Commun. Comput. Phys.
doi: 10.4208/cicp.250509.211009a

Vol. **8**, No. 5, pp. 1242-1263
November 2010

Trigonometric WENO Schemes for Hyperbolic Conservation Laws and Highly Oscillatory Problems

Jun Zhu[1] and Jianxian Qiu[2,*]

[1] *College of Science, Nanjing University of Aeronautics and Astronautics, Nanjing, Jiangsu 210016, China.*
[2] *Department of Mathematics, Nanjing University, Nanjing, Jiangsu 210093, China.*

Received 25 May 2009; Accepted (in revised version) 21 October 2009

Communicated by Chi-Wang Shu

Available online 28 July 2010

Abstract. In this paper, we use trigonometric polynomial reconstruction, instead of algebraic polynomial reconstruction, as building blocks for the weighted essentially non-oscillatory (WENO) finite difference schemes to solve hyperbolic conservation laws and highly oscillatory problems. The goal is to obtain robust and high order accurate solutions in smooth regions, and sharp and non-oscillatory shock transitions. Numerical results are provided to illustrate the behavior of the proposed schemes.

AMS subject classifications: 65M06, 65M99, 35L65

Key words: TWENO scheme, hyperbolic conservation laws, highly oscillatory problem, finite difference scheme.

1 Introduction

In this paper, we investigate using trigonometric polynomial reconstruction as building blocks for the weighted essentially non-oscillatory (WENO) finite difference schemes, termed as TWENO schemes, to solve hyperbolic conservation laws:

$$\begin{cases} u_t + f_x(u) = 0, \\ u(x,0) = u_0(x). \end{cases} \tag{1.1}$$

Hyperbolic conservation laws appear often in applications, such as in gas dynamics and modelling of shallow waters, among many others. As a result, devising robust, accurate and efficient methods for numerically solving these problems is of considerable importance and has attracted the interest of many researchers and practitioners. In 1959,

*Corresponding author. *Email addresses:* zhujun@nuaa.edu.cn (J. Zhu), jxqiu@nju.edu.cn (J. Qiu)

Godunov [4] proposed a first-order numerical scheme for solving hyperbolic conservation laws. In order to achieve uniform high order of accuracy, Harten and Osher [6] gave a weaker version of the TVD (Total Variation Diminishing) [5] criterion, and also obtained the essentially non-oscillatory (ENO) type schemes. The key idea of ENO schemes is to apply the most smooth stencil among all candidate stencils to approximate the variables at cell boundaries to a high order of accuracy and to avoid oscillations near discontinuities. In 1994, Liu et al. [9] proposed a Weighted ENO (WENO) scheme that was constructed from the *r-th* order ENO schemes to obtain $(r+1)$-*th* order accuracy. Then in 1996, Jiang and Shu [7] proposed a framework to construct finite difference WENO schemes from the *r-th* order (in L^1-norm sense) ENO schemes to get $(2r-1)$-*th* order accuracy. It gave a new way of measuring the smoothness indicators and emulated the ideas of minimizing the total variation of the approximation in [14]. Recently, Zhang and Shu [16] constructed the third-order WENO scheme on three-dimensional tetrahedral meshes. A key idea in WENO schemes is the exploitation of a linear combination of lower order fluxes or reconstruction to obtain a higher order approximation. Both ENO and WENO schemes use the idea of adaptive stencils to automatically achieve high order accuracy and non-oscillatory property near discontinuities. For the system case, WENO schemes are based on local characteristic decompositions and flux splitting to avoid spurious oscillations. They have been used successfully in many applications, especially for problems containing shocks and/or complicated smooth solution structures, such as compressible turbulence simulations and aeroacoustics.

Wave-like phenomena or highly oscillatory problems are encountered quite often in nature. However, little attention has been paid to trigonometric essentially non-oscillatory schemes, which appear to be more suitable for the simulation of such problems. In 1976, Baron [1] studied trigonometric interpolation and presented certain Neville-like algorithms. Muhlbach [10–13] studied general basis function, including trigonometric interpolation in Newton form. However, their work cannot be applied to ENO type schemes directly. The methodology used cannot obey the rule of adding one interpolation point to the stencil once a time but two. Christofi [3] provided a new trigonometric reconstruction methodology that can add interpolation points one by one and can also possess necessary symmetries to be used in ENO schemes.

In this paper, following the ideas in [3,7,14], we construct a kind of finite difference TWENO scheme which is of 5-*th* order accurate. The main differences between [3] and this work are the way of measuring the smoothness of the trigonometric polynomials and the form of the reconstruction that the schemes are ultimately based on. A Newton form is employed in [3], but a Lagrange form used in this work that seems suitable for improving the WENO type schemes. Compared to [3], the new scheme with the same stencils can achieve an even higher order of accuracy in smooth regions and less oscillations in discontinuous regions. Compared to the WENO schemes of [7] and [14], one major advantage of the new TWENO scheme is its good performance for the wave-like and highly oscillatory problems.

The organization of this paper is as follows. In Section 2, we review and construct

finite difference TWENO schemes. In Section 3, we present extensive numerical tests to verify the accuracy and stability of the proposed schemes. Concluding remarks are given in Section 4.

2 Finite difference TWENO schemes

In this section we consider one-dimensional hyperbolic conservation laws (1.1). The semi-discretization is written in the form:

$$\frac{du}{dt}=L(u), \tag{2.1}$$

where $L(u)$ is the high order spatial discrete representation of $-f_x(u)$.

For simplicity of presentation, we assume that the mesh is uniformly distributed into several cells $I_i=[x_{i-1/2},x_{i+1/2}]$, with the cell size $x_{i+1/2}-x_{i-1/2}=h$, and cell centers $x_i=\frac{1}{2}(x_{i+1/2}+x_{i-1/2})$. We also denote $u_i(t)=u(x_i,t)$. Then, the right hand side of (2.1) can be written as in [7]:

$$L(u_i(t))=-\frac{1}{h}(\hat{f}_{i+1/2}-\hat{f}_{i-1/2}), \tag{2.2}$$

where $\hat{f}_{i+1/2}$ is a numerical flux which is a high order approximation of the flux $f(u)$ at the boundary $x_{i+1/2}$ of cell I_i. If we take the numerical flux $\hat{f}_{i+1/2}$ to be the $(2r+1)$-*th* order approximation to $v_{i+1/2}=v(x_{i+1/2})$, where $v(x)$ is defined implicitly by:

$$f(u(x))=\frac{1}{h}\int_{x-h/2}^{x+h/2}v(\xi)d\xi, \tag{2.3}$$

then the right-hand side of (2.2) is the $(2r+1)$-*th* order approximation to $-f_x(u)$ at $x=x_i$.

For a general flux, we split it into two parts:

$$f(u)=f^+(u)+f^-(u),\quad \text{with}\quad \frac{df^+(u)}{du}\geq 0\quad\text{and}\quad \frac{df^-(u)}{du}\leq 0.$$

In this paper, we use the simplest Lax-Friedrichs splitting:

$$f^{\pm}(u)=\frac{1}{2}(f(u)\pm\alpha u), \tag{2.4}$$

where α is taken as $\alpha=\max_u|f'(u)|$ over the whole range of u. Let $\hat{f}^+_{i+1/2}$ and $\hat{f}^-_{i+1/2}$ be the numerical fluxes at $x_{i+1/2}$ obtained from (2.3) for positive and negative parts of $f(u)$, respectively. We can define $\hat{f}_{i+1/2}=\hat{f}^+_{i+1/2}+\hat{f}^-_{i+1/2}$.

Now, we describe the procedure of reconstruction of $\hat{f}^+_{i+1/2}$. From the definition of $v(x)$ in (2.3) for $f^+(u)$, we have:

$$f^+(u_i)=\frac{1}{h}\int_{x_{i-1/2}}^{x_{i+1/2}}v(\xi)d\xi=v_i,$$

where v_i is the cell average of $v(x)$ on the I_i.

Step 1. We choose the following stencil: $T=\{I_{i-r},\cdots,I_{i+r}\}$. We define the primitive function of $v(x)$ [14]:

$$V(x)=\int_{-\infty}^{x} v(\xi)d\xi, \tag{2.5}$$

$$V(x_{i+1/2})=\sum_{j=-\infty}^{i}\int_{I_j} v(\xi)d\xi=h\sum_{j=-\infty}^{i} v_j. \tag{2.6}$$

Thus, with the cell averages $\{v_j\}$, we know the primitive function $V(x)$ at the $x_{i+1/2}$ exactly. If we denote the proper trigonometric polynomial $P(x)$ that interpolates $V(x)$ at the $(2r+2)$ points $x_{i-(2r+1)/2},\cdots,x_{i+(2r+1)/2}$ and denote its derivative by $p(x)=P'(x)$, then it is easy to verify that:

$$\begin{aligned}\frac{1}{h}\int_{I_j} p(\xi)d\xi&=\frac{1}{h}\int_{I_j} P'(\xi)d\xi=\frac{1}{h}\big(P(x_{j+1/2})-P(x_{j-1/2})\big)\\&=\frac{1}{h}\big(V(x_{j+1/2})-V(x_{j-1/2})\big)\\&=\frac{1}{h}\left(\int_{-\infty}^{x_{j+1/2}} v(\xi)d\xi-\int_{-\infty}^{x_{j-1/2}} v(\xi)d\xi\right)\\&=\frac{1}{h}\int_{I_j} v(\xi)d\xi=v_j,\quad j=i-r,\cdots,i+r.\end{aligned} \tag{2.7}$$

This implies that $p(x)$ is the desired trigonometric polynomial. Next, we use the Lagrange form of the interpolating trigonometric polynomial [3]:

$$\begin{aligned}l_n(x)=&\cos\left(\frac{1}{2}(x-x_{i+(2n-(2r+3))/2})\right)\\&\times\prod_{k=1,k\neq n}^{2r+2}\frac{\sin\left(\frac{1}{2}(x-x_{i+(2k-(2r+3))/2})\right)}{\sin\left(\frac{1}{2}(x_{i+(2n-(2r+3))/2}-x_{i+(2k-(2r+3))/2})\right)},\quad n=1,\cdots,2r+2,\end{aligned} \tag{2.8a}$$

$$P(x)=\sum_{n=1}^{2r+2} V(x_{i-(2r+3-2n)/2})l_n(x). \tag{2.8b}$$

One can verify that the following relations are valid:

$$\sum_{n=1}^{2r+2} l_n(x)=1, \tag{2.9a}$$

$$P(x)-V(x_{i-(2r+1)/2})=\sum_{n=1}^{2r+2}\Big(V(x_{i+(2n-(2r+3))/2})-V(x_{i-(2r+1)/2})\Big)l_n(x), \tag{2.9b}$$

$$V(x_{i+(2n-(2r+3))/2})-V(x_{i-(2r+1)/2})=h\sum_{j=-r}^{n-2r} v_{i+j},\quad n\geq r. \tag{2.9c}$$

Taking the derivative on both sides of the above relation (2.9b), we can obtain

$$p(x)=\frac{\partial}{\partial x}\left(\sum_{n=1}^{2r+2}\Big(V(x_{i+(2n-(2r+3))/2})-V(x_{i-(2r+1)/2})\Big)l_n(x)\right). \tag{2.10}$$

Step 2. In this step, we set $n=1,\cdots,r+1$. We divide the stencil T into $(r+1)$ smaller stencils: $S_1=\{I_{i-r},\cdots,I_i\},\cdots, S_{r+1}=\{I_i,\cdots,I_{i+r}\}$. Just as in Step 1, we can construct the trigonometric polynomials $p_n(x)$ in the associated smaller stencils to approximate the function $v(x)$.

The values of the functions $p_n(x)$ at the point $x_{i+1/2}$ of cell I_i can be written as a linear combination of $\{v_j\}$. For example, when $r=1$, we have

$$p_1(x_{i+1/2})=\frac{h}{2\sin(h)}(-v_{i-1}+v_i+2v_i\cos(h)), \tag{2.11a}$$

$$p_2(x_{i+1/2})=\frac{h}{2\sin(h)}(v_i+v_{i+1}). \tag{2.11b}$$

When $r=2$, we have

$$\begin{aligned}p_1(x_{i+1/2})=&-\beta(h)\Big(2v_{i-1}-3v_i+(3v_{i-1}-v_{i-2}-5v_i)\cos(h)\\&+(v_{i-1}-v_{i-2}-3v_i)\cos(2h)+v_{i-1}\cos(3h)\Big),\end{aligned} \tag{2.12a}$$

$$p_2(x_{i+1/2})=\beta(h)\Big(2v_i+v_{i+1}+(-v_{i-1}+2v_i+v_{i+1})\cos(h)+v_i\cos(2h)\Big), \tag{2.12b}$$

$$p_3(x_{i+1/2})=\beta(h)\Big(v_i+2v_{i+1}+(v_i+2v_{i+1}-v_{i+2})\cos(h)+v_{i+1}\cos(2h)\Big). \tag{2.12c}$$

where and also in (2.13) below, $\beta(h)=h/[\sin(2h)(2+4\cos(h))]$.

When $r=3$, we have

$$\begin{aligned}p_1(x_{i+1/2})=&-\beta(h)\Big(5v_{i-1}-3v_{i-2}+v_{i-3}-5v_i+2(4v_{i-1}-2v_{i-2}+v_{i-3}-5v_i)\cos(h)\\&+(6v_{i-1}-4v_{i-2}-6v_i)\cos(2h)+(2v_{i-1}-2v_{i-2}-4v_i)\cos(3h)+2v_{i-1}\cos(4h)\Big),\end{aligned} \tag{2.13a}$$

$$\begin{aligned}p_2(x_{i+1/2})=&\beta(h)\Big(-v_{i-1}+v_{i-2}+3v_i+v_{i+1}+(-2v_{i-1}+6v_i+2v_{i+1})\cos(h)\\&-2(v_{i-1}-v_i)\cos(2h)+2v_i\cos(3h)\Big),\end{aligned} \tag{2.13b}$$

$$p_3(x_{i+1/2})=\beta(h)\Big(-v_{i-1}+v_i+v_{i+1}-v_{i+2}+4(v_i+v_{i+1})\cos(h)+2(v_i+v_{i+1})\cos(2h)\Big), \tag{2.13c}$$

$$\begin{aligned}p_4(x_{i+1/2})=&\beta(h)\Big(v_i+3v_{i+1}-v_{i+2}+v_{i+3}+2(v_i+3v_{i+1}-v_{i+2})\cos(h)\\&+2(v_{i+1}-v_{i+2})\cos(2h)+2v_{i+1}\cos(3h)\Big).\end{aligned} \tag{2.13d}$$

Then we find the linear weights, such that the following equation is valid:

$$p(x_{i+1/2}) = \sum_{n=1}^{r+1} \gamma_n p_n(x_{i+1/2}).$$

For example, when $r=1$, we have

$$\gamma_1 = \frac{\cos(h)}{1+2\cos(h)}, \quad \gamma_2 = \frac{1+\cos(h)}{1+2\cos(h)}. \tag{2.14}$$

When $r=2$, we have

$$\gamma_1 = \frac{1}{2(1+2\cos(h)+\cos(2h)+\cos(3h))}, \tag{2.15a}$$

$$\gamma_2 = \frac{\cos(h)+\cos(2h)+\cos(3h)}{1+2\cos(h)+\cos(2h)+\cos(3h)}, \tag{2.15b}$$

$$\gamma_3 = \frac{1+2\cos(h)}{2(1+2\cos(h)+\cos(2h)+\cos(3h))}. \tag{2.15c}$$

When $r=3$, we have

$$\gamma_1 = \frac{1}{s}\cos(2h), \tag{2.16a}$$

$$\gamma_2 = \frac{1}{s}\Big(2+3\cos(h)+2\cos(2h)+2\cos(3h)+\cos(4h)+\cos(5h)+\cos(6h)\Big), \tag{2.16b}$$

$$\gamma_3 = \frac{1}{s}\Big(2+4\cos(h)+4\cos(2h)+3\cos(3h)+3\cos(4h)+\cos(5h)+\cos(6h)\Big), \tag{2.16c}$$

$$\gamma_4 = \frac{1}{s}\Big(1+\cos(h)+\cos(2h)+\cos(3h)\Big), \tag{2.16d}$$

where $s=5+8\cos(h)+8\cos(2h)+6\cos(3h)+4\cos(4h)+2\cos(5h)+2\cos(6h)$.

For smaller stencils S_n, we compute the smoothness indicators, denoted by β_n, which measure how smooth the functions $p_n(x)$ are in the target cell I_i. The smaller these smoothness indicators, the smoother the functions are in the target cell. We use the same recipe for the smoothness indicators as in [7]:

$$\beta_n = \sum_{\alpha=1}^{r} \int_{I_i} h^{2\alpha-1} \Big(\frac{d^\alpha p_n(x)}{dx^\alpha}\Big)^2 dx. \tag{2.17}$$

We then compute the nonlinear weights based on the linear weights and the smoothness indicators [14]:

$$\omega_n = \frac{\overline{\omega}_n}{\sum_{k=1}^{r+1}\overline{\omega}_k}, \quad \overline{\omega}_n = \frac{\gamma_n}{\sum_{k=1}^{r+1}(\varepsilon+\beta_k)^2}, \tag{2.18}$$

where γ_n are the linear weights determined in the above step, and ε is a small positive number to avoid overflow. We use $\varepsilon = 10^{-6}$ in all the computations in this paper. The

final reconstruction of the numerical flux which approximates the positive part of $f(u)$ at $x=x_{i+1/2}$ is given by

$$\hat{f}^{+}_{i+1/2}=\sum_{n=1}^{r+1}\omega_n p_n\left(x_{i+\frac{1}{2}}\right).$$

The procedure for the reconstruction of $\hat{f}^{-}_{i+1/2}$ is a mirror symmetry to that of $\hat{f}^{+}_{i+1/2}$ with respect to $x_{i+1/2}$.

Step 3. The semidiscrete scheme (2.1) is then discretized in time by a TVB Runge-Kutta method [15]; for example the 4*th*-order version is given by

$$\begin{cases} u^{(1)}=u^n+\dfrac{1}{2}\Delta tL(u^n), \\ u^{(2)}=u^n+\dfrac{1}{2}\Delta tL(u^{(1)}), \\ u^{(3)}=u^n+\Delta tL(u^{(2)}), \\ u^{n+1}=-\dfrac{1}{3}u^n+\dfrac{1}{3}u^{(1)}+\dfrac{2}{3}u^{(2)}+\dfrac{1}{3}u^{(3)}+\dfrac{1}{6}\Delta tL(u^{(3)}). \end{cases} \tag{2.19}$$

Remark 2.1. For systems of conservation laws, such as the Euler equations of gas dynamics, all of the reconstructions are performed in the local characteristic directions to avoid oscillations.

