpha = \alpha_D = \min\{\alpha \mid d'(\alpha) = -1\} \)

\[ S(x) \]
\[ d(\alpha) \]
\[ d'(\alpha) \]

**Automatic thresholding : distribution function (L-meas \{S(x) > \alpha\})**

- Hard to define \(\alpha\) using only \(d\)
- However, one can set \(\alpha = \alpha_D\) or \(\alpha = \alpha_P\)
  - either \(\alpha = \alpha_P = \min\{\alpha \mid d''(\alpha) = 0\}\)

---

Automatic thresholding: distribution function ($L$-meas $\{S(x) > \alpha\}$)

- Hard to define $\alpha$ using only $d$
- However, one can set $\alpha = \alpha_D$ or $\alpha = \alpha_P$
- In practice, $d$ is defined by $d(\alpha) = \sum_{j=0}^{M-1} d_j \mathbb{1}_{(\alpha_j,\alpha_{j+1})}(\alpha)$ with

$$d_j = \# \{ k ; S_k^m > \alpha_j \} \text{ and } (\alpha_j)_{0 \leq j \leq M} = \left( S_m \left( \frac{j}{M} \right)^\beta \right)_{0 \leq j \leq M} \quad \beta \geq 1$$

Automatic thresholding: distribution function ($L$-meas \{ $S(x) > \alpha$ \})

- Hard to define $\alpha$ using only $d$
- However, one can set $\alpha = \alpha_D$ or $\alpha = \alpha_P$
- $\alpha$ is usually too small (too sensitive to $\beta$)!!! → detects small fluctuations
Automatic thresholding: distribution function ($L$-meas $\{ S(x) > \alpha \}$)

- Hard to define $\alpha$ using only $d$
- However, one can set $\alpha = \alpha_D$ or $\alpha = \alpha_P$
- $\alpha$ is usually too small (too sensitive to $\beta$) → detects small fluctuations
- We propose to set

$$\alpha = \alpha_{PE} = \min\{ \alpha \in [0, \bar{S}] \mid \max\{ \alpha d(\alpha) \} \}$$

![Graphs](image)

Pons, K., Ersoy, M. Adaptive mesh refinement method. Part 1: Automatic thresholding based on a distribution function

Pons, K., Ersoy, M. Adaptive mesh refinement method. Part 2: Application to tsunamis propagation
Why $\alpha d(\alpha)$?

**Properties**

Assume that $S$ is twice differentiable and has $p$ local maxima. Then, there exists

$$\forall k = 0 \ldots p - 1, \ \alpha_{k+1}^{**} \in (\alpha_k^*, \alpha_{k+1}^*) \text{ such that } d''(\alpha_{k+1}^{**}) = 0$$

and the function $f$ has $p$ local maxima $\overline{\alpha}_1, \ldots, \overline{\alpha}_p$ such that

$$\forall k = 1 \ldots p, \ \overline{\alpha}_k \in (\alpha_k^{**}, \alpha_k^*).$$

As a consequence $\alpha_{PE} > \alpha_P > \alpha_D$. 
**Illustration:** $\alpha_{PE}$ threshold for “discontinuous” flows

**Figure:** The function $f$ for the mesh refinement criterion

$$S(x) = 200 \exp(-1000(x - 1.25)^2) + 1.25 \exp(-5(x - 3.75)^2)$$ representing a shock type solution

$$\alpha_{PE} = 0.83974 \quad \text{and} \quad S_m = 2.4353$$
Illustration: $\alpha_{PE}$ threshold for “smooth” flows

Figure: The function $f$ for the mesh refinement criterion $S(x) = 2\exp(-10(x - 1.25)^2) + 1.25\exp(-5(x - 3.75)^2)$ representing a smooth flow.
1 **Adaptive Mesh Refinement method**
   - Generality
   - An example

2 **Automatic mesh refinement threshold**

3 **Numerical simulations**
Numerical validation

A Dam-break problem for the Saint-Venant equations:

\[
\begin{cases}
    \partial_t h + \partial_x (hu) = 0, \\
    \partial_t (hu) + \partial_x (hu^2 + gh^2/2) = -ghZ'(x)
\end{cases}
\]

where

\( h(t, x) \): density
\( u(t, x) \): velocity of the water column
\( g \): gravity strength
\( Z \): topography

