2008

Adaptive filtering and limiting in compact high order methods for multiscale gas dynamics and MHD systems

Helen C. Yee
NASA Ames Research Center, yee@nas.nasa.gov

Bjorn Sjögreen
Lawrence Livermore National Laboratories, sjogreen2@llnl.gov

Available online at www.sciencedirect.com

ScienceDirect

Computers & Fluids 37 (2008) 593–619

computers & fluids

www.elsevier.com/locate/compfluid

Adaptive filtering and limiting in compact high order methods for multiscale gas dynamics and MHD systems

H.C. Yee [a,*], B. Sjögreen [b]

[a] NASA Ames Research Center, Moffett Field, CA 94035, USA
[b] NADA, KTH, Stockholm, Sweden

Available online 24 August 2007

Dedicated to Prof. Hafez's on his 60th birthday

Abstract

The adaptive multistep linear and nonlinear filters for multiscale shock/turbulence gas dynamics and magnetohydrodynamics (MHD) flows of the authors are extended to include compact high order central differencing as the spatial base scheme. The adaptive mechanism makes used of multiresolution wavelet decomposition of the computed flow data as sensors for numerical dissipative control. The objective is to expand the work initiated in [Yee HC, Sjögreen B. Nonlinear filtering in compact high order schemes. In: Proceedings of the 19th ICNSP and 7th APPTC conference; 2005; J Plasma Phys 2006;72:833–36] and compare the performance of adaptive multistep filtering in compact high order schemes with adaptive filtering in standard central (non-compact) schemes for multiscale problems containing shock waves.
Published by Elsevier Ltd.

1. Introduction

High order compact spatial discretizations in conjunction with linear high order compact filters are methods of choice for many incompressible and low speed turbulent/acoustic flows due to their advantage of requiring a very low number of grid points per wavelength and flexibility in geometry handling. On the other hand, for unaveraged, unsteady compressible viscous flows containing shock/shear waves, it was observed that the use of even very high order shock-capturing schemes is still too dissipative for turbulence and transition predictions, especially for direct numerical simulations (DNS) and large Eddy simulations (LES). Methods commonly used for shock/turbulence interactions relying on switching between spectral or high order compact schemes and shock-capturing schemes are not practical for multiscale shock/turbulence interactions. One shortcoming of this type of hybridization is that the numerical solution might experience a non-smooth transition at the switch to a different type of scheme. For 2D and 3D complex shock wave and shear surface interactions, the switch mechanism can become less trivial and frequent switching between these two types of schemes can further promote numerical instability beyond the induced instability from the inherent strong nonlinearity and the presence of multiscale physical processes that are dominant features of the subject flow in question. Our highly parallelizable adaptive multistep linear and nonlinear filter schemes do not rely on switching between schemes to avoid the related numerical instability [33,40–44,46]. Instead of solely relying on very high order high-resolution shock-capturing methods for accuracy, the filter schemes take advantage of the effectiveness of the nonlinear dissipation contained in good shock-capturing schemes and standard linear filters (and/or high order linear dissipation) as post-processing stabilizing mechanisms at locations where needed.

* Corresponding author. Tel.: + 1 650 604 4769; fax: +1 650 604 4377.
E-mail addresses: yee@nas.nasa.gov (H.C. Yee), bjorns@nada.kth.se (B. Sjögreen).

0045-7930/$ - see front matter Published by Elsevier Ltd.
doi:10.1016/j.compfluid.2007.07.015

The adaptive filter method consists of two steps, a full time step using a spatially high order non-dissipative (or very low dissipative) base scheme, followed by adaptive multistep filter consisting of the products of wavelet based flow sensors and linear and nonlinear numerical dissipations to filter the solution. The numerical dissipation control idea is very general and can be used in conjunction with spectral, spectral element [16], finite element, discontinuous Galerkin [28], finite volume and finite difference spatial base schemes. The type of shock-capturing scheme used as nonlinear dissipation is very general and can be any dissipative portion of high resolution TVD, MUSCL, ENO, or WENO shock-capturing methods [39,15,28]. The shock-capturing dissipations usually contain flux limiters. The linear filter can be the standard spectral or compact filter, or the product of a high order linear dissipation and an appropriate flow sensor. By design, the flow sensors, spatial base schemes and linear and nonlinear dissipation models are standalone modules. Therefore, a whole class of low dissipative high order filter schemes can be derived at ease.

To clarify some of the terms and for generality of discussion we denote, loosely, the standard spectral filter, compact filter and non-compact high order linear numerical dissipation as high order linear numerical dissipations (or linear filter). In contrast, we denote the dissipative portion of any high resolution shock-capturing scheme as nonlinear numerical dissipation, since these dissipations are nonlinear even if one applies the scheme to a linear conservation law. When nonlinear dissipations are applied in a filter approach (to be discussed), we denote the approach as nonlinear filters. When high order linear dissipation, (e.g., the product of a flow sensor and the AD8 term) is applied in a filter approach, we denote the approach as high order linear filter. Although nonlinear numerical dissipations can suppress spurious high frequency oscillations, they might not be as effective as the standard high order linear dissipations (or linear filters). With appropriate wavelet flow sensors, locations of spurious high frequency oscillations, locations of shocks and high gradient regions, and locations of large vortices or vortex sheets can be detected separately. The appropriate numerical dissipations are then applied to these locations with the remaining regions free of numerical dissipation. (see [33,41,43] for a discussion.)

It is noted that earlier numerical experiments by the authors [33,40,41,43] and collaborators [22,1] indicate that inclusion of the flow sensors as an integral part of the shock-capturing dissipation limiting process or high order linear dissipation process can improve numerical accuracy over the original standard shock-capturing schemes. However, this improvement in accuracy is not as pronounced as the inclusion of the flow sensors as part of the filter approach.

In the finite difference approach, high order compact and high order central (non-compact) spatial discretizations are natural choices for the spatial base schemes. Numerical experiments using the standard sixth-order central base scheme with nonlinear filtering indicate improved accuracy over the standard shock-capturing schemes, standard shock-capturing scheme with flow sensors, and hybrid schemes mentioned earlier. In light of the fact that compact schemes are less compatible with parallel computations and thus require more CPU time than standard non-compact central schemes, the true efficiency and accuracy performance of compact base schemes under our framework of filtering and limiting is not certain. The objective is to expand the work initiated in [45] and to compare the performance of multistep filtering in compact high order schemes with filtering in standard central (non-compact) schemes for multiscale problems containing shock waves.

2. Adaptive filtering and limiting in high order methods

In this section, the scheme for the MHD system in uniform Cartesian grids is summarized. The scheme for gas dynamics is the same except without the three extra magnetic field equations. The high order formulation in generalized moving coordinates with freestream preservation is reported in Vinokur and Yee [37].

2.1. Conservative symmetrizable MHD systems

Consider the 3D conservative and symmetrizable [12,27] (non-conservative) forms of the ideal compressible MHD equations in Cartesian geometry,

$$U_t + \nabla \cdot \mathbf{F} = 0 \quad \text{(conservative)}, \tag{1}$$

$$U_t + \nabla \cdot \mathbf{F} = S \quad \text{(symmetrizable)}, \tag{2}$$

$$U = \begin{pmatrix} \rho \\ \rho\mathbf{u} \\ e \\ \mathbf{B} \end{pmatrix}; \quad \mathbf{F} = \begin{pmatrix} \rho\mathbf{u} \\ \rho\mathbf{u}\mathbf{u}^{\mathrm{T}} + (p + B^2/2)I - \mathbf{B}\mathbf{B}^{\mathrm{T}} \\ \mathbf{u}(e + p + B^2/2) - \mathbf{B}(\mathbf{u}^{\mathrm{T}}\mathbf{B}) \\ \mathbf{B}\mathbf{u}^{\mathrm{T}} - \mathbf{u}\mathbf{B}^{\mathrm{T}} \end{pmatrix};$$

$$S = -(\nabla \cdot \mathbf{B}) \begin{pmatrix} 0 \\ \mathbf{B} \\ \mathbf{u}^{\mathrm{T}}\mathbf{B} \\ \mathbf{u} \end{pmatrix}. \tag{3}$$

Here the velocity vector $\mathbf{u} = (u, v, w)^{\mathrm{T}}$, the magnetic field vector $\mathbf{B} = (B_x, B_y, B_z)^{\mathrm{T}}$, ρ is the density, and e is the total energy. The notation $B^2 = B_x{}^2 + B_y{}^2 + B_z{}^2$ is used. The superscript "T" indicates the transpose of the subject column vector. $\mathbf{a}^{\mathrm{T}}\mathbf{b}$ denotes the inner product between the vectors $\mathbf{a}$ and $\mathbf{b}$. The divergence of the outer product of two vectors, $\nabla \cdot \mathbf{a}\mathbf{b}^{\mathrm{T}}$, is a vector whose ith component is

$$\sum_{j=1} 3 \frac{\partial}{\partial x_j} a_i b_j.$$

The pressure is related to the other variables by

$$p = (\gamma - 1)\left(e - \frac{1}{2}\rho(u^2 + v^2 + w^2) - \frac{1}{2}(B_x^2 + B_y^2 + B_z^2)\right).$$

The magnetic pressure is proportional to B^2. For plasmas and monatomic gases, $\gamma = 5/3$. The vector on the right hand side of Eq. (2) is the non-conservative portion of the symmetrizable MHD equations and is frequently referred to in the literature as a source term vector.

The conservative and symmetrizable forms of the non-ideal compressible MHD [9] systems (viscous, resistive and Hall MHD) are

$$U_t + \nabla \cdot \mathbf{F} = \mathbf{F}_v,$$
$$U_t + \nabla \cdot \mathbf{F} = \mathbf{F}_v + S,$$
$$\mathbf{F}_v = \begin{bmatrix} 0 & \operatorname{div}\tau & f_{v5} & \frac{1}{\sigma}(\triangle\mathbf{B} - \nabla\operatorname{div}\mathbf{B}) - \beta_\mathrm{h}\nabla\times((\nabla\times\mathbf{B})\times\mathbf{B}) \end{bmatrix}^\mathrm{T}.$$

The fifth component of $\mathbf{F}_v$ is

$$f_{v5} = \operatorname{div}(\mathbf{u}^\mathrm{T}\tau) + \operatorname{div}\mathbf{h} - \frac{1}{\sigma}\operatorname{div}((\nabla\times\mathbf{B})\times\mathbf{B}) - \beta_\mathrm{h}\operatorname{div}(((\nabla\times\mathbf{B})\times\mathbf{B})\times\mathbf{B}).$$

The vector $\mathbf{F}_v$ includes viscosity, resistivity, and Hall effect with τ being the viscous stress tensor, σ the conductivity coefficient, β_h the strength of the Hall effect and $\mathbf{h}$ the heat flux. The plasma β is β_p = (plasma pressure/magnetic pressure).

Without loss of generality we will describe our numerical methods for the inviscid x-flux of the ideal MHD Eq. (1) on a uniform grid. The schemes to be discussed, for the most part, only spell out the x-component terms with the y- and z-components omitted. Let $A(U)$ denote the Jacobian $\partial F/\partial U$ with the understanding that the present F and S are the inviscid x-component of the 3D description above. We also write the non-conservative term S in Eq. (2) in the x-direction as $N(U)U_x$.

An important ingredient in our high order filter method is the use of the dissipative portion of high-resolution shock-capturing schemes as part of the nonlinear filters for accurately capturing discontinuities. If the dissipative portion of higher order Lax–Friedrichs or Nessyahu–Tadmor [23] type of shock-capturing schemes is not employed (see [39] for a discussion), these nonlinear filters usually involve the use of field-by-field approximate Riemann solvers.

Seven of the eigenvalues and eigenvectors are identical for the "conservative" Jacobian matrix A and the "symmetrizable" Jacobian matrix $(A - N)$ [10]. For ease of reference, we refer to the distinct eigenvalue (eigenvector) between the conservative and symmetrizable MHD as the eighth eigenvalue (eigenvector). The eighth eigenvector of A of the conservative system associated with the degenerate zero eigenvalue can sometimes coincide with one of the other eigenvectors, thereby making it difficult to obtain a Roe-type approximate Riemann solver for the multi-dimensional conservative MHD. On the other hand, the eigenvectors of the symmetrizable Jacobian $A^* = (A - N)$ always form a complete basis, and can be obtained from analytical formulas [12,27] for 1D or higher. Here, a Roe-type average state developed in Gallice [10] for the multi-D symmetrizable MHD is employed to solve both the conservative and symmetrizable systems Eqs. (1) and (2). This form is an improvement over the Brio and Wu [2] and Powell [27] forms. See the multistep filter section for more discussion on the rationale of employing symmetrizable eigenvectors to solve the conservative system.

2.2. Description of high order filter methods

For non-ideal MHD, we apply the spatial base scheme for the first derivative twice for the second derivatives in the viscous terms (similarly for the resistive and Hall terms). Basically, the filter method consists of two steps, a divergence-free preserving spatial base scheme step (not involving the use of approximate Riemann solvers or flux limiters) and a multistep filter (usually involving the use of approximate Riemann solvers and flux limiters). The high order spatial base scheme to approximate the flux derivative of the ideal MHD is very general. Spectral, spectral element, finite element, discontinuous Galerkin, compact and non-compact schemes are possible candidates. In order to have good shock-capturing capability and improved nonlinear stability related to spurious high frequency oscillations, a multistep filter approach consisting of a high order nonlinear filter and a high order linear filter was investigated in [41,43,45]. The nonlinear filter consists of the product of an artificial compression method indicator or wavelet sensor [33] and the nonlinear dissipative portion of a high-resolution shock-capturing scheme. The high order linear filter consists of the product of another sensor and a spectral-like filter or a high order centered linear dissipative operator that is compatible with the order of the base scheme being used.

2.2.1. Divergence-free preserving base scheme step

The first step of the numerical method consists of a time step via a high order non-dissipative spatial and high order temporal base scheme operator L^*. After the completion of a full time step of the base scheme step, the solution is denoted by U^*

$$U^* = L^*(U^n), \tag{4}$$

where U^n is the numerical solution vector at time level n. The spatial base scheme can be, e.g., any of the sixth-order or higher central or compact discretizations. For strong shock interactions and/or steep gradient flows, a small amount of high order linear dissipation can be added to the base scheme step to reