Adaptive multi scale scheme based on numerical density of entropy production for conservation laws.

Mehmet Ersoy [1], Frédéric Golay [2] and Lyudmyla Yushchenko [3]

Workshop MTM2011-29306, 18-19 February 2013

1. *Mehmet.Ersoy@univ-tln.fr*
2. *Frederic.Golay@univ-tln.fr*
3. *Lyudmyla.Yushchenko@univ-tln.fr*

Outline of the talk

We focus on general non linear hyperbolic conservation laws

$$\begin{cases} \dfrac{\partial \boldsymbol{w}}{\partial t} + \dfrac{\partial \boldsymbol{f}(\boldsymbol{w})}{\partial x} = 0,\ (t,x) \in \mathbb{R}^+ \times \mathbb{R} \\ \boldsymbol{w}(0,x) = \boldsymbol{w}_0(x),\ x \in \mathbb{R}. \end{cases} \tag{1}$$

where

$\boldsymbol{w} \in \mathbb{R}^d$	:	vector state,
$\boldsymbol{f}$	:	flux governing the physical description of the flow.

It is well-known, even if the initial data are smooth, that : at a finite time :

- solutions develop complex discontinuous structure
- uniqueness is lost

Serre D., *Systems of conservation laws*, (99) ; Eymard R., Gallouët T., Herbin R., *The finite volume method*, (00) ;

We focus on general non linear hyperbolic conservation laws

$$\begin{cases} \dfrac{\partial \boldsymbol{w}}{\partial t} + \dfrac{\partial \boldsymbol{f}(\boldsymbol{w})}{\partial x} = 0,\ (t,x) \in \mathbb{R}^+ \times \mathbb{R} \\ \boldsymbol{w}(0,x) = \boldsymbol{w}_0(x),\ x \in \mathbb{R}. \end{cases} \tag{1}$$

where

$\boldsymbol{w} \in \mathbb{R}^d$	:	vector state,
$\boldsymbol{f}$	:	flux governing the physical description of the flow.

It is well-known, even if the initial data are smooth, that : at a finite time :

- solutions develop complex discontinuous structure
- uniqueness is lost

and is recovered (weak physical solution) by completing the system (1) with an entropy inequality of the form :

$$\frac{\partial s(\boldsymbol{w})}{\partial t} + \frac{\partial \psi(\boldsymbol{w})}{\partial x} \leqslant 0$$

where (s, ψ) stands for a convex entropy-entropy flux pair.

Serre D., *Systems of conservation laws*, (99) ; Eymard R., Gallouët T., Herbin R., *The finite volume method*, (00) ;

The role of the entropy inequality

The inequality

$$\frac{\partial s(\boldsymbol{w})}{\partial t} + \frac{\partial \psi(\boldsymbol{w})}{\partial x} \leqslant 0$$

provides information on the regularity of the solution :

- $\equiv 0$ if solutions are smooth
- < 0 if solutions are discontinuous

Houston P., Mackenzie J.A., Süli E., Warnecke G., *Numer. Math.*, (99) ; Karni S., Kurganov A., Petrova G., *J. Comp. Phys.*, (02) ; Puppo G., *SIAM J. Sci. Comput.* ICOSAHOM, (03) ; Puppo G., Semplice M., *Commun. Comput. Phys.*, (03) ; Golay F., *C.R. Mécanique*, (09) ;

The role of the entropy inequality

The inequality

$$\frac{\partial s(\boldsymbol{w})}{\partial t} + \frac{\partial \psi(\boldsymbol{w})}{\partial x} \leqslant 0$$

provides information on the regularity of the solution :

- $\equiv 0$ if solutions are smooth
- < 0 if solutions are discontinuous

Numerical approximation of this inequality, called *numerical density of entropy production*, measure the amount of violation of the entropy equation (as a measure of the local residual).

Houston P., Mackenzie J.A., Süli E., Warnecke G., *Numer. Math.*, (99) ; Karni S., Kurganov A., Petrova G., *J. Comp. Phys.*, (02) ; Puppo G., *SIAM J. Sci. Comput.* ICOSAHOM, (03) ; Puppo G., Semplice M., *Commun. Comput. Phys.*, (03) ; Golay F., *C.R. Mécanique*, (09) ;

The inequality

$$\frac{\partial s(\boldsymbol{w})}{\partial t} + \frac{\partial \psi(\boldsymbol{w})}{\partial x} \leqslant 0$$

provides information on the regularity of the solution :

- $\equiv 0$ if solutions are smooth
- < 0 if solutions are discontinuous

Numerical approximation of this inequality, called *numerical density of entropy production*, measure the amount of violation of the entropy equation (as a measure of the local residual).
As a consequence, the numerical density of entropy production provides information on the need :

- to coarsen the mesh if solutions are smooth
- locally refine the mesh if solutions are discontinuous

Houston P., Mackenzie J.A., Süli E., Warnecke G., *Numer. Math.*, (99) ; Karni S., Kurganov A., Petrova G., *J. Comp. Phys.*, (02) ; Puppo G., *SIAM J. Sci. Comput.* ICOSAHOM, (03) ; Puppo G., Semplice M., *Commun. Comput. Phys.*, (03) ; Golay F., *C.R. Mécanique*, (09) ;

The inequality

$$\frac{\partial s(\boldsymbol{w})}{\partial t} + \frac{\partial \psi(\boldsymbol{w})}{\partial x} \leqslant 0$$

provides information on the regularity of the solution :

- $\equiv 0$ if solutions are smooth
- < 0 if solutions are discontinuous

Numerical approximation of this inequality, called *numerical density of entropy production*, measure the amount of violation of the entropy equation (as a measure of the local residual).
As a consequence, the numerical density of entropy production provides information on the need :

- to coarsen the mesh if solutions are smooth
- locally refine the mesh if solutions are discontinuous

Conclusion : an intrinsic *a posteriori* error indicator $\Longrightarrow$ *automatic mesh refinement*

Houston P., Mackenzie J.A., Süli E., Warnecke G., *Numer. Math.*, (99) ; Karni S., Kurganov A., Petrova G., *J. Comp. Phys.*, (02) ; Puppo G., *SIAM J. Sci. Comput.* ICOSAHOM, (03) ; Puppo G., Semplice M., *Commun. Comput. Phys.*, (03) ; Golay F., *C.R. Mécanique*, (09) ;

An important time restriction

Explicit adaptive schemes are well-know to be time consuming due to a CFL stability condition since :

the CFL imposes an upper bound on $\frac{\delta t}{h}$ where δt is the time step and h the finest mesh size.

Müller S., Stiriba Y., *SIAM J. Sci. Comput.*, (07) ; Tan Z., Zhang Z., Huang Y., Tang T., *J. Comp. Phys.*, (04) ; Ersoy M., Golay F., Yushchenko L., *CEJM*, (13) ;

An important time restriction, local time stepping approach

Explicit adaptive schemes are well-know to be time consuming due to a CFL stability condition since :

the CFL imposes an upper bound on $\frac{\delta t}{h}$ where δt is the time step and h the finest mesh size.

Nevertheless, the cpu-time can be significantly reduced using the local time stepping algorithm

Müller S., Stiriba Y., *SIAM J. Sci. Comput.*, (07) ; Tan Z., Zhang Z., Huang Y., Tang T., *J. Comp. Phys.*, (04) ; Ersoy M., Golay F., Yushchenko L., *CEJM*, (13) ;

Explicit adaptive schemes are well-know to be time consuming due to a CFL stability condition since :

the CFL imposes an upper bound on $\frac{\delta t}{h}$ where δt is the time step and h the finest mesh size.

Nevertheless, the cpu-time can be significantly reduced using the local time stepping algorithm

Aims : save the cpu-time
by making use of the

- local time stepping algorithm

Müller S., Stiriba Y., *SIAM J. Sci. Comput.*, (07) ; Tan Z., Zhang Z., Huang Y., Tang T., *J. Comp. Phys.*, (04) ; Ersoy M., Golay F., Yushchenko L., *CEJM*, (13) ;

An important time restriction, local time stepping approach & Aims

Explicit adaptive schemes are well-know to be time consuming due to a CFL stability condition since :

the CFL imposes an upper bound on $\frac{\delta t}{h}$ where δt is the time step and h the finest mesh size.

Nevertheless, the cpu-time can be significantly reduced using the local time stepping algorithm

Aims : save the cpu-time keeping the order of accuracy
by making use of the

- automatic mesh refinement algorithm
- local time stepping algorithm

Müller S., Stiriba Y., *SIAM J. Sci. Comput.*, (07) ; Tan Z., Zhang Z., Huang Y., Tang T., *J. Comp. Phys.*, (04) ; Ersoy M., Golay F., Yushchenko L., *CEJM*, (13) ;

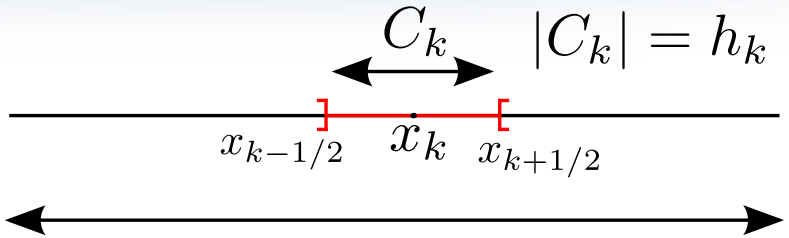


Figure: a cell C_k

Integrating

$$\begin{cases} \dfrac{\partial \boldsymbol{w}}{\partial t} + \dfrac{\partial \boldsymbol{f}(\boldsymbol{w})}{\partial x} & = & 0 \\ \dfrac{\partial s(\boldsymbol{w})}{\partial t} + \dfrac{\partial \psi(\boldsymbol{w})}{\partial x} & \leqslant & 0 \end{cases}$$

