Chin. Ann. Math.
27B(5), 2006, 539–548
DOI: 10.1007/s11401-005-0216-7

Chinese Annals of Mathematics, Series B

Corrected Nonconservative Schemes

Olivier PIRONNEAU*

Abstract Derivatives of discontinuities being Dirac singularities, it is usually not possible to multiply them by discontinuous functions. However in the context of conservation laws we have shown in a recent paper that it can be done. We shall make use of this new framework to revisit some upwind methods, mostly characteristic schemes, and show that they can be corrected to be conservative and to work on difficult problems such as Euler's equations for fluids. Numerous numerical results are given.

Keywords Upwinding, Non-conservative, Finite volume method, Compressible Euler
2000 MR Subject Classification 35A25, 65N99

0 Introduction

Non-conservative schemes have been discarded long ago for not giving the proper shock conditions and positions.

To illustrate this difficulty consider Burgers' equation

$$\partial_t u + \partial_x \frac{u^2}{2} = 0, \quad \forall\, x \in \mathbb{R} \times (0,T), \quad u(x,0) = u^0(x), \quad \forall\, x \in \mathbb{R}. \tag{0.1}$$

A shock at $x(0)$ in the initial condition u^0 propagates at speed $\dot{x} = \bar{u}$, the mean value of u at the shock position $x(t)$ at time t.

If everything is smooth away from the shock then, except at $(x(t), t)$, (0.1) is equivalent to

$$\partial_t u + u\partial_x u = 0. \tag{0.2}$$

But at the shock indeed, $u\partial_x u$ does not make sense.

Multiplied by u^{p-1} for any $p \neq 1$ it can be rewritten as

$$\frac{1}{p}\partial_t u^p + \frac{1}{p+1}\partial_x u^{p+1} = 0. \tag{0.3}$$

However, denoting by $[\,\cdot\,]$ the jump across a shock, the condition at the shock for this equation (Rankine-Hugoniot condition) is

$$[u^p]\dot{x} = \frac{p}{p+1}[u^{p+1}], \tag{0.4}$$

which, of course, is not the same as $\dot{x} = \bar{u}$; for instance if $u^- = 0$, $\frac{u^+}{2} \neq \frac{pu^+}{p+1}$ if $p \neq 1$.

Manuscript received May 30, 2005.
*LJLL, University of Paris VI & IUF, 4 Place Jussieu, 75252 Paris Cedex 05, France.
E-mail: pironneau@ann.jussieu.fr

In [1] a calculus was given which, for example, allows to expand $\partial_x u^2$ in terms of u and $\partial_x u$:

$$\partial_x u^2 = 2\bar{u}\partial_x u \text{ or more generally } \partial_x f(u) = \Big(\int_0^1 f'(u^- + s[u])ds\Big)\partial_x u. \tag{0.5}$$

These identities apply only to functions which are C^1 except on non intersecting lines (shocks) on which u has left and right traces $u^\pm$.

Obviously (0.5) is an identity when u is C^1; when there is a shock at x^* then $\partial_x f(u)$ has a Dirac mass at x^* of weight $[f(u)]$. On the right in (0.5) there is also a Dirac mass but of weight $[u]\int_0^1 f'(u^- + s[u])ds$ which, by the way, is also $[f(u)]$. Thus (0.5) is an equality in the sense of distribution theory.

Notation

For any continuous real valued function g we shall denote

$$\overline{g(u)} := \int_0^1 g(u^- + s(u^+ - u^-))ds \tag{0.6}$$

whenever $u : \mathbb{R}^d \to \mathbb{R}$ is continuous except on a smooth curve γ of $\mathbb{R}^d$ but has left and right traces $u^\pm$ on γ.

Notice that $g(u)$ is defined everywhere, not only on γ and that it is the usual value when u is continuous. Notice also that $\bar{u} = \frac{u^+ + u^-}{2}$ and that if G is a primitive of g and $d = 1$, $\overline{g(u)} = [G(u)]/[u]$.

1 Characteristic Schemes for Burgers' Equation

Consider again (0.1). It is also

$$\partial_t u + \bar{u}\partial_x u = 0, \tag{1.1}$$

because at a shock $x(t)$ the first term has a Dirac mass equal to $-[u]\dot{x}$ while the second term has one of weight $\bar{u}[u]$; the sum must be zero: that is precisely the Rankine Hugoniot condition.

Recall once more that here $\bar{u}$ is defined everywhere and is u except at discontinuities where it is $\frac{u^+ + u^-}{2}$.

