Implementation and Testing a Finite Difference Method for the Compressible Euler's Gas Equations

Timothée Schmoderer

28 août 2017

Contents

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES ROUEN

University of Cologne

1 Introduction

In this work we are interested in solving numerically the one dimensional Euler's gas equations :

$$\frac{\partial}{\partial t}\begin{pmatrix}\rho\\ \rho u\\ E\end{pmatrix}+\frac{\partial}{\partial x}\begin{pmatrix}\rho u\\ P+\rho u^2\\ u(E+P)\end{pmatrix}=0 \qquad \forall x\in\Omega,\ \forall t\geq 0 \tag{1}$$

Plus boundary conditions for : $\rho,\ u,$ and P

The system is completed with the following equation of state :

$$E=\frac{P}{\gamma-1}+\frac{\rho u^2}{2}$$

We note $U=\begin{pmatrix}\rho\\ \rho u\\ E\end{pmatrix}$ the state vector and $f(U)=\begin{pmatrix}\rho u\\ \rho u^2+P\\ u(E+P)\end{pmatrix}$ the flux vector. Such as the system (1) is rewritten :

$$\frac{\partial U}{\partial t}+\frac{\partial f(U)}{\partial x}=0$$

Here are the possible boundary conditions we will meet (assume $\Omega=[a,b]$) :

1. **Periodic :** What append at one side of the domain is reported at the other side :

$$\Phi(a,t)=\Phi(b,t) \qquad \Phi=u,\ \rho \text{ and } P \tag{2}$$

2. **Solid wall :** The fluid will behave like there is two indestructible wall at the two sides of the domain :

$$\begin{aligned} u(a,t)=u(b,t)&=0\\ \frac{\partial\Phi}{\partial x}(a,t)=\frac{\partial\Phi}{\partial x}(b,t)&=0 \qquad \Phi=\rho \text{ and } P\end{aligned} \tag{3}$$

3. **Inlet :** this condition prescribe the value of the flow at the boundary x_B, like there is a source of fluid :

$$\Phi(x_B,t)=\Phi_{in}(t) \qquad \Phi=u,\ \rho \text{ and } P$$

4. **Outflow :** All the fluids that reaches the boundary x_B is evacuate outside :

$$\frac{\partial\Phi}{\partial x}(x_B,t)=0 \qquad \Phi=u,\ \rho \text{ and } P$$

The goal is to implement the scheme describe in [1], check that the convergence rate is 2 and to test the scheme on different cases. The difficulty in solving this equations is that even when the initials conditions are smooth, shock (discontinuities) could occurs in the solution. Shocks are characterised numerically with an infinite gradient which computer doesn't like. All the problem is to find methods that capture shocks.

2 Implementation

2.1 Cells and Nodes

Choose a non-zero integer N, and divide Ω into N equals grid "cells" (Figure 1), let x_i be the cell-centers and Δx be the cell's width.

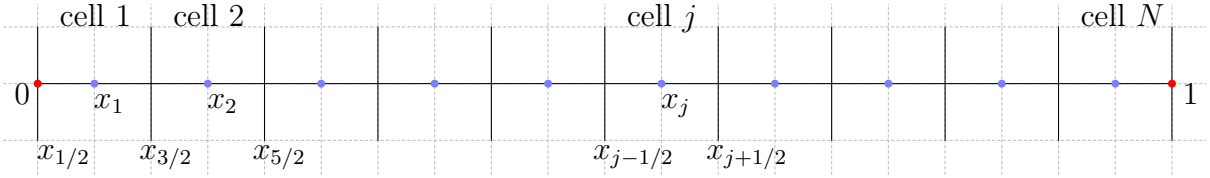


FIGURE 1 – Cells construction on Ω

We denote by $x_{j\pm1/2}$ the cells interfaces : $x_j \pm \dfrac{\Delta x}{2}$. And we note U_j the approximation of $U(x_j, t)$.

2.2 Algorithm

The goal is to get the following equations :

$$\frac{dU_j}{dt} = -\frac{\hat{f}_{j+1/2} - \hat{f}_{j-1/2}}{\Delta x} \qquad j = 1, \dots, N \tag{4}$$

From this form we can understand the general ingredient of the method. Providing the value of U_j for a certain time t we can then compute the fluxes at the cells interfaces. Then the time-variation of the state is equal to the difference of fluxes (it seems very logical, isn't it ?). Because we know the sate only in the center of the cell, the main difficulty is to get a good approximation of the flux value at the cells interfaces.

As proposed in [1], we compute the $\{\hat{f}_{j\pm1/2}\}$ as follows :

1. Compute the spectral radius of the Jacobian at $x = x_j$. Analytic computation (see [2] for details) let us know that for compressible Euler equation this is :

$$a_j = |u_j| + c_j$$

2. We split the flux function f as :

$$f(U_j) = f_j^+ + f_j^- \qquad f_j^\pm = \frac{1}{2}\left(f(U_j) \pm a_j U_j\right)$$

3. We compute the slopes in each cells using the minmod function :

$$(f_x)_j^\pm = minmod\left(\theta\frac{f_j^\pm - f_{j-1}^\pm}{\Delta x}, \frac{f_{j+1}^\pm - f_{j-1}^\pm}{2\Delta x}, \theta\frac{f_{j+1}^\pm - f_j^\pm}{\Delta x}\right)$$

Where $\theta \in [1, 2]$ is a parameter that control the amount of numerical dissipation, it will be taken at 1.5.
The *minmod* function is defined by :

$$minmod(a, b, c) = \begin{cases} \min(a, b, c) & \text{if } a > 0,\ b > 0 \text{ and } c > 0 \\ \max(a, b, c) & \text{if } a < 0,\ b < 0 \text{ and } c < 0 \\ 0 & \text{else} \end{cases}$$

4. Construct f^E and f^W as :

$$f_j^E = f_j^+ + \frac{\Delta x}{2}(f_x)_j^+ \qquad f_j^W = f_j^- - \frac{\Delta x}{2}(f_x)_j^-$$

5. Finally :

$$\hat{f}_{j+1/2} = f_j^E + f_{j+1}^W \quad \hat{f}_{j-1/2} = f_{j-1}^E + f_j^W$$

The interesting part here is the *minmod* function. This function is the shock capturing procedure. At step *2*, the split fluxes have the following property :

$$\frac{\partial f^+}{\partial x} \geq 0 \qquad \frac{\partial f^-}{\partial x} \leq 0$$

Hence the *minmod* function is a switch on this property. If it doesn't hold, it means we are in a sensible region, better to set the slopes to 0 to keep a constant flux, else we take the closest value from zeros in order to avoid the infinite gradient.

2.3 Ghost Nodes

While presenting the scheme I haven't discuss on which nodes we have to apply the method. Obviously there is no problem when we are in the middle of the domain, problems arise when we come close to the boundaries. In order to achieve the method, two ghosts point are needed on each side of the domain, with value induced by the boundary conditions. Why two ? Remember that we want values at each cell's interfaces. Then for instance to get $\hat{f}_{1-1/2}$ we will need f_0^E so at this point we need one ghost cell left. Moreover to get f_0^E we will need to get the slopes for $j = 0$ then we might need $f_{-1}^{\pm}$ in the *minmod* function. So one more ghost cell.

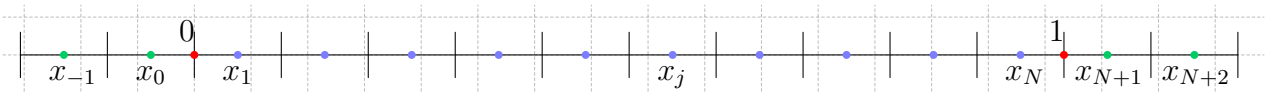


FIGURE 2 – Ω with ghost points

The whole method can be summarize in this array (red are ghost values, and dot show for

what j we can compute the value) :

cell	-1	0	1	2	$\dots$	j	$\dots$	$N-1$	N	$N+1$	$N+2$
a_j	·	·	·	·	·	·	·	·	·	·	·
$f^{\pm}$	·	·	·	·	·	·	·	·	·	·	·
$f^{\pm}_j - f^{\pm}_{j-1}$		·	·	·	·	·	·	·	·	·	
$f^{\pm}_{j+1} - f^{\pm}_{j-1}$		·	·	·	·	·	·	·	·	·	
$f^{\pm}_{j+1} - f^{\pm}_j$		·	·	·	·	·	·	·	·	·	
$(f_x)^{\pm}$		·	·	·	·	·	·	·	·	·	
f^E		·	·	·	·	·	·	·	·	·	
f^W		·	·	·	·	·	·	·	·	·	
$\hat{f}_{j+1/2}$			·	·	·	·	·	·	·		
$\hat{f}_{j-1/2}$			·	·	·	·	·	·	·		
$\frac{dU_j}{dt}$			·	·	·	·	·	·	·		

2.4 Boundary condition

The boundary conditions are reflected in the value set in the ghost cells. I will describe how *wall* and *periodic* conditions are set, because, *inlet* conditions cause no problem and *outflow* is easily deduce from the wall conditions.

2.4.1 Periodic conditions

The periodic conditions describe the gas as if it is moving in a circle. As we want the value at $x = 0$ and $x = 1$ to coincide, the following figure (3) allow us to understand what happen.