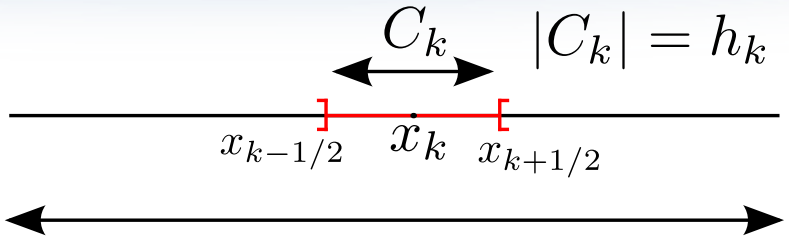


Figure: a cell C_k

over each cells $C_k \times (t_n, t_{n+1})$ we obtain :

$$\int_{C_k} \boldsymbol{w}\left(t_{n+1}, x\right)\, dx - \int_{C_k} \boldsymbol{w}\left(t_n, x\right) dx + \int_{t_n}^{t_{n+1}} \boldsymbol{f}(\boldsymbol{w}(t, x_{i+1/2})) - \boldsymbol{f}(\boldsymbol{w}(t, x_{i-1/2}))dt = 0$$

$$S = \int_{C_k} s(\boldsymbol{w}(t_{n+1}, x))dx - \int_{C_k} s(\boldsymbol{w}(t_n, x))dx + \int_{t_n}^{t_{n+1}} \psi(\boldsymbol{w}(t, x_{i+1/2})) - \psi(\boldsymbol{w}(t, x_{i-1/2}))dt$$

where is the density of entropy production and should satisfy

$$S \leqslant 0.$$

Choosing

$$\boldsymbol{F}_{k+1/2}(\boldsymbol{w}_k^n, \boldsymbol{w}_{k+1}^n) \text{ as a suitable approximation of } \frac{1}{\delta t_n}\int_{t_n}^{t_{n+1}} \boldsymbol{f}(\boldsymbol{w}(x_{k\pm 1/2}, s)\, ds,$$

noting $\delta t_n = t_{n+1} - t_n$ and

$$\boldsymbol{w}_k^n \simeq \frac{1}{h_k}\int_{C_k} \boldsymbol{w}\left(x, t_n\right)\, dx$$

we obtain :

$$\boldsymbol{w}_k^{n+1} = \boldsymbol{w}_k^n - \frac{\delta t_n}{h_k}\left(\boldsymbol{F}_{k+1/2}^n - \boldsymbol{F}_{k-1/2}^n\right) \ ,$$

Choosing

$$\boldsymbol{F}_{k+1/2}(\boldsymbol{w}_k^n, \boldsymbol{w}_{k+1}^n) \text{ as a suitable approximation of } \frac{1}{\delta t_n}\int_{t_n}^{t_{n+1}} \boldsymbol{f}(\boldsymbol{w}(x_{k\pm 1/2}, s)\, ds,$$

noting $\delta t_n = t_{n+1} - t_n$ and

$$\boldsymbol{w}_k^n \simeq \frac{1}{h_k}\int_{C_k} \boldsymbol{w}\left(x, t_n\right)\, dx$$

we obtain :

$$\boldsymbol{w}_k^{n+1} = \boldsymbol{w}_k^n - \frac{\delta t_n}{h_k}\left(\boldsymbol{F}_{k+1/2}^n - \boldsymbol{F}_{k-1/2}^n\right) \ ,$$

Using the same discretisation, we get :

$$S_k^n = \frac{s_k^{n+1} - s_k^n}{\delta t_n} + \frac{\psi_{k+1/2}^n - \psi_{k-1/2}^n}{h_k} \ ,$$

called the numerical density of entropy production.

The numerical density of entropy production :

- if the solution is smooth, S_k^n fails to be 0.
- if the solution is discontinuous, S_k^n should be negative.

The numerical density of entropy production :

- if the solution is smooth, S_k^n fails to be 0.
- if the solution is discontinuous, S_k^n should be negative.

Thus, in both cases, S_k^n provides local information

- on the accuracy of the scheme
- on the need to refine the mesh

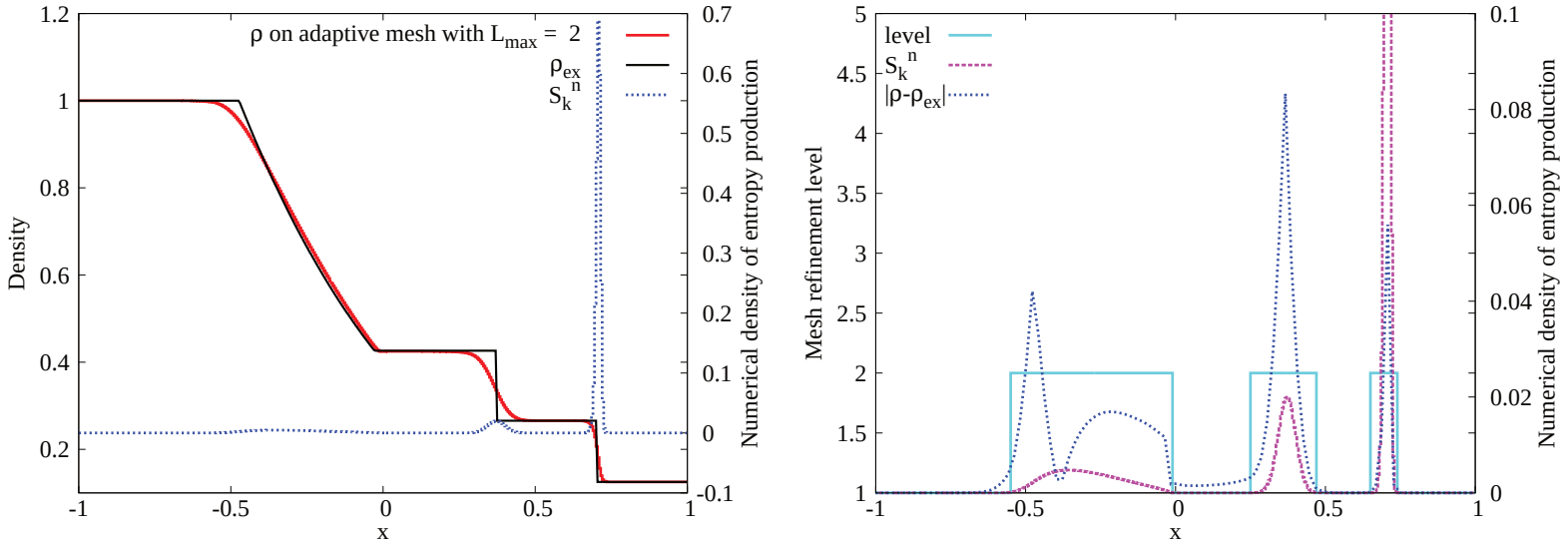
The numerical density of entropy production :

- if the solution is smooth, S_k^n fails to be 0.
- if the solution is discontinuous, S_k^n should be negative.

Thus, in both cases, S_k^n provides local information

- on the accuracy of the scheme
- on the need to refine the mesh

and it will be used as a mesh refinement indicator :



(a) Density and numerical density of entropy production ($-S_k^n$).

(b) Numerical density of entropy production and local error ($-S_k^n$).

FIGURE: Numerical example

In particular, one has :

Theorem

Consider a pth convergent scheme for Equations (1) and discretise the entropy inequality as Equations (1). Let S_k^n be the corresponding numerical density of entropy production and $\Delta = \lambda h$ be a fixed time step where h stands for the meshsize.
Then

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

In particular, one has :

Theorem

Consider a pth convergent scheme for Equations (1) and discretise the entropy inequality as Equations (1). Let S_k^n be the corresponding numerical density of entropy production and $\Delta = \lambda h$ be a fixed time step where h stands for the meshsize.
Then

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

and the following property is satisfied :

Properties

Consider a monotone scheme. Then, for almost every k, every n,

$$S_k^n \leqslant 0.$$

In particular, one has :

Theorem

Consider a pth convergent scheme for Equations (1) and discretise the entropy inequality as Equations (1). Let S_k^n be the corresponding numerical density of entropy production and $\Delta = \lambda h$ be a fixed time step where h stands for the meshsize.
Then

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

and the following property is satisfied :

Properties

Consider a monotone scheme. Then, for almost every k, every n,

$$S_k^n \leqslant 0.$$

Let us remark that :
even if locally S_k^n can take positive value, from the previous results, one has

$$S_k^n \leqslant C\Delta t^q, \quad q \geqslant p\ .$$

Let us consider the transport equation :

$$\left\{ \begin{array}{lcl} w_t + w_x & = & 0 \\ w(0,x) & = & w_0(x) \end{array} \right.$$

and the Godunov scheme :

$$\left\{ \begin{array}{lcl} w_k^{n+1} & = & w_k^n + \dfrac{\delta t}{\delta x}\left(w_k^n - w_{k-1}^n\right) \\ S_k^{n+1} & = & \dfrac{s(w_k^{n+1}) - s(w_k^n)}{\delta t} + \dfrac{\psi(s(w_k^n)) - \psi(s(w_{k-1}^n))}{\delta x} \end{array} \right.$$

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

Let us consider the transport equation :

$$\begin{cases} w_t + w_x &= 0 \\ w(0,x) &= w_0(x) \end{cases}$$

and the Godunov scheme :

$$\begin{cases} w_k^{n+1} &= w_k^n + \dfrac{\delta t}{\delta x}\left(w_k^n - w_{k-1}^n\right) \\ S_k^{n+1} &= \dfrac{s(w_k^{n+1}) - s(w_k^n)}{\delta t} + \dfrac{\psi(s(w_k^n)) - \psi(s(w_{k-1}^n))}{\delta x} \end{cases}$$

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

Substituting w_k^{n+1} into S_k^{n+1}, one find

$$S_k^{n+1} = -\varepsilon \left(\frac{w_k^n - w_{k-1}^n}{\delta x}\right)^2 \qquad \text{with } \varepsilon = \delta x \left(1 - \frac{\delta t}{\delta x}\right) > 0.$$

Let us consider the transport equation :

$$\begin{cases} w_t + w_x &= 0 \\ w(0,x) &= w_0(x) \end{cases}$$

and the Godunov scheme :

$$\begin{cases} w_k^{n+1} &= w_k^n + \dfrac{\delta t}{\delta x}\left(w_k^n - w_{k-1}^n\right) \\ S_k^{n+1} &= \dfrac{s(w_k^{n+1}) - s(w_k^n)}{\delta t} + \dfrac{\psi(s(w_k^n)) - \psi(s(w_{k-1}^n))}{\delta x} \end{cases}$$

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

Substituting w_k^{n+1} into S_k^{n+1}, one find

$$S_k^{n+1} = -\varepsilon \left(\frac{w_k^n - w_{k-1}^n}{\delta x}\right)^2 \leqslant 0 \text{ with } \varepsilon = \delta x \left(1 - \frac{\delta t}{\delta x}\right) > 0.$$

