

Residual/Entropy viscosity

Jean-Luc Guermond

Department of Mathematics
Texas A&M University
and LIMS/CNRS, Orsay

Seminar, LLNL
March 23, 2009,
Livermore, CA



Acknowledgments

Collaborators: Richard Pasquetti, Univ. Nice, Bojan Popov, Texas A&M University



Outline

1 TRANSPORT EQUATION



Outline

- 1 TRANSPORT EQUATION
- 2 NONLINEAR SCALAR CONSERVATION LAWS



Outline

- 1 TRANSPORT EQUATION
- 2 NONLINEAR SCALAR CONSERVATION LAWS
- 3 COMPRESSIBLE EULER EQUATIONS



OUTLINE



The PDE

- Solve the transport equation

$$\partial_t u + \beta \cdot \nabla u = 0, \quad u|_{t=0} = u_0, \quad +\text{BCs}$$



The PDE

- Solve the transport equation

$$\partial_t u + \beta \cdot \nabla u = 0, \quad u|_{t=0} = u_0, \quad +\text{BCs}$$

- Use continuous finite elements of degree p .



The PDE

- Solve the transport equation

$$\partial_t u + \beta \cdot \nabla u = 0, \quad u|_{t=0} = u_0, \quad +\text{BCs}$$

- Use continuous finite elements of degree p .
- Deviate as little possible from Galerkin.



Some background

- Rk 1 **Godonov's theorem**: Linear numerical schemes for solving the linear transport equation having the property of not generating new extrema (monotone scheme), can be at most first-order accurate.
 \Rightarrow higher-order (at least second-order) monotone schemes must be **nonlinear**. Ex: slope limiters.



Some background

- Rk 1 **Godonov's theorem**: Linear numerical schemes for solving the linear transport equation having the property of not generating new extrema (monotone scheme), can be at most first-order accurate.
⇒ higher-order (at least second-order) monotone schemes must be **nonlinear**. Ex: slope limiters.
- Rk 2 To guarantee uniqueness for conservation laws need **entropy** inequality.



Some background

- Rk 1 **Godonov's theorem**: Linear numerical schemes for solving the linear transport equation having the property of not generating new extrema (monotone scheme), can be at most first-order accurate.
 \Rightarrow higher-order (at least second-order) monotone schemes must be **nonlinear**. Ex: slope limiters.
- Rk 2 To guarantee uniqueness for conservation laws need **entropy** inequality.
- Slope limiters \Leftrightarrow add **nonlinear viscosity**.



The idea

Need viscosity }
Need entropy inequality } \Rightarrow viscosity \sim entropy residual.



The idea

$$\left. \begin{array}{l} \text{Need viscosity} \\ \text{Need entropy inequality} \end{array} \right\} \Rightarrow \text{viscosity} \sim \text{entropy residual.}$$

- No entropy production across contact discontinuities \Rightarrow no viscosity added in contact discontinuities.



The idea

$$\left. \begin{array}{l} \text{Need viscosity} \\ \text{Need entropy inequality} \end{array} \right\} \Rightarrow \text{viscosity} \sim \text{entropy residual.}$$

- No entropy production across contact discontinuities \Rightarrow no viscosity added in contact discontinuities.
- Viscosity \sim residual (Hughes-Mallet (1986) Johnson-Szepessy (1990))
- Add entropy to formulation (For Hamilton-Jacobi equations Guermond-Popov (2007))



The algorithm

- Define entropy residual $D_h := \partial_t u_h^2 + \beta \cdot \nabla u_h^2$,



The algorithm

- Define entropy residual $D_h := \partial_t u_h^2 + \beta \cdot \nabla u_h^2$,
- Define local mesh size of cell K : $h_K = \text{diam}(K)/p$



The algorithm

- Define entropy residual $D_h := \partial_t u_h^2 + \beta \cdot \nabla u_h^2$,
- Define local mesh size of cell K : $h_K = \text{diam}(K)/\rho$
- Construct a wave speed associated with this residual on each mesh cell K :

$$v_K := h_K \|D_h\|_{\infty, K} / \|u_h^2\|_{\infty, K}$$



The algorithm

- Define entropy residual $D_h := \partial_t u_h^2 + \beta \cdot \nabla u_h^2$,
- Define local mesh size of cell K : $h_K = \text{diam}(K)/p$
- Construct a wave speed associated with this residual on each mesh cell K :

$$v_K := h_K \|D_h\|_{\infty, K} / \|u_h^2\|_{\infty, K}$$

- Define entropy viscosity on each mesh cell K :

$$\nu_K := c_1 h_K \min(\|\beta\|_{\infty, K}, c_2 v_K)$$



The algorithm

- Define entropy residual $D_h := \partial_t u_h^2 + \beta \cdot \nabla u_h^2$,
- Define local mesh size of cell K : $h_K = \text{diam}(K)/\rho$
- Construct a wave speed associated with this residual on each mesh cell K :

$$v_K := h_K \|D_h\|_{\infty, K} / \|u_h^2\|_{\infty, K}$$

- Define entropy viscosity on each mesh cell K :

$$\nu_K := c_1 h_K \min(\|\beta\|_{\infty, K}, c_2 v_K)$$

- Solution method: Galerkin + entropy viscosity:

$$\int_{\Omega} (\partial_t u_h + \beta \cdot \nabla u_h) v_h dx + \sum_K \int_K \nu_K \nabla u_h \nabla v_h dx = 0, \quad \forall v_h$$



The algorithm + time discretization

- Use an explicit time stepping: BDF, RK3, RK4, etc.



The algorithm + time discretization

- Use an explicit time stepping: BDF, RK3, RK4, etc.
- Idea: make the viscosity explicit.