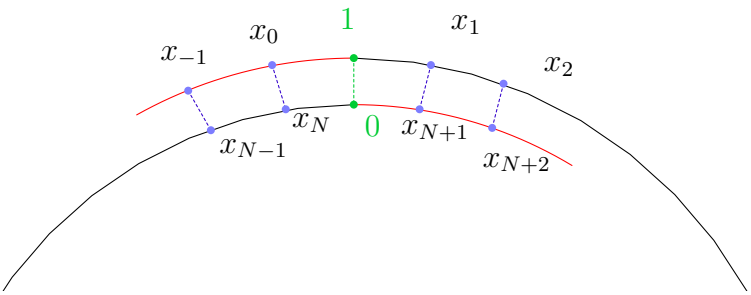


FIGURE 3 – Ω seen with periodic boundary conditions

Then it is easy to deduce the conditions :

$$\begin{array}{llllll} \rho_0 & = \rho_N & u_0 & = u_N & P_0 & = P_N \\ \rho_{-1} & = \rho_{N-1} & u_{-1} & = u_{N-1} & P_{-1} & = P_{N-1} \end{array}$$

That is to say for the state vector :

$$U_0 = U_N \qquad U_{-1} = U_{N-1}$$

2.4.2 Wall conditions

From equations 3 we deduce the value in the value of ρ, u and P at the two ghost cells centers (I only describe them for left boundary the right case is the same.) :

$$\begin{array}{llllll} \rho_0 & = \rho_1 & u_0 & = -u_1 & P_0 & = P_1 \\ \rho_{-1} & = \rho_2 & u_{-1} & = -u_2 & P_{-1} & = P_2 \end{array}$$

Which in term of U is the following :

$$U_0 = \begin{pmatrix} U_1^1 \\ -U_1^2 \\ U_1^3 \end{pmatrix} \qquad U_{-1} = \begin{pmatrix} U_2^1 \\ -U_2^2 \\ U_2^3 \end{pmatrix}$$

where the upperscript is the denotes the i-th component.

2.5 Integration of the semi-discrete system

Once we get the semi discrete scheme (4) we have to numerically integrate it. Based on [3, 4, 5] and as suggests by the author of the method I implemented the following 3rd order SSP Runge-Kutta method :

$$\begin{aligned} u^{(1)} &= u^n + \Delta t L(u^n) \\ u^{(2)} &= \frac{3}{4}u^n + \frac{1}{4}u^{(1)} + \frac{1}{4}\Delta t L(u^{(1)}) \\ u^{n+1} &= \frac{1}{3}u^n + \frac{2}{3}u^{(2)} + \frac{2}{3}\Delta t L(u^{(2)}) \end{aligned} \tag{5}$$

Where the L operator is the one that gives us the second member.

Courant's number : In this work the best Courant's number we can choose is 1. Thus we get the bigger time step we could expect :

$$\Delta t = \frac{\Delta x}{\max_{x\in\Omega} a}$$

3 Numerical Experiment for regular nodes distribution

In this section we will show by experimentation that the method is of order 2 as announced and demonstrate the efficiency of the method on several typical example. All computations are done with 1000 nodes. Computation done with higher order method could be found in [6], thus we can convince ourselves that the method is working well.

3.1 Error and convergence

Because we don't know the analytic solution we adopt the *Manufactured Solution* strategy to check the convergence rate of the method :

1. We choose $q(x,t) \in \mathbb{R}^3$ as smooth as possible to be the solution of equations (1).
2. We put q in these equations and get, in general, a non zeros result (otherwise, it will mean we have an analytical solution, which is in general not possible). Let's call the rest $\mathcal{S}(x,t)$:
$$\frac{\partial q}{\partial t} + \frac{\partial f(q)}{\partial x} = \mathcal{S}$$
3. We now consider the modified problem :

$$\begin{gathered} \frac{\partial U}{\partial t} + \frac{\partial f(U)}{\partial x} = \mathcal{S} \\ U(x,0) = q(x,0) \\ \text{Plus boundary conditions to be discusses bellow} \end{gathered} \tag{6}$$

   For which we know the solution to be q.
4. We apply our method to this problem. It requires minor modification in the code to add the source term.
5. We can now compare the numerical solution to the analytical one. And compute the convergence rate.

A common choice in this approach is to choose periodic boundary conditions.

3.1.1 Case 1

This case could be found in [7] (example 3.3).
For this first case we want the solution to be :

$$\begin{aligned} \rho(x,t) &= 1 + 0.2\sin(2\pi(x-t)) \\ u(x,t) &= 1 \\ P(x,t) &= 1 \end{aligned}$$

Then a little computation using the equation of state gives the solution for the energy :

$$E(x,t) = \frac{\gamma+1}{2(\gamma-1)} + 0.1\sin(2\pi(x-t))$$

Hence the initial conditions are the following :

$$\begin{aligned} \rho(x,0) &= 1 + 0.2\sin(2\pi x) \quad \forall x \in \Omega \\ u(x,0) &= 1 \quad \forall x \in \Omega \\ P(x,0) &= 1 \quad \forall x \in \Omega \end{aligned} \tag{7}$$

In this particular case, the source term is identically zero :

$$\mathcal{S}(x,t) = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

We make computation for 100, 200, 300, 400, 600, 800, 1200, 1600, 2400, 3200, 4000 6400, 9600, 12800, 19200, 25600, 38400, 51200 nodes and up to 1000 iterations. Hence we are able to compare the exact solution for the density and the numerical approximation. Figures 4 and 5 shows the log of the error against the log of the number of nodes.

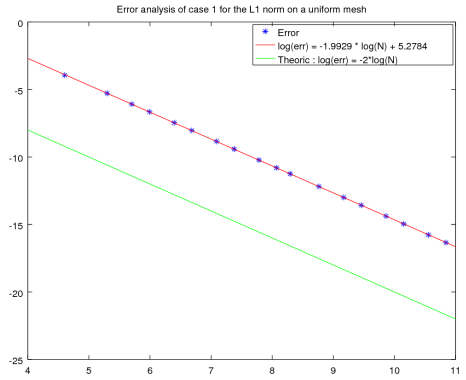


FIGURE 4 – Convergence rate for manufactured solution # 1 in L^1 norm

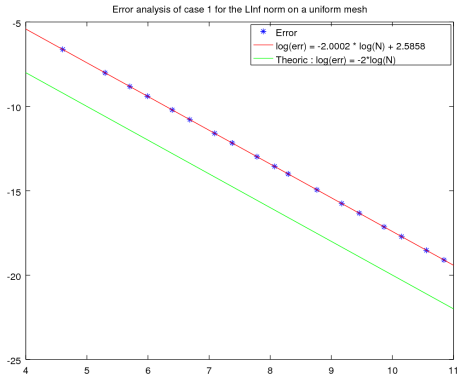


FIGURE 5 – Convergence rate for manufactured solution # 1 in L^∞ norm

Where the L^1 and L^∞ norms are the following :

$$\|U^{ex} - U^a\|_1 = \sum_{j=1}^{N} |U^{ex}(x_j) - U_j^a| \qquad \|U^{ex} - U^a\|_\infty = \max_{j=1..N} |U^{ex}(x_j) - U_j^a|$$

The $\log error$ perfectly fit a -2 slope line. Hence as we hoped, the method seems to be of order 2.

3.1.2 Case 2

The idea behind this case comes mainly from [8]
The second case will comfort us in the fact tat the method is of order 2. We want the solution to be :

$$\begin{aligned} \rho(x,t) &= 2 + 0.1\sin(2\pi(x-t)) \\ u(x,t) &= 1 \\ E(x,t) &= 2 + 0.1\cos(2\pi(x-t)) \end{aligned}$$

Then, the initial conditions are the following :

$$\begin{aligned} u(x,0) &= 1 \quad \forall x \in \Omega \\ \rho(x,0) &= 2 + 0.1\sin(2\pi x) \quad \forall x \in \Omega \\ P(x,0) &= \frac{\gamma - 1}{20}\left(20 + 2\cos(2\pi x) - sin(2\pi x)\right) \quad \forall x \in \Omega \end{aligned} \tag{8}$$

And the source term is given by :

$$\mathcal{S}(x,t) = (1-\gamma)\pi(2\rho(x,t) + E(x,t) - 6)\begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} \tag{9}$$

We again make computation for 100, 200, 300, 400, 600, 800, 1200, 1600, 2400, 3200, 4000 6400, 9600, 12800, 19200, 25600, 38400, 51200 nodes with step of 100 nodes up to 1000 iterations. Hence we are able to compare the exact solution and the numerical approximation. Figures 6 and 7 shows the log of the error against the log of the number of nodes.

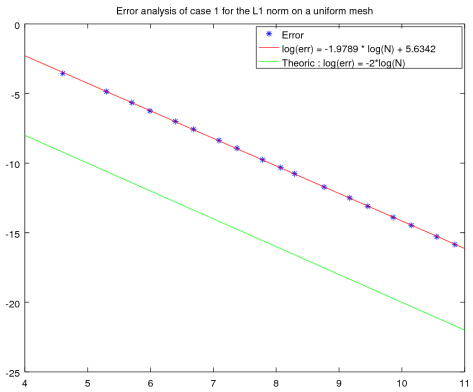


FIGURE 6 – Convergence rate for manufactured solution # 2 in L^1 norm

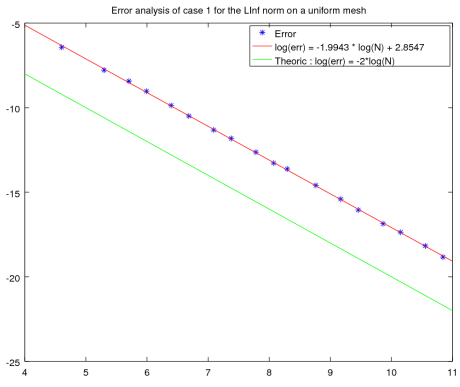


FIGURE 7 – Convergence rate for manufactured solution # 2 in L^∞ norm

Again the convergence rate is really near 2. We can conclude that the method it of order 2 as announced.

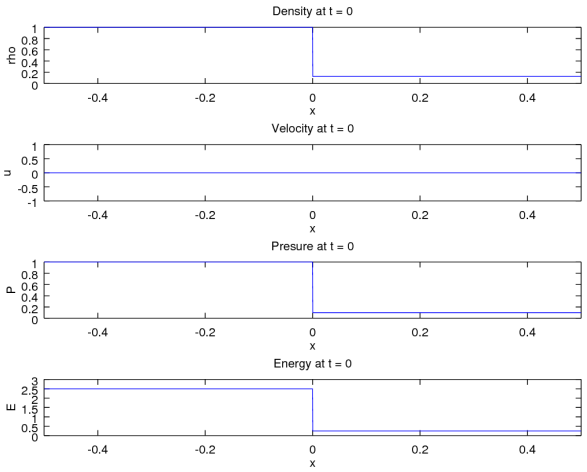
Having satisfying ourselves with the convergence rate we can now observe the method working on several typical example.

3.2 Sod shock tube

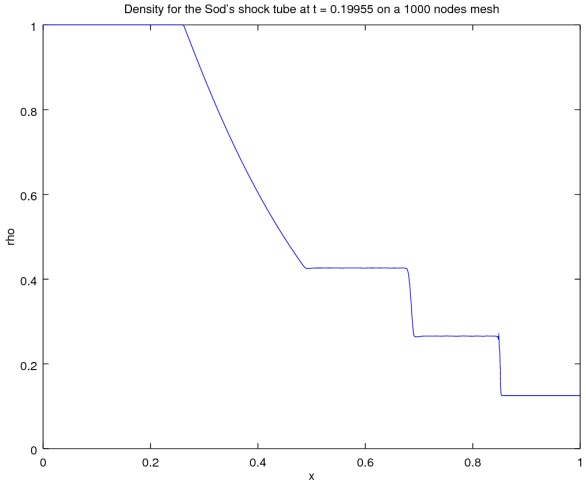
We begin our collection of test with a popular one. The domain is $\Omega = [0, 1]$, and the initial conditions are :

$$
\begin{aligned}
u(x,0) &= 0 \qquad \forall x \in \Omega \\
\rho(x,0) &= \begin{cases} 1.0 & x \in [0, 0.5] \\ 0.125 & x \in [0.5, 1] \end{cases} \\
P(x,0) &= \begin{cases} 1.0 & x \in [0, 0.5] \\ 0.1 & x \in [0.5, 1] \end{cases}
\end{aligned}
\tag{10}
$$

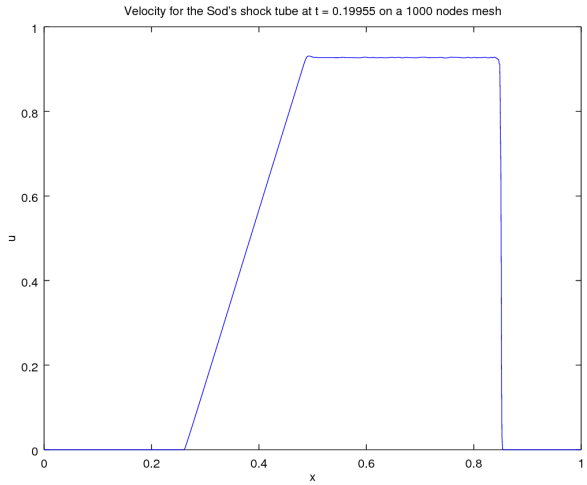
On this we apply wall boundary conditions and look at the result around $t = 0.2$.



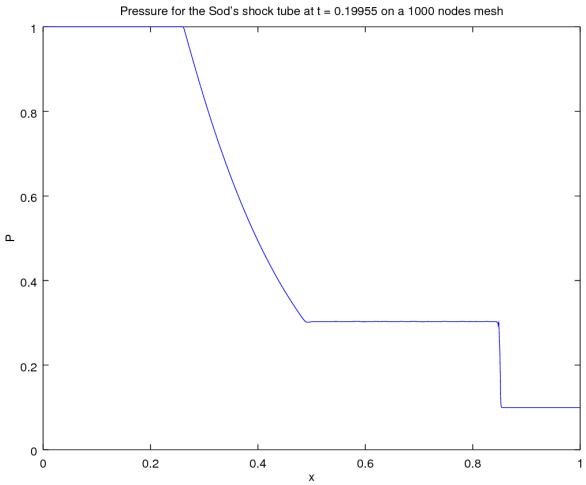
(a) Initial conditions



(b) Density



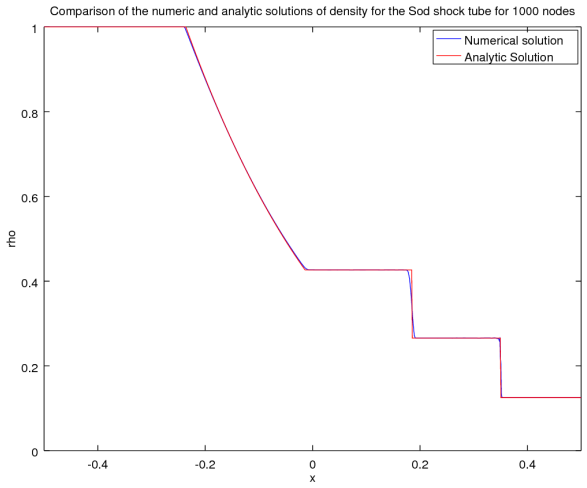
(c) Velocity



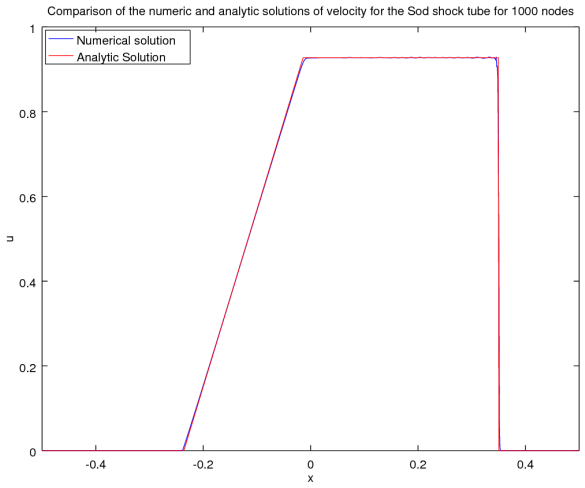
(d) Pressure

FIGURE 8 – Result for the Sod's shock tube

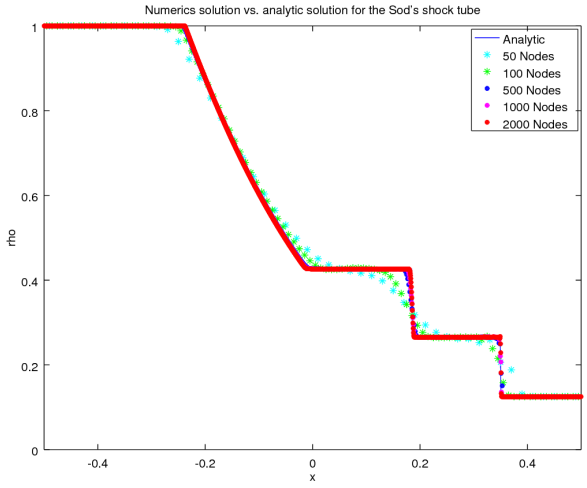
Based on [9] (and the code in [10]) we can compute the analytic solution for this case. On the figure bellow we can see the difference between the analytic solution and the approximation we computed :



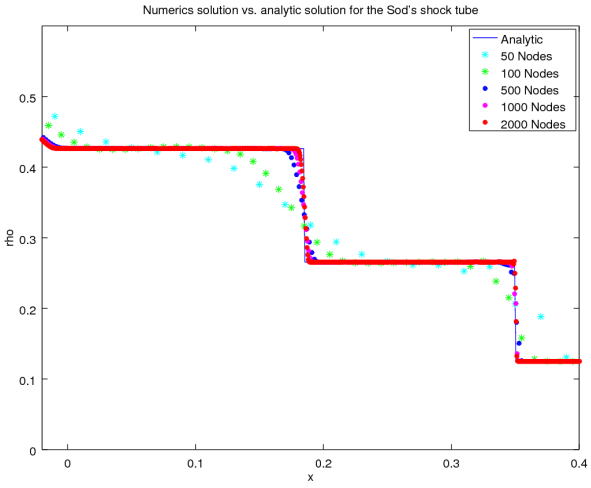
(a) Density



(b) Velocity



(c) Multi nodes



(d) Multi nodes zoom

FIGURE 9 – Result for the Sod's shock tube with the analytic solution and for different mesh size

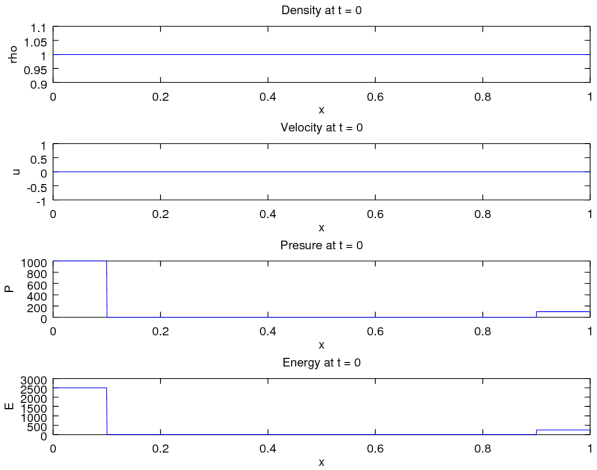
As we can see the method is not perfect. However the results are quiet really good and close to the analytic solution. We see on figures 9-(c) and 9-(d) that the more there is nodes, the closest is the numerical result to the analytic solution. This valid our intuition that the method converged and due to Lax's theorem we know that this equation is a weak-solution of the initials PDE's problem.

3.3 Interacting blast wave

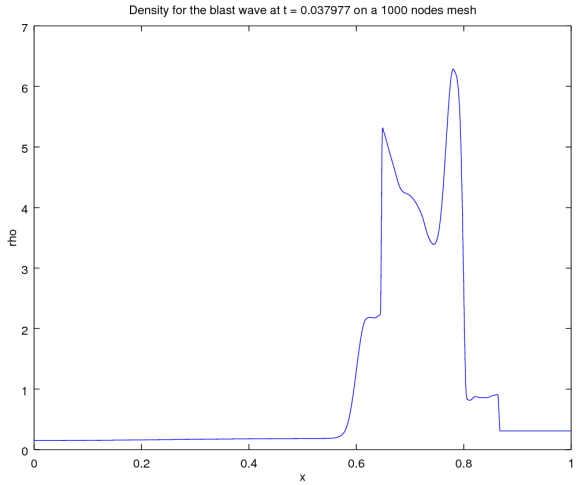
See [11] and [7] (example 3.7). Initial conditions :

$$\begin{aligned} u(x,0) &= 0 \qquad \forall x \in \Omega \\ \rho(x,0) &= 1 \qquad \forall x \in \Omega \\ P(x,0) &= \begin{cases} 1000 & x \in [0, 0.1] \\ 0.01 & x \in [0.1, 0.9] \\ 100 & x \in [0.9, 1] \end{cases} \end{aligned} \tag{11}$$

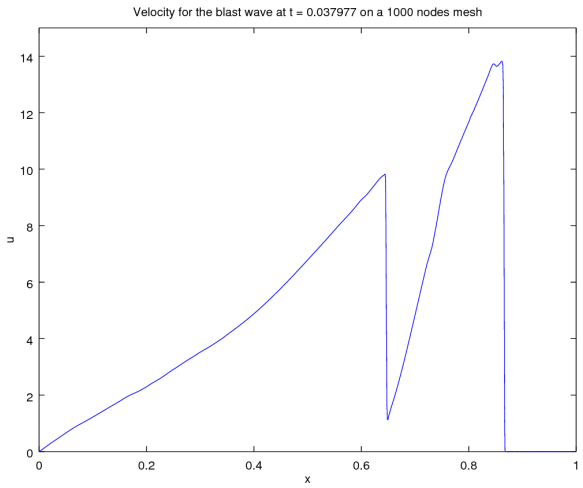
We use again solid wall boundary conditions and watch the result around $t = 0.038$.



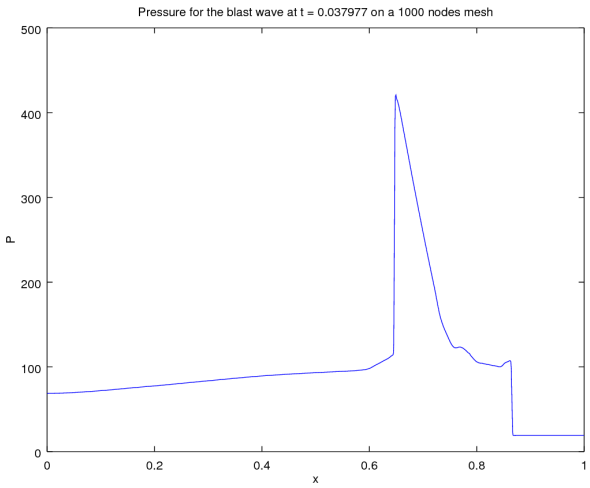
(a) Initial conditions



(b) Density



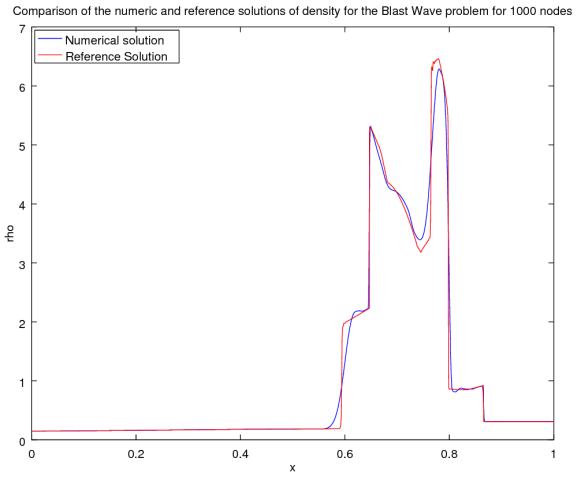
(c) Velocity



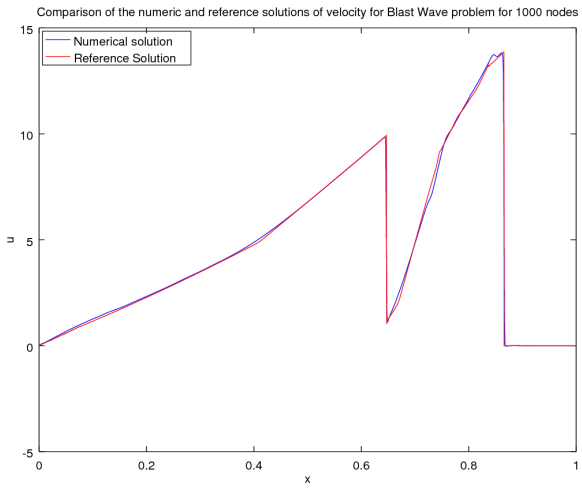
(d) Pressure

FIGURE 10 – Result for the blast wave

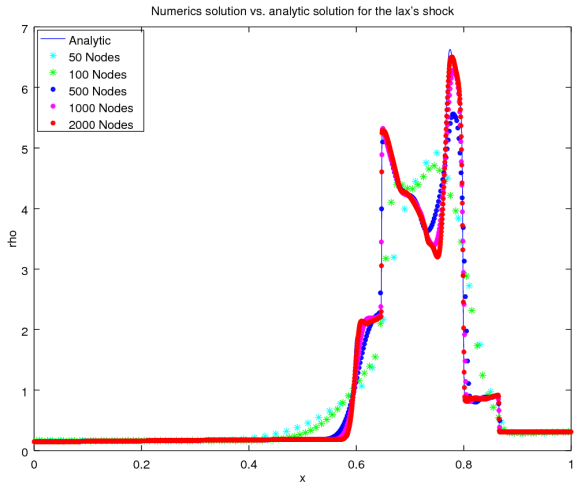
Unfortunately there is no analytic solution for this case. Hence, in order to observe the convergence of the method I used a WENO-5 method, from [12], to compute a reference solution with 16000 nodes.



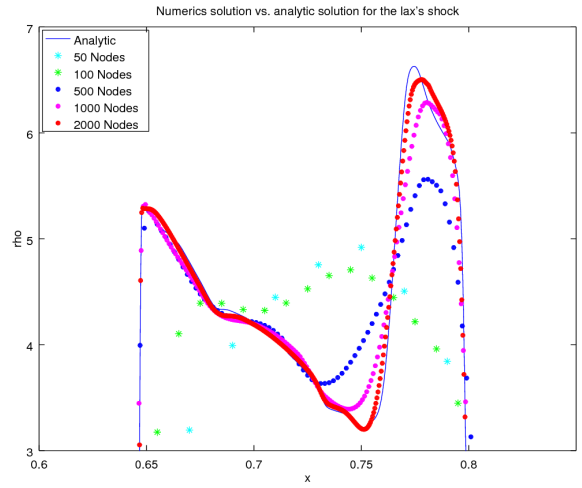
(a) Density



(b) Velocity



(c) Multi nodes



(d) Multi nodes zoom

FIGURE 11 – Result for the blast wave problem with the reference solution and for different mesh size

We can observe that the solution computed is close from the reference one. But we got a lot more errors that in the previous case. This is probably due to the complexity of the case and the numerical pollution from Matlab execution. However we get the general forms of the solution and the shocks are well disposed so the made did not compute all wrong.

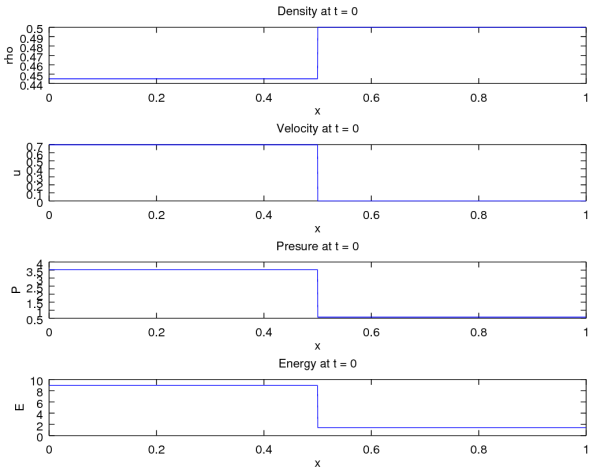
3.4 Lax's shock tube

See [7] (example 3.5). The initial conditions are :

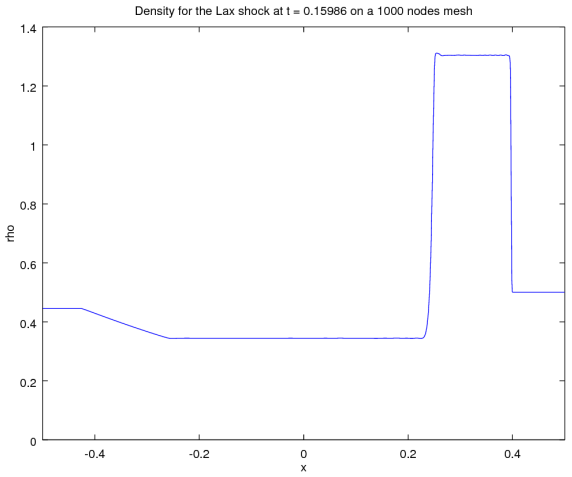
$$u(x,0) = \begin{cases} 0.698 & x \in [0, 0.5] \\ 0 & x \in [0.5, 1] \end{cases}$$
$$\rho(x,0) = \begin{cases} 0.445 & x \in [0, 0.5] \\ 0.5 & x \in [0.5, 1] \end{cases} \quad (12)$$
$$P(x,0) = \begin{cases} 3.528 & x \in [0, 0.5] \\ 0.571 & x \in [0.5, 1] \end{cases}$$

We use inlet conditions on the left and solid wall conditions in the right boundary and watch the result around $t = 0.16$. Inlet conditions are :