Notations

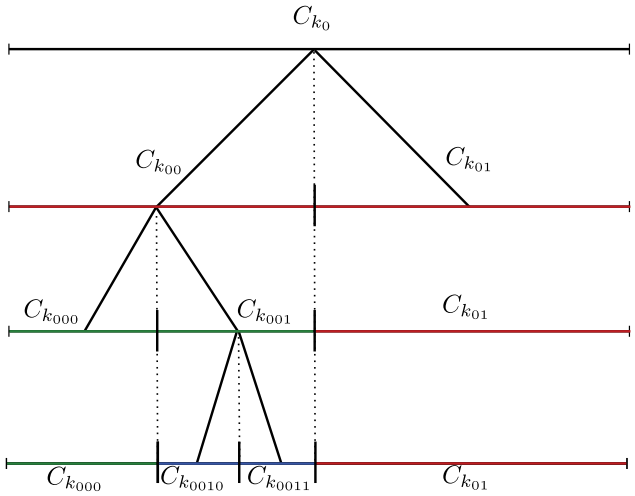


Figure: Example of hierarchical dyadic tree with three different cells

C_{k_0}	:	macro cell,
b	:	binary number which contains the hierarchical information of a sub-cell,
C_{k_b}	:	sub-cell of C_{k_0},
$L_{\max}$	:	maximum level of refinement C_{k_0},
$\mathrm{length(b)}$	:	level of refinement

Mesh refinement process : refinement& unrefinement

- define a mesh refinement parameter $\bar{S}$, say, the mean value over the domain Ω :

$$\bar{S} = \frac{1}{|\Omega|} \sum_{k_b} S^n_{k_b}$$

Mesh refinement process : refinement& unrefinement

- define a mesh refinement parameter $\bar{S}$, say, the mean value over the domain Ω :

$$\bar{S} = \frac{1}{|\Omega|} \sum_{k_b} S^n_{k_b}$$

- define two coefficients

$$\begin{array}{lcl} 0 \leqslant \alpha_{min} \leqslant 1 & : & \text{ratio of mesh coarsening,} \\ 0 \leqslant \alpha_{max} \leqslant 1 & : & \text{ratio of mesh refinement,} \end{array}$$

Mesh refinement process : refinement& unrefinement

- define a mesh refinement parameter $\bar{S}$, say, the mean value over the domain Ω :

$$\bar{S} = \frac{1}{|\Omega|} \sum_{k_b} S^n_{k_b}$$

- define two coefficients

$$\begin{array}{lcl} 0 \leqslant \alpha_{min} \leqslant 1 & : & \text{ratio of mesh coarsening,} \\ 0 \leqslant \alpha_{max} \leqslant 1 & : & \text{ratio of mesh refinement,} \end{array}$$

Then, for each cell C_{k_b} :

- if $S^n_{k_b} > \bar{S}\alpha_{max}$, the mesh is refined and split into two sub-cells $C_{k_{b_0}}$ and $C_{k_{b_1}}$,
- if $S^n_{k_{b_0}} < \bar{S}\alpha_{min}$ and $S^n_{k_{b_1}} < \bar{S}\alpha_{min}$, the mesh is coarsened into a cell C_{k_b}.

Mesh refinement process : refinement& unrefinement

- define a mesh refinement parameter $\bar{S}$, say, the mean value over the domain Ω :

$$\bar{S} = \frac{1}{|\Omega|} \sum_{k_b} S^n_{k_b}$$

- define two coefficients

$$\begin{array}{lcl} 0 \leqslant \alpha_{min} \leqslant 1 & : & \text{ratio of mesh coarsening,} \\ 0 \leqslant \alpha_{max} \leqslant 1 & : & \text{ratio of mesh refinement,} \end{array}$$

Then, for each cell C_{k_b} :

- if $S^n_{k_b} > \bar{S}\alpha_{max}$, the mesh is refined and split into two sub-cells $C_{k_{b_0}}$ and $C_{k_{b_1}}$,
- if $S^n_{k_{b_0}} < \bar{S}\alpha_{min}$ and $S^n_{k_{b_1}} < \bar{S}\alpha_{min}$, the mesh is coarsened into a cell C_{k_b}.

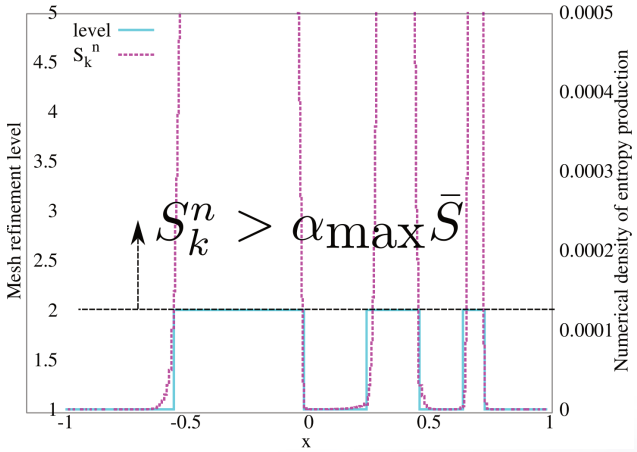
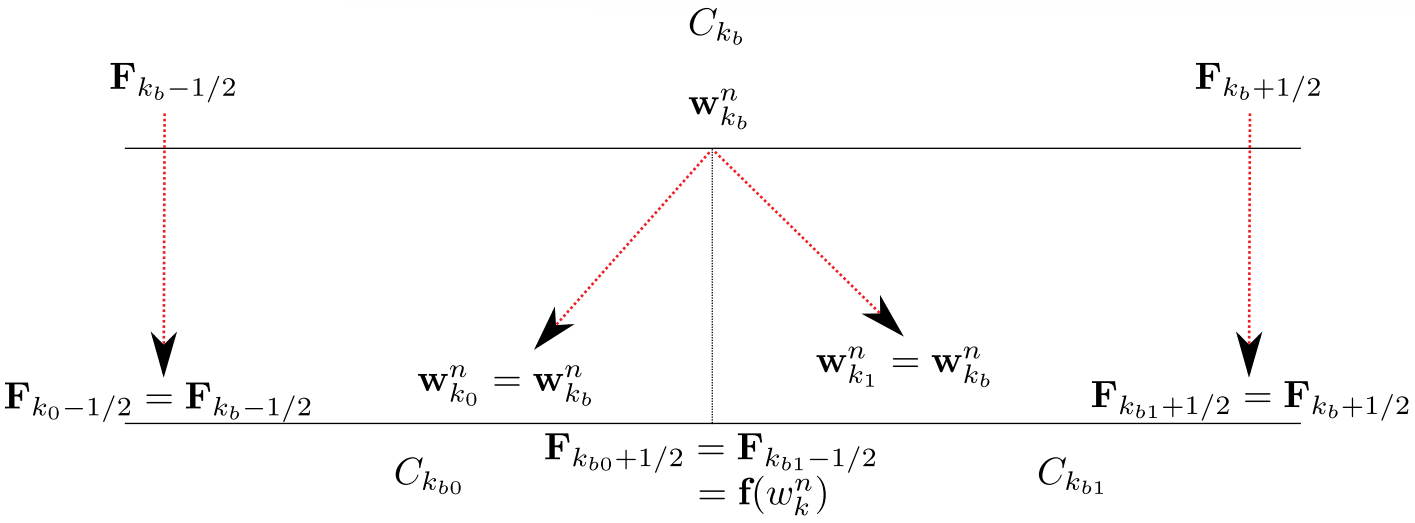
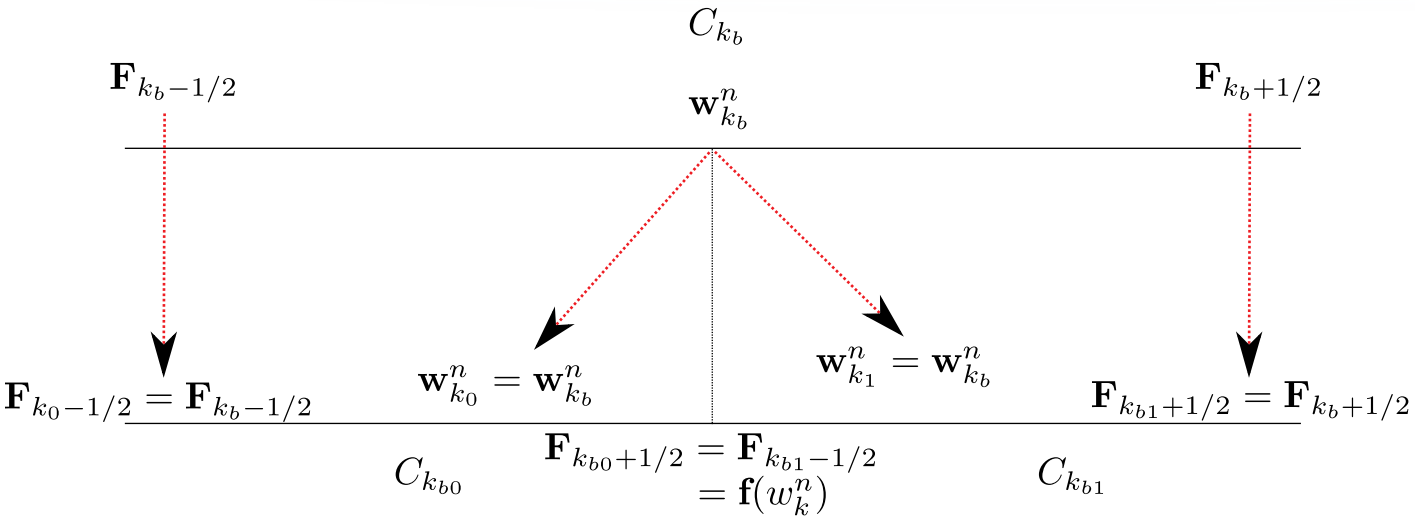