For given τ and z, let $t \to X(t)$ be the solution of

$$\dot{X}(t) = \bar{u}(X(t), t), \quad \forall\, t \in (0, \tau) \cup (\tau, T), \quad X(\tau) = z. \tag{1.2}$$

Then (1.1) is also, formally at least,

$$\frac{d}{dt}u(X(t), t) = 0 \quad \text{at } \tau = t,\ z = x. \tag{1.3}$$

So a possible scheme for (0.1) is

$$u^{m+1}(x) = u^m(X^m(x)), \quad \text{where } X^m(x) = x - \bar{u}^m(x)\delta t. \tag{1.4}$$

Such schemes, which is essentially an extention of Lax' scheme for arbitrary time step δt, have been analyzed in [2, 3, 5] etc. They are quite easy to program and are energy preserving when the integrals are computed exactly.

Entropy conditions should be satisfied in the limit because the scheme is dissipative. Indeed by Taylor expansion on (1.4)

$$u^{m+1} = u^m - \delta t \partial_x \bar{u}^m + \frac{\delta t^2}{2}(\bar{u}^m)^2 \partial_{xx} u + \cdots$$

so that the PDE equivalent to the scheme is

$$\partial_t u + \bar{u}\partial_x u - \frac{\bar{u}^2 \delta t}{2}\partial_{xx} u + o(\delta t) = 0.$$

A spacial discretization leads to

$$u_i^{m+1} = u_{j(i)}^m (1 - \alpha_i) + u_{j(i)+1}^m \alpha_i, \tag{1.5}$$

where $j(i)$ is the integer part of $\frac{x_i - \bar{u}_i^m \delta t}{\delta x}$ and α_i is the remainder part.

Naturally one can approximate $\bar{u}_i \approx \frac{u_{i+1}+u_{i-1}}{2}$. One can also use this formula only when $u_{i+1} - u_{i-1} \gg \delta x$ and let it be u_i otherwise.

The advantages of such a scheme are obvious; as $\alpha_i \in (0, 1)$ we have

$$\min_j u_j^m \le u_i^{m+1} \le \max_j u_j^m. \tag{1.6}$$

And at the level of (1.4) the conservation of integrals can be analyzed by formula like

$$\int_{\mathbb{R}} u^{m+1} = \int_{\mathbb{R}} u^m (X^m(x)) dx = \int_{\mathbb{R}} u^m(y) \det(\nabla X^m)^{-1}(y) dy. \tag{1.7}$$

The scheme works very well (see Figure 1), despite its non conservative appearance, but this is because it is, in fact, an approximation of (1.1) which is a conservative expression of (0.1). Now suppose that we apply the same to (0.3) written in terms of $v = u^p$:

$$\partial_t v + \frac{p}{p+1}\partial_x v^{1+\frac{1}{p}} = 0. \tag{1.8}$$

According to (0.5) it is

$$\partial_t v + \overline{v^{\frac{1}{p}}}\partial_x v = 0. \tag{1.9}$$

So (1.5) applies to v except that the convective velocity which defines $j(i)$ and α_i is

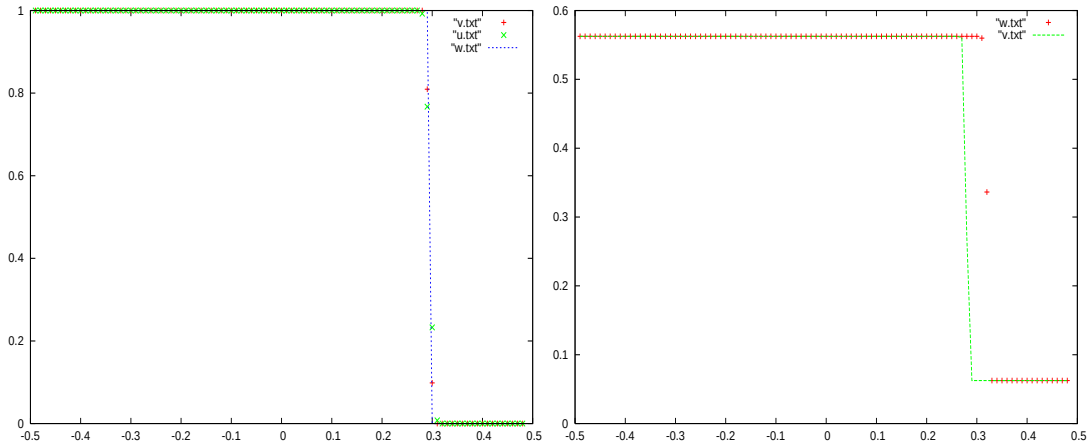
$$\overline{v^{\frac{1}{p}}} = \frac{p}{p+1}\frac{(v^+)^{1+\frac{1}{p}} - (v^-)^{1+\frac{1}{p}}}{v^+ - v^-}.$$