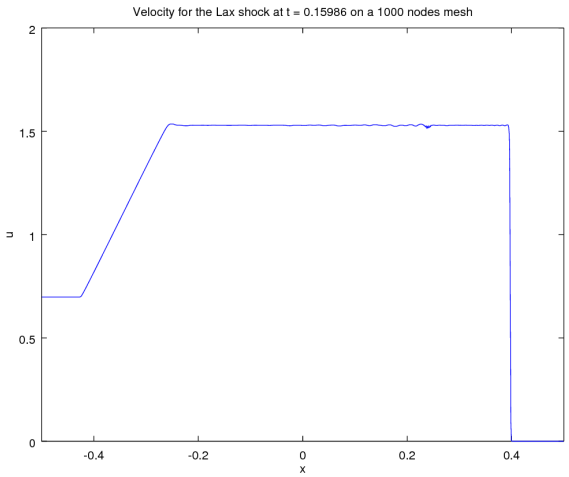
$$u(0,t) = 0.698 \quad \rho(0,t) = 0.445 \quad P(0,t) = 3.528$$



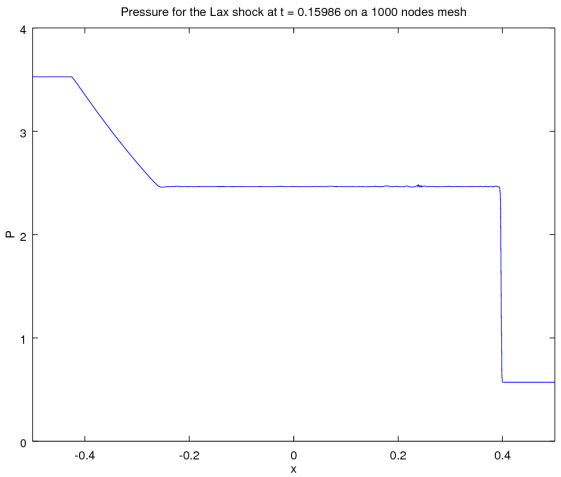
(a) Initial conditions



(b) Density



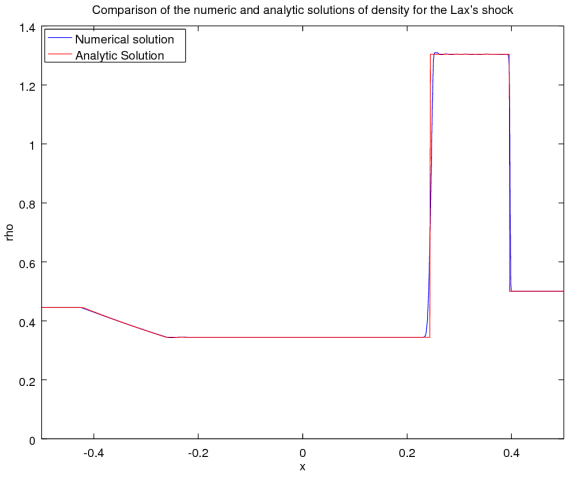
(c) Velocity



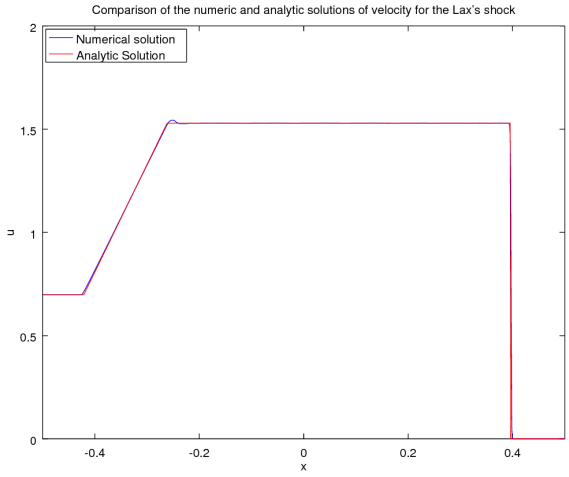
(d) Pressure

FIGURE 12 – Result for the Lax's shock tube

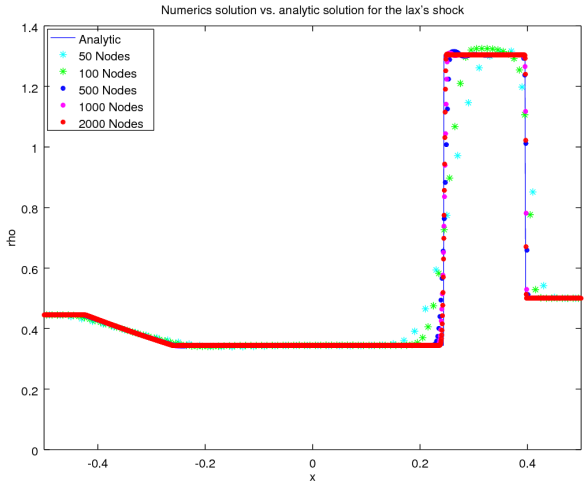
For this case, as for the Sod's shock tube, there exists an analytic solutions. Thus we can compare our method to this solution.



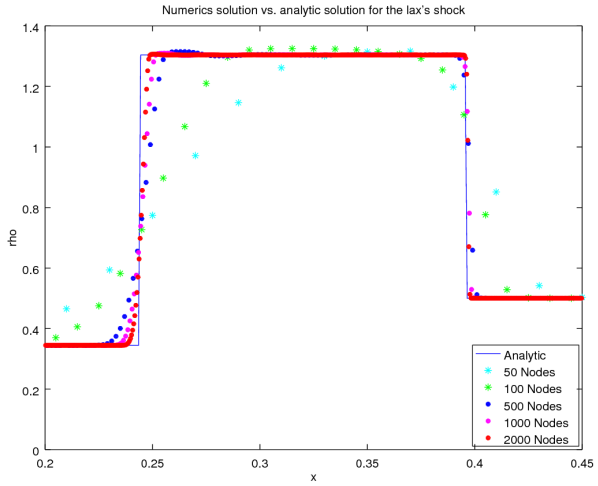
(a) Density



(b) Velocity



(c) Multi nodes



(d) Multi nodes zoom

FIGURE 13 – Result for the Lax's shock tube with the analytic solution and for different mesh size

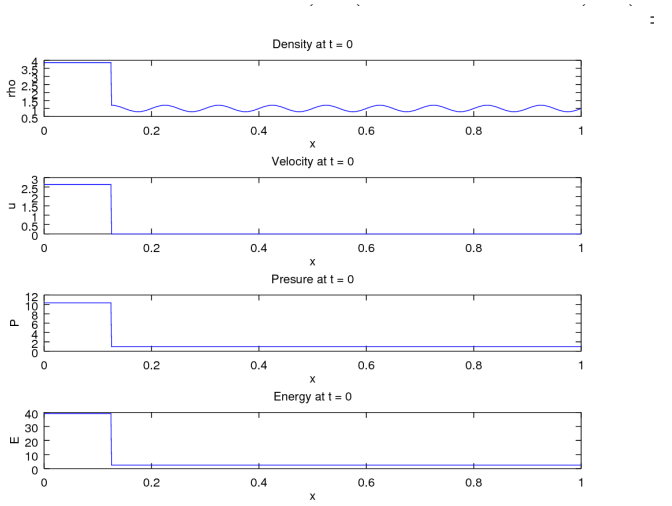
Again, we observe that the method fit well the analytic solution and captured the shocks.

3.5 Shu-Osher's problem

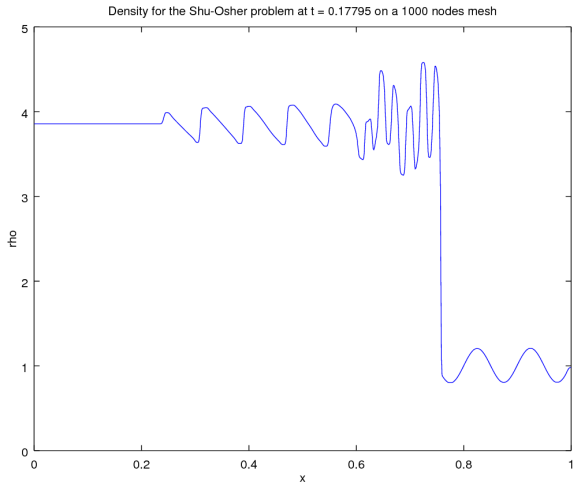
See [13] and [7] (example 3.6). The initial conditions are :

$$
\begin{aligned}
u(x,0) &= \begin{cases} 2.629369 & x \in [0, 0.125] \\ 0 & x \in [0.125, 1] \end{cases} \\
\rho(x,0) &= \begin{cases} 3.857143 & x \in [0, 0.125] \\ 1 + 0.2\sin(20\pi x) & x \in [0.125, 1] \end{cases} \\
P(x,0) &= \begin{cases} 31/3 & x \in [0, 0.125] \\ 1 & x \in [0.125, 1] \end{cases}
\end{aligned}
\tag{13}
$$

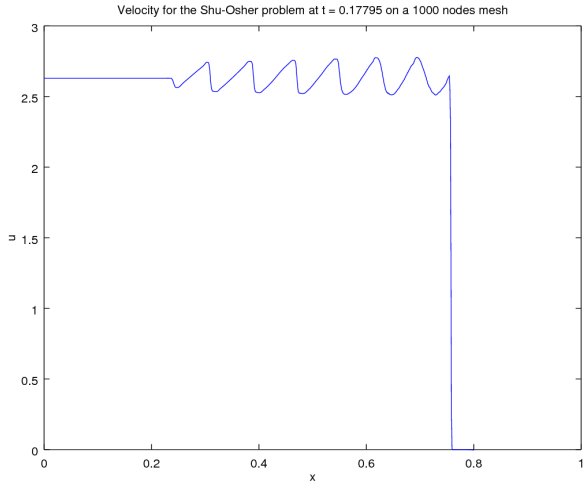
This time we use inlet conditions on the left and outflow conditions on the right. The inlet conditions are :