Figure: Example of $\bar{S}$ $(-S^n_k)$

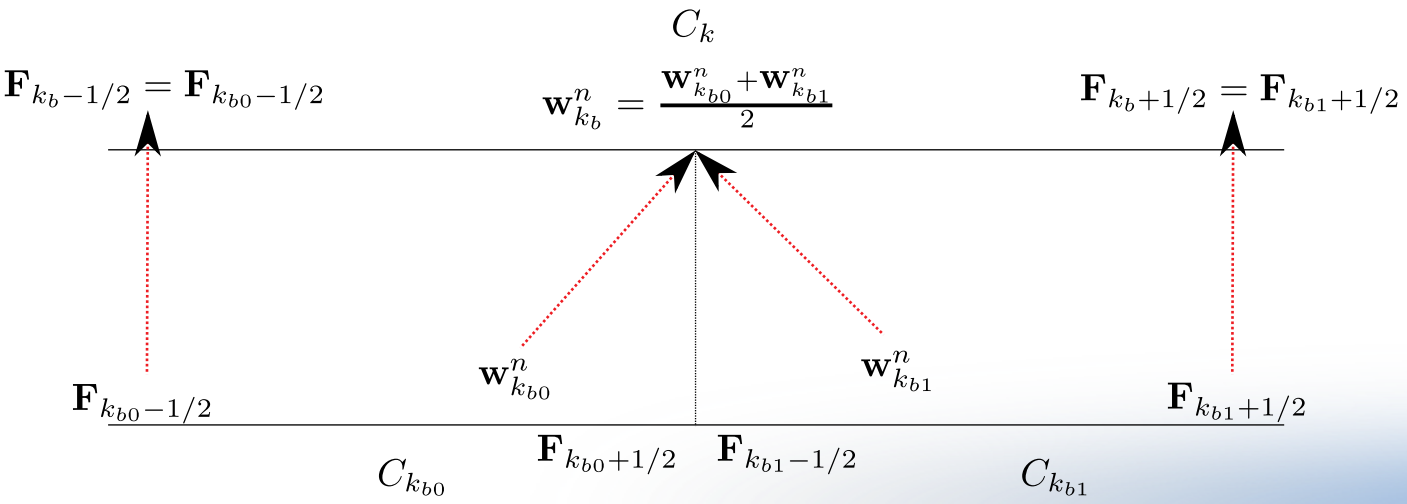
- If a cell C_{k_b} is split into two sub-cells $C_{k_{b0}}$ and $C_{k_{b1}}$, new subcells are initialized as follows :



- If a cell C_{k_b} is split into two sub-cells $C_{k_{b0}}$ and $C_{k_{b1}}$, new subcells are initialized as follows :



- if two sub-cells $C_{k_{b0}}$ and $C_{k_{b1}}$ is merged, the new cell C_{k_b} is initialized as follows :



Notations :

Let

$\Delta t_n = 2^N \delta t_n$	:	be the macro time step
δt_n	:	be themicro time step
L_k	:	be thelevel of refinement of the kth cell C_k defined by
	:	$2^{N-L_k} h_{\min} \leqslant h_k < 2^{N+1-L_k} h_{\min}$
$N = \log_2 \left(\frac{h_{\max}}{h_{\min}} \right) + 1$	:	be the maximum level of refinement

Notations :

Let

$\Delta t_n = 2^N \delta t_n$	:	be the macro time step
δt_n	:	be themicro time step
L_k	:	be thelevel of refinement of the kth cell C_k defined by
	:	$2^{N-L_k} h_{\min} \leqslant h_k < 2^{N+1-L_k} h_{\min}$
$N = \log_2 \left(\frac{h_{\max}}{h_{\min}} \right) + 1$	:	be the maximum level of refinement

Principle :

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

Let us note

$$\delta F^n_{k-1,k,k+1} := \left(\boldsymbol{F}^n_{k+1/2}(\boldsymbol{w}_k, \boldsymbol{w}_{k+1}) - \boldsymbol{F}^n_{k-1/2}(\boldsymbol{w}_{k-1}, \boldsymbol{w}_k)\right)$$

the flux differences and let us consider a first order scheme.

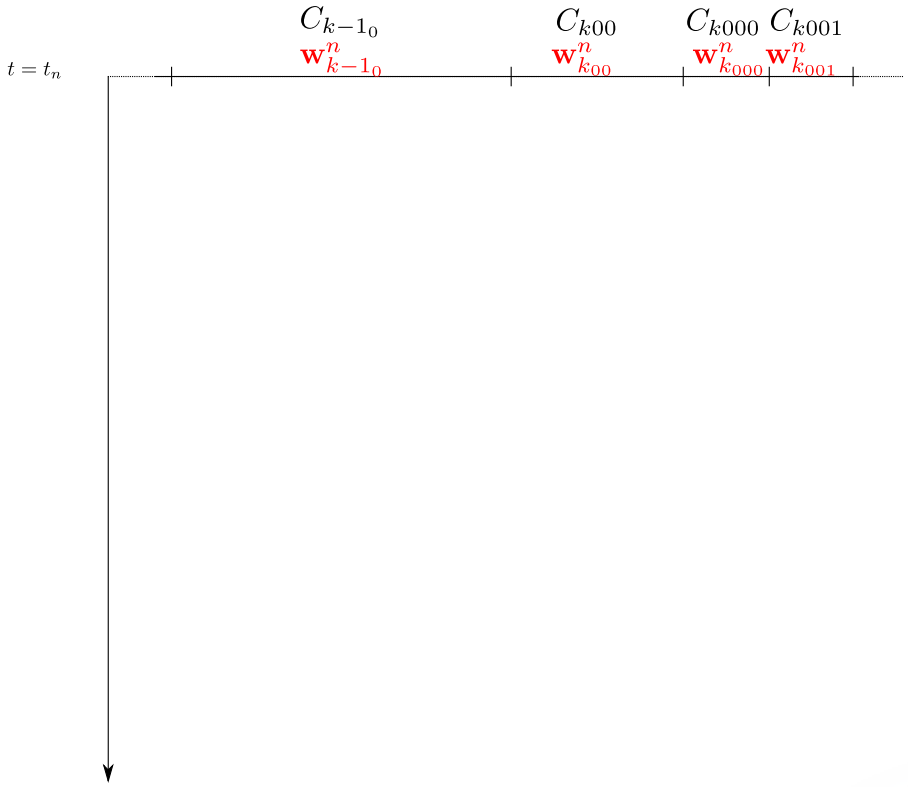
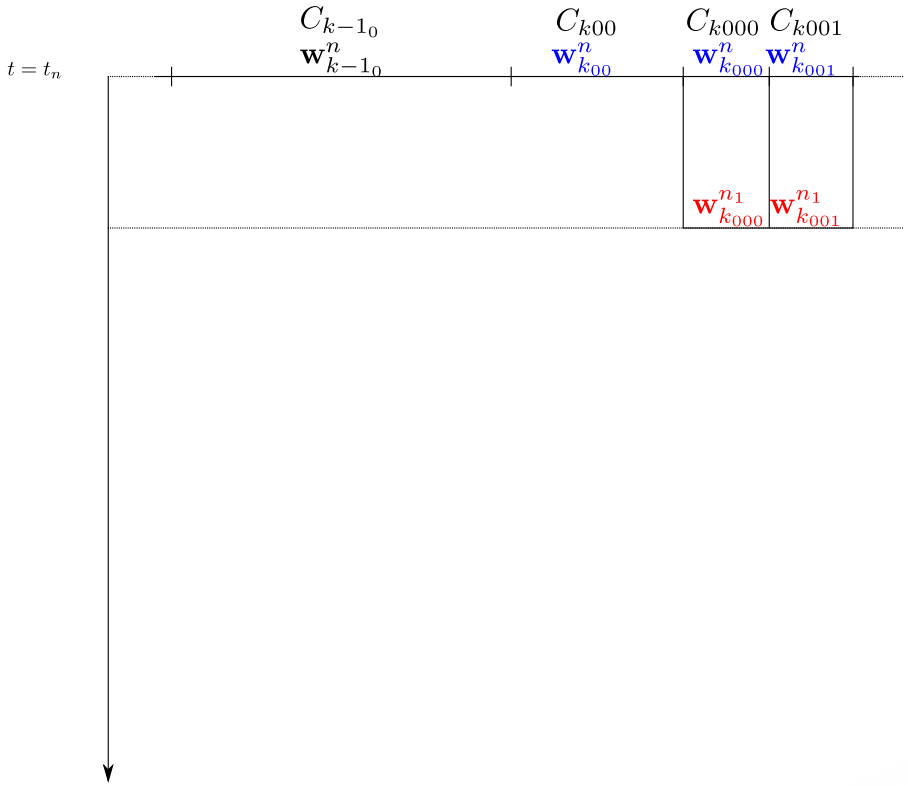


Figure: $t = t_n$

Let us note

$$\delta F^n_{k-1,k,k+1} := \left(\boldsymbol{F}^n_{k+1/2}(\boldsymbol{w}_k, \boldsymbol{w}_{k+1}) - \boldsymbol{F}^n_{k-1/2}(\boldsymbol{w}_{k-1}, \boldsymbol{w}_k)\right)$$

the flux differences and let us consider a first order scheme.



$$\boldsymbol{w}^{n_1}_{k_{000}} = \boldsymbol{w}^n_{k_{000}} - \frac{\delta t_n}{h_{k_{000}}} \delta \boldsymbol{F}^n_{k_{00},k_{000},k_{001}}$$