The algorithm + time discretization

- EX: BDF2



The algorithm + time discretization

- EX: BDF2
- Evaluate D_h on each cell at time step t^n using u_h^n , u_h^{n-1} , u_h^{n-2}

$$\|D_h\|_{\infty, K} := \left\| \frac{3(u_h^n)^2 - 4(u_h^{n-1})^2 + (u_h^{n-2})^2}{2\Delta t} + \beta \cdot \nabla (u_h^n)^2 \right\|_{\infty, K}$$



The algorithm + time discretization

- EX: BDF2
- Evaluate D_h on each cell at time step t^n using u_h^n , u_h^{n-1} , u_h^{n-2}

$$\|D_h\|_{\infty, K} := \left\| \frac{3(u_h^n)^2 - 4(u_h^{n-1})^2 + (u_h^{n-2})^2}{2\Delta t} + \beta \cdot \nabla (u_h^n)^2 \right\|_{\infty, K}$$



The algorithm + time discretization

- EX: BDF2
- Evaluate D_h on each cell at time step t^n using $u_h^n, u_h^{n-1}, u_h^{n-2}$

$$\|D_h\|_{\infty, K} := \left\| \frac{3(u_h^n)^2 - 4(u_h^{n-1})^2 + (u_h^{n-2})^2}{2\Delta t} + \beta \cdot \nabla (u_h^n)^2 \right\|_{\infty, K}$$

- Evaluate entropy viscosity on each mesh cell K :

$$\nu_K := c_1 h_K \min(\|\beta\|_{\infty, K}, c_2 h_K \|D_h\|_{\infty, K} / \|u_h^2\|_{\infty, K})$$



The algorithm + time discretization

- EX: BDF2
- Evaluate D_h on each cell at time step t^n using $u_h^n, u_h^{n-1}, u_h^{n-2}$

$$\|D_h\|_{\infty, K} := \left\| \frac{3(u_h^n)^2 - 4(u_h^{n-1})^2 + (u_h^{n-2})^2}{2\Delta t} + \beta \cdot \nabla (u_h^n)^2 \right\|_{\infty, K}$$

- Evaluate entropy viscosity on each mesh cell K :

$$\nu_K := c_1 h_K \min(\|\beta\|_{\infty, K}, c_2 h_K \|D_h\|_{\infty, K} / \|u_h^2\|_{\infty, K})$$

- Compute u_h^{n+1}

$$\begin{aligned} \frac{3}{2\Delta t} \int_{\Omega} u_h^{n+1} v_h \, d\mathbf{x} &= \int_{\Omega} \frac{1}{2\Delta t} (4u_h^n - u_h^{n-1}) - \beta \cdot \nabla (2u_h^n - u_h^{n-1}) v_h \, d\mathbf{x} \\ &\quad - \sum_K \int_K \nu_K \nabla (2u_h^n - u_h^{n-1}) \nabla v_h \, d\mathbf{x}, \quad \forall v_h \end{aligned}$$



Theory for linear steady equations

- Consider

$$\alpha u + \beta \nabla u = f, \quad u|_{\Gamma^-} = 0.$$



Theory for linear steady equations

- Consider

$$\alpha u + \beta \nabla u = f, \quad u|_{\Gamma^-} = 0.$$

Theorem

Let u_h be the finite element approximation. If u in $H^1(\Omega)$ then

$\|u - u_h\|_{L^2} \leq c h^{\frac{1}{2}} \|u\|_{H^1}$ if D_h based on residual.

$\|u - u_h\|_{L^2} \leq c h^{\frac{1}{4}} \|u\|_{H^1}$ if D_h based on quadratic entropy.



Theory for linear steady equations

- Consider

$$\alpha u + \beta \nabla u = f, \quad u|_{\Gamma^-} = 0.$$

Theorem

Let u_h be the finite element approximation. If u in $H^1(\Omega)$ then

$\|u - u_h\|_{L^2} \leq c h^{\frac{1}{2}} \|u\|_{H^1}$ if D_h based on residual.

$\|u - u_h\|_{L^2} \leq c h^{\frac{1}{4}} \|u\|_{H^1}$ if D_h based on quadratic entropy.

Theorem

In one space dimension with \mathbb{P}_1 finite element and residual viscosity,

$$\|u_h\|_{L^\infty} \leq (1 + ch^{\frac{1}{2}} \log(1/h)) \|f\|_{L^\infty}.$$



Numerical tests, smooth case, time-dependent transport

- $\Omega = \{(x, y) \in \mathbb{R}^2, \sqrt{x^2 + y^2} \leq 1\} := B(0, 1),$



Numerical tests, smooth case, time-dependent transport

- $\Omega = \{(x, y) \in \mathbb{R}^2, \sqrt{x^2 + y^2} \leq 1\} := B(0, 1)$,
- Speed: rotation about origin, angular speed 2π



Numerical tests, smooth case, time-dependent transport

- $\Omega = \{(x, y) \in \mathbb{R}^2, \sqrt{x^2 + y^2} \leq 1\} := B(0, 1)$,
- Speed: rotation about origin, angular speed 2π
- $u(x, y) = \frac{1}{2} \left(1 - \tanh \left(\frac{(x - r_0 \cos(2\pi t))^2 + (y - r_0 \sin(2\pi t))^2}{a^2} - 1 \right) + 1 \right)$,



Numerical tests, smooth case, time-dependent transport

- $\Omega = \{(x, y) \in \mathbb{R}^2, \sqrt{x^2 + y^2} \leq 1\} := B(0, 1)$,
- Speed: rotation about origin, angular speed 2π
- $u(x, y) = \frac{1}{2} \left(1 - \tanh \left(\frac{(x - r_0 \cos(2\pi t))^2 + (y - r_0 \sin(2\pi t))^2}{a^2} - 1 \right) + 1 \right)$,
- $a = 0.3, r_0 = 0.4$



Numerical tests, smooth case, time-dependent transport

- $\Omega = \{(x, y) \in \mathbb{R}^2, \sqrt{x^2 + y^2} \leq 1\} := B(0, 1)$,
- Speed: rotation about origin, angular speed 2π
- $u(x, y) = \frac{1}{2} \left(1 - \tanh \left(\frac{(x - r_0 \cos(2\pi t))^2 + (y - r_0 \sin(2\pi t))^2}{a^2} - 1 \right) + 1 \right)$,
- $a = 0.3, r_0 = 0.4$