Numerical validation

(a) A priori error

(b) Numerical density of entropy production

(c) A posteriori error

(d) Gradient of $h$

Figure: Numerical results for the water height at time $t = 2$ s
**Numerical validation**

(a) A priori error  
(b) Numerical density of entropy production  
(c) A posteriori error  
(d) Gradient of $h$

**Figure:** Time evolution of the threshold parameter and the mean value $S_m$
Test case I: Solitary wave propagation over a two dimensional reef

Figure: Experimental settings and wave gauges locations


**Test case I: Solitary wave propagation over a two dimensional reef**

<table>
<thead>
<tr>
<th></th>
<th>Adaptive mesh simulation</th>
<th>Uniform mesh simulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulation time</td>
<td>240</td>
<td>240</td>
</tr>
<tr>
<td>Number of cells</td>
<td>200-560</td>
<td>1000</td>
</tr>
<tr>
<td>Re-meshing time step</td>
<td>0.05 s</td>
<td>not applicable</td>
</tr>
<tr>
<td>Time order integration</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Space order integration</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>CFL</td>
<td>0.99</td>
<td>0.99</td>
</tr>
</tbody>
</table>

**Table:** Numerical parameters
**Test case I: Solitary wave propagation over a two dimensional reef**

![Graph](image)

(a) $\tilde{t} = 55.03$

**Figure:** Surface profiles of solitary wave propagation over an exposed reef crest. Confrontation of experimental data (blue circles) to numerical data computed on a uniform grid (solid green line) and on an adaptive grid (solid red lines). The solid cyan line represents the mesh level and the black one the bathymetry.
**Test case I: Solitary wave propagation over a two dimensional reef**

**Figure:** Surface profiles of solitary wave propagation over an exposed reef crest. Confrontation of experimental data (blue circles) to numerical data computed on a uniform grid (solid green line) and on an adaptive grid (solid red lines). The solid cyan line represents the mesh level and the black one the bathymetry.
**Test case I: Solitary wave propagation over a two dimensional reef**

![Graph](image)

**(a) \( \tilde{t} = 69.13 \)**

**Figure:** Surface profiles of solitary wave propagation over an exposed reef crest. Confrontation of experimental data (blue circles) to numerical data computed on a uniform grid (solid green line) and on an adaptive grid (solid red lines). The solid cyan line represents the mesh level and the black one the bathymetry.
**Test case I: Solitary wave propagation over a two dimensional reef**

![Graph](image.png)

(a) $\tilde{t} = 70.68$

**Figure:** Surface profiles of solitary wave propagation over an exposed reef crest. Confrontation of experimental data (blue circles) to numerical data computed on a uniform grid (solid green line) and on an adaptive grid (solid red lines). The solid cyan line represents the mesh level and the black one the bathymetry.
Test case I: Solitary wave propagation over a two dimensional reef

Figure: Surface profiles of solitary wave propagation over an exposed reef crest. Confrontation of experimental data (blue circles) to numerical data computed on a uniform grid (solid green line) and on an adaptive grid (solid red lines). The solid cyan line represents the mesh level and the black one the bathymetry.
**Test case I: Solitary wave propagation over a two dimensional reef**

![Graph](image)

(a) $\tilde{t} = 125.03$

**Figure:** Surface profiles of solitary wave propagation over an exposed reef crest. Confrontation of experimental data (blue circles) to numerical data computed on a uniform grid (solid green line) and on an adaptive grid (solid red lines). The solid cyan line represents the mesh level and the black one the bathymetry.
Test case I: Solitary wave propagation over a two dimensional reef

Figure: Surface profiles of solitary wave propagation in time at wave gauges 1 to 6.