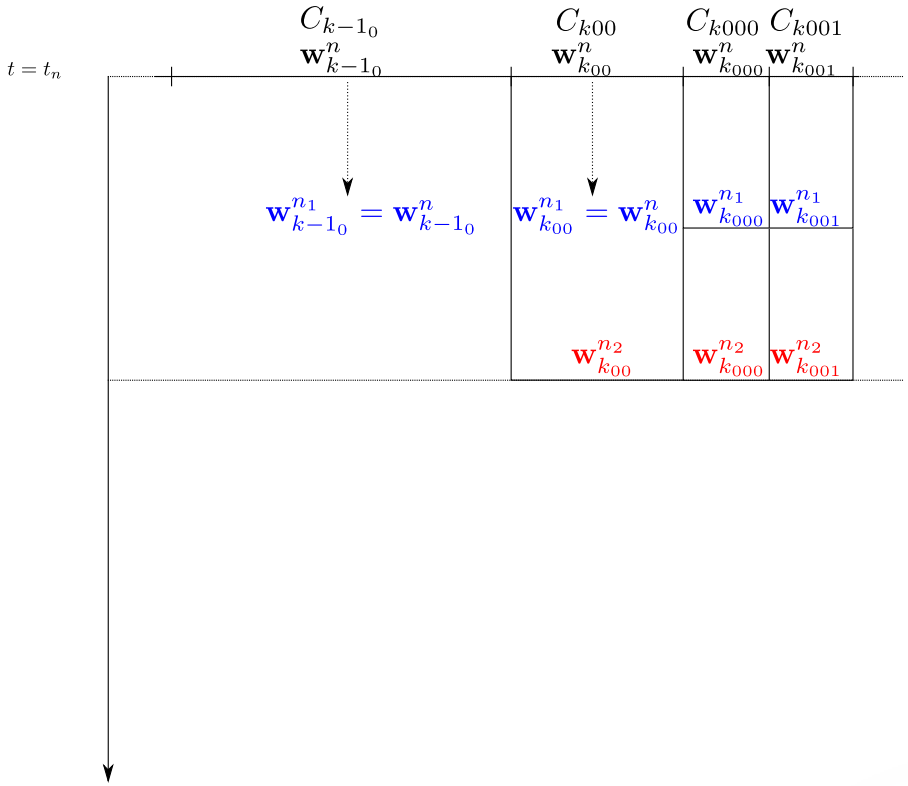
$$\boldsymbol{w}^{n_1}_{k_{001}} = \boldsymbol{w}^n_{k_{001}} - \frac{\delta t_n}{h_{k_{001}}} \delta \boldsymbol{F}^n_{k_{000},k_{001},k+1_b}$$

Figure: $t_{n_1} = t_n + \delta t_n$

Let us note

$$\delta F^n_{k-1,k,k+1} := \left(\boldsymbol{F}^n_{k+1/2}(\boldsymbol{w}_k, \boldsymbol{w}_{k+1}) - \boldsymbol{F}^n_{k-1/2}(\boldsymbol{w}_{k-1}, \boldsymbol{w}_k)\right)$$

the flux differences and let us consider a first order scheme.



$$\boldsymbol{w}^{n_2}_{k00} = \boldsymbol{w}^{n_1}_{k00} - \frac{\delta t_n}{h_{k00}} \delta \boldsymbol{F}^{n_1}_{k-1_0,k00,k000}$$

$$\boldsymbol{w}^{n_2}_{k000} = \boldsymbol{w}^{n_1}_{k000} - \frac{\delta t_n}{h_{k000}} \delta \boldsymbol{F}^{n_1}_{k00,k000,k001}$$

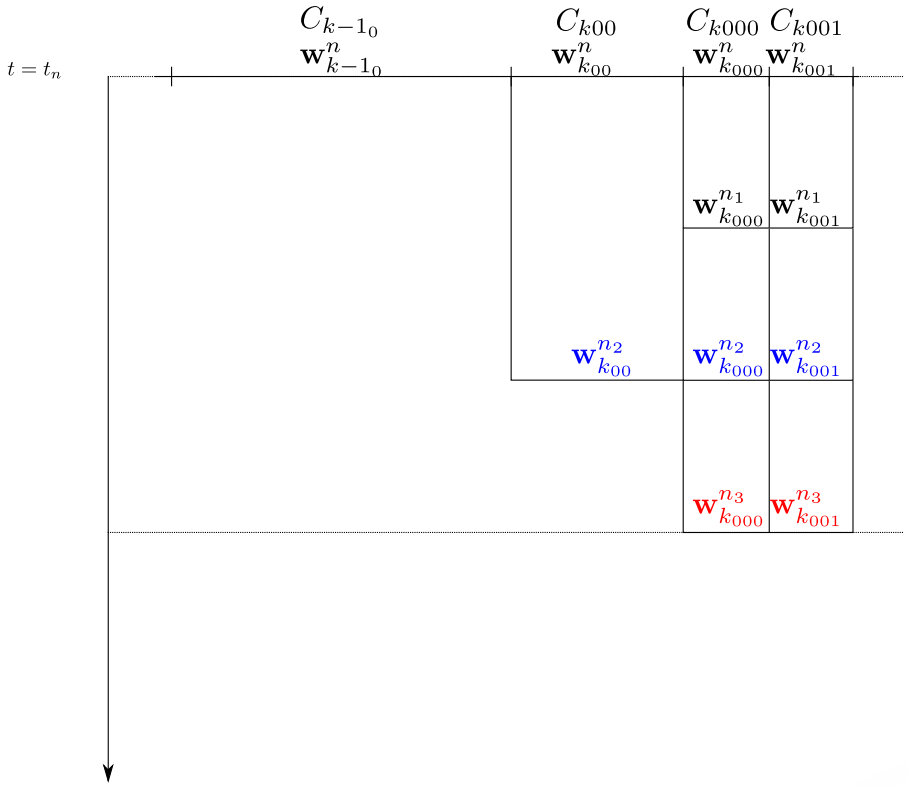
$$\boldsymbol{w}^{n_2}_{k001} = \boldsymbol{w}^{n_1}_{k001} - \frac{\delta t_n}{h_{k001}} \delta \boldsymbol{F}^{n_1}_{k000,k001,k+1_b}$$

Figure: $t_{n_2} = t_n + 2\delta t_n$

Let us note

$$\delta F^n_{k-1,k,k+1} := \left(\boldsymbol{F}^n_{k+1/2}(\boldsymbol{w}_k, \boldsymbol{w}_{k+1}) - \boldsymbol{F}^n_{k-1/2}(\boldsymbol{w}_{k-1}, \boldsymbol{w}_k)\right)$$

the flux differences and let us consider a first order scheme.



$$\boldsymbol{w}^{n_3}_{k_{000}} = \boldsymbol{w}^{n_2}_{k_{000}} - \frac{\delta t_n}{h_{k_{000}}} \delta \boldsymbol{F}^{n_2}_{k_{00},k_{000},k_{001}}$$

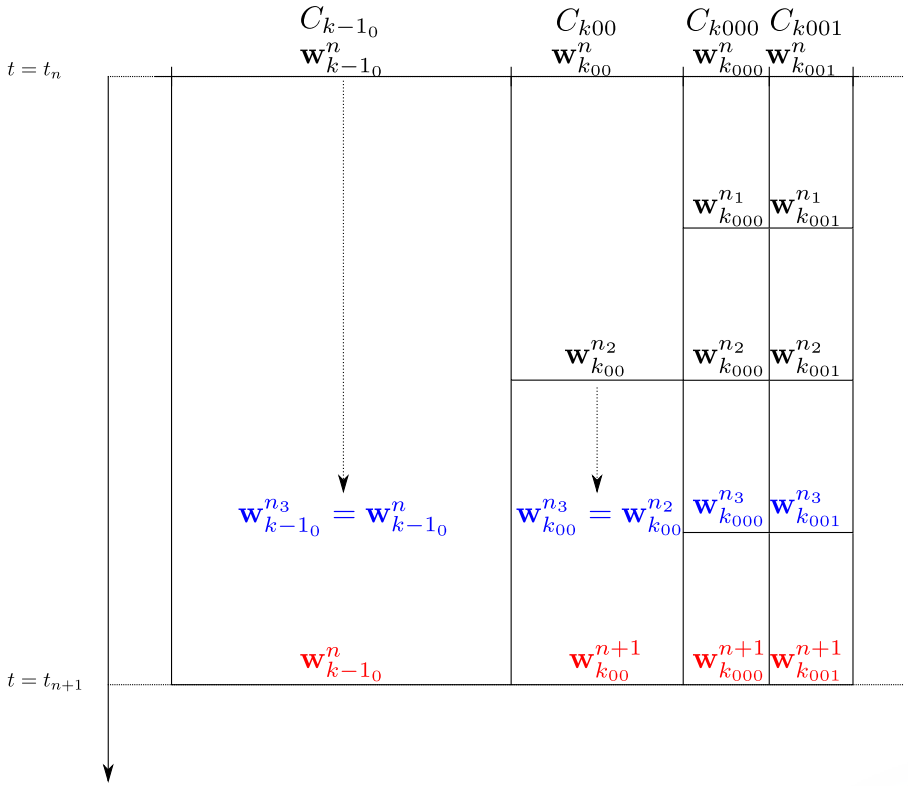
$$\boldsymbol{w}^{n_3}_{k_{001}} = \boldsymbol{w}^{n_2}_{k_{001}} - \frac{\delta t_n}{h_{k_{001}}} \delta \boldsymbol{F}^{n_2}_{k_{000},k_{001},k+1_b}$$

Figure: $t_{n_3} = t_n + 3\delta t_n$

Let us note

$$\delta F^n_{k-1,k,k+1} := \left(\boldsymbol{F}^n_{k+1/2}(\boldsymbol{w}_k, \boldsymbol{w}_{k+1}) - \boldsymbol{F}^n_{k-1/2}(\boldsymbol{w}_{k-1}, \boldsymbol{w}_k)\right)$$

the flux differences and let us consider a first order scheme.



$$\boldsymbol{w}^{n+1}_{k-1_0} = \boldsymbol{w}^{n_3}_{k-1_0} - \frac{\delta t_n}{h_{k-1_0}} \delta \boldsymbol{F}^{n_3}_{k-2_b, k-1_0, k_{00}}$$

$$\boldsymbol{w}^{n+1}_{k_{00}} = \boldsymbol{w}^{n_3}_{k_{00}} - \frac{\delta t_n}{h_{k_{00}}} \delta \boldsymbol{F}^{n_3}_{k-1_0, k_{00}, k_{000}}$$

$$\boldsymbol{w}^{n+1}_{k_{000}} = \boldsymbol{w}^{n_3}_{k_{000}} - \frac{\delta t_n}{h_{k_{000}}} \delta \boldsymbol{F}^{n_3}_{k_{00}, k_{000}, k_{001}}$$

$$\boldsymbol{w}^{n+1}_{k_{001}} = \boldsymbol{w}^{n_3}_{k_{001}} - \frac{\delta t_n}{h_{k_{001}}} \delta \boldsymbol{F}^{n_3}_{k_{000}, k_{001}, k+1_b}$$

Figure: $t_{n+1} = t_n + 4\delta t_n$

Time and space high order extensions can be easily implemented in multi-scale framework :

- Time integration using Adams-Bashforth integration technique. For example, the second order Adams-Bashforth method is :

$$\boldsymbol{w}_k(t_{n+1}) = \boldsymbol{w}_k(t_n) - \frac{\delta t_n}{h_k}\ \delta \boldsymbol{F}_k(t_n) - \frac{\delta t_n^2}{2\delta t_{n-1}\ h_k}\left(\delta \boldsymbol{F}_k(t_n) - \delta \boldsymbol{F}_k(t_{n-1})\right)\ .$$

with

$$S_k^n := \frac{s(\boldsymbol{w}_k(t_{n+1})) - s(\boldsymbol{w}_k(t_n))}{\delta t_n} + \frac{\delta \psi_k(t_n)}{h_k}\ + \frac{\delta t_n}{2\delta t_{n-1}\ h_k}\left(\delta \psi_k(t_n) - \delta \psi_k(t_{n-1})\right)$$

Time and space high order extensions can be easily implemented in multi-scale framework :

- Time integration using Adams-Bashforth integration technique. For example, the second order Adams-Bashforth method is :

$$\boldsymbol{w}_k(t_{n+1}) = \boldsymbol{w}_k(t_n) - \frac{\delta t_n}{h_k}\ \delta \boldsymbol{F}_k(t_n) - \frac{\delta t_n^2}{2\delta t_{n-1}\ h_k}\left(\delta \boldsymbol{F}_k(t_n) - \delta \boldsymbol{F}_k(t_{n-1})\right)\ .$$

with

$$S_k^n := \frac{s(\boldsymbol{w}_k(t_{n+1})) - s(\boldsymbol{w}_k(t_n))}{\delta t_n} + \frac{\delta \psi_k(t_n)}{h_k} + \frac{\delta t_n}{2\delta t_{n-1}\ h_k}\left(\delta \psi_k(t_n) - \delta \psi_k(t_{n-1})\right)$$

- For instance, a second order MUSCL (Monotone Upstream-centered Schemes) reconstruction :

$$\begin{cases} \boldsymbol{w}_{k_{b0}}^n = \boldsymbol{w}_{k_b}^n - \dfrac{h_k}{4}\dfrac{\partial \boldsymbol{w}_{k_b}^n}{\partial x}\ , \\[2ex] \boldsymbol{w}_{k_{b1}}^n = \boldsymbol{w}_{k_b}^n + \dfrac{h_k}{4}\dfrac{\partial \boldsymbol{w}_{k_b}^n}{\partial x}\ . \end{cases}$$

Numerical solutions are computed in the case of the one-dimensional gas dynamics equations for ideal gas :

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \frac{\partial \rho u}{\partial x} &= 0 \\ \frac{\partial \rho u}{\partial t} + \frac{\partial \left(\rho u^2 + p\right)}{\partial x} &= 0 \\ \frac{\partial \rho E}{\partial t} + \frac{\partial \left(\rho E + p\right) u}{\partial x} &= 0 \\ p &= (\gamma - 1)\rho\varepsilon \end{aligned}$$

where

$\rho(t,x)$	:	density
$u(t,x)$	:	velocity
$p(t,x)$	:	pressure
$\gamma := 1.4$	:	ratio of the specific heats
$E(\varepsilon, u)$	:	total energy

where $E := \varepsilon + \dfrac{u^2}{2}$ (where ε is the internal specific energy).

Using the conservative variables $\boldsymbol{w} = (\rho, \rho u, \rho E)^t$, we classically define the convex continuous entropy

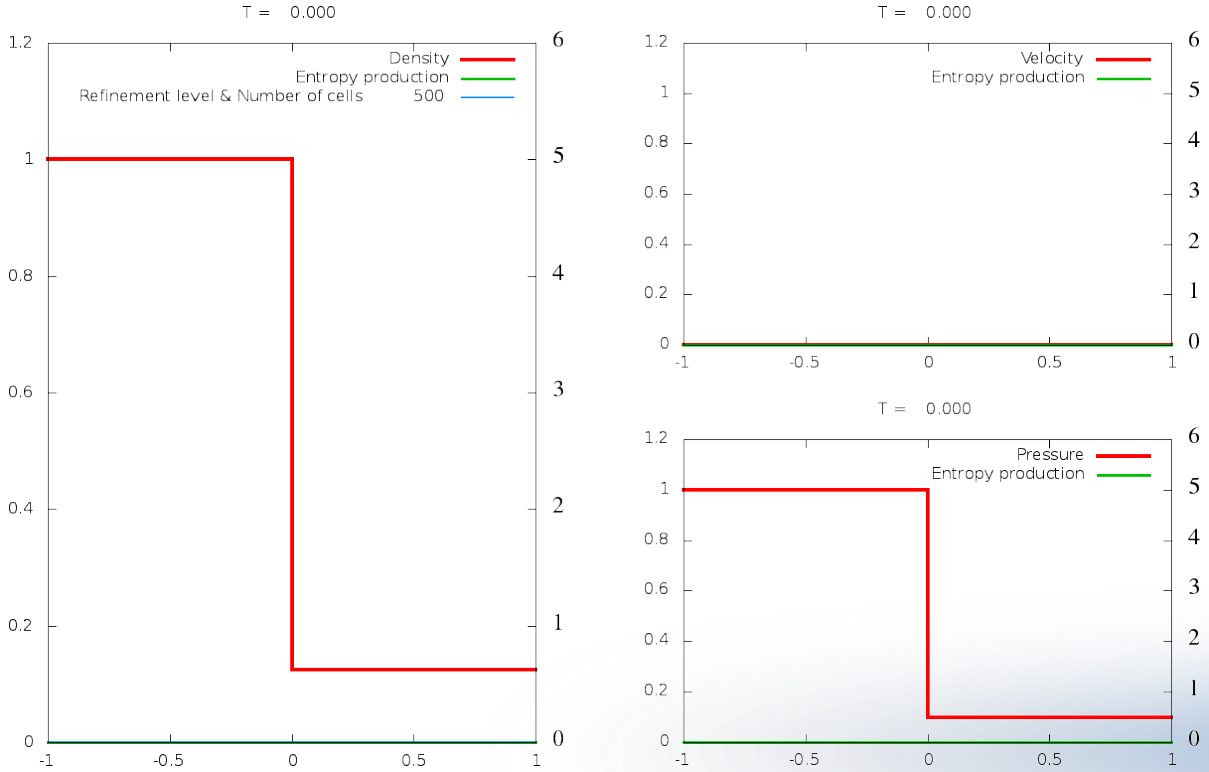
$$s(\boldsymbol{w}) = -\rho \ln\left(\frac{p}{\rho^\gamma}\right) \text{ of flux } \psi(\boldsymbol{w}) = u\, s(\boldsymbol{w}) \ .$$

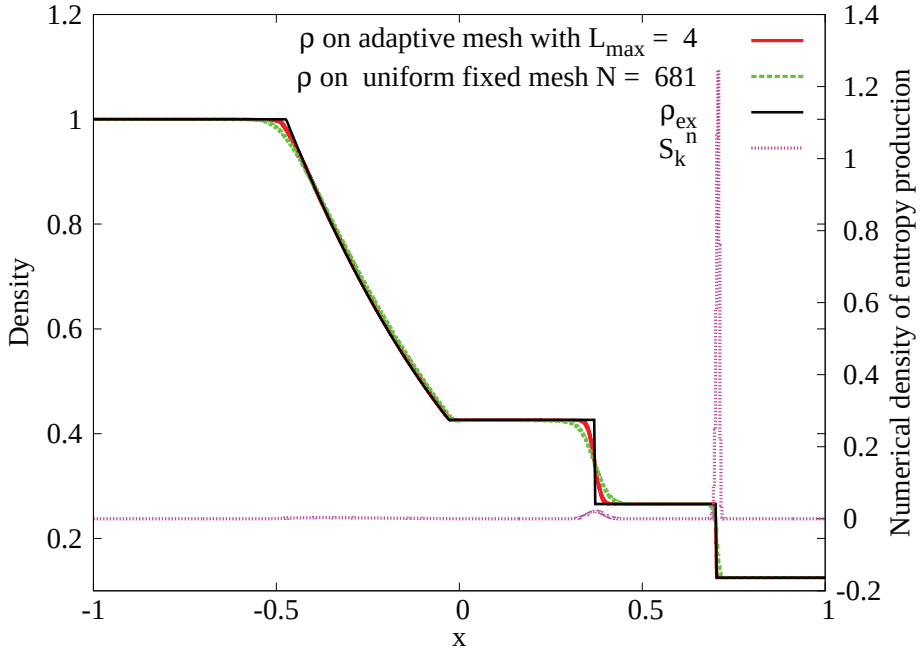
We have used the Godunov solver and displayed $-S$ instead of S. All tests have been performed on *Intel(R) Core(TM) i5-2500 CPU @ 3.30GHz*

Parameters :

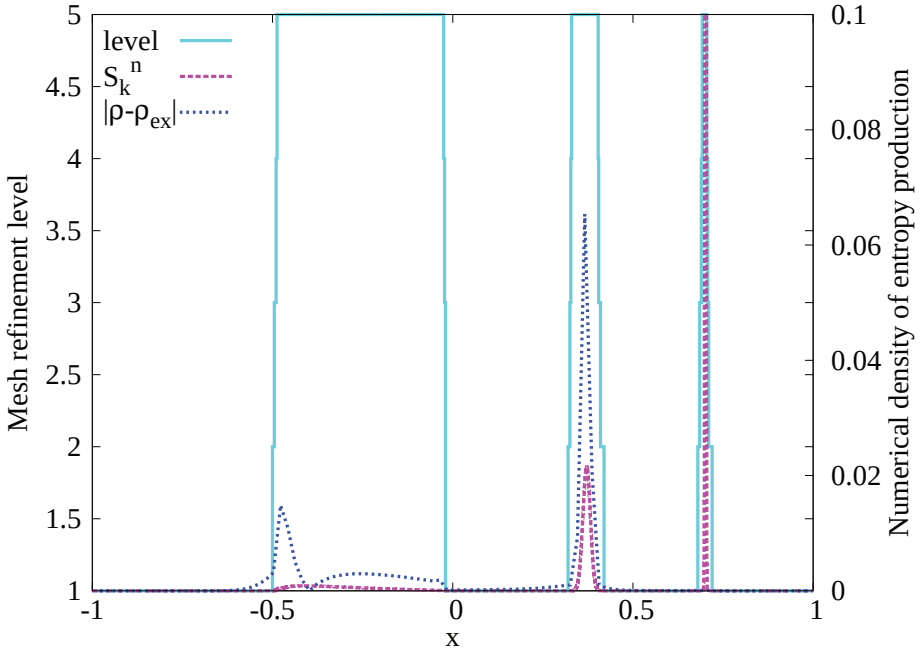
Mesh refinement parameter $\alpha_{\max}$	:	0.01 ,
Mesh coarsening parameter $\alpha_{\min}$	:	0.001 ,
Mesh refinement parameter $\bar{S}$	:	$\frac{1}{\lvert\Omega\rvert}\sum_{k_b} S^n_{k_b}$
CFL	:	0.25,
Simulation time (s)	:	0.4,
Initial number of cells	:	200,
Maximum level of mesh refinement	:	$L_{\max}$.