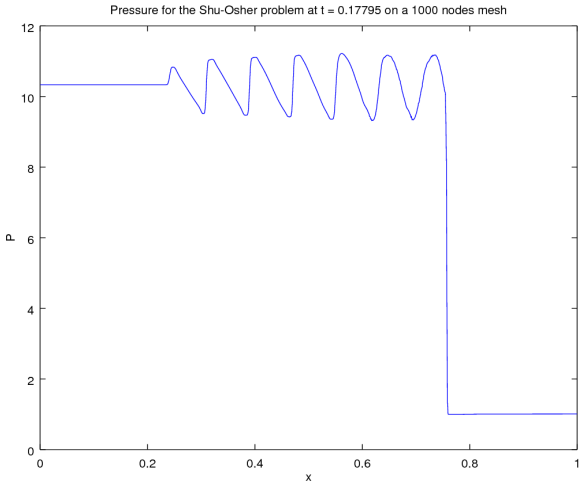
(a) Initial conditions



(b) Density



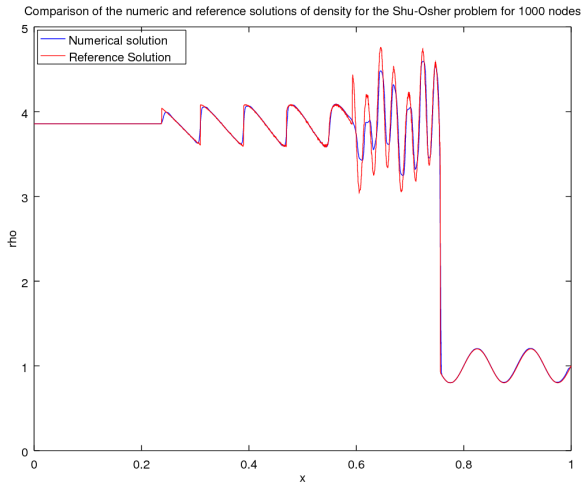
(c) Velocity



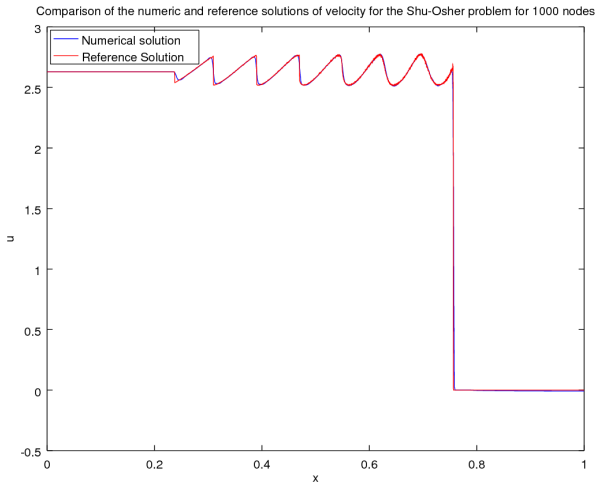
(d) Pressure

FIGURE 14 – Result for the Shu-Osher's problem

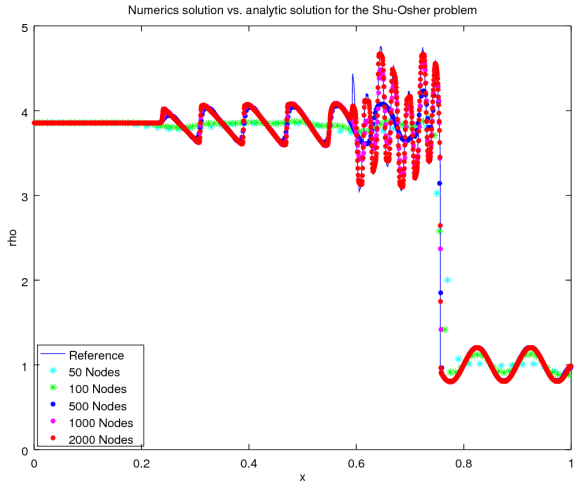
Again we don't know any analytic solution for this case, so the reference solution is given by the WENO-5 method on 16000 nodes.



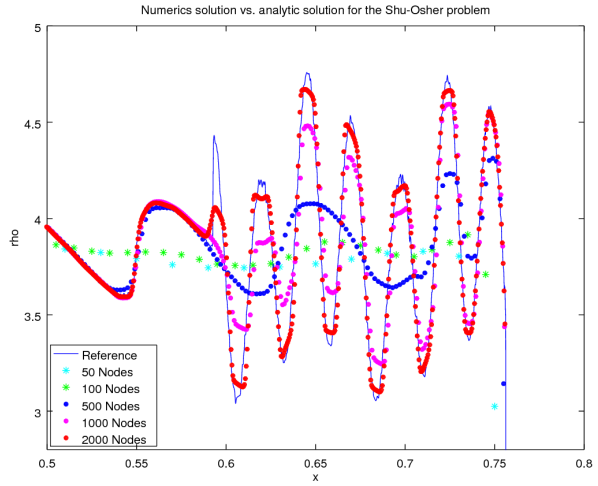
(a) Density



(b) Velocity



(c) Multi nodes



(d) Multi nodes zoom

FIGURE 15 – Result for the Shu-Osher problem with the analytic solution and for different mesh size

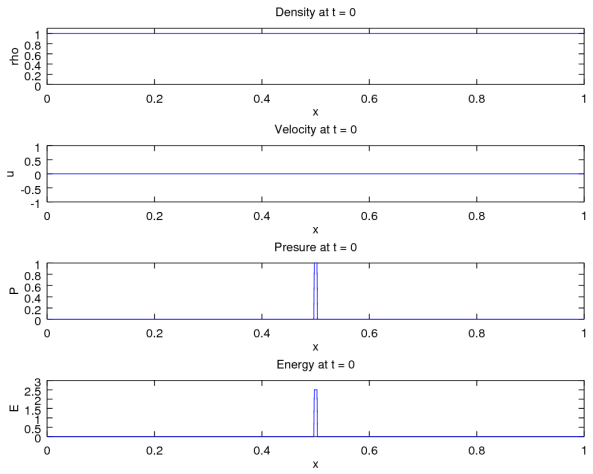
We definitely observe that the method converge to the reference solution.

3.6 Sedov explosion

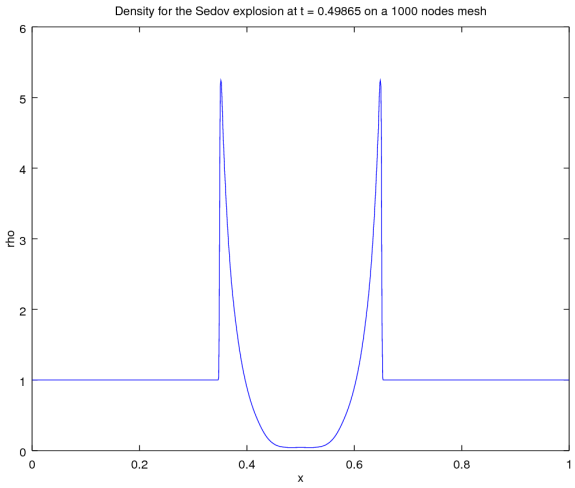
The initial conditions are :

$$
\begin{aligned}
u(x,0) &= 0 \qquad \forall x \in \Omega \\
\rho(x,0) &= 1 \qquad \forall x \in \Omega \\
P(x,0) &= \begin{cases} 1 & x \in [0.5 - 3.5\dfrac{\Delta x}{2}, 0.5 + 3.5\dfrac{\Delta x}{2}] \\ 10^{-5} & \text{else} \end{cases}
\end{aligned}
\tag{14}
$$

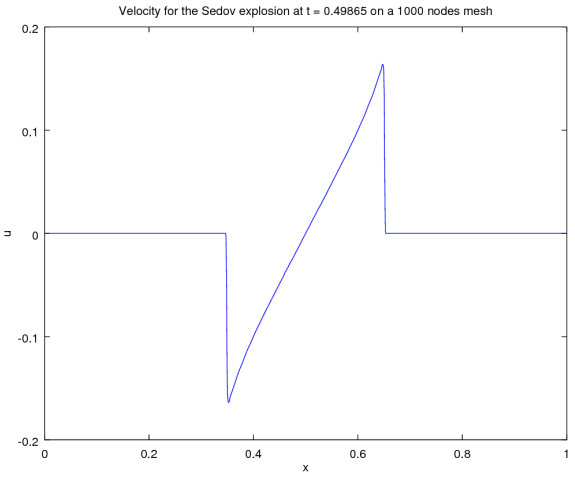
We use again solid wall boundary conditions and watch the result around $t = 0.005$.



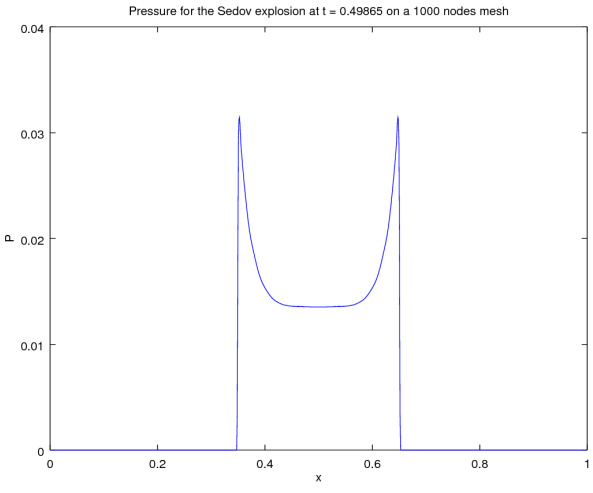
(a) Initial conditions



(b) Density



(c) Velocity



(d) Pressure

FIGURE 16 – Result for the Sedov's explosion

4 Adaptation on non-regular meshes

We are now interested in adapting our method to non regular meshes. After presenting the new mesh disposition, we will perform again the error test and several example to illustrate the robustness of the method.

4.1 The nodes

The particular distribution of nodes we are interested in is the Gauss-Lobatto nodes. First we defined the n-th Legendre polynomial as follows.

$$\begin{aligned} P_0(x) &= 1 \qquad \forall x \in [-1,1] \\ P_1(x) &= x \qquad \forall x \in [-1,1] \\ (n+1)P_{n+1}(x) &= (2n+1)xP_n(x) - nP_{n-1}(x) \qquad \forall x \in [-1,1] \end{aligned} \tag{15}$$

The nodes are the $N-2$ roots of the derivative of $(n-1)$-th Legendre polynomial with the two extremities of the domain :

$$-1 = x_1 < x_i \ : \ P'_{n-1}(x_i) = 0 \quad i \in [\![2, N-1]\!] < x_N = 1$$

We can easily shifted them into $[a,b]$ with the formula :

$$x'_i = \frac{b-a}{2}x_i + \frac{a+b}{2}$$

The nodes are computed with the Haylley's method [1], see [14] for further details. The recursion is the following :

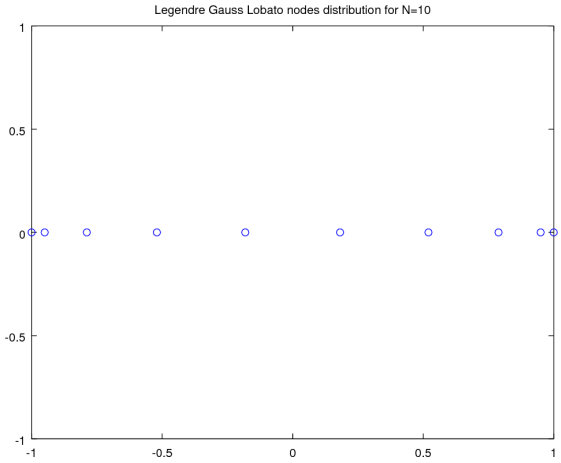
$$\begin{aligned} x_0 &= \left[-1, \left(1 - \frac{3(N-1)}{8N^3}\right)\cos\left(\frac{4j+1}{4N+1}\pi\right), 1\right] \quad j \in [\![1, N-1]\!] \\ x_{n+1} &= x_n - 2 * \frac{P'_{n-1}(x_n)P''_{n-1}(x_n)}{2[P''_{n-1}(x_n)]^2 - P'_{n-1}(x_n)P'''_{n-1}(x_n)} \end{aligned} \tag{16}$$

And the derivative are given by :

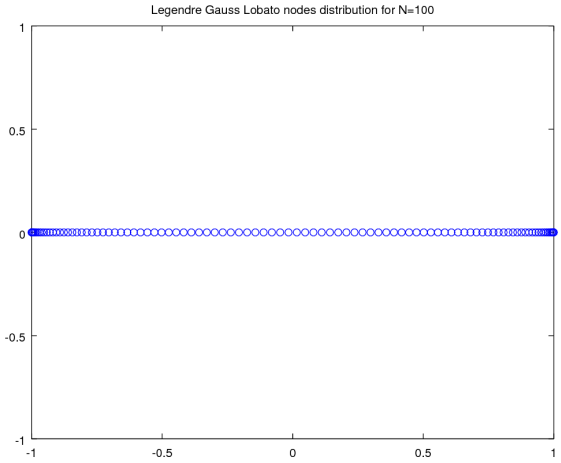
$$\begin{aligned} P'_n(x) &= \frac{n(P_{n-1}(x) - xP_n(x))}{1-x^2} \\ P''_n(x) &= \frac{2xP'_n(x) - n(n+1)P_n}{1-x^2} \\ P'''_n(x) &= \frac{2xP''_n(x) - (n(n+1)-2)P_n}{1-x^2} \end{aligned}$$

As we can see on the picture bellow, the distribution is dense on the extremities of the domain and really spaced in the center.

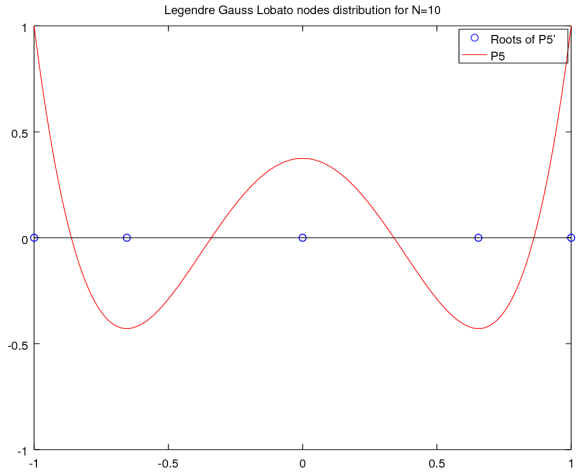
1. A method of order 3 similar to the method of Newton.



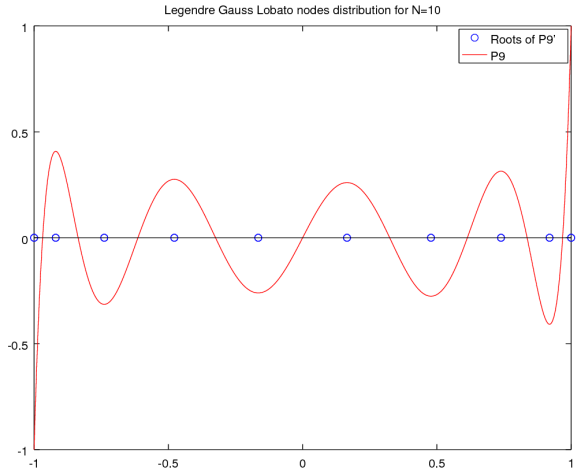
(a) Mesh $N = 10$



(b) Mesh $N = 100$



(c) Mesh $N = 5$ with P_5



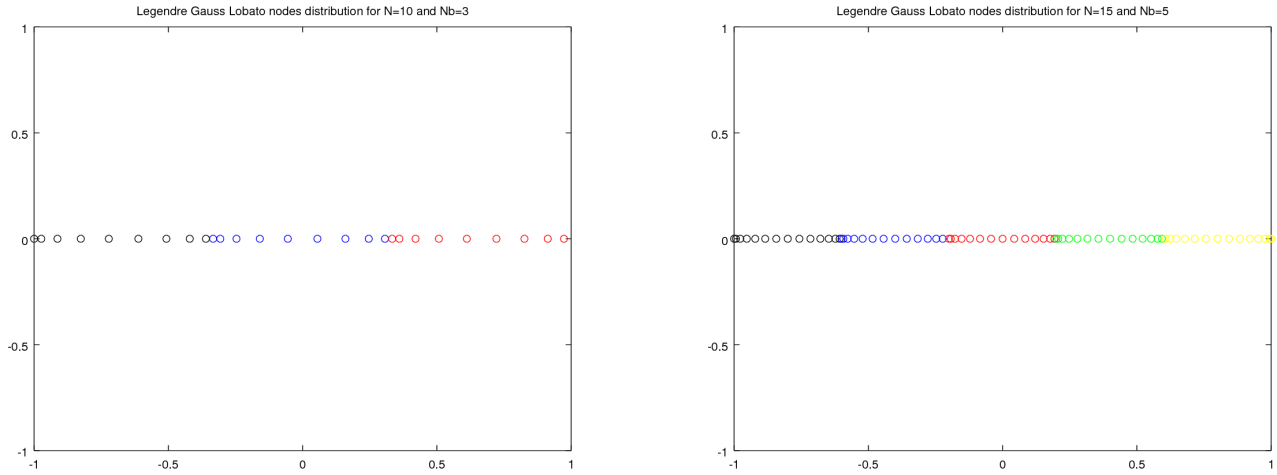
(d) Mesh $N = 10$ with P_9

FIGURE 17 – Legendre-Gauss-Lobato mesh

Also on figure 17-(c) and 17-(d) we observe that the nodes we computed fit the extrema of the polynomial that is to say we get the roots of the derivative polynomial.

4.2 Division in blocks

Before we continue, I precise that we computed the cell edges of our domain and by taking the mean value for two consecutive nodes we get the cell's centers.
We divide our interval $[a, b]$ into N_b equal blocks, and apply this mesh in each block.



(a) Mesh $N = 10$ in $N_b = 3$ blocks

(b) Mesh $N = 15$ in $N_b = 5$ blocks

FIGURE 18 – Legendre-Gauss-Lobato mesh divide in blocks

5 Numerical experiment for non regular nodes distribution

Experiment are made first with one block and then with 5 blocks. We do not present again the initial conditions which are the same as in section 3.
Before beginning, a preliminary remarks about Δt. As we saw in section 2.5, the time step depends on the mesh size. Because we don't have a regular mesh here, we use the following integration time step :

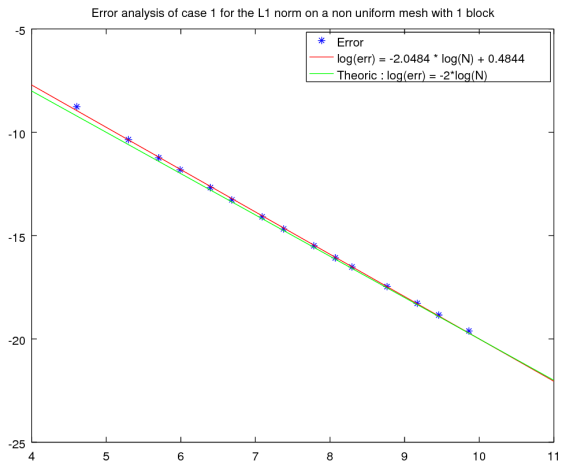
$$\Delta t = \frac{\min \Delta x}{\max a}$$

5.1 Error and convergence

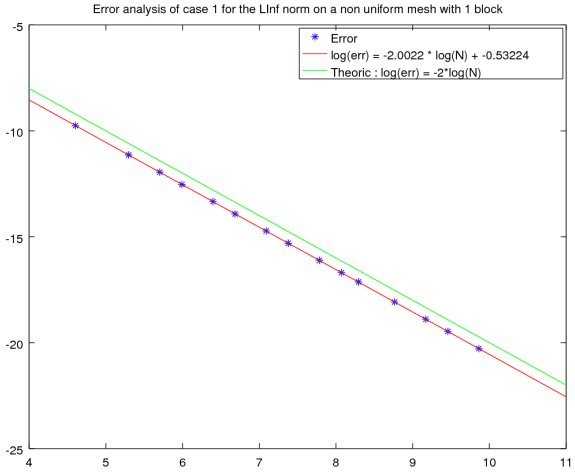
As before we computed the convergence rate with the method of the manufactured solutions. We take the two same cases as before :

$$\begin{aligned}
\text{Case 1 : } & \begin{cases} \rho(x,0) = 1 + 0.2\sin(2\pi x) & \forall x \in \Omega \\ u(x,0) = 1 & \forall x \in \Omega \\ P(x,0) = 1 & \forall x \in \Omega \end{cases} \\
& \mathcal{S}_1(x,t) = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \\
\text{Case 2 : } & \begin{cases} u(x,0) = 1 & \forall x \in \Omega \\ \rho(x,0) = 2 + 0.1\sin(2\pi x) & \forall x \in \Omega \\ P(x,0) = \dfrac{\gamma - 1}{20}\left(20 + 2\cos(2\pi x) - sin(2\pi x)\right) & \forall x \in \Omega \end{cases} \\
& \mathcal{S}_2(x,t) = (1-\gamma)\pi(2\rho(x,t) + E(x,t) - 6)\begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}
\end{aligned} \tag{17}$$

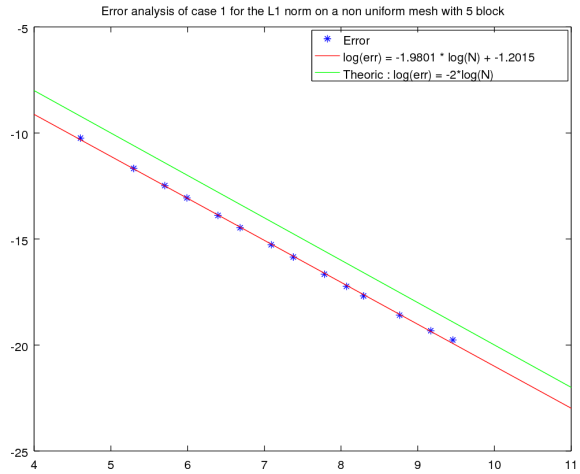
The convergence rate is the same as for the uniform case. But it is surprising that we got such perfect fitting to straight lines. We could imagine that due to this particular distribution, the error would fluctuate more.



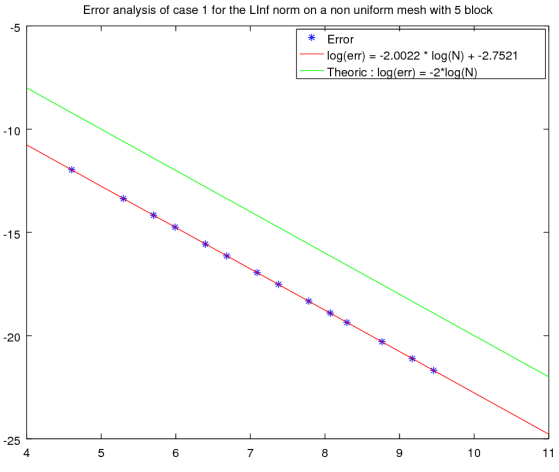
(a) Error L^1 with 1 block



(b) Error L^∞ with 1 block

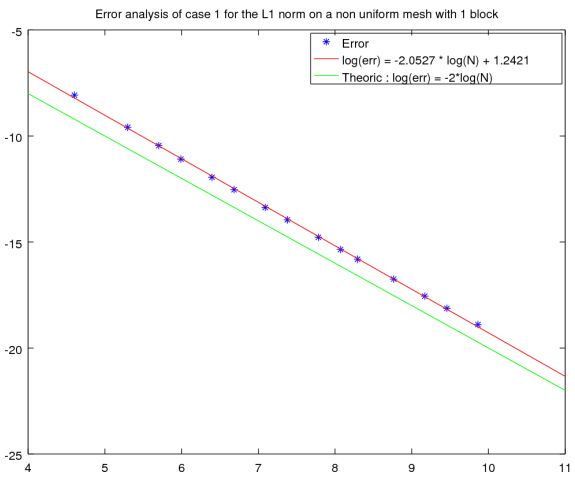


(c) Error L^1 with 5 blocks

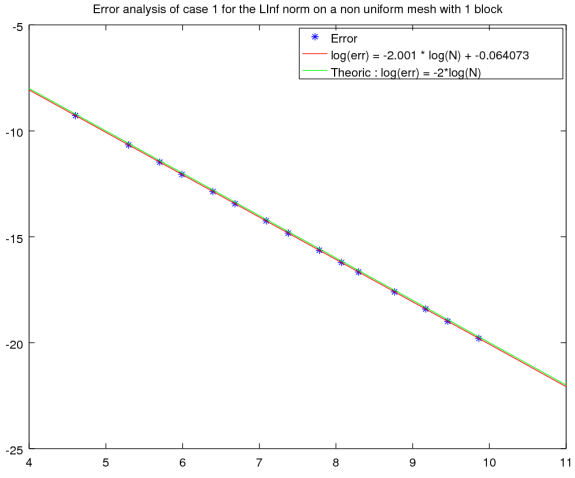


(d) Error L^∞ with 5 blocks

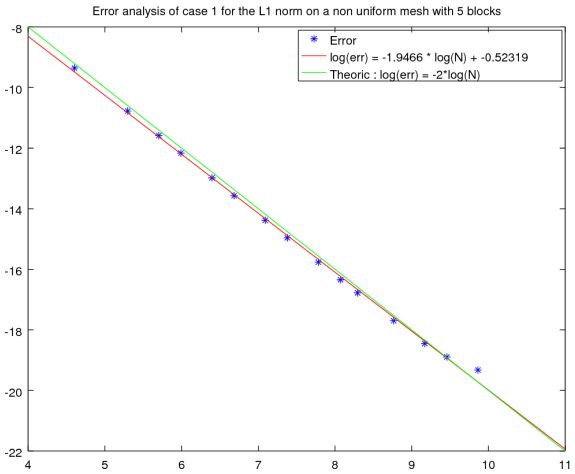
FIGURE 19 – Error for the Case 1 for non uniform meshes



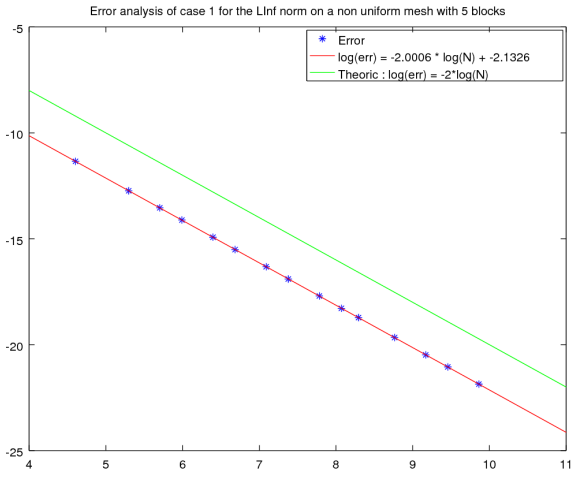
(a) Error L^1 with 1 block



(b) Error L^∞ with 1 block



(c) Error L^1 with 5 blocks



(d) Error L^∞ with 5 blocks

FIGURE 20 – Error for the Case 2 for non uniform meshes

5.2 Sod shock tube

5.3 Interacting Blast Wave

6 Concluding remarks

All this example demonstrate the efficiency of the method. In each case we got results really similar to other results computed with higher order method. We also see the efficiency of the method to capture discontinuities as for example in density for the Sod's shock tube (Figure 9).

To conclude on a personal feeling about this work, I think the most difficult part was to work through the literature and to understand all the differences between finite differences scheme and finite volume scheme. Because in 1D they have really similar form. Then I need a few more time to understand the data structure needed for implementation, especially the nodes numbering (because from one paper to another there is always two king of numbering). But since I have understood the trick with the nodes numbering, the implementation was straight forward.

Joke :
Have you already wondered why, in deriving exp, the function remains the same ?
Because she saw what Euler did to the governing equations of the fluids dynamics and decided to remain quiet.

References

[1] Kurganov Alexander Russo Giovanni Coco Armando, Chertock Alina. A second-order finite-difference method for compressible fluids in domains with moving boundaries. 2017.

[2] Michael Zingale. Notes on the euler equations. *hydro by example*, 2013.

[3] Xinghui Zhong. Strong stability-preserving (ssp) high-order time discretization methods, Sep 2009.

[4] Chi-Wang Shu and Stanley Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes. *Journal of Computational Physics*, 77(2) :439 – 471, 1988.

[5] Sigal Gottlieb, Chi-Wang Shu, and Eitan Tadmor. Strong stability-preserving high-order time discretization methods. *SIAM Review*, 43(1) :89–112, 2001.

[6] *FLASH User's Guide*.

[7] Jun Zhu and Jianxian Qiu. A new fifth order finite difference weno scheme for solving hyperbolic conservation laws. *Journal of Computational Physics*, 318 :110–121, 2016.

[8] Gregor J Gassner. A kinetic energy preserving nodal discontinuous galerkin spectral element method. *International Journal for Numerical Methods in Fluids*, 76(1) :28–50, 2014.

[9] The riemann (or sod) shock-tube problem. http://www.phys.lsu.edu/~tohline/PHYS7412/sod.html .

[10] The riemann (or sod) shock-tube problem. https://fr.mathworks.com/matlabcentral/fileexchange/48734-riemannexact-p1-rho1-u1-p4-rho4-u4-tol- .

[11] A. Baeza, P. Mulet, and D. Zorío. High order boundary extrapolation technique for finite difference methods on complex domains with cartesian meshes. *Journal of Scientific Computing*, 66(2) :761–791, Feb 2016.

[12] Sigal Gottlieb, Julia S Mullen, and Steven J Ruuth. A fifth order flux implicit weno method. *Journal of Scientific Computing*, 27(1-3) :271–287, 2006.

[13] Shu-osher shock tube problem. http://www.ttctech.com/Samples/shockwave/shockwave.htm .

[14] Wikipedia. Halley's method — wikipedia, the free encyclopedia. https://en.wikipedia.org/w/index.php?title=Halley%27s_method&oldid=787009965 , 2017. [Online ; accessed 25-July-2017].