- (a) $x = 17.6\,\text{m}$
- (b) $x = 50.4\,\text{m}$
- (c) $x = 58.1\,\text{m}$
- (d) $x = 65.2\,\text{m}$
**Test case I: Solitary wave propagation over a two dimensional reef**

![Graphs showing threshold and number of cells over time](image)

**Figure:** Time evolution of the mesh refinement threshold and the number of cells: 95 s against 210 s
**Test case II:** Solitary wave propagation over an irregular 3-d shallow shelf

**Figure:** Experimental settings


Test case II: Solitary wave propagation over an irregular 3-d shallow shelf

<table>
<thead>
<tr>
<th></th>
<th>Adaptive mesh simulation</th>
<th>Uniform mesh simulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulation time</td>
<td>30 s</td>
<td>30 s</td>
</tr>
<tr>
<td>Number of blocks</td>
<td>128</td>
<td>128</td>
</tr>
<tr>
<td>Number of cells</td>
<td>7,500-25,000</td>
<td>33,000</td>
</tr>
<tr>
<td>Re-meshing time step</td>
<td>0.25 s</td>
<td>not applicable</td>
</tr>
<tr>
<td>Time order integration</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Space order integration</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>CFL</td>
<td>0.5</td>
<td>0.5</td>
</tr>
</tbody>
</table>

**Table:** Numerical parameters
Test case II: Solitary wave propagation over an irregular 3-d shallow shelf

Figure: Numerical water height (coloration is issue from the kinetic energy)
Test case II: Solitary wave propagation over an irregular 3-d shallow shelf

(a) $t=2.5\text{s}$

Figure: Numerical water height (coloration is issue from the kinetic energy)
**Test case II: Solitary wave propagation over an irregular 3-d shallow shelf**

**Figure:** Numerical water height (coloration is issue from the kinetic energy)

(a) $t=5.75\text{s}$
Test case II: Solitary wave propagation over an irregular 3-d shallow shelf

(a) $t=23.75s$

Figure: Numerical water height (coloration is issued from the kinetic energy)
Test case II: Solitary wave propagation over an irregular 3-d shallow shelf

(a) WG2 results

(b) WG4 results

(c) WG6 results

(d) WG7 results
**Test case II: Solitary wave propagation over an irregular 3-d shallow shelf**

![Graph](image)

**Figure:** Time evolution of the mesh refinement threshold and the number of cells: speed up the computation by 2.5 time
Test case III: Tsunami runup onto a complex three dimensional Monai-Walley beach

(a) Top view

(b) Side view

Figure: Settings

Kévin Pons, Mehmet Ersoy, Frédéric Golay, Richard Marcer. Adaptive mesh refinement method. Part 2: Application to tsunamis propagation
# Test case III: Tsunami runup onto a complex three dimensional Monai-Walley beach

<table>
<thead>
<tr>
<th></th>
<th>Adaptive mesh simulation</th>
<th>Uniform mesh simulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulation time</td>
<td>30 s</td>
<td>30 s</td>
</tr>
<tr>
<td>Number of blocks</td>
<td>240</td>
<td>240</td>
</tr>
<tr>
<td>Number of cells</td>
<td>8,000–40,000</td>
<td>62,000</td>
</tr>
<tr>
<td>Re-meshing time step</td>
<td>0.25 s</td>
<td>not applicable</td>
</tr>
<tr>
<td>Time order integration</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>Space order integration</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>CFL</td>
<td>0.5</td>
<td>0.5</td>
</tr>
</tbody>
</table>

**Table:** Numerical parameters
Test case III: Tsunami runup onto a complex three dimensional Monai-Walley beach

Figure: Numerical water height (coloration is issue from the kinetic energy)
Test case III: Tsunami runup onto a complex three dimensional Monai-Walley beach

Figure: Numerical water height (coloration is issue from the kinetic energy)
Test case III: Tsunami runup onto a complex three dimensional Monai-Walley beach

Figure: Numerical water height (coloration is issue from the kinetic energy)
Test case III: Tsunami runup onto a complex three dimensional Monai-Walley beach

Figure: Numerical water height (coloration is issue from the kinetic energy)
Test case III: Tsunami runup onto a complex three dimensional Monai-Walley beach

Figure: Free surface results at different positions: experimental data versus numerical simulation with and without mesh adaptivity
Test case III: Tsunami runup onto a complex three dimensional Monai-Walley beach

Figure: Time evolution of the mesh refinement threshold and the number of cells: speed up the computation by 3 time
Thank you

for your

attention
Example

Let us consider the transport equation:

\[
\begin{align*}
    w_t + w_x &= 0 \\
    w(0, x) &= w_0(x)
\end{align*}
\]
**Example**