Numerical tests, smooth case, time-dependent transport

- $\Omega = \{(x, y) \in \mathbb{R}^2, \sqrt{x^2 + y^2} \leq 1\} := B(0, 1)$,
- Speed: rotation about origin, angular speed 2π
- $u(x, y) = \frac{1}{2} \left(1 - \tanh \left(\frac{(x - r_0 \cos(2\pi t))^2 + (y - r_0 \sin(2\pi t))^2}{a^2} - 1 \right) + 1 \right)$,
- $a = 0.3, r_0 = 0.4$

h	\mathbb{P}_1 Stab.				\mathbb{P}_1 Gal.			
	L^2	rate	L^1	rate	L^2	rate	L^1	rate
2.00E-1	2.5893E-1	-	3.6139E-1	-	1.7055E-1	-	2.7910E-1	-
1.00E-1	9.7934E-2	1.403	1.3208E-1	1.452	7.2792E-2	1.228	1.1463E-1	1.284
5.00E-2	1.9619E-3	2.320	2.7310E-3	2.274	1.0993E-2	2.727	1.8321E-2	2.645
2.50E-2	3.5360E-4	2.472	5.1335E-3	2.411	2.0080E-3	2.453	3.4351E-3	2.415
1.25E-2	6.4959E-4	2.445	1.0061E-3	2.351	4.6904E-4	2.098	7.9146E-4	2.118
1.00E-2	3.9226E-4	2.261	6.3555E-4	2.058	3.0943E-4	1.864	5.3982E-4	1.715
6.25E-3	1.4042E-4	2.186	2.3829E-4	2.087	1.2295E-4	1.964	2.1743E-4	1.935

Table: \mathbb{P}_1 approximation.



Numerical tests, smooth case, time-dependent transport

h	\mathbb{P}_2 Stab.				\mathbb{P}_2 Gal.			
	L^2	rate	L^1	rate	L^2	rate	L^1	rate
2.00E-1	4.1390E-2	-	7.0823E-2	-	4.8881E-2	-	9.0057E-2	-
1.00E-1	1.2442E-2	1.734	2.1057E-2	1.750	1.6434E-2	1.572	2.9270E-2	1.621
5.00E-2	2.5126E-3	2.307	4.6317E-3	2.185	3.0119E-3	2.447	5.6097E-3	2.383
2.50E-2	6.0450E-4	2.055	1.1379E-3	2.025	6.6025E-4	2.190	1.2455E-3	2.171
1.25E-2	1.6707E-4	1.855	3.0991E-4	1.876	1.7662E-4	1.902	3.2785E-4	1.926
1.00E-2	1.0510E-4	2.077	1.9546E-4	2.066	1.0970E-4	2.134	2.0413E-4	2.123
6.25E-3	4.1979E-5	1.953	7.7808E-5	1.960	4.3198E-5	1.983	8.0071E-5	1.991

Table: \mathbb{P}_2 approximation.



Convergence saturation

- The method seems to be second-order for \mathbb{P}_1 .



Convergence saturation

- The method seems to be second-order for \mathbb{P}_1 .
- The method seems to be second-order **only** with \mathbb{P}_2 .



Convergence saturation

- The method seems to be second-order for \mathbb{P}_1 .
- The method seems to be second-order **only** with \mathbb{P}_2 .
- The key is that: when solution is smooth, viscosity goes away!



Convergence saturation

- The method seems to be second-order for \mathbb{P}_1 .
 - The method seems to be second-order **only** with \mathbb{P}_2 .
 - The key is that: when solution is smooth, viscosity goes away!
-
- Similar behavior for ENO schemes (either ENO2 or ENO4 and higher are used).



Numerical tests, BV solution, time-dependent transport

- $\Omega = \{(x, y) \in \mathbb{R}^2, \sqrt{x^2 + y^2} \leq 1\} := B(0, 1),$



Numerical tests, BV solution, time-dependent transport

- $\Omega = \{(x, y) \in \mathbb{R}^2, \sqrt{x^2 + y^2} \leq 1\} := B(0, 1)$,
- Speed: rotation about origin, angular speed 2π



Numerical tests, BV solution, time-dependent transport

- $\Omega = \{(x, y) \in \mathbb{R}^2, \sqrt{x^2 + y^2} \leq 1\} := B(0, 1)$,
- Speed: rotation about origin, angular speed 2π
- $u(x, y) = \chi_{B(0, a)}(\sqrt{(x - r_0 \cos(2\pi t))^2 + (y - r_0 \sin(2\pi t))^2})$,



Numerical tests, BV solution, time-dependent transport

- $\Omega = \{(x, y) \in \mathbb{R}^2, \sqrt{x^2 + y^2} \leq 1\} := B(0, 1)$,
- Speed: rotation about origin, angular speed 2π
- $u(x, y) = \chi_{B(0, a)}(\sqrt{(x - r_0 \cos(2\pi t))^2 + (y - r_0 \sin(2\pi t))^2})$,
- $a = 0.3, r_0 = 0.4$



Numerical tests, BV solution, time-dependent transport

- $\Omega = \{(x, y) \in \mathbb{R}^2, \sqrt{x^2 + y^2} \leq 1\} := B(0, 1)$,
- Speed: rotation about origin, angular speed 2π
- $u(x, y) = \chi_{B(0, a)}(\sqrt{(x - r_0 \cos(2\pi t))^2 + (y - r_0 \sin(2\pi t))^2})$,
- $a = 0.3, r_0 = 0.4$