But we can also keep $w = \frac{(v^+)^{\frac{1}{p}} + (v^-)^{\frac{1}{p}}}{2}$ and have a scheme which integrates correctly (0.1). Results are shown on Figure 1.

2 The Shallow Water Equations

Consider

$$\partial_t u + \partial_x\Big(\frac{u^2}{2} + p(\rho)\Big) = 0, \quad \partial_t \rho + \partial_x(\rho u) = 0 \tag{2.1}$$



(1) (2)

Figure 1 (1) Numerical solution with scheme (1.5) of Burgers' equation (0.1) with 3 different time steps, $u^0(x) = a$ when $x < -0.3$ and b otherwise, $a = 1,\ b = 0,\ \delta x = \frac{1}{100},\ \delta t = 0.01, 0.02,\ 0.03$ and the solution is shown at $T = 0.4$. The scheme is exact for $\delta t = 0.02$. (2) Solution of (1.9) with $p = 2,\ a = 0.75,\ b = 0.25$, with the right shock speed (dots) and the speed of $p = 1$ (solid line). In the first case the shock is, as it ought to be, at 0.34, in the second it is also correctly predicted at 0.3.

with initial conditions $u(x,0) = 0$ and $\rho(x,0) = 1 - \frac{H(x)}{2}$ where H is the Heaviside function.

Let

$$D_t = \partial_t + \bar{u}\partial_x.$$

Then (2.1) is also

$$D_t u + \overline{p'(\rho)}\ \partial_x \rho = 0, \quad D_t \rho + \bar{\rho}\,\partial_x u = 0, \tag{2.2}$$

because, according to (0.5)

$$\partial_x(\rho u) = \bar{u}\partial_x\rho + \bar{\rho}\partial_x u. \tag{2.3}$$

With $X^m(x) = x - \overline{u^m}(x)\delta t$, a simple implicit scheme is

$$u^{m+1} + \delta t\,\overline{p'(\rho^m)}\ \partial_x \rho^{m+1} = u^m o X^n, \quad \rho^{m+1} + \delta t\,\overline{\rho^m}\ \partial_x u^{m+1} = \rho^m o X^n. \tag{2.4}$$

Then central finite difference in space is applied. To compute $\bar{u}$ we use half of the left plus right values.

On Figure 2, a comparison is shown between the solution with a Riemann solver and the solution with the scheme above when $p = \rho$. It is shown also that if $\overline{\rho^m}$ and $\overline{u^m}$ are not computed with central differences, an error is observed on the shock speed.

Consider now the case $p = \log\rho$. Then it is tempting to use $e = \log\rho$. The algorithm becomes

$$u^{m+1} + \delta t\partial_x e^{m+1} = u^m o X^n, \quad e^{m+1} + \delta t\partial_x u^{m+1} = e^m o X^n. \tag{2.5}$$

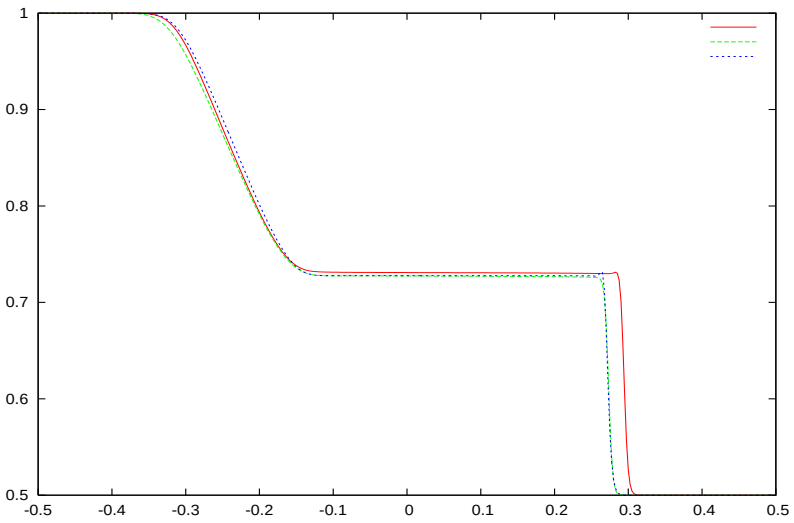


Figure 2 Comparison between 2 finite difference formulae (centered or upwind for overlined quantities) with the solution obtained with a scheme based on a Riemann solver (400 points, $T = 0.3$, 300 time steps). The position of the shock with the upwind formula for the overlined quantities is wrong; with centered formulae, the result coincides with those with the Riemann solver.