Remark 2.2. For two-dimensional problems, the reconstructions can be performed by a dimension-by-dimension fashion.

3 Numerical tests

In this section we present only the results of numerical tests of the 5*th*-order scheme described in the previous section.

Example 3.1. We solve the following linear scalar equation:

$$\begin{cases} u_t+u_x=0, & -\pi\le x\le\pi,\ t>0, \\ u(x,0)=u_0(x), \end{cases} \tag{3.1}$$

with the initial condition $u(x,0)=\sin(x)$ and periodic boundary condition. We compute the solution up to $t=1$. The errors and numerical orders of accuracy of the TWENO and WENO [7] schemes are shown in Table 1 and the numerical error against CPU time is plotted in Fig. 1. Table 1 demonstrates that the theoretical order is actually achieved and TWENO can exhibit smaller errors than WENO for the same mesh level. Fig. 1 shows that TWENO needs less CPU time than WENO does to obtain the same quantities of L^1 and L^∞ errors. It is therefore concluded that TWENO is more efficient than WENO for this test case.

Table 1: Example 3.1: L^1 and L^∞ errors. TWENO scheme and WENO scheme at $t=1$.

	TWENO scheme				WENO scheme			
grid points	L^1 error	order	L^∞ error	order	L^1 error	order	L^∞ error	order
10	5.72E-5		8.71E-5		5.72E-3		1.22E-2	
20	1.34E-6	5.41	2.10E-6	5.36	2.57E-4	4.47	4.66E-4	4.70
40	3.43E-8	5.28	5.42E-8	5.28	7.60E-6	5.08	1.59E-5	4.86
80	8.69E-10	5.30	1.36E-9	5.30	2.27E-7	5.06	4.90E-7	5.02
160	2.17E-11	5.32	3.41E-11	5.32	6.95E-9	5.03	1.43E-8	5.09
320	5.40E-13	5.33	8.48E-13	5.33	2.16E-10	5.00	4.01E-10	5.15
640	1.34E-14	5.33	2.10E-14	5.33	6.71E-12	5.01	1.16E-11	5.10

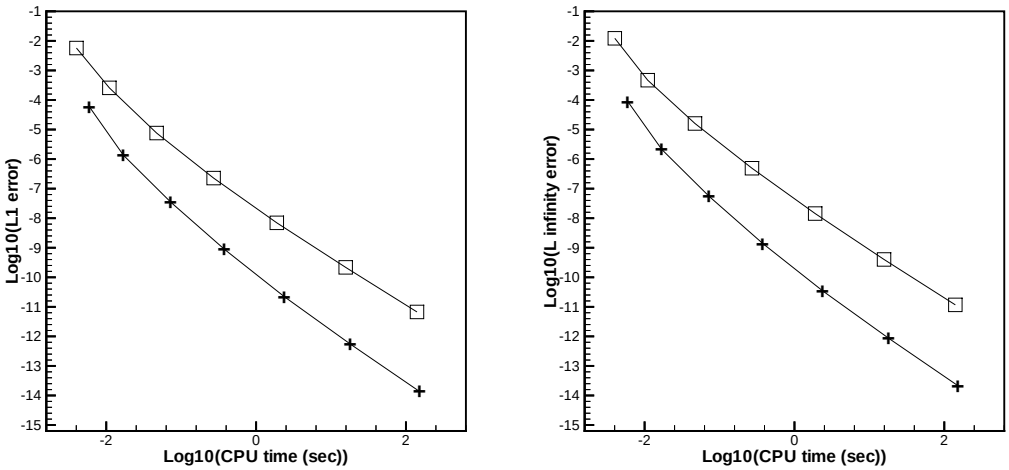

Figure 1: Example 3.1: computing time and error. Plus signs with a solid line denote the results of TWENO scheme; squares with a solid line denote the results of WENO scheme.

Example 3.2. We solve the following linear scalar equation:

$$\begin{cases} u_t+u_x=0 & -\infty<x<\infty,\ t>0, \\ u(x,0)=u_0(x), \end{cases} \tag{3.2}$$

with the initial condition $u(x,0)=x^5$. In this test case, we adopt a large computational domain at the initial time stage, and shrink computational domain at each time step to avoid implementation of boundary conditions. We compute the solutions up to $t=1$. The errors and numerical orders of accuracy of the TWENO and WENO of [7] are shown in Table 2, and the numerical error against CPU time is plotted in Fig. 2. It is observed in Table 2 that the theoretical order is actually achieved but TWENO may exhibit slightly bigger errors than WENO. Fig. 2 shows that TWENO needs more CPU time than WENO does to obtain the same quantities of L^1 and L^∞ errors, as the function basis of the solutions are in the algebraic polynomial space and not in the trigonometric polynomial

Table 2: Example 3.2: L^1 and L^∞ errors. TWENO scheme and WENO scheme at $t=1$.

	TWENO scheme				WENO scheme			
grid points	L^1 error	order	L^∞ error	order	L^1 error	order	L^∞ error	order
20	7.45E-2		2.56E-1		6.23E-2		2.21E-1	
40	3.05E-3	4.61	1.34E-2	4.25	2.40E-3	4.70	1.27E-2	4.12
80	1.10E-4	4.79	7.78E-4	4.11	8.69E-5	4.79	7.65E-4	4.06
160	2.98E-6	5.21	3.24E-5	4.59	2.25E-6	5.27	3.20E-5	4.58
320	9.02E-8	5.05	1.21E-6	4.74	6.92E-8	5.03	1.20E-6	4.74
640	2.85E-9	4.98	3.52E-8	5.11	2.22E-9	4.96	3.46E-8	5.11

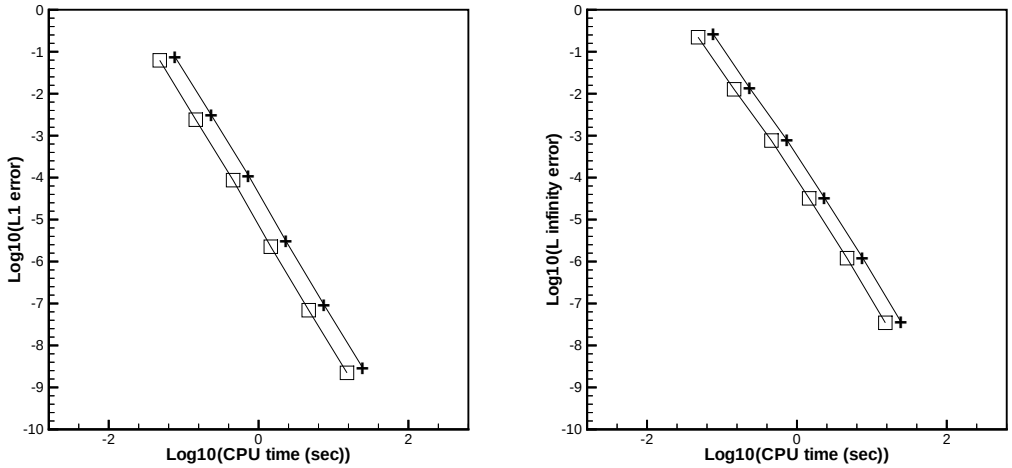

Figure 2: Example 3.2: computing time and error. Plus signs with a solid line denote the results of TWENO scheme; squares with a solid line denote the results of WENO scheme.

space any more. However, it is observed that the differences given by the two schemes are very small.

Example 3.3. We solve the following linear scalar equation:

$$\begin{cases} u_t+u_x+u_y=0 & -\pi\le x\le\pi,\ -\pi\le y\le\pi,\ t>0, \\ u(x,y,0)=u_0(x,y,), \end{cases} \tag{3.3}$$

with the initial condition $u(x,y,0)=\sin(x+y)$ and periodic boundary conditions. We compute the solution up to $t=1$. The errors and numerical orders of accuracy given by the TWENO and WENO [7] schemes are shown in Table 3, and the numerical error against CPU time is plotted in Fig. 3. We can observe that the theoretical order is actually achieved and the TWENO scheme can get better results and is more efficient than WENO in this test case.

Table 3: Example 3.3: L^1 and L^∞ errors. TWENO scheme and WENO scheme at $t=1$.

	TWENO scheme				WENO scheme			
grid points	L^1 error	order	L^∞ error	order	L^1 error	order	L^∞ error	order
10×10	1.12E-4		1.73E-4		1.22E-2		1.98E-2	
20×20	2.74E-6	5.35	4.34E-6	5.31	4.76E-4	4.68	8.89E-4	4.47
40×40	7.04E-8	5.28	1.10E-7	5.29	1.47E-5	5.01	3.04E-5	4.86
80×80	1.75E-9	5.32	2.76E-9	5.31	4.45E-7	5.04	9.44E-7	5.00
160×160	4.35E-11	5.33	6.84E-11	5.33	1.38E-8	5.00	2.84E-8	5.05
320×320	1.08E-12	5.33	1.69E-12	5.33	4.32E-10	5.00	8.08E-10	5.13
640×640	2.68E-14	5.33	4.21E-14	5.33	1.34E-11	5.00	2.33E-11	5.11

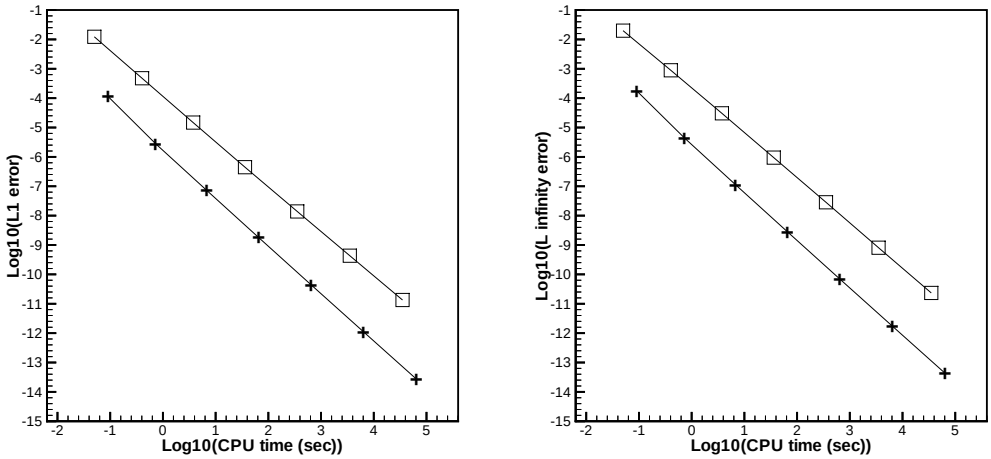

Figure 3: Example 3.3: computing time and error. Plus signs with a solid line denote the results of TWENO scheme; squares with a solid line denote the results of WENO scheme.

Example 3.4. We solve the 1D Euler equations

$$\frac{\partial}{\partial t}\begin{pmatrix}\rho\\ \rho u\\ E\end{pmatrix}+\frac{\partial}{\partial x}\begin{pmatrix}\rho u\\ \rho u^2+p\\ u(E+p)\end{pmatrix}=0, \tag{3.4}$$

where ρ is density, u is the velocity in x-direction, E is total energy, and p is pressure. The initial conditions are:

$$\rho(x,0)=1+0.2\sin(x),\quad u(x,0)=1,\quad p(x,0)=1,$$

and boundary conditions are periodic. We compute the density solution up to $t=2$.

The errors and numerical orders of accuracy of the TWENO and WENO [7] schemes are shown in Table 4 and the numerical error against CPU time is presented in Fig. 4. We can observe that the theoretical order is actually achieved and the TWENO scheme can get better results. It is observed from Fig. 4 that TWENO is more efficient than WENO in this test case at the fine mesh levels.

Table 4: Example 3.4: L^1 and L^∞ errors. TWENO scheme and WENO scheme at $t=2$.

	TWENO scheme				WENO scheme			
grid points	L^1 error	order	L^∞ error	order	L^1 error	order	L^∞ error	order
10	4.86E-3		1.22E-2		2.56E-3		4.18E-3	
20	5.29E-5	6.52	1.83E-4	6.05	1.06E-4	4.58	1.92E-4	4.43
40	7.35E-7	6.16	4.55E-6	5.33	3.26E-6	5.03	6.45E-6	4.89
80	9.48E-9	6.27	9.64E-8	5.56	9.56E-8	5.09	1.88E-7	5.09
160	8.73E-11	6.76	1.03E-9	6.53	2.92E-9	5.03	5.22E-9	5.17
320	1.00E-12	6.44	5.76E-12	7.49	8.76E-11	5.05	1.48E-10	5.13
640	1.41E-14	6.14	5.04E-14	6.83	2.40E-12	5.18	3.95E-12	5.23

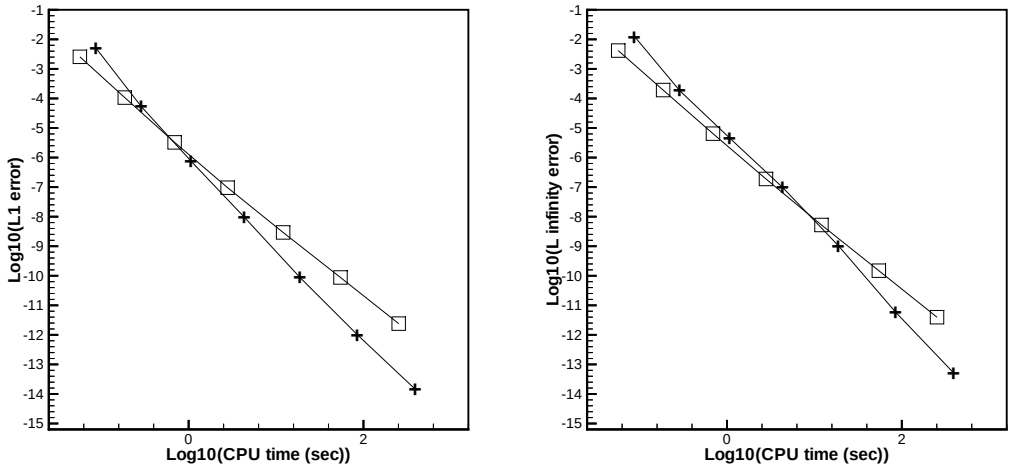

Figure 4: Example 3.4: computing time and error. Plus signs with a solid line denote the results of TWENO scheme; squares with a solid line denote the results of WENO scheme.

Example 3.5. We solve the 2D Euler equations

$$\frac{\partial}{\partial t}\begin{pmatrix}\rho\\ \rho u\\ \rho v\\ E\end{pmatrix}+\frac{\partial}{\partial x}\begin{pmatrix}\rho u\\ \rho u^2+p\\ \rho uv\\ u(E+p)\end{pmatrix}+\frac{\partial}{\partial y}\begin{pmatrix}\rho v\\ \rho uv\\ \rho v^2+p\\ v(E+p)\end{pmatrix}=0, \tag{3.5}$$

where ρ is density; u and v are the velocity components in the x- and y-directions, respectively; E is total energy; and p is pressure. The initial conditions are:

$$\rho(x,y,0)=1+0.2\sin(x+y),\quad u(x,y,0)=0.7,\quad v(x,y,0)=0.3,\quad p(x,y,0)=1,$$

and boundary conditions are periodic in both directions. We compute the density solution up to $t=2$. The errors and numerical orders of accuracy of the TWENO and WENO [7] schemes are shown in Table 5 and the numerical error against CPU time is

Table 5: Example 3.5: L^1 and L^∞ errors. TWENO scheme and WENO scheme at $t=2$.

	TWENO scheme				WENO scheme			
grid points	L^1 error	order	L^∞ error	order	L^1 error	order	L^∞ error	order
10×10	7.98E-3		2.27E-2		2.50E-3		4.16E-3	
20×20	4.71E-5	7.40	1.86E-4	6.92	1.02E-4	4.61	1.90E-4	4.45
40×40	6.72E-7	6.12	4.30E-6	5.43	3.12E-6	5.03	6.33E-6	4.91
80×80	7.64E-9	6.45	7.15E-8	5.91	9.32E-8	5.06	1.79E-7	5.14
160×160	6.83E-11	6.80	5.26E-10	7.08	2.78E-9	5.06	4.60E-9	5.28
320×320	7.65E-13	6.48	2.93E-12	7.48	7.66E-11	5.18	1.23E-10	5.22

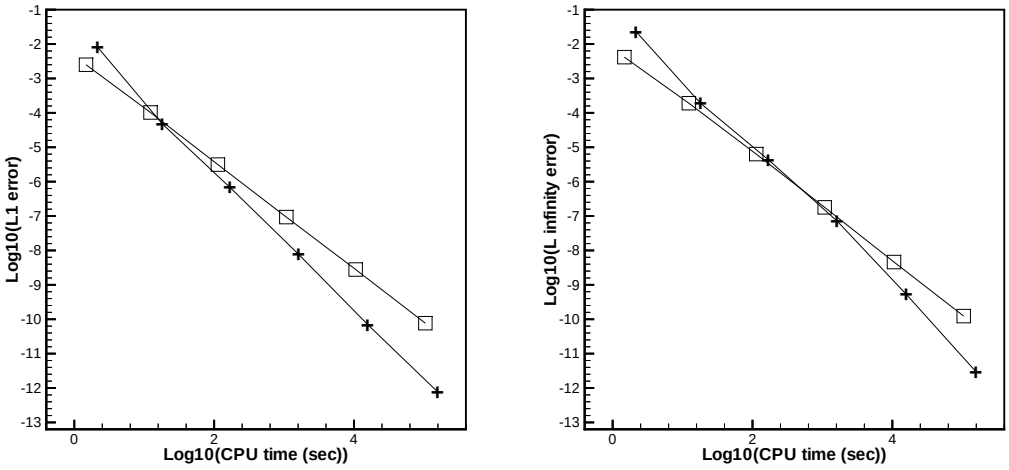

Figure 5: Example 3.5: computing time and error. Plus signs with a solid line denote the results of TWENO scheme; squares with a solid line denote the results of WENO scheme.

presented in Fig. 5. We can observe that the theoretical order is actually achieved and the TWENO scheme can get better results. Moreover, TWENO is more efficient than WENO in this test case at the fine mesh levels.

Example 3.6. We use function values $H(x_{j+\frac{1}{2}})$ from each of the test functions H listed in [3]:

$$H(x)=\begin{cases} x, & 0\le x<\pi, \\ 2\pi-x, & \pi\le x<2\pi, \end{cases} \qquad H'(x)=\begin{cases} 1, & 0\le x<\pi, \\ -1, & \pi\le x<2\pi, \end{cases} \tag{3.6a}$$

$$H(x)=\begin{cases} x+\sin^4(x), & 0\le x<\pi, \\ 2\pi-(x+\sin^4(x)), & \pi\le x<2\pi, \end{cases}$$

$$H'(x)=\begin{cases} 1+4\sin^3(x)\cos(x), & 0\le x<\pi, \\ -(1+4\sin^3(x)\cos(x)), & \pi\le x<2\pi. \end{cases} \tag{3.6b}$$

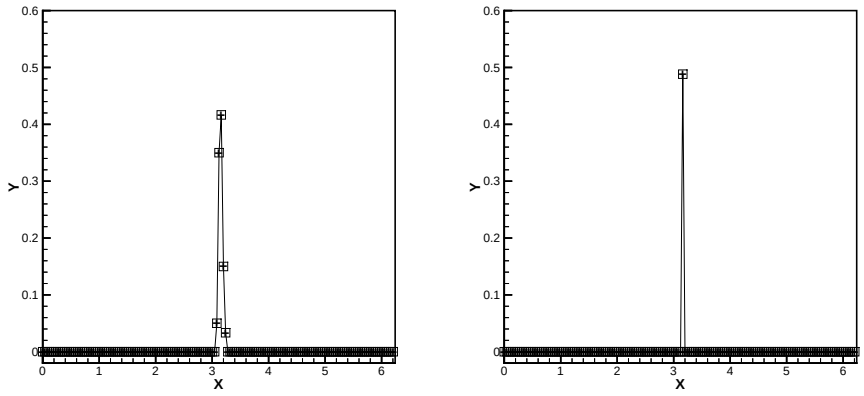

Figure 6: Error comparisons of the function $H'(x)$ of the form (3.6a). Left: the pointwise errors $|H'(x_{j+1/2})-\hat{f}_{j+1/2}|$ of the 4th degree trigonometric polynomial reconstruction to the derivative $H'(x)$ are depicted by plus signs and a solid line; those of the 4th degree algebraic polynomial are depicted by squares and a solid line. Right: the pointwise errors $|H'(x_{j+1/2})-\hat{f}_{j+1/2}|$ of the TWENO scheme are depicted by plus signs and a solid line; those of the WENO scheme are given by squares and a solid line.

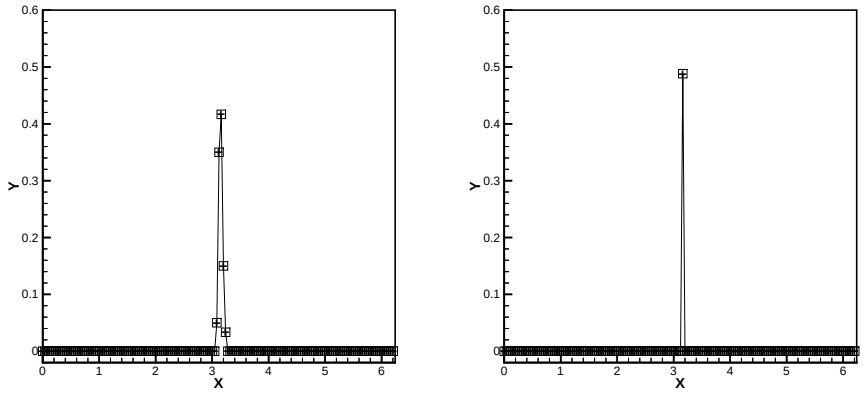

Figure 7: Same as Fig. 6 except for $H'(x)$ given by (3.6b).

Moreover, for $0\leq x<2\pi$:

$$H(x)=\sin^7(x), \qquad H'(x)=7\sin^6(x)\cos(x), \tag{3.7a}$$

$$H(x)=0.01\sin(3x), \quad H'(x)=0.03\cos(3x), \tag{3.7b}$$

$$H(x)=x^2\cos(4x), \qquad H'(x)=2x\cos(4x)-4x^2\sin(4x), \tag{3.7c}$$

$$H(x)=e^{\sin(x)}, \qquad H'(x)=\cos(x)e^{\sin(x)}. \tag{3.7d}$$

For every example, we present two graphs as follows: The first graph shows the pointwise errors $|H'(x_{j+1/2})-\hat{f}_{j+1/2}|$ obtained by using the two approximations with 4th degree trigonometric and algebraic polynomial reconstructions, respectively. The second graph shows the pointwise errors $|H'(x_{j+1/2})-\hat{f}_{j+1/2}|$, obtained by using the TWENO and WENO schemes, respectively. These plots are displayed in Figs. 6 through 11. In

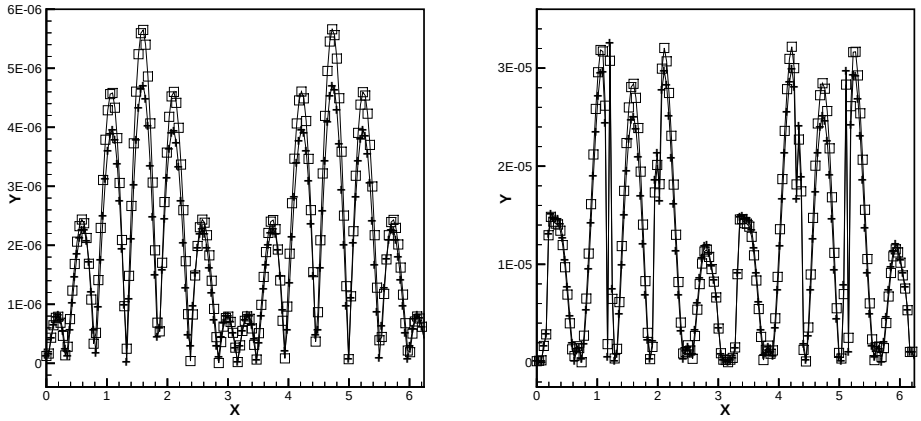

Figure 8: Same as Fig. 6 except for $H'(x)$ given by (3.7a).

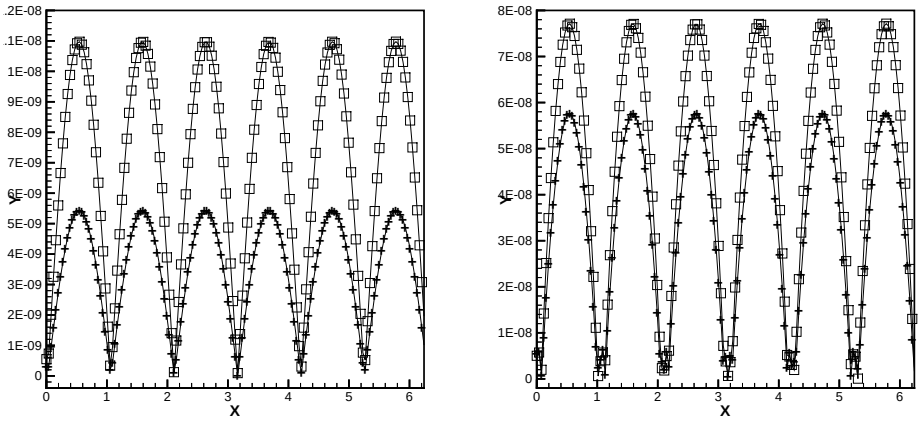

Figure 9: Same as Fig. 6 except for $H'(x)$ given by (3.7b).

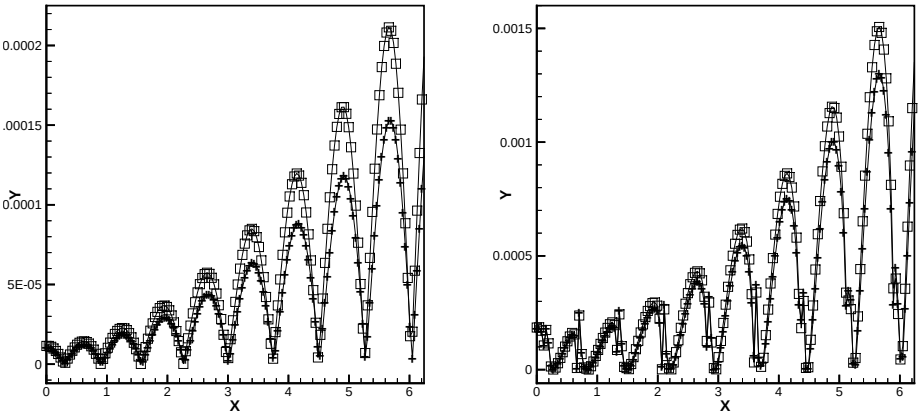

Figure 10: Same as Fig. 6 except for $H'(x)$ given by (3.7c).