Numerical tests, BV solution, time-dependent transport

- $\Omega = \{(x, y) \in \mathbb{R}^2, \sqrt{x^2 + y^2} \leq 1\} := B(0, 1)$,
- Speed: rotation about origin, angular speed 2π
- $u(x, y) = \chi_{B(0,a)}(\sqrt{(x - r_0 \cos(2\pi t))^2 + (y - r_0 \sin(2\pi t))^2})$,
- $a = 0.3, r_0 = 0.4$

h	\mathbb{P}_1 Stab.				\mathbb{P}_1 Gal.			
	L^2	rate	L^1	rate	L^2	rate	L^1	rate
2.00E-1	1.3578E-1	-	7.0021E-2	-	1.2550E-1	-	7.7194E-2	-
1.00E-1	1.1366E-1	0.257	5.1486E-2	0.444	1.3336E-1	-	8.4934E-2	-
5.00E-2	8.1621E-2	0.478	2.7808E-2	0.889	8.8637E-2	0.590	5.5299E-2	0.619
2.50E-2	6.1286E-2	0.413	1.6309E-2	0.770	6.6938E-2	0.405	3.8872E-2	0.509
1.25E-2	4.6861E-2	0.387	9.7747E-3	0.739	5.0986E-2	0.393	2.7757E-2	0.486
1.00E-2	4.2952E-2	0.380	8.2369E-3	0.767	4.6521E-2	0.411	2.4603E-2	0.541
6.25E-3	3.6033E-2	0.374	5.8287E-3	0.736	3.8774E-2	0.387	1.9427E-2	0.503

Table: \mathbb{P}_1 approximation.



Numerical tests, BV solution, time-dependent transport

h	\mathbb{P}_2 Stab.				\mathbb{P}_2 Gal.			
	L^2	rate	L^1	rate	L^2	rate	L^1	rate
2.00E-1	1.0930E-1	-	4.3373E-2	-	1.2695E-1	-	7.6539E-2	-
1.00E-1	7.3222E-2	0.578	2.3771E-2	0.868	7.7805E-2	0.706	5.0175E-2	0.609
5.00E-2	5.5707E-2	0.394	1.3704E-2	0.795	5.7999E-2	0.424	3.4561E-2	0.538
2.50E-2	4.2522E-2	0.389	8.0365E-3	0.770	4.5870E-2	0.338	2.5407E-2	0.444
1.25E-2	3.2409E-2	0.392	4.6749E-3	0.782	3.6951E-2	0.312	1.8913E-2	0.426
1.00E-2	2.9812E-2	0.374	3.9421E-3	0.764	3.4146E-2	0.354	1.7187E-2	0.429
6.25E-3	2.4771E-2	0.394	2.7200E-3	0.790	2.8974E-2	0.350	1.4406E-2	0.376

Table: \mathbb{P}_2 approximation.



Nonlinear scalar conservation laws

- Solve

$$\partial_t u + \partial_x f(u) + \partial_y g(u) = 0 \quad u|_{t=0} = u_0, \quad +\text{BCs.}$$



Nonlinear scalar conservation laws

- Solve

$$\partial_t u + \partial_x f(u) + \partial_y g(u) = 0 \quad u|_{t=0} = u_0, \quad +\text{BCs.}$$

- The unique entropy solution satisfies

$$\partial_t E(u) + \partial_x F(u) + \partial_y G(u) \leq 0$$

for all entropy pair $E(u)$, $F(u) = \int E'(u)f'(u)du$,
 $G(u) = \int E'(u)g'(u)du$



Nonlinear scalar conservation laws

- Solve

$$\partial_t u + \partial_x f(u) + \partial_y g(u) = 0 \quad u|_{t=0} = u_0, \quad +\text{BCs.}$$

- The unique entropy solution satisfies

$$\partial_t E(u) + \partial_x F(u) + \partial_y G(u) \leq 0$$

for all entropy pair $E(u)$, $F(u) = \int E'(u)f'(u)du$,
 $G(u) = \int E'(u)g'(u)du$

- Use continuous finite elements of degree p .



Nonlinear scalar conservation laws

- Solve

$$\partial_t u + \partial_x f(u) + \partial_y g(u) = 0 \quad u|_{t=0} = u_0, \quad +\text{BCs.}$$

- The unique entropy solution satisfies

$$\partial_t E(u) + \partial_x F(u) + \partial_y G(u) \leq 0$$

for all entropy pair $E(u)$, $F(u) = \int E'(u)f'(u)du$,
 $G(u) = \int E'(u)g'(u)du$

- Use continuous finite elements of degree p .
- Deviate as little possible from Galerkin.



2D scalar nonlinear conservation laws

- Choose one entropy $E(u)$



2D scalar nonlinear conservation laws

- Choose one entropy $E(u)$
- Define entropy residual, $D_h(u) := \partial_t E(u) + \partial_x F(u) + \partial_y G(u)$



2D scalar nonlinear conservation laws

- Choose one entropy $E(u)$
- Define entropy residual, $D_h(u) := \partial_t E(u) + \partial_x F(u) + \partial_y G(u)$
- Define local mesh size of cell K : $h_K = \text{diam}(K)/p$



2D scalar nonlinear conservation laws

- Choose one entropy $E(u)$
- Define entropy residual, $D_h(u) := \partial_t E(u) + \partial_x F(u) + \partial_y G(u)$
- Define local mesh size of cell K : $h_K = \text{diam}(K)/p$
- Construct a speed associated with residual on each cell K :

$$v_K := h_K \|D_h\|_{\infty, K} / \|E(u_h)\|_{\infty, K}$$



2D scalar nonlinear conservation laws

- Choose one entropy $E(u)$
- Define entropy residual, $D_h(u) := \partial_t E(u) + \partial_x F(u) + \partial_y G(u)$
- Define local mesh size of cell K : $h_K = \text{diam}(K)/p$
- Construct a speed associated with residual on each cell K :

$$v_K := h_K \|D_h\|_{\infty, K} / \|E(u_h)\|_{\infty, K}$$

- Compute maximum local wave speed:

$$\beta_K = \|\sqrt{f'(u)^2 + g'(u)^2}\|_{\infty, K}$$



2D scalar nonlinear conservation laws

- Choose one entropy $E(u)$
- Define entropy residual, $D_h(u) := \partial_t E(u) + \partial_x F(u) + \partial_y G(u)$
- Define local mesh size of cell K : $h_K = \text{diam}(K)/p$
- Construct a speed associated with residual on each cell K :

$$v_K := h_K \|D_h\|_{\infty, K} / \|E(u_h)\|_{\infty, K}$$

- Compute maximum local wave speed:
 $\beta_K = \|\sqrt{f'(u)^2 + g'(u)^2}\|_{\infty, K}$
- Define entropy viscosity on each mesh cell K :

$$\nu_K := c_1 h_K \min(\beta_K, c_2 v_K)$$



2D scalar nonlinear conservation laws

- Choose one entropy $E(u)$
- Define entropy residual, $D_h(u) := \partial_t E(u) + \partial_x F(u) + \partial_y G(u)$
- Define local mesh size of cell K : $h_K = \text{diam}(K)/p$
- Construct a speed associated with residual on each cell K :

$$v_K := h_K \|D_h\|_{\infty, K} / \|E(u_h)\|_{\infty, K}$$

- Compute maximum local wave speed:
 $\beta_K = \|\sqrt{f'(u)^2 + g'(u)^2}\|_{\infty, K}$
- Define entropy viscosity on each mesh cell K :

$$\nu_K := c_1 h_K \min(\beta_K, c_2 v_K)$$

- Solution method: Galerkin + entropy viscosity:

$$\int_{\Omega} (\partial_t u_h + \partial_x f(u_h) + \partial_y g(u_h)) v_h dx + \sum_K \int_{K_{\square}} \nu_K \nabla u_h \nabla v_h dx = 0,$$



Convergence tests, 2D Burgers

- Solve 2D Burgers

$$\partial_t u + \partial_x \left(\frac{1}{2} u^2 \right) + \partial_y \left(\frac{1}{2} u^2 \right) = 0$$



Convergence tests, 2D Burgers

- Solve 2D Burgers

$$\partial_t u + \partial_x \left(\frac{1}{2} u^2 \right) + \partial_y \left(\frac{1}{2} u^2 \right) = 0$$

- Subject to the following initial condition

$$u(x, y, 0) = u^0(x, y) = \begin{cases} -0.2 & \text{if } x < 0.5 \text{ and } y > 0.5 \\ -1 & \text{if } x > 0.5 \text{ and } y > 0.5 \\ 0.5 & \text{if } x < 0.5 \text{ and } y < 0.5 \\ 0.8 & \text{if } x > 0.5 \text{ and } y < 0.5 \end{cases}$$



Convergence tests, 2D Burgers

- Solve 2D Burgers

$$\partial_t u + \partial_x \left(\frac{1}{2} u^2 \right) + \partial_y \left(\frac{1}{2} u^2 \right) = 0$$

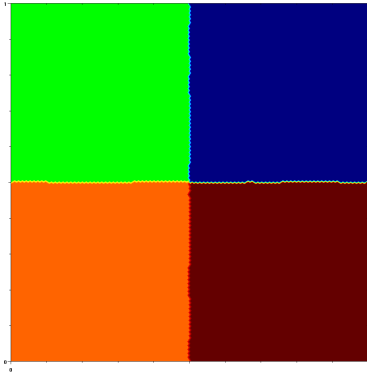
- Subject to the following initial condition

$$u(x, y, 0) = u^0(x, y) = \begin{cases} -0.2 & \text{if } x < 0.5 \text{ and } y > 0.5 \\ -1 & \text{if } x > 0.5 \text{ and } y > 0.5 \\ 0.5 & \text{if } x < 0.5 \text{ and } y < 0.5 \\ 0.8 & \text{if } x > 0.5 \text{ and } y < 0.5 \end{cases}$$

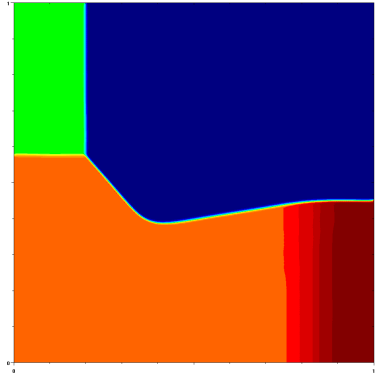
- Compute solution in $(0, 1)^2$ at $t = \frac{1}{2}$.



Convergence tests, 2D Burgers



Initial data



P_1 FE, $3 \cdot 10^4$ nodes

Convergence tests, 2D Burgers

h	\mathbb{P}_1, h weight				$\mathbb{P}_1, h^{1/2}$ weight			
	L^2	rate	L^1	rate	L^2	rate	L^1	rate
5.00E-2	2.3651E-1	–	9.3661E-2	–	2.3569E-1	–	9.2986E-2	–
2.50E-2	1.7653E-1	0.422	4.9934E-2	0.907	1.7991E-1	0.390	5.1945E-2	0.840
1.25E-2	1.2788E-1	0.465	2.5990E-2	0.942	1.3303E-1	0.436	2.8164E-2	0.883
6.25E-3	9.3631E-2	0.449	1.3583E-2	0.936	1.0023E-1	0.408	1.5359E-2	0.875
3.12E-3	6.7498E-2	0.472	6.9797E-3	0.961	7.3418E-2	0.449	8.1661E-3	0.911

Table: Burgers, \mathbb{P}_1 approximation.



Convergence tests, 2D Burgers

h	\mathbb{P}_2, h weight				$\mathbb{P}_2, h^{1/2}$ weight			
	L^2	rate	L^1	rate	L^2	rate	L^1	rate
5.00E-2	1.8068E-1	–	5.2531E-2	–	1.8523E-1	–	5.4902E-2	–
2.50E-2	1.2956E-1	0.480	2.7212E-2	0.949	1.3754E-1	0.429	2.9643E-2	0.889
1.25E-2	9.5508E-2	0.440	1.4588E-2	0.899	1.0342E-1	0.411	1.6183E-2	0.873
6.25E-3	6.8806E-2	0.473	7.6435E-3	0.932	7.6828E-2	0.428	8.6782E-3	0.899

Table: Burgers, \mathbb{P}_2 approximation.



Buckley Leverett, \mathbb{P}_2 FE

- Solve $\partial_t u + \partial_x f(u) + \partial_y g(u) = 0$.