Let us consider the transport equation:

\[
\begin{align*}
  w_t + w_x &= 0 \\
  w(0, x) &= w_0(x)
\end{align*}
\]

and the Godunov scheme:

\[
\begin{align*}
  w_{k}^{n+1} &= w_{k}^{n} - \frac{\delta t}{\delta x} (w_{k}^{n} - w_{k-1}^{n})
\end{align*}
\]
Example

Let us consider the transport equation:

\[
\begin{aligned}
    w_t + w_x &= 0 \\
    w(0, x) &= w_0(x)
\end{aligned}
\]

and the Godunov scheme:

\[
\begin{aligned}
    w_{k}^{n+1} &= w_{k}^{n} - \frac{\delta t}{\delta x} (w_{k}^{n} - w_{k-1}^{n}) \\
    S_{k}^{n+1} &= \frac{s(w_{k}^{n+1}) - s(w_{k}^{n})}{\delta t} + \frac{\psi(s(w_{k}^{n})) - \psi(s(w_{k-1}^{n}))}{\delta x}
\end{aligned}
\]

with \( s(w) = w^2 \) and \( \psi(w) = w^2 \).
Example

Let us consider the transport equation:

\[
\begin{aligned}
&w_t + w_x = 0 \\
&w(0, x) = w_0(x)
\end{aligned}
\]

and the Godunov scheme:

\[
\begin{cases}
&w_{k}^{n+1} = w_{k}^{n} - \frac{\delta t}{\delta x} (w_{k}^{n} - w_{k-1}^{n}) \\
&S_{k}^{n+1} = \frac{s(w_{k}^{n+1}) - s(w_{k}^{n})}{\delta t} + \frac{\psi(s(w_{k}^{n})) - \psi(s(w_{k}^{n-1}))}{\delta x}
\end{cases}
\]

with \(s(w) = w^2\) and \(\psi(w) = w^2\).

Substituting \(w_{k}^{n+1}\) into \(S_{k}^{n+1}\), we get

\[
S_{k}^{n+1} = -\varepsilon \left( \frac{w_{k}^{n} - w_{k-1}^{n}}{\delta x} \right)^2 \leq 0 \text{ with } \varepsilon = \delta x \left( 1 - \frac{\delta t}{\delta x} \right) > 0.
\]
Explicit adaptive schemes: time consuming due to the restriction

\[ \|w\| \frac{\delta t}{h} \leq 1, \quad h = \min_k h_k \]
Explicit adaptive schemes: time consuming due to the restriction
\[ \|w\| \frac{\delta t}{h} \leq 1, \quad h = \min_k h_k \]

Local time stepping algorithm: save the cpu-time
- Sort cells in groups w.r.t. to their level

Explicit adaptive schemes: time consuming due to the restriction

\[ \|w\| \frac{\delta t}{h} \leq 1, \quad h = \min_k h_k \]

Local time stepping algorithm: save the cpu-time
- Sort cells in groups w.r.t. to their level
- Update the cells following the local time stepping algorithm.

with

$$\delta F_{k-1,k,k+1}^n := \left( F_{k+1/2}^n (w_k, w_{k+1}) - F_{k-1/2}^n (w_{k-1}, w_k) \right)$$
**Figure**: $t_{n_1} = t_n + \delta t_n$

with

$$\delta F^n_{k-1,k,k+1} := \left( F^n_{k+1/2}(w_{k}, w_{k+1}) - F^n_{k-1/2}(w_{k-1}, w_{k}) \right)$$
\[
\begin{align*}
\mathbf{w}^{n_2}_{k-1_0} &= \mathbf{w}^{n_1}_{k-1_0} \\
\mathbf{w}^{n_2}_{k00} &= \mathbf{w}^{n_1}_{k00} \\
\mathbf{w}^{n_2}_{k000} &= \mathbf{w}^{n_1}_{k000} \\
\mathbf{w}^{n_2}_{k001} &= \mathbf{w}^{n_1}_{k001}
\end{align*}
\]

**Figure:** \( t_{n_2} = t_n + 2\delta t_n \)

with

\[
\delta F^n_{k-1,k,k+1} := \left( F^n_{k+1/2}(\mathbf{w}_k, \mathbf{w}_{k+1}) - F^n_{k-1/2}(\mathbf{w}_{k-1}, \mathbf{w}_k) \right)
\]
\[ w_{k00}^{n3} = w_{k00}^{n2} - \delta t_n \frac{\delta F_{k000,k000,k01}^{n2}}{h_{k000}} \]

\[ w_{k01}^{n3} = w_{k01}^{n2} - \delta t_n \frac{\delta F_{k000,k001,k+1}^{n2}}{h_{k001}} \]