It is easy to see that this scheme is unconditionally L^2 stable as it conserves energy. However the scheme is not conservative because

$$\frac{\partial_x \rho}{\bar{\rho}} \neq \partial_x \log \rho.$$

To our surprise very little difference can be observed numerically from the two schemes. Nevertheless one can make the scheme conservative by replacing (2.5) by

$$u^{m+1} + \delta t w^m \partial_x e^{m+1} = u^m o X^n, \quad e^{m+1} + \frac{\delta t}{w^m} \partial_x u^{m+1} = e^m o X^n$$
$$\text{with } w = [\exp e^{m+1}](\overline{\exp e^{m+1}}\ [e^{m+1}])^{-1}. \tag{2.6}$$

Even though $w = 1$ when e is continuous, w is defined everywhere, not only at the shock, so it is easy to implement numerically.

Remark 2.1 Notice the extreme simplicity of a scheme like (2.2) which, if programmed in `C++`, is only half a page (see Appendix A).

3 Isentropic 2D Flows

Isentropic inviscid flow is given by

$$\partial_t \rho + \nabla \cdot (\rho u) = 0, \quad \partial_t (\rho u) + \nabla \cdot (\rho u \otimes u + \rho^\gamma) = 0 \tag{3.1}$$

with ρ and u given at time zero.

Let us assume that the flow is quasi-stationary near the boundaries. Then, on a bounded domain Ω, $u\cdot n$ must be prescribed on $\Gamma := \partial\Omega$, (n is its normal vector), and u, ρ also at points of Γ where $u\cdot n < 0$.

Consider the following time-discretization and weak formulation

$$\begin{aligned}\frac{1}{\overline{\rho^m}\delta t}(\rho^{m+1} - \rho^m oX^m) + \nabla\cdot u^{m+1} = 0,\\ \frac{\overline{(\rho^m)^{2-\gamma}}}{\gamma\delta t}(u^{m+1} - u^m o\widetilde{X}^m) + \nabla\rho^{m+1} = 0,\end{aligned} \tag{3.2}$$

where $\widetilde{X}$ is computed with the velocity $\frac{\overline{\rho u}}{\bar\rho}$.

Let V_h be the finite element space of piecewise affine continuous functions which are equal to zero on the boundaries Γ_D where $u\cdot n < 0$. Then denoting by $\Gamma_N = \Gamma - \Gamma_D$, we approximate (3.2) by searching $(u_h^{m+1},\ \rho_h^{m+1})$ such that

$$u_h^{m+1} - u_{\Gamma_D} \in V_h$$

and

$$\begin{aligned}\int_\Omega \frac{\rho_h^{m+1} w_h}{\overline{\rho_h^m}} - \delta t\Big(\int_\Omega u_h^{m+1}\cdot\nabla w_h - \int_{\Gamma_M} u_h^{m+1}\cdot n w_h\Big) = \int_\Omega \frac{\rho_h^m oX^m w_h}{\overline{\rho_h^m}}\,,\\ \frac{1}{\gamma}\int_\Omega \overline{(\rho_h^m)^{2-\gamma}}\ u_h^{m+1} v_h + \delta t\int_\Omega \nabla\cdot\rho_h^{m+1} v_h = \frac{1}{\gamma}\int_\Omega \overline{(\rho_h^m)^{2-\gamma}}\ u_h^m oX^m v_h\end{aligned} \tag{3.3}$$

for all $w_h,\ v_{hi}$ in $V_h,\ j = 1, 2$. Here Γ_M is the part of Γ_N which is transparent to incoming waves (non reflecting boundary). By contrast on $\Gamma_N\backslash\Gamma_M$ the formulation imposes $u\cdot n = 0$ and shocks will be reflected. The leading idea in constructing such scheme is to keep the symmetry between the grad and div terms in the two equations so as to obtain an energy-like conservation property when $w_h = \rho_h$ and $v_h = u_h$.