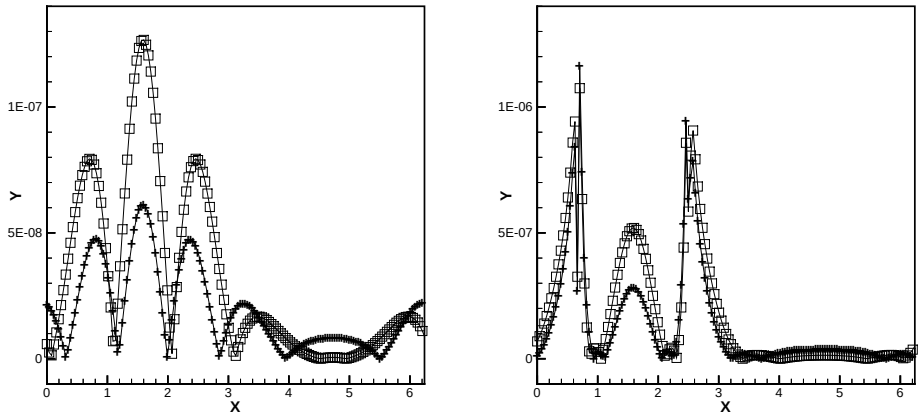

Figure 11: Same as Fig. 6 except for $H'(x)$ given by (3.7d).

each example, we let $x_j = 2\pi j/N$ and set $N = 160$. As expected, our numerical experiments for the test functions (3.6) and (3.7a)-(3.7c) demonstrate that the results using the trigonometric polynomials in the reconstruction procedure are better than those obtained by using the algebraic polynomials. However, when exponential functions, see (3.7d), none of the scheme is uniformly better than the other one.

Example 3.7. We solve the 1D Euler equations with Riemann initial condition for the Lax problem:

$$(\rho,u,p)^T = \begin{cases} (0.445,0.698,3.528)^T, & x \le 0, \\ (0.5,0,0.571)^T, & x > 0. \end{cases} \tag{3.8}$$

For $t = 0.16$, we present in Fig. 12 the exact solution and the computed solutions for the density ρ obtained with the TWENO and WENO schemes, respectively, using 200

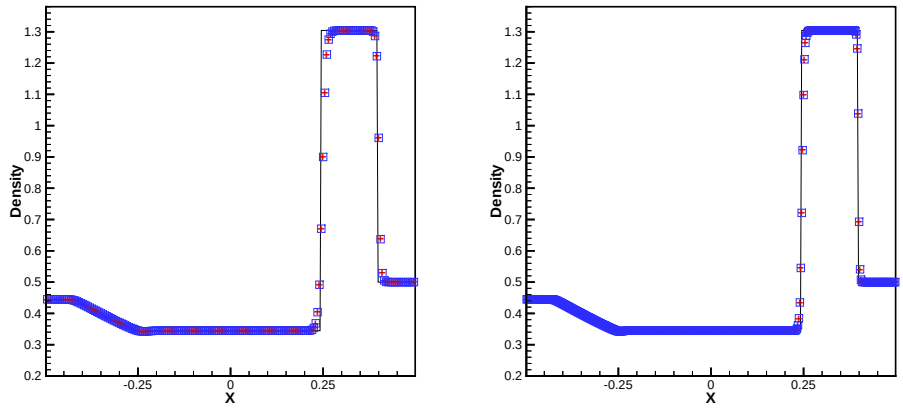

Figure 12: The Lax problem solved by TWENO scheme and WENO scheme at $t = 0.16$. Left: 200 grid points; Right: 400 grid points. Solid line: exact solution; plus signs: TWENO results; squares: WENO results.

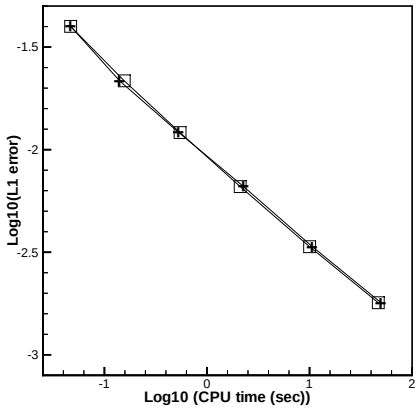

Figure 13: Computing time against error for the Lax problem. Plus signs with a solid line denote the TWENO results; and squares with a solid line denote the WENO results.

and 400 grid points. The corresponding L_1 error against CPU time graph is plotted in Fig. 13. The data points correspond to the cases using 50, 100, 200, 400, 800, and 1600 grid points in both TWENO and WENO schemes. It is observed that the computational results obtained with TWENO and WENO schemes are similar.

Example 3.8. We solve the 1D Euler equations with a moving Mach=3 shock interacting with sine waves in density:

$$(\rho,u,p)^T=\begin{cases}(3.857143,2.629369,10.333333)^T, & x<-4,\\ (1+0.2\sin(5x),0,1)^T, & x\geq -4.\end{cases} \tag{3.9}$$

For $t=1.8$, we present in Fig. 14 the computed density ρ along with a reference solution. The former is obtained with the TWENO and WENO schemes, respectively, using 200

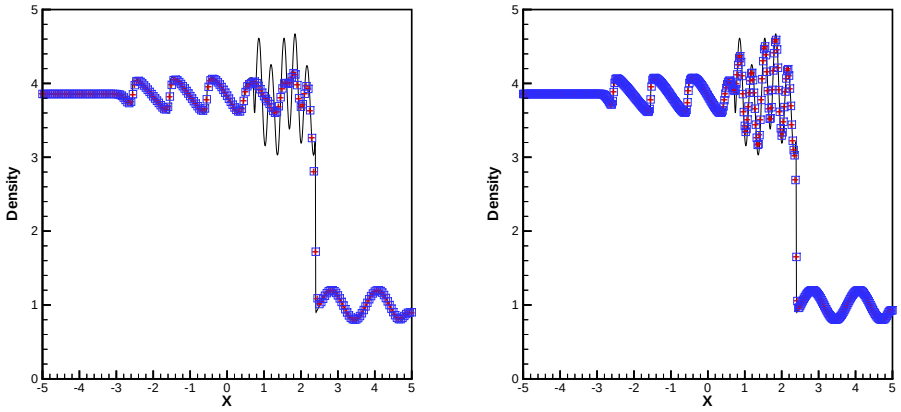

Figure 14: The shock density wave interaction problem solved by TWENO scheme and WENO scheme at $t=1.8$. Left: 200 grid points; Right: 400 grid points. Solid line: reference solution; plus signs: TWENO results; squares: WENO results.

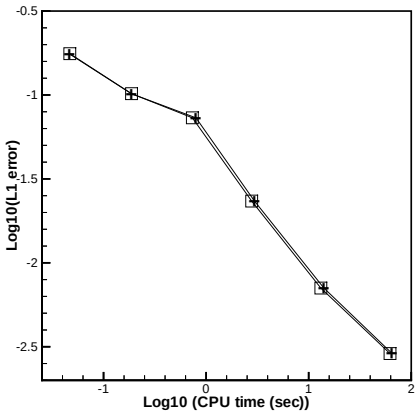

Figure 15: Computing time against error for the shock density wave interaction problem. Plus signs with a solid line denote the TWENO results; and squares with a solid line denote the WENO results.

and 400 grid points; and the latter is given by using the WENO scheme using 2000 grid points. We then provide the L_1 error against the CPU time in Fig. 15. The data points correspond to the cases using 50, 100, 200, 400, 800, and 1600 grid points for the TWENO and WENO schemes. The reference solution is obtained using WENO scheme with 16000 grid points. Again, it is observed that the numerical results obtained using the two schemes are similar.

Example 3.9. 2D Euler equations for the Riemann problem [2, 8]. We solve the Euler equations (3.5) in a computational domain of $[0,1]\times[0,1]$ and set the initial conditions as:

$$(\rho,u,v,p)^T=\begin{cases}(0.5313,0,0,0.4)^T, & x>0.5,\ y>0.5,\\ (1,0.7276,0,1)^T, & x<0.5,\ y>0.5,\\ (0.8,0,0,1)^T, & x<0.5,\ y<0.5,\\ (1,0,0.7276,1)^T, & x>0.5,\ y<0.5,\end{cases} \tag{3.10a}$$

$$(\rho,u,v,p)^T=\begin{cases}(1.1,0,0,1.1)^T, & x>0.5,\ y>0.5,\\ (0.5065,0.8939,0,0.35)^T, & x<0.5,\ y>0.5,\\ (1.1,0.8939,0.8939,1.1)^T, & x<0.5,\ y<0.5,\\ (0.5065,0,0.8939,0.35)^T, & x>0.5,\ y<0.5,\end{cases} \tag{3.10b}$$

$$(\rho,u,v,p)^T=\begin{cases}(1,0.1,0,1)^T, & x>0.5,\ y>0.5,\\ (0.5313,0.8276,0,0.4)^T, & x<0.5,\ y>0.5,\\ (0.8,0.1,0,0.4)^T, & x<0.5,\ y<0.5,\\ (0.5313,0.1,0.7276,0.4)^T, & x>0.5,\ y<0.5,\end{cases} \tag{3.10c}$$

$$(\rho,u,v,p)^T=\begin{cases}(0.5313,0.1,0.1,0.4)^T, & x>0.5,\ y>0.5,\\ (1.0222,-0.6179,0.1,1)^T, & x<0.5,\ y>0.5,\\ (0.8,0.1,0.1,1)^T, & x<0.5,\ y<0.5,\\ (1,0.1,0.8276,1)^T, & x>0.5,\ y<0.5,\end{cases} \tag{3.10d}$$

$$(\rho,u,v,p)^T=\begin{cases}(1,0.75,-0.5,1)^T, & x>0.5,\ y>0.5,\\ (2,0.75,0.5,1)^T, & x<0.5,\ y>0.5,\\ (1,-0.75,0.5,1)^T, & x<0.5,\ y<0.5,\\ (3,-0.75,-0.5,1)^T, & x>0.5,\ y<0.5.\end{cases} \tag{3.10e}$$

In Fig. 16, we show the computational results for density at (1) $t=0.25$, (2) $t=0.25$, (3) $t=0.3$, (4) $t=0.2$, (5) $t=0.3$, respectively. We can see that most of the flow features are captured well for all these Riemann problems.