**Figure:** \( t_{n3} = t_n + 3\delta t_n \)

with

\[ \delta F_{k-1,k,k+1}^n := \left( F_{k+1/2}^n(w_k, w_{k+1}) - F_{k-1/2}^n(w_{k-1}, w_k) \right) \]
\[ w_{n+1}^{n+1} = w_{n+1}^{n} - \frac{\delta t_n}{h_{k-10}} \delta F_{k-2b,k-10,k00}^{n3} \]

\[ w_{n+1}^{n0} = w_{n0}^{n} - \frac{\delta t_n}{h_{k00}} \delta F_{k-10,k00,k000}^{n3} \]

\[ w_{n+1}^{n00} = w_{n00}^{n} - \frac{\delta t_n}{h_{k000}} \delta F_{k000,k000,k001}^{n3} \]

\[ w_{n+1}^{n01} = w_{n01}^{n} - \frac{\delta t_n}{h_{k001}} \delta F_{k000,k001,k1+b}^{n3} \]

with

\[ \delta F_{k-1,k,k+1}^{n} := \left( F_{k+1/2}(w_{k}, w_{k+1}) - F_{k-1/2}(w_{k-1}, w_{k}) \right) \]
LOCAL TIME STEPPING ALGORITHM

\[
\text{foreach } i \in \{1, 2^N\} \text{ do}
\]

Let \( j \) be the biggest integer such that \( 2^j \) divides \( i \)

\[
\text{foreach interface } x_{k+1/2} \text{ such that } L_{k+1/2} \geq N - j \text{ do}
\]

1. compute the integral of \( F_{k+1/2}(t) \) on the time interval \( 2^{N-L_{k+1/2}} \delta t_n \),
2. distribute \( F_{k+1/2}(t_n) \) to the two adjacent cells,
3. update only the cells of level greater than \( N - j \).

end
Shu and Osher test case

Mesh coarsening parameter $\alpha$ : $0.001$,

Mesh refinement parameter $\bar{S}$ : $\frac{1}{|\Omega|} \sum_{k_b} S_{k_b}^n$,

CFL : $0.219$,

Simulation time ($s$) : $0.18$,

Initial number of cells : $500$,

Maximum level of mesh refinement : $L_{\text{max}} = 4$.
Efficiency of the local time stepping method

| Method | $\mathcal{P}$ | $||\rho - \rho_{ref}||_{L^1}$ | CPU-time | $N_{L_{max}}$ | Maximum number of cells |
|--------|---------------|-----------------|---------|---------------|------------------------|
| AB1    | 0.288         | $4.74 \times 10^{-2}$ | 181     | 1574          | 2308                   |
| AB1M   | 0.288         | $4.80 \times 10^{-2}$ | 120     | 1572          | 2314                   |

**Table:** Shu and Osher test case: comparison of numerical schemes of order 1
<table>
<thead>
<tr>
<th>( P )</th>
<th>( | \rho - \rho_{ref} |_{L_1^x} )</th>
<th>cpu-time</th>
<th>( N_{L_{\text{max}}} )</th>
<th>maximum number of cells</th>
</tr>
</thead>
<tbody>
<tr>
<td>AB1</td>
<td>0.288</td>
<td>4.74 ( 10^{-2} )</td>
<td>181</td>
<td>1574</td>
</tr>
<tr>
<td>AB1M</td>
<td>0.288</td>
<td>4.80 ( 10^{-2} )</td>
<td>120</td>
<td>1572</td>
</tr>
<tr>
<td>AB2</td>
<td>0.287</td>
<td>2.75 ( 10^{-2} )</td>
<td>170</td>
<td>1391</td>
</tr>
</tbody>
</table>