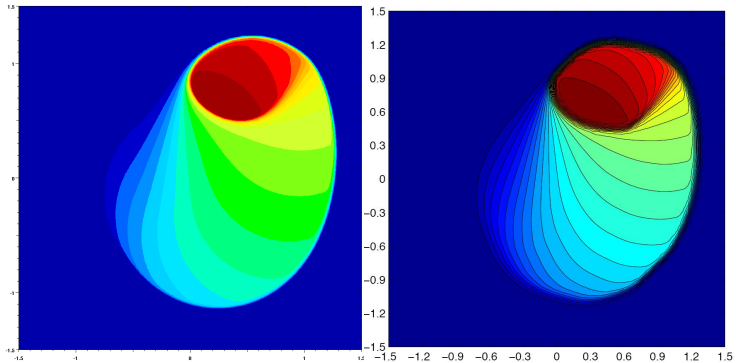
$$f(u) = \frac{u^2}{u^2 + (1-u)^2}, \quad g(u) = f(u)(1 - 5(1-u)^2)$$

Non-convex fluxes (composite waves)

$$u(x, y, 0) = \begin{cases} 1, & \sqrt{x^2 + y^2} \leq 0.5 \\ 0, & \text{else} \end{cases}$$



Buckley Leverett, \mathbb{P}_2 FE



KPP (WENO + superbee limiter fails), \mathbb{P}_2 FE

- Solve $\partial_t u + \partial_x f(u) + \partial_y g(u) = 0$.

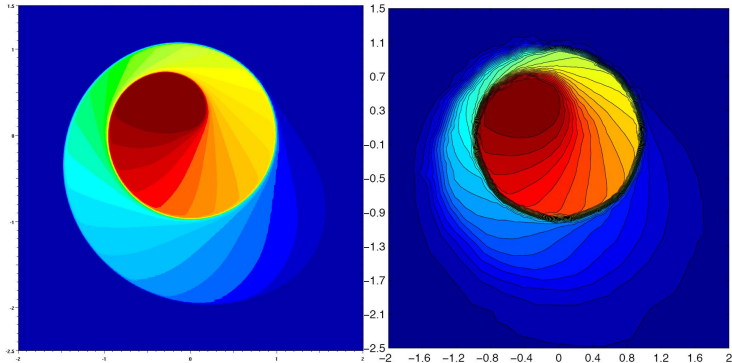
$$f(u) = \sin(u), \quad g(u) = \cos(u)$$

Non-convex fluxes (composite waves)

$$u(x, y, 0) = \begin{cases} \frac{7}{2}\pi, & \sqrt{x^2 + y^2} \leq 1 \\ \frac{1}{4}\pi, & \text{else} \end{cases}$$



KPP (WENO + superbee limiter fails), \mathbb{P}_2 FE



Euler flows

- Solve compressible Euler equations

$$\partial_t \rho + \nabla \cdot (\rho \mathbf{u}) = 0$$

$$\partial_t (\rho \mathbf{u}) + \nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u} + p \mathbb{I}) = 0$$

$$\partial_t (E) + \nabla \cdot (\mathbf{u}(E + p)) = 0$$

$$\rho e = E - \frac{1}{2} \rho \mathbf{u}^2, \quad T = (\gamma - 1)e \quad T = \frac{p}{\rho}$$

Initial data + BCs

- Use continuous finite elements of degree p .



Euler flows

- Solve compressible Euler equations

$$\partial_t \rho + \nabla \cdot (\rho \mathbf{u}) = 0$$

$$\partial_t (\rho \mathbf{u}) + \nabla \cdot (\rho \mathbf{u} \otimes \mathbf{u} + p \mathbb{I}) = 0$$

$$\partial_t (E) + \nabla \cdot (\mathbf{u}(E + p)) = 0$$

$$p e = E - \frac{1}{2} \rho \mathbf{u}^2, \quad T = (\gamma - 1) e \quad T = \frac{p}{\rho}$$

Initial data + BCs

- Use continuous finite elements of degree p .
- Deviate as little possible from Galerkin.



The algorithm

- Compute the entropy $S_h = \frac{\rho_h}{\gamma-1} \log(\rho_h/\rho_h^\gamma)$



The algorithm

- Compute the entropy $S_h = \frac{\rho_h}{\gamma-1} \log(\rho_h/\rho_h^\gamma)$
- Define entropy residual, $D_h := \partial_t S_h + \nabla \cdot (\mathbf{u}_h S_h)$



The algorithm

- Compute the entropy $S_h = \frac{\rho_h}{\gamma-1} \log(\rho_h/\rho_h^\gamma)$
- Define entropy residual, $D_h := \partial_t S_h + \nabla \cdot (\mathbf{u}_h S_h)$
- Define local mesh size of cell K : $h_K = \text{diam}(K)/p$



The algorithm

- Compute the entropy $S_h = \frac{\rho_h}{\gamma-1} \log(\rho_h/\rho_h^\gamma)$
- Define entropy residual, $D_h := \partial_t S_h + \nabla \cdot (\mathbf{u}_h S_h)$
- Define local mesh size of cell K : $h_K = \text{diam}(K)/p$
- Construct a speed associated with residual on each cell K :

$$v_K := h_K \|D_h\|_{\infty, K} / \|S_h\|_{\infty, K}$$



The algorithm

- Compute the entropy $S_h = \frac{\rho_h}{\gamma-1} \log(\rho_h/\rho_h^\gamma)$
- Define entropy residual, $D_h := \partial_t S_h + \nabla \cdot (\mathbf{u}_h S_h)$
- Define local mesh size of cell K : $h_K = \text{diam}(K)/p$
- Construct a speed associated with residual on each cell K :

$$v_K := h_K \|D_h\|_{\infty, K} / \|S_h\|_{\infty, K}$$

- Compute maximum local wave speed:
 $\beta_K = \| \|\mathbf{u}\| + (\gamma T)^{\frac{1}{2}} \| \|_{\infty, K}$