Figures 3 shows the flow around a disk at various Mach numbers. Symmetry is exploited and computation is performed on half of the domain only.

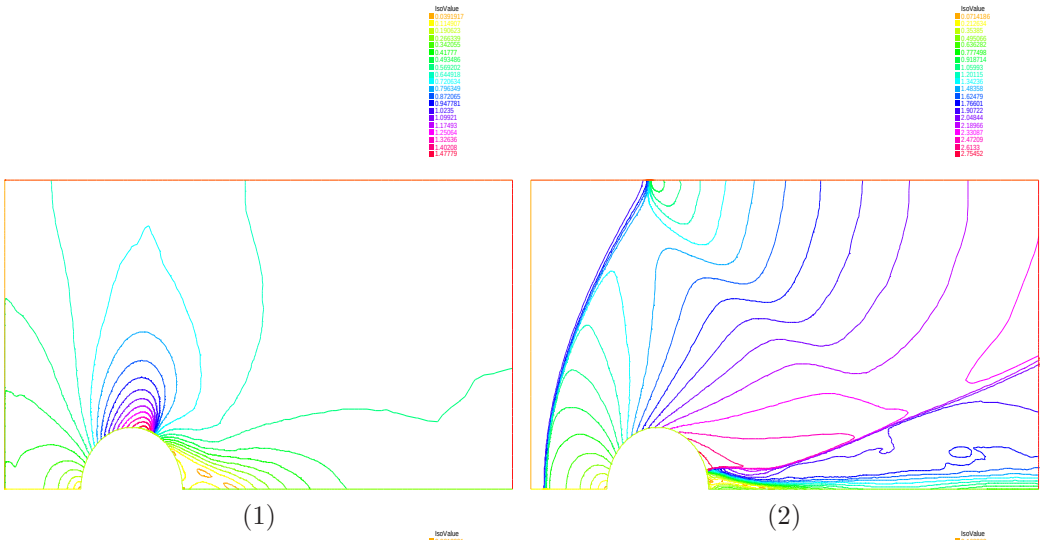
Then the same method is applied for the computation of flows around a NACA0012 airfoil (see Figure 3).

Remark 3.1 Here again the program (in freefem script) of Appendix B shows the simplicity of the algorithm.

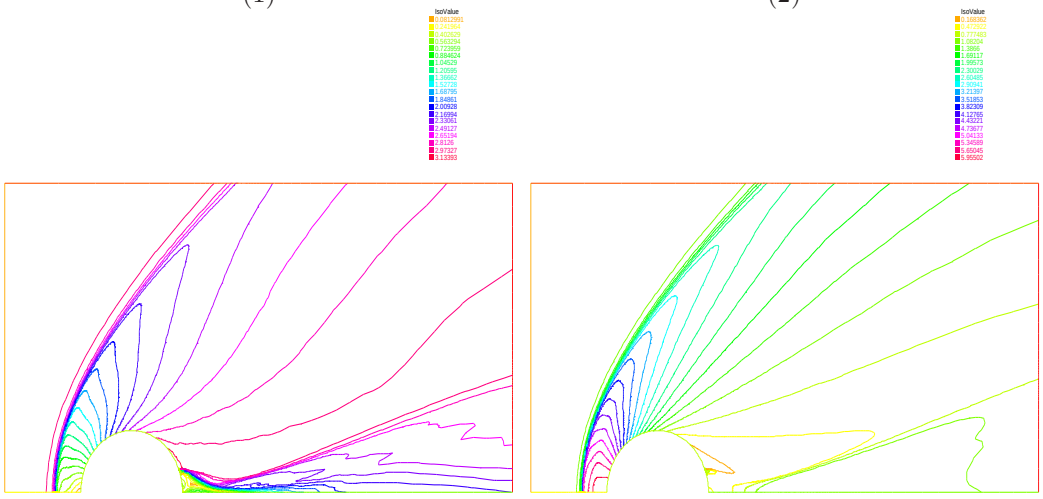
Finally when $\gamma = 1$ we make the transformation $e = \log\rho$ and write (3.2) as

$$\begin{aligned}\frac{1}{\delta t}(e^{m+1} - e^m oX^m) + \frac{1}{\eta^m}\nabla\cdot u^{m+1} = 0,\\ \frac{1}{\delta t}(u^{m+1} - u^m o\widetilde{X}^m) + \eta^m\nabla e^{m+1} = 0.\end{aligned} \tag{3.4}$$