The initial conditions are :





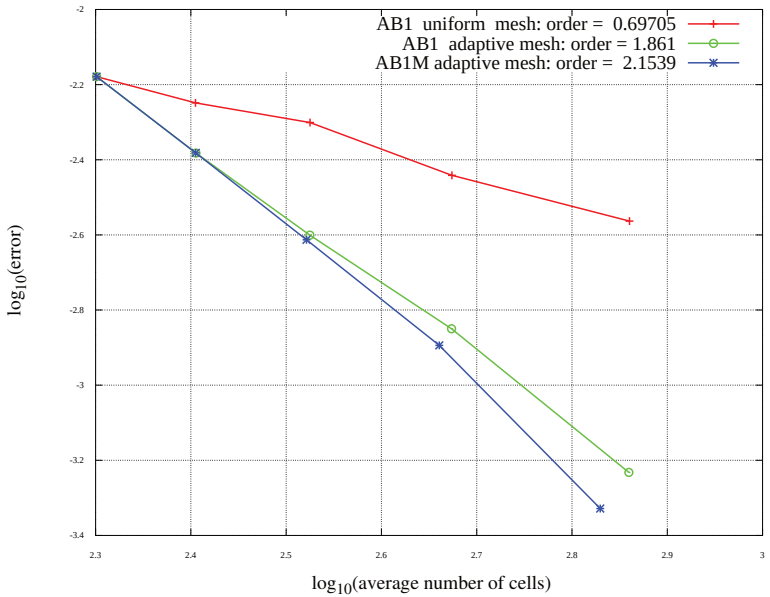
(a) Density and numerical density of entropy production.



(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 $L_{\max} = 5$ and the AB1 scheme on a uniform fixed grid of 681 cells.

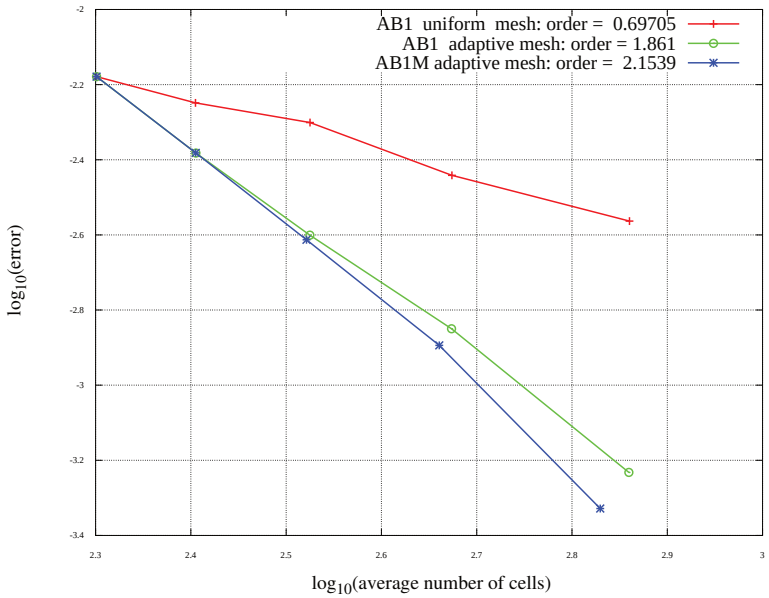
Accuracy& Convergence&CPU time



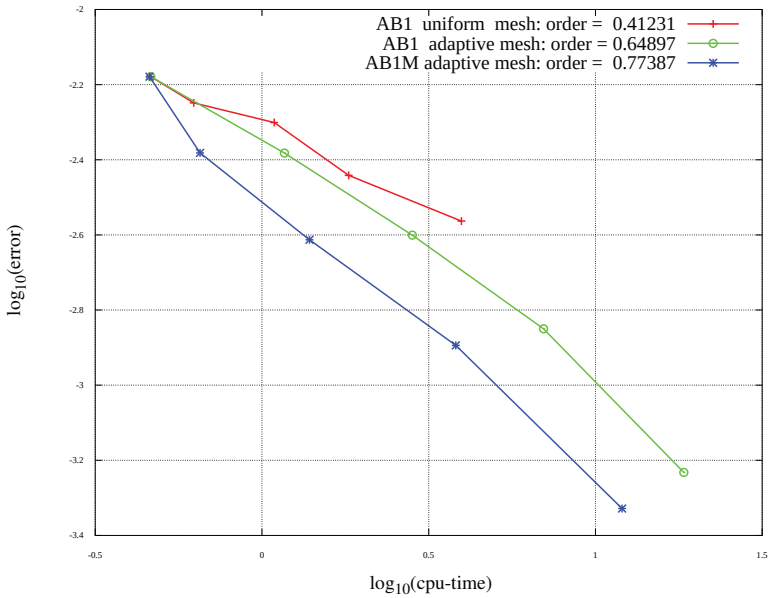
(a) $\|\rho_{ex} - \rho\|_{l^1_t l^1_x}$ with respect to the averaged number of cells for the schemes of order 1.

Figure: Sod's shock tube problem : convergence rate

Accuracy&Convergence& CPU time



(a) $\|\rho_{ex} - \rho\|_{l^1_t l^1_x}$ with respect to the averaged number of cells for the schemes of order 1.



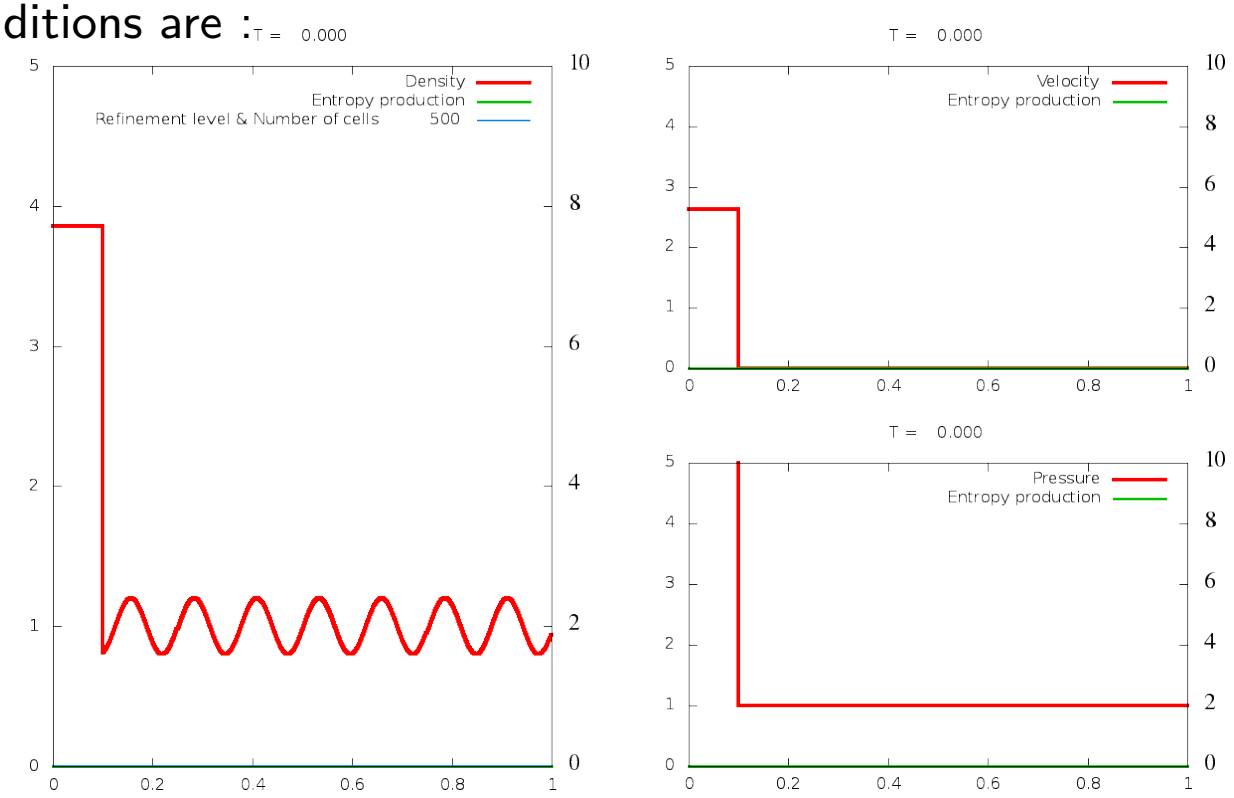
(b) cpu-time–$\|\rho_{ex} - \rho\|_{l^1_t l^1_x}$ with respect to the cpu-time for the schemes of order 1.