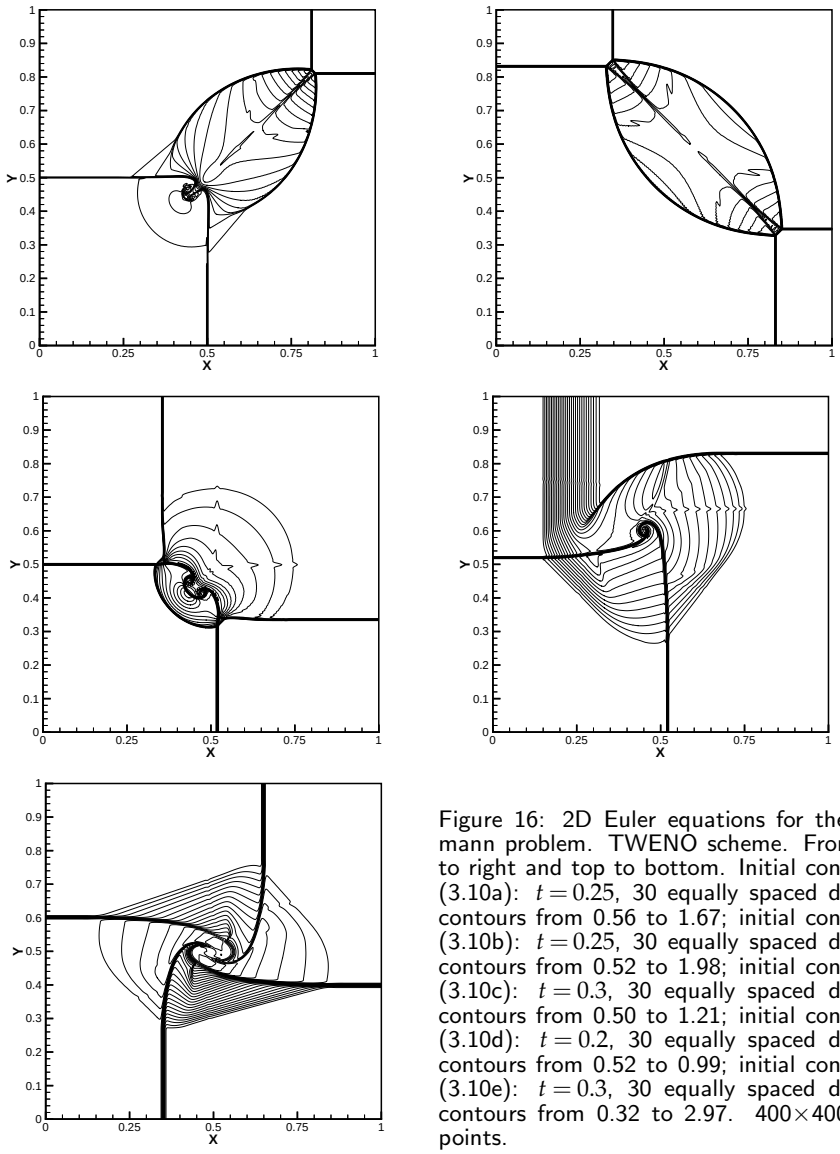

Figure 16: 2D Euler equations for the Riemann problem. TWENO scheme. From left to right and top to bottom. Initial condition (3.10a): $t=0.25$, 30 equally spaced density contours from 0.56 to 1.67; initial condition (3.10b): $t=0.25$, 30 equally spaced density contours from 0.52 to 1.98; initial condition (3.10c): $t=0.3$, 30 equally spaced density contours from 0.50 to 1.21; initial condition (3.10d): $t=0.2$, 30 equally spaced density contours from 0.52 to 0.99; initial condition (3.10e): $t=0.3$, 30 equally spaced density contours from 0.32 to 2.97. 400×400 grid points.

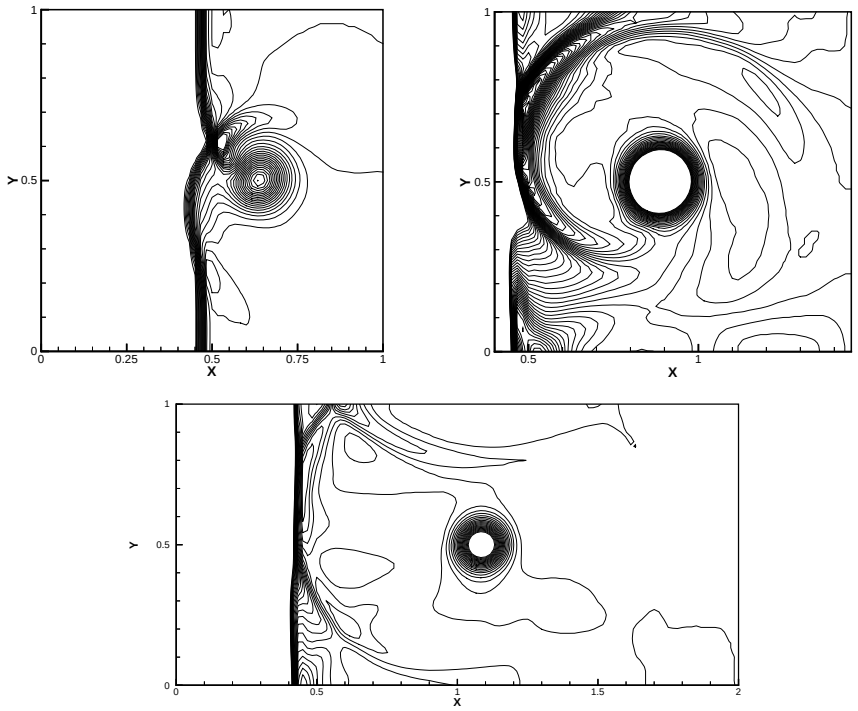

Figure 17: Shock vortex interaction problem. TWENO scheme. Top left: $t=0.35$, 30 equally spaced pressure contours from 1.02 to 1.33, region $[0,1]\times[0,1]$; Top right: $t=0.6$, 90 equally spaced pressure contours from 1.19 to 1.37, region $[0.4,1.45]\times[0,1]$; Bottom: $t=0.8$, 30 equally spaced pressure contours from 1.1 to 1.3, region $[0,2]\times[0,1]$. 120×60 grid points.

Example 3.10. Shock vortex interaction problem [7]. We solve the Euler equations (3.5) in a computational domain of $[0,2]\times[0,1]$. A stationary Mach 1.1 shock is positioned at $x=0.5$ and normal to the x-axis. The left state of the shock is $(\rho,u,v,p)^T=(1,\sqrt{\gamma},0,1)^T$. A small vortex is superposed to the flow left to the shock and the center is at $(x_c,y_c)=(0.25,0.5)$. We describe the vortex as a perturbation of the velocity, temperature and entropy of the mean flow and we denote it by the tilde values:

$$\begin{cases} \tilde{u}=\varepsilon\tau e^{\alpha(1-\tau^2)}\sin\theta, \quad \tilde{v}=-\varepsilon\tau e^{\alpha(1-\tau^2)}\cos\theta, \\ \tilde{T}=-\dfrac{(\gamma-1)\varepsilon^2 e^{2\alpha(1-\tau^2)}}{4\alpha\gamma}, \quad \tilde{S}=0, \end{cases} \tag{3.11}$$

where $\tau=r/r_c$ and $r=\sqrt{(x-x_c)^2+(y-y_c)^2}$. Here $\varepsilon=0.3$, $r_c=0.05$ and $\alpha=0.204$. The results are shown at $t=0.35$, $t=0.6$ and $t=0.8$. We present the numerical results in different regions in Fig. 17. We see that the method gives a good resolution for both vortex and shock. We also can see that the reflection is well captured at $t=0.8$.