**Table:** Shu and Osher test case: comparison of numerical schemes of order 1 and 2.
\begin{table}[ht]
\centering
\begin{tabular}{|c|c|c|c|c|c|}
\hline
 & $\mathcal{P}$ & $\|\rho - \rho_{ref}\|_{L_1^x}$ & cpu-time & $N_{L_{\text{max}}}$ & maximum number of cells \\
\hline
AB1 & 0.288 & $4.74 \times 10^{-2}$ & 181 & 1574 & 2308 \\
AB1M & 0.288 & $4.80 \times 10^{-2}$ & 120 & 1572 & 2314 \\
\hline
AB2 & 0.287 & $2.75 \times 10^{-2}$ & 170 & 1391 & 2023 \\
AB2M & 0.286 & $2.74 \times 10^{-2}$ & 108 & 1357 & 1994 \\
\hline
\end{tabular}
\caption{Shu and Osher test case: comparison of numerical schemes of order 1 and 2}
\end{table}
Table: Shu and Osher test case: comparison of numerical schemes of order 1 and 2

<table>
<thead>
<tr>
<th></th>
<th>$P$</th>
<th>$|\rho - \rho_{ref}|_{L_x}^1$</th>
<th>cpu-time</th>
<th>$N_{L_{max}}$</th>
<th>maximum number of cells</th>
</tr>
</thead>
<tbody>
<tr>
<td>AB1</td>
<td>0.288</td>
<td>$4.74 \times 10^{-2}$</td>
<td>181</td>
<td>1574</td>
<td>2308</td>
</tr>
<tr>
<td>AB1M</td>
<td>0.288</td>
<td>$4.80 \times 10^{-2}$</td>
<td>120</td>
<td>1572</td>
<td>2314</td>
</tr>
<tr>
<td>AB2</td>
<td>0.287</td>
<td>$2.75 \times 10^{-2}$</td>
<td>170</td>
<td>1391</td>
<td>2023</td>
</tr>
<tr>
<td>AB2M</td>
<td>0.286</td>
<td>$2.74 \times 10^{-2}$</td>
<td>108</td>
<td>1357</td>
<td>1994</td>
</tr>
<tr>
<td>RK2</td>
<td>0.285</td>
<td>$2.08 \times 10^{-2}$</td>
<td>299</td>
<td>1375</td>
<td>2005</td>
</tr>
</tbody>
</table>

\[ \| \rho - \rho_{ref} \|_1 \leq \mathcal{P} \]

<table>
<thead>
<tr>
<th></th>
<th>$\mathcal{P}$</th>
<th>$| \rho - \rho_{ref} |_1$</th>
<th>cpu-time</th>
<th>$N_{L_{\text{max}}}$</th>
<th>maximum number of cells</th>
</tr>
</thead>
<tbody>
<tr>
<td>AB1</td>
<td>0.288</td>
<td>$4.74 \times 10^{-2}$</td>
<td>181</td>
<td>1574</td>
<td>2308</td>
</tr>
<tr>
<td>AB1M</td>
<td>0.288</td>
<td>$4.80 \times 10^{-2}$</td>
<td>120</td>
<td>1572</td>
<td>2314</td>
</tr>
<tr>
<td>AB2</td>
<td>0.287</td>
<td>$2.75 \times 10^{-2}$</td>
<td>170</td>
<td>1391</td>
<td>2023</td>
</tr>
<tr>
<td>AB2M</td>
<td>0.286</td>
<td>$2.74 \times 10^{-2}$</td>
<td>108</td>
<td>1357</td>
<td>1994</td>
</tr>
<tr>
<td>RK2</td>
<td>0.285</td>
<td>$2.08 \times 10^{-2}$</td>
<td>299</td>
<td>1375</td>
<td>2005</td>
</tr>
</tbody>
</table>

**Table:** Shu and Osher test case: comparison of numerical schemes of order 1 and 2

Reference solution & Numerical results

(a) Density and numerical density of entropy production.

(b) Zoom on oscillating region.

Figure: Shu and Osher test case.
Main difficulty: mesh and data structure.

For fast computation, the following are required:

- parallel treatment
- hierarchical grids
TWO AND THREE DIMENSIONAL CASE: BB-AMR

- Main difficulty: mesh and data structure.
  Some interesting issues:
  - 2D quad-tree [1],
  - 3D octree [2],
  - 2/3D anisotropic AMR [3].

TWO AND THREE DIMENSIONAL CASE: BB-AMR

- Main difficulty: mesh and data structure.
- The strategy adopted:
Main difficulty: mesh and data structure.

The strategy adopted:

1. **1 domain = 1 block = 1 cpu**: “failure” → synchronization depends on the finest domain

2. **1 domain = n blocks = 1 cpu**: “good compromise” → each domain has almost the same number of cells → “better” synchronization

It certainly exists better strategies...
Main difficulty: mesh and data structure.

The strategy adopted:

1. Domain = 1 block = 1 CPU: "failure" → synchronization depends on the finest domain

2. Domain = n × blocks = 1 CPU: "good compromise" → each domain has almost the same number of cells → "better" synchronization
TWO AND THREE DIMENSIONAL CASE: BB-AMR

- Main difficulty: mesh and data structure.
- The strategy adopted:
  1. 1 domain = 1 block = 1 cpu: “failure” → synchronization depends on the finest domain
  2. 1 domain = n × blocks = 1 cpu: “good compromise” → each domain has almost the same number of cells → “better” synchronization
  3. It certainly exists better strategy . . .
TWO AND THREE DIMENSIONAL CASE : BB-AMR

- Main difficulty: mesh and data structure.
- The strategy adopted:
- Management of domain’s interfaces, projection step, . . .
How it works?

- each domain has almost the same number of cells
How it works?

- each domain has almost the same number of cells
- domain are defined using Cuthill-McKee numbering
How it works?

- each domain has almost the same number of cells
- domain are defined using Cuthill-McKee numbering

\[
\begin{array}{ccc}
6 & 9 & \\
3 & 5 & 8 \\
1 & 2 & 4 & 7 \\
\end{array}
\]
How it works?

- each domain has almost the same number of cells
- domain are defined using Cuthill-McKee numbering

\[
\begin{array}{ccc}
6 & 9 & \\
5 & 8 & \\
D_1 & 4 & 7
\end{array}
\]
How it works?

- Each domain has almost the same number of cells
- Domain are defined using Cuthill-McKee numbering

```
 3  6  9
D1 2 5 8
   1 4 7
```
DOMAIN = N × BLOCKS = 1 CPU

How it works?

- each domain has almost the same number of cells
- domain are defined using Cuthill-McKee numbering

\[ \begin{array}{ccc}
D_1 & 6 & 9 \\
D_2 & 5 & 8 \\
& 4 & 7 \\
\end{array} \]
How it works?

- each domain has almost the same number of cells
- domain are defined using Cuthill-McKee numbering
DOMAIN = N x BLOCKS = 1 CPU

How it works?

- each domain has almost the same number of cells
- domain are defined using Cuthill-McKee numbering

Gain is important and numerical stability is conserved!
How it works?

- each domain has almost the same number of cells
- domain are defined using Cuthill-McKee numbering
How it works?

- each domain has almost the same number of cells
- domain are defined using Cuthill-McKee numbering
- more sophisticated numbering exists . . .
Each domain has almost the same number of cells. Domains are defined using Cuthill-McKee numbering. More sophisticated numbering exists, but re-numbering and re-meshing are expensive. The mesh should be kept constant on a time interval to ensure numerical stability.
How it works?

- each domain has almost the same number of cells
- domain are defined using Cuthill-McKee numbering
- more sophisticated numbering exists . . .
- re-numbering and re-meshing being expensive
  - the mesh should kept constant on a time interval
  - AMR time-step computed through the smallest block and not the smallest cell

\[ T_{n+1} - T_n = \Delta T_{\text{AMR}} \] is given by the CFL

\[ \Delta T_{\text{AMR}} \leq \beta \frac{\min_k h_{\text{block}_k}}{\max_k \| u_{\text{block}_k} \|}, \quad 0 < \beta \leq 1. \]

\[
\begin{array}{c|c|c}
T_0 & \delta t & T_1 \\
\hline
\text{AMR} & \text{AMR} & \text{AMR}
\end{array}
\]
How it works?

- each domain has almost the same number of cells
- domain are defined using Cuthill-McKee numbering
- more sophisticated numbering exists . . .
- re-numbering and re-meshing being expensive
  - the mesh should kept constant on a time interval
  - AMR time-step computed through the smallest block and not the smallest cell
  - Gain is important and numerical stability is conserved!
Examples:

- A two dimensional example of BB-AMR with 3 domains and 9 blocks.
Examples:

- A two dimensional example of BB-AMR with 3 domains and 9 blocks.

- A three dimensional example of BB-AMR with 3 domains and 27 blocks.