Figure: Sod's shock tube problem : convergence rate and cpu-time

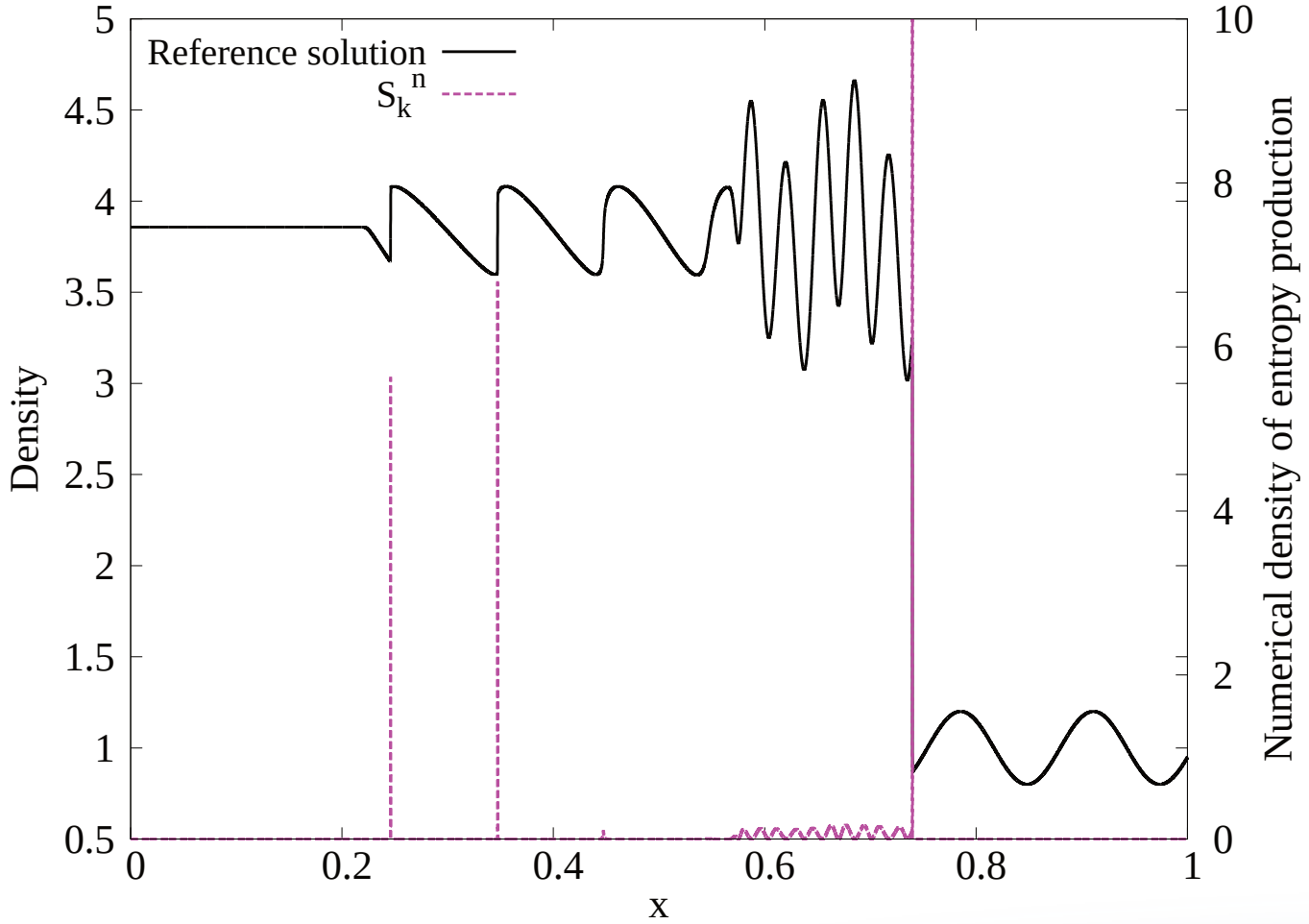
Parameters :

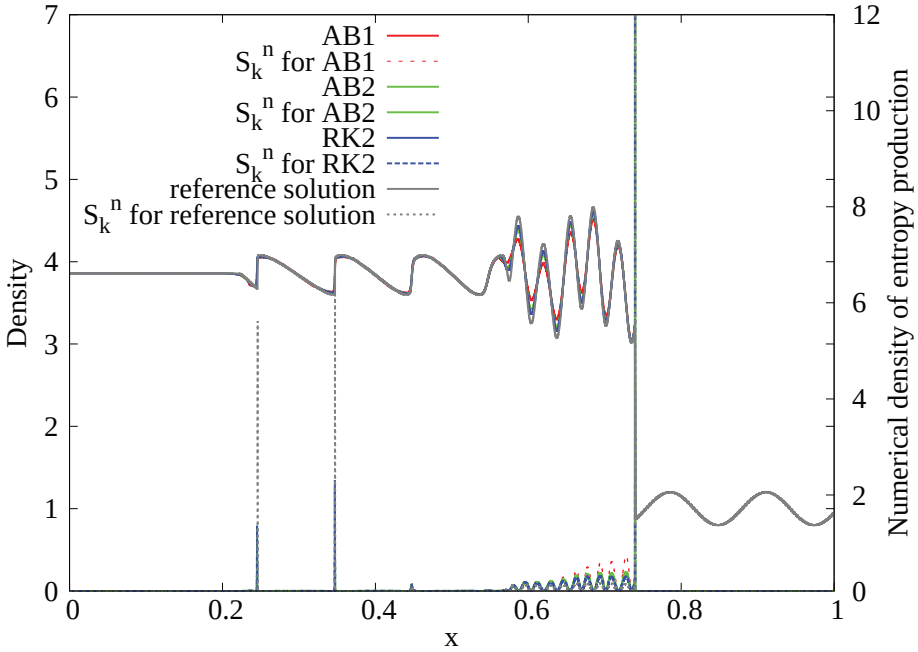
CFL	:	0.219,
Simulation time (s)	:	0.18,
Initial number of cells	:	500,
Maximum level of mesh refinement	:	$L_{\max} = 4$.

The initial conditions are :

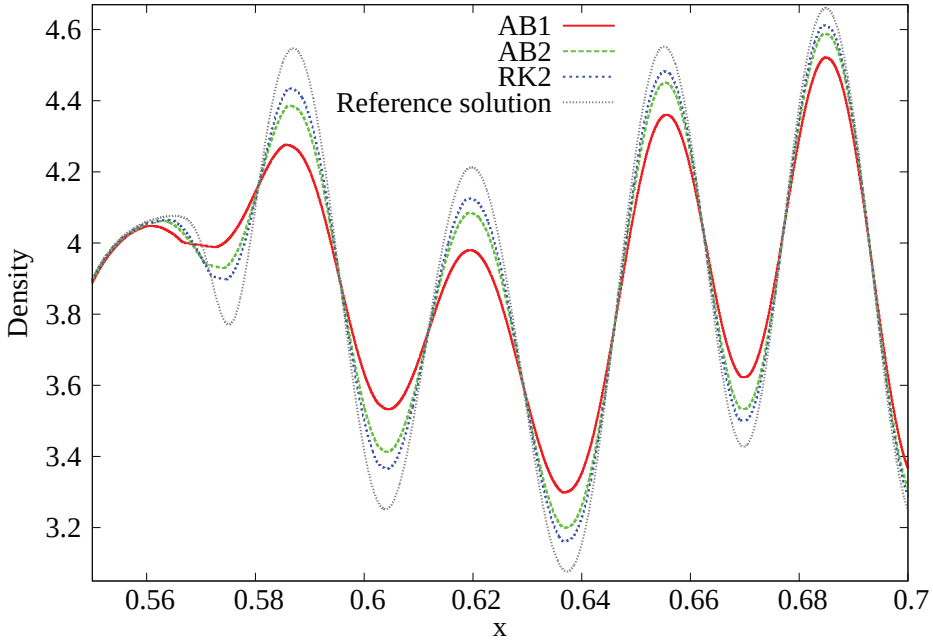


Reference solution computed on a fine grid with 100 000 cells





(a) Density and numerical density of entropy production.



(b) Zoom on oscillating region.

Figure: Shu Osher test case.

	$\mathcal{P}$	$\|\rho - \rho_{ref}\|_{l^1_x}$	cpu-time	$N_{L_{\max}}$	maximum number of cells
AB1	0.288	$4.74\,10^{-2}$	181	1574	2308
AB1M	0.288	$4.80\,10^{-2}$	120	1572	2314
AB2	0.287	$2.75\,10^{-2}$	170	1391	2023
AB2M	0.286	$2.74\,10^{-2}$	108	1357	1994
RK2	0.285	$2.08\,10^{-2}$	299	1375	2005

Table: Comparison of numerical schemes of order 1 and 2

With

- local time stepping (to compute fast)
- the numerical density of entropy production (to pilot locally the mesh)

With

- local time stepping (to compute fast)
- the numerical density of entropy production (to pilot locally the mesh)

we have obtain an efficient adaptive algorithm :

- ✅ accurate
- ✅ reduction of the computational time
- ✅ implementation of first and second order in space and time

With

- local time stepping (to compute fast)
- the numerical density of entropy production (to pilot locally the mesh)

we have obtain an efficient adaptive algorithm :

- ✅ accurate
- ✅ reduction of the computational time
- ✅ implementation of first and second order in space and time

We plan

- 🐢 to improve the efficiency to capture accurately contact discontinuities (artificial entropy)
- 🐢 to extend this work for 2D/3D numerical applications.

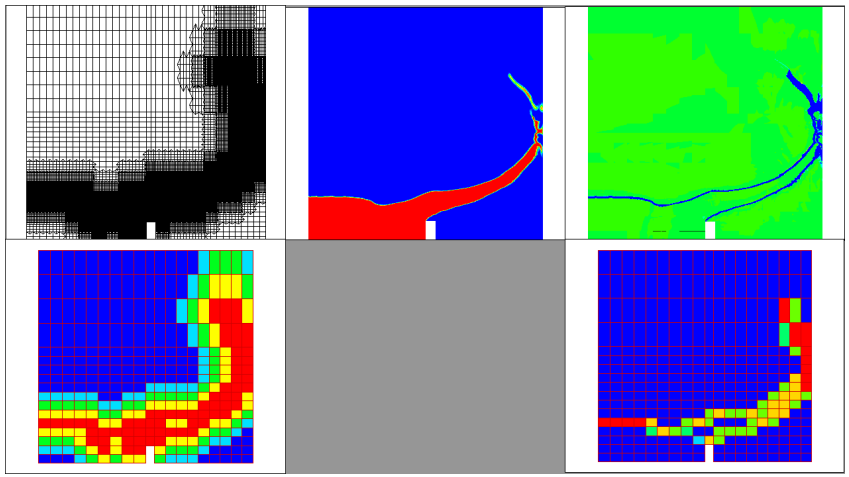


FIGURE: 2D dambreak problem : $\alpha_{\max} = 0.1$, $\alpha_{\max} = 0.2$, $L_{\max} = 5$ using $\bar{S}$ and 321 domains. (top left : mesh, top middle : ρ, top right : S_k^n, bottom left : level, bottom right : $\frac{1}{|D|}\int_D S_k^n$)

Thank you

for your

attention

foreach $i \in \{1, 2^N\}$ **do**

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

foreach *interface* $x_{k+1/2}$ *such that* $\mathcal{L}_{k+1/2} \geqslant N - j$ **do**

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

end

end