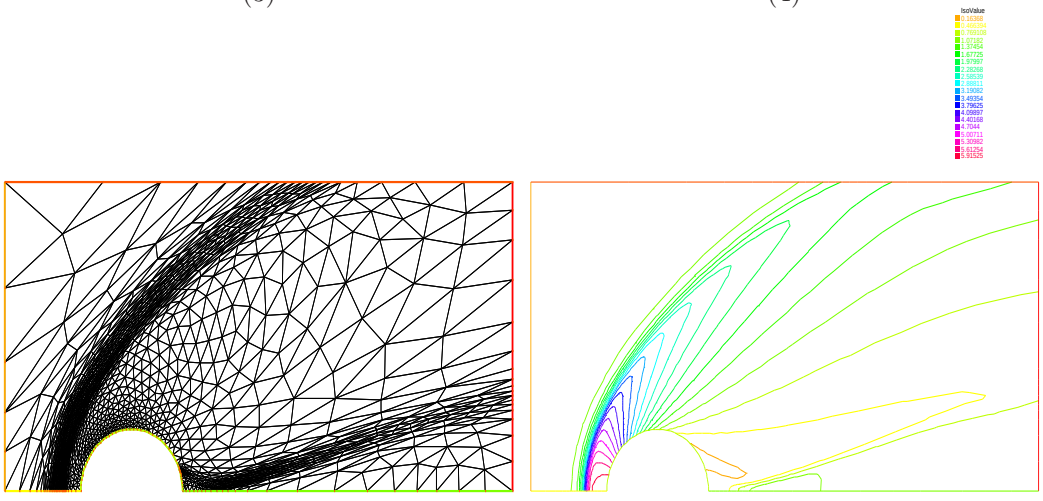
We made two computations, one with $\eta = 1$ because the scheme is fast and conserves energy and one with the correct value $\eta = \frac{[\exp e]}{\overline{\exp e}\,[e]}$. The results show that the computation is wrong with $\eta = 1$ everywhere and the shock is not at the correct place while with the correct η the results are the same as with (3.3), thought admittedly not as good. In this case conservativity is important!



(1) (2)

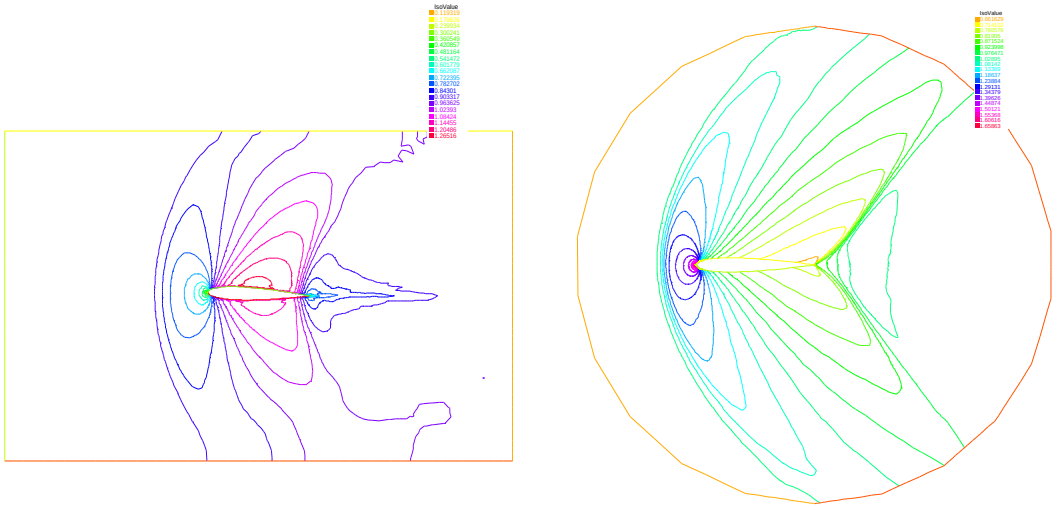


(3) (4)