Example 3.11. Double Mach reflection problem. We solve the Euler equations (3.5) in a computational domain of $[0,4]\times[0,1]$. A reflection wall lies at the bottom of the domain

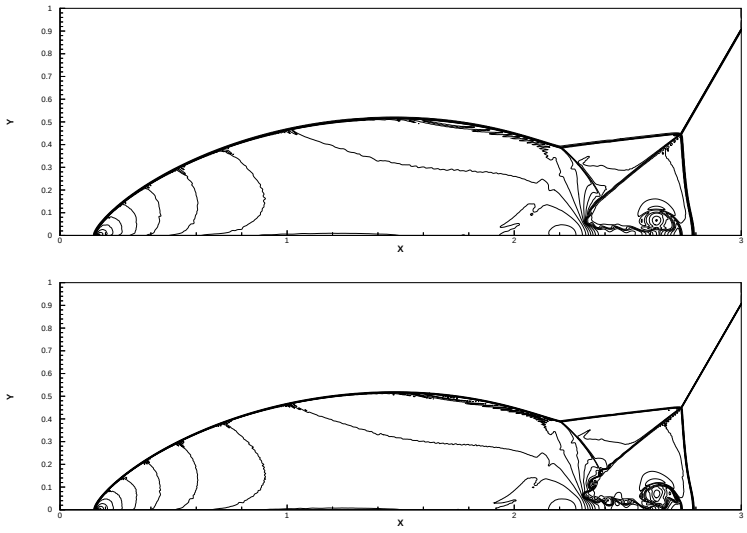

Figure 18: Double Mach reflection problem. TWENO scheme. $t=0.2$. 30 equally spaced density contours from 1.5 to 22.7. Top: 1600×400 grid points; Bottom: 2400×600 grid points.

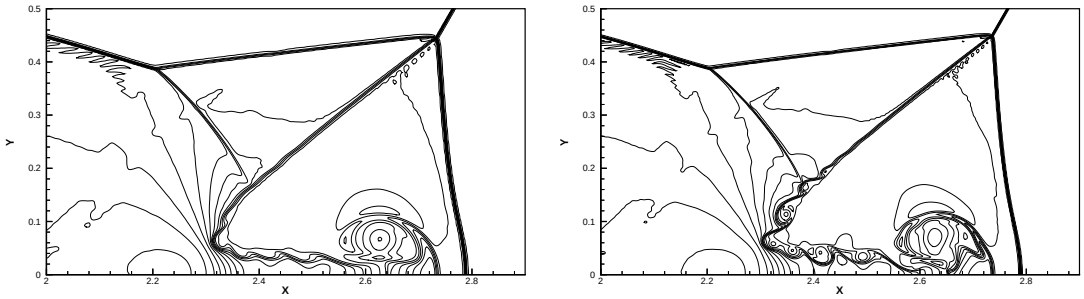

Figure 19: Double Mach reflection problem, zoom in. TWENO scheme. $t=0.2$. 30 equally spaced density contours from 1.5 to 22.7. Left: 1600×400 grid points; Right: 2400×600 grid points.

starting from $x=\frac{1}{6}$, y=0, making a 60^o angle with the x-axis. The reflection boundary condition is used at the wall, while for the rest of the bottom boundary (the part from $x=0$ to $x=\frac{1}{6}$) the exact post-shock condition is imposed. The top boundary uses the exact motion of the Mach 10 shock. We present the density at $t=0.2$ in the region $[0,3]\times[0,1]$ and the blow-up region around the double Mach stems in Fig. 18 and Fig. 19, respectively. All pictures show the density contours with 30 equal spaced contour lines from 1.5 to 22.7. We can see that most of the flow features are captured well, and the scheme resolves the two Mach stems well.

Example 3.12. A Mach 3 wind tunnel with a step. The setup of the problem is as follows: The wind tunnel is 1 length unit wide and 3 length units long. The step is 0.2 length units high and is located 0.6 length units from the left end of the tunnel. Initially, a right going Mach 3 flow is used. Reflective boundary conditions are applied along the walls of

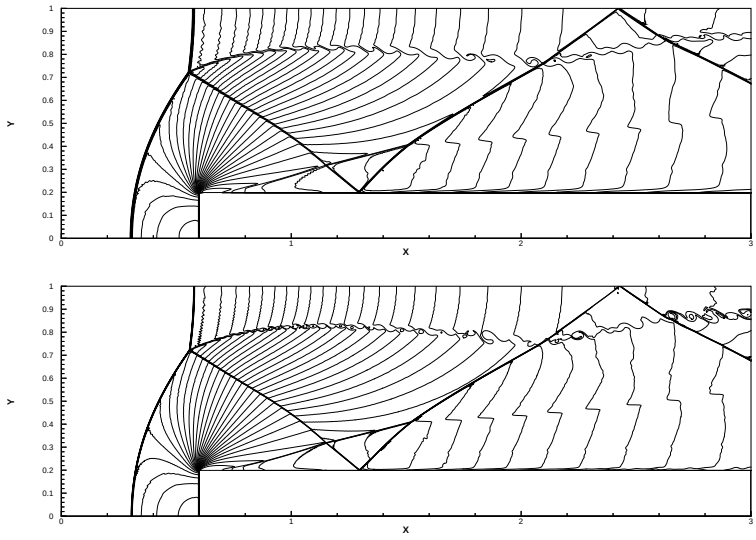

Figure 20: A Mach 3 wind tunnel with a step problem. TWENO scheme. $t=4.0$. 30 equally spaced density contours from 0.32 to 6.15. Top: 1200×400 grid points; Bottom: 2400×800 grid points.

the tunnel and inflow and outflow boundary conditions are applied at the entrance and the exit, respectively. Fig. 20 presents the numerical results at $t=4$ in the whole solution domain $[0,3]\times[0,1]$. The pictures show the density contours with 30 equal spaced contour lines from 0.32 to 6.15. We can see that the scheme performs well with good resolution, both the shock and contact discontinuous are well captured.

4 Concluding remarks

In this paper, we constructed a class of high order finite difference TWENO schemes for hyperbolic conservation laws and highly oscillatory problems. The key ideas of TWENO schemes are the use of trigonometric polynomial interpolation in the reconstruction phase and the methodology of weighted essentially non-oscillatory properties. We use the procedure presented in [14] to obtain the values of linear weights, smoothness indicators and nonlinear weights. Also in these cases, the function basis are not in the algebraic polynomial space but in the trigonometric polynomial space instead. Compared to the original trigonometric essentially non-oscillatory schemes [3], the major advantages of the new TWENO schemes are as follows. They are robust in computations for problems with strong shocks and can obtain the same order of accuracy with more compact stencils. Moreover, they use less logical "if" and are very efficient on vector computers for parallel computing. Compared to the original WENO schemes in [14], the more oscillations for the trigonometric polynomials are used, the better the TWENO schemes work. Numerical experiments for Euler equations of compressible gas dynamics and other cases are presented to show the effectiveness of the proposed schemes.

Acknowledgments

The research was supported by NSFC grants 10671091, 10811120283 and the European project ADIGMA on the development of innovative solution algorithms for aerodynamic simulations. Additional support was provided by USA NSF DMS-0820348 while J. Qiu was in residence at Department of Mathematical Sciences, Rensselaer Polytechnic Institute.

References

[1] W. Baron, Zur trigonometrischen interpolation, Computing, 16 (1976), 319-328.
[2] M. Brio, A.R. Zakharian and G.M. Webb, Two dimensional Riemann solver for Euler equations of gas dynamics, J. Comput. Phys., 167 (2001), 177-195.
[3] S. Christofi, The study of building blocks for essentially non-oscillatory (ENO) schemes, PhD. thesis, Division of Applied Mathematics, Brown University, 1996.
[4] S.K. Godunov, A finite-difference scheme for the numerical computation of discontinuous solutions of the equations of fluid dynamics, Matthematicheskii sbornik, 47(3) (1959), 271-290.
[5] A. Harten, High resolution schemes for hyperbolic conservation laws, J. Comput. Phys., 49(3) (1983), 357-393.
[6] A. Harten and S. Osher, Uniformly high-order accurate non-oscillatory schemes, IMRC Technical Summary Rept, 2823, Univ. of Wisconsin, Madison, WI, 1985.
[7] G.S. Jiang and C.W. Shu, Efficient implementation of weighted ENO schemes, J. Comput. Phys., 126 (1996), 202-228.
[8] P.D. Lax and X.D. Liu, Solution of two dimensional Riemann problems of gas dynamics by positive schemes, SIAM J. Sci. Comput., 19(2) (1998), 319-340.
[9] X.D. Liu, S. Osher and T. Chan, Weighted essentially non-oscillatory schemes, J. Comput. Phys., 115 (1994), 200-212.
[10] G. Muhlbach, A recurrence formula for generalized divided differences and some applications, J. Apprxn., Th 9 (1973), 165-172.
[11] G. Muhlbach, Newton-und-Hermite-interpolation mit Cebysev-systemen,Z. Angew. Math. Mech., 54 (1974), 541-550.
[12] G. Muhlbach, The general Neville-Aitken-algorithm and some applications, Num. Math., 31 (1978), 97-110.
[13] G. Muhlbach, The general recurrence relation for divided differences and the general Newton-interpolation-algorithm with applications to trigonometric interpolation, Num. Math., 32 (1979), 393-408.
[14] C.W. Shu, Essentially non-oscillatory and weighted essentially non-oscillatory schemes for hyperbolic conservation laws, in Advanced Numerical Approximation of Nonlinear Hyperbolic Equations, edited by A. Quarteroni, Editor, Lecture Notes in Mathematics, CIME subseries, Springer-Verlag, Berlin/New York; ICASE Report 97-65.
[15] C.W. Shu and S. Osher, Efficient implementation of essentially non-oscillatory shock capturing schemes, J. Comput. Phys., 77 (1988), 439-471.
[16] Y.T. Zhang and C.W. Shu, Third order WENO scheme on three dimensional tetrahedral meshes, Commun. Comput. Phys., 5 (2009), 836-848.