The algorithm

- Compute the entropy $S_h = \frac{\rho_h}{\gamma-1} \log(\rho_h/\rho_h^\gamma)$
- Define entropy residual, $D_h := \partial_t S_h + \nabla \cdot (\mathbf{u}_h S_h)$
- Define local mesh size of cell K : $h_K = \text{diam}(K)/p$
- Construct a speed associated with residual on each cell K :

$$v_K := h_K \|D_h\|_{\infty, K} / \|S_h\|_{\infty, K}$$

- Compute maximum local wave speed:
 $\beta_K = \| \|\mathbf{u}\| + (\gamma T)^{\frac{1}{2}} \| \|_{\infty, K}$
- Define entropy viscosity and thermal conductivity on each mesh cell K :

$$\mu_K := c_1 \rho_h h_K \min(\beta_K, c_2 v_K), \quad \kappa_K = \mathcal{P} \mu_K$$



The algorithm

- Compute the entropy $S_h = \frac{\rho_h}{\gamma-1} \log(\rho_h/\rho_h^\gamma)$
- Define entropy residual, $D_h := \partial_t S_h + \nabla \cdot (\mathbf{u}_h S_h)$
- Define local mesh size of cell K : $h_K = \text{diam}(K)/p$
- Construct a speed associated with residual on each cell K :

$$v_K := h_K \|D_h\|_{\infty, K} / \|S_h\|_{\infty, K}$$

- Compute maximum local wave speed:
 $\beta_K = \| \|\mathbf{u}\| + (\gamma T)^{\frac{1}{2}} \| \|_{\infty, K}$
- Define entropy viscosity and thermal conductivity on each mesh cell K :

$$\mu_K := c_1 \rho_h h_K \min(\beta_K, c_2 v_K), \quad \kappa_K = \mathcal{P} \mu_K$$

- Solution method: Galerkin + entropy viscosity + thermal conductivity



1D burgers + Fourier

- Solution method: Fourier + BDF4 + entropy viscosity



1D burgers + Fourier

- Solution method: Fourier + BDF4 + entropy viscosity

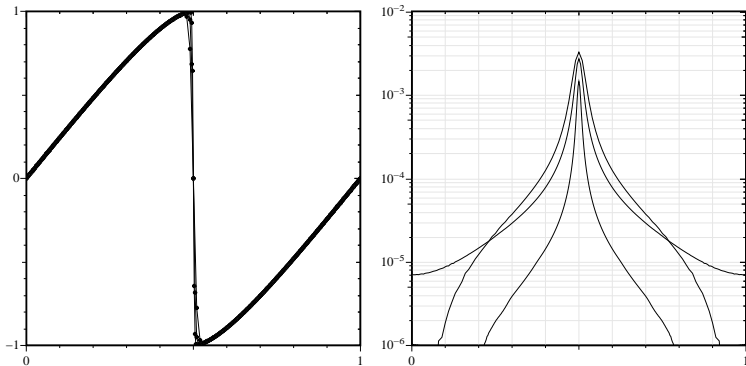


Figure: Left: u_N and right: $\nu_N(u_N)$, for Burgers at $t = 0.25$ with $N = 50, 100,$ and 200 .

1D Nonconvex flux + Fourier

- Consider $\partial_t + \partial_x f(u) = 0$, $u(x, 0) = u_0(x)$

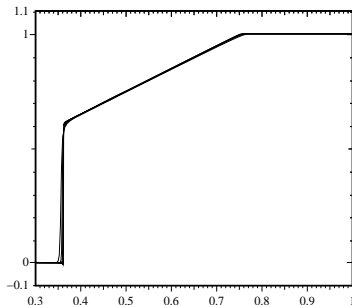
$$f(u) = \begin{cases} \frac{1}{4}u(1-u) & \text{if } u < \frac{1}{2}, \\ \frac{1}{2}u(u-1) + \frac{3}{16} & \text{if } u \geq \frac{1}{2}, \end{cases} \quad u_0(x) = \begin{cases} 0, & x \in (0, 0.25], \\ 1, & x \in (0.25, 1] \end{cases}$$



1D Nonconvex flux + Fourier

- Consider $\partial_t + \partial_x f(u) = 0$, $u(x, 0) = u_0(x)$

$$f(u) = \begin{cases} \frac{1}{4}u(1-u) & \text{if } u < \frac{1}{2}, \\ \frac{1}{2}u(u-1) + \frac{3}{16} & \text{if } u \geq \frac{1}{2}, \end{cases} \quad u_0(x) = \begin{cases} 0, & x \in (0, 0.25], \\ 1, & x \in (0.25, 1] \end{cases}$$



Non-convex flux problem
 u_N at $t = 1$ with $N = 200$,
 400, 800, and 1600.



Euler flows + Fourier

- Solution method: Fourier + BDF4 + entropy viscosity



Euler flows + Fourier

- Solution method: Fourier + BDF4 + entropy viscosity

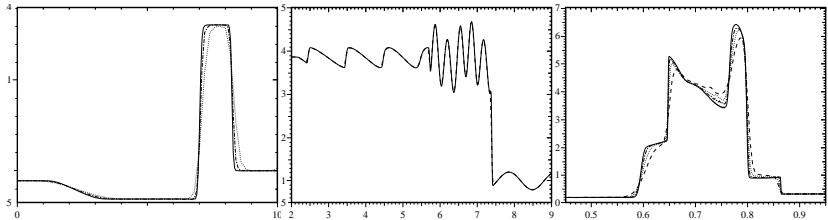


Figure: Lax shock tube, $t = 1.3$, 50, 100, 200 points. Shu-Osher shock tube, $t = 1.8$, 400, 800 points. Right: Woodward-Collela blast wave, $t = 0.038$, 200, 400, 800, 1600 points.



Mach 3 Wind Tunnel with a Step, \mathbb{P}_1 finite elements

- Mach 3 Wind Tunnel with a Step (Standard Benchmark since Woodward and Colella (1984))



Mach 3 Wind Tunnel with a Step, \mathbb{P}_1 finite elements

- Mach 3 Wind Tunnel with a Step (Standard Benchmark since Woodward and Colella (1984))
- Inflow boundary, density 1.4, pressure 1, and x -velocity 3, (Mach =3)



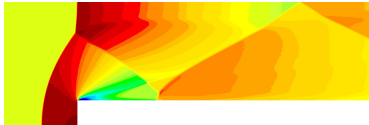
Mach 3 Wind Tunnel with a Step, \mathbb{P}_1 finite elements

- Mach 3 Wind Tunnel with a Step (Standard Benchmark since Woodward and Colella (1984))
- Inflow boundary, density 1.4, pressure 1, and x -velocity 3, (Mach =3)

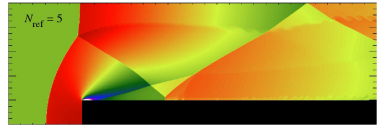


Mach 3 Wind Tunnel with a Step, \mathbb{P}_1 finite elements

- Mach 3 Wind Tunnel with a Step (Standard Benchmark since Woodward and Colella (1984))
- Inflow boundary, density 1.4, pressure 1, and x -velocity 3, (Mach =3)



\mathbb{P}_1 FE, $1.3 \cdot 10^5$ nodes
Log(density)

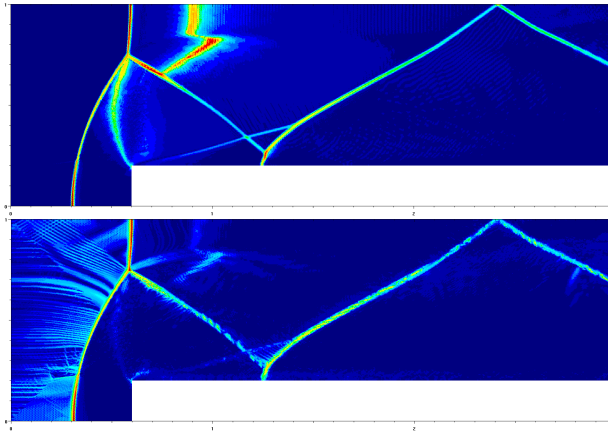


Flash Code, adaptive *PPM*,
 $\sim 4.9 \cdot 10^6$ nodes

movie, www.math.duke.edu/~ying



Mach 3 Wind Tunnel with a Step



Viscosity for
mass

Viscosity for
momentum



Mach 10 Double Mach reflection

- Right-moving Mach 10 shock makes 60° angle with x-axis (Standard Benchmark, Woodward and Colella (1984))



Mach 10 Double Mach reflection

- Right-moving Mach 10 shock makes 60° angle with x-axis (Standard Benchmark, Woodward and Colella (1984))
- Shock interacts with flat plate $x \in (\frac{1}{6}, +\infty)$.



Mach 10 Double Mach reflection

- Right-moving Mach 10 shock makes 60° angle with x-axis (Standard Benchmark, Woodward and Colella (1984))
- Shock interacts with flat plate $x \in (\frac{1}{6}, +\infty)$.
- The unshocked fluid $\rho = 1.4$, $p = 1$, and $\mathbf{u} = 0$



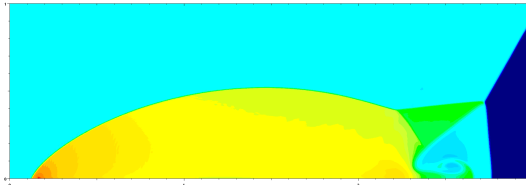
Mach 10 Double Mach reflection

- Right-moving Mach 10 shock makes 60° angle with x-axis (Standard Benchmark, Woodward and Colella (1984))
- Shock interacts with flat plate $x \in (\frac{1}{6}, +\infty)$.
- The unshocked fluid $\rho = 1.4$, $p = 1$, and $\mathbf{u} = 0$



Mach 10 Double Mach reflection

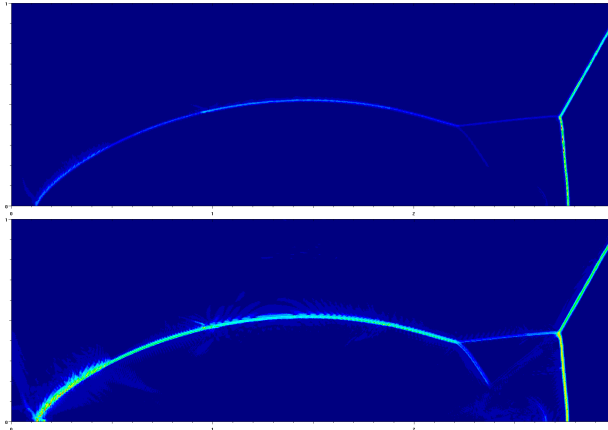
- Right-moving Mach 10 shock makes 60° angle with x-axis (Standard Benchmark, Woodward and Colella (1984))
- Shock interacts with flat plate $x \in (\frac{1}{6}, +\infty)$.
- The unshocked fluid $\rho = 1.4$, $p = 1$, and $\mathbf{u} = 0$



\mathbb{P}_1 FE, $1.5 \cdot 10^5$ nodes, $t = 0.2$
[movie, mngrid.ucsd.edu/~akritsuk](http://movie.mngrid.ucsd.edu/~akritsuk)



Mach 10 Double Mach reflection

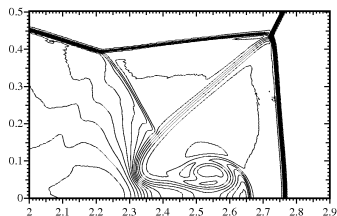


Viscosity for
mass

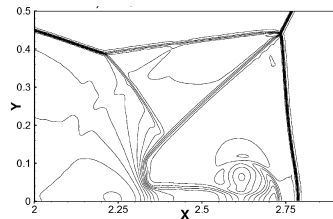
Viscosity for
momentum



Mach 10 Double Mach reflection



\mathbb{P}_1 FE, $1.5 \cdot 10^5$ nodes



WENO5, $\sim 1.7 \cdot 10^5$ nodes