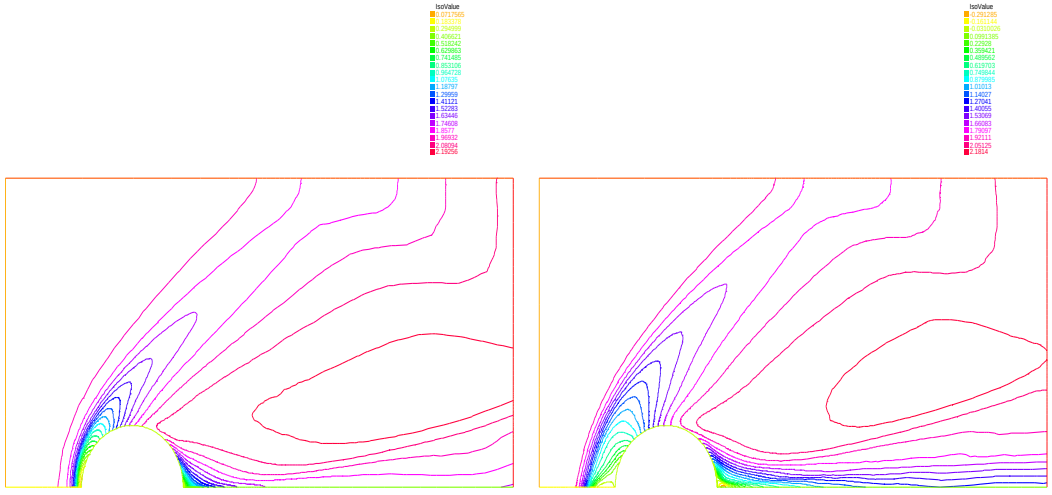
(5) (6)

Figure 3 Flow around a disk computed with scheme (3.3) and $\gamma = 1.4$. From top to bottom, left to right we have the Mach lines at inflow Mach =0.7, then 2 and then 3. At Mach 3 we display also the density ρ. Each computation is performed with mesh adaptivity. The first 20 iterations are done with coarse meshes, then from iterations 20 to 30 we use between 12000 to 20000 vertices. Already with 1200 vertices at iteration 8 the results are very good: the last 2 pictures (bottom) display the mesh and the density at inflow Mach=3 at iteration 8. To obtain the last picture (8 iterations) takes less than a minute on a Pentium M PC at 2 MHz using `freefem++`[4], starting from $u^0 = 0, \rho^0 = 1$.



(1) (2)

Figure 4 Flow around a NACA0012 airfoil showing the density computed with scheme (3.3) and $\gamma = 1.4$ at inflow Mach of 0.9 and 1.2 with 2 degree of incidence angle.



(1) (2)

Figure 5 Flow around a disk at Mach 2 with $\gamma = 1.4$ computed with (3.4). (1) $\eta = 1$ and the shock is not at the right place. (2) η is what it should be for correct Rankine-Hugoniot conditions and the results are similar to those of Figure 3.

4 Extension to the Full Euler Equations for Fluids

There are naturally several ways to extend the scheme to the Euler equations for Fluids:

$$\begin{aligned} &\partial_t \rho + \bar{u}\nabla\rho + \bar{\rho}\,\nabla\cdot u = 0,\\ &\bar{\rho}\Big(\partial_t u + \frac{\overline{\rho u}}{\bar{\rho}}\,\nabla u\Big) + \nabla p = 0,\\ &\partial_t p + \bar{u}\nabla p + (\gamma-1)\bar{p}\,\nabla\cdot u = 0. \end{aligned} \tag{4.1}$$

One possibility is to couple u, p and then update ρ, i.e.,

$$\begin{aligned} &\frac{1}{(\gamma-1)\delta t\bar{p}^m}(p^{m+1} - p^m oX^m) + \nabla\cdot u^{m+1} = 0,\\ &\frac{\bar{\rho}^m}{\delta t}(u^{m+1} - u^m o\widetilde{X}^m) + \nabla p^{m+1} = 0,\\ &\rho^{m+1} = \rho^m oX^m + \frac{\bar{\rho}^m}{(\gamma-1)\bar{p}^m}(p^{m+1} - p^m oX^m). \end{aligned} \tag{4.2}$$

A numerical result is given on Figure 4.

Figure 6 Pressure for a Euler flow around a disk at Mach 2 computed by (4.2)

5 Conclusion

It is true that expanding the products by the chain rule for derivatives leads to a loss of conservativity. By using explicitly the jumps of the discontinuous functions one can recover the correct Rankine-Hugoniot conditions and this does not require shock tracking. This procedure is powerful to generate new schemes and we have shown in this paper that Characteristic-Galerkin schemes can be extended in this fashion to conservation laws; the final scheme is robust and quite simple to implement. The details of the generalization to the full Euler equations will be presented later.

Appendix A

Implementation in C++ of Algorithm (1.5)

```
 int main(){  // p =r^gamma
    const double T=0.3, dt=0.001, gamma = 2;
    const int N=400,  M=int(T/dt);
    const double dx = 1./N, dtdx2=dt/dx/2;
    double r[N],u[N],r1[N],u1[N],x[N];

    for(int i=0;i<N;i++){ // initial condition
        x[i] = double(i)/N-0.5;
        if(i<N/2)   {r[i] =1;}  else    { r[i]=0.5;}
        u[i]=0;  r1[i]=r[i]; u1[i]=u[i];
    }
    for(int m=0;m<M;m++){   // time iterations
      for(int k=0;k<10;k++) //Gauss-Seidel iter. for linear syst.
        for(int i=1;i<N-1;i++){
            double uk = (u[i+1]+u[i-1])/2;
            double xk = 0.5+x[i]-uk*dt;
            int j = xk*N;
             if(j==xk*N) j=j-1;
             if(j>=N-1)j=N-2; else if(j<=0) j = 0;
            double xjn = xk*N-j;
            double r1oX = r1[j]*(1-xjn) + r1[j+1]*xjn ;
            double u1oX = u1[j]*(1-xjn) + u1[j+1]*xjn ;
            double rro=(r1[i-1]+r1[i+1])/2,
                   uro = gamma*pow(rro,gamma-2);
            u[i] = u1oX -dtdx2*uro*(r[i+1]-r[i-1]);
            r[i] = r1oX -dtdx2*rro*(u[i+1]-u[i-1]);
        }
      for(int i=1;i<N-1;i++){ u1[i]=u[i];r1[i]=r[i];}
    }
} // print results here
```

Appendix B

Implementation in the `freefem++` language of Algorithm (3.3).

```
real pi2 = atan(1.0)*2, R=0.3, x1=-3,x2=1.0,y2=1.5; border
a1(t=x1,-2-R){x=t; y=0;} border a2(t=-2+R,x2){x=t; y=0;} border
a3(t=0,y2){x=x2;y=t;} border a4(t=x2,x1){x=t;y=y2;} border
a5(t=y2,0){x=x1;y=t;} border bb(t=pi2*2,0) { x=-2+R*cos(t);
y=R*sin(t); }

mesh Th = buildmesh(a1(30)+a2(40)+a3(20)+a4(60)+a5(20)+bb(30));
plot(Th,wait=1); real dt=0.02, u0= 2*sqrt(1.4), v0=0,
visc=0.00125;

fespace Vh(Th,P1); Vh u,v,r,r1,u1,v1,rh,uh,vh,fro;

problem eul(u,v,r,uh,vh,rh) = int2d(Th)(
(fro*(u*uh+v*vh)+r*rh/r1)/dt
        + (dx(r)*uh+ dy(r)*vh - dx(rh)*u - dy(rh)*v)
) + int2d(Th)(-(rh*convect([u1,v1],-dt,r1)/r1
            + fro*(uh*convect([u1,v1],-dt,u1)
            + vh*convect([u1,v1],-dt,v1)))/dt)
+ int1d(Th,3)(rh*u) + int1d(Th,4)(rh*v) + on(5,r=1) + on(5,u=u0) +
on(1,2,5,v=0);

u1= u0; v1= v0; r1 = 1;

for(int k=0;k<200;k++){ // the time loop
    fro = pow(r1,0.6)/1.4;  eul;
    u1=u;r1=abs(r);
    v1=v*(y<1.4)+abs(v)*(y>=1.4); // to avoid reflexion from top
    plot(r,wait=0,value=1);}
```

References

[1] Bardos, C. and Pironneau, O., A formalism for the differentiation of conservation laws, *C. R. Acad. Sci. Paris, Série I*, **335**(10), 2002, 839–845.

[2] Pironneau, O., Finite Element Methods for Fluids, Wiley, 1989.

[3] Douglas, J. and Russell, T. F., Numerical methods for convection dominated diffusion problems based on combining the method of characteristics with finite element or finite difference procedures, *SIAM J. Numer. Anal.*, **19**(5), 1982, 871–885.

[4] Hecht, F., Freefem++ Manual. http://www.freefem.org

[5] Yabe, T., Ishikawa, T., Wang, P. Y., et al., A universal solver for hyperbolic equations by cubic-polynomial interpolation, II, Two- and three-dimensional solvers, *Comput. Phys. Comm.*, **66**(2-3), 1991, 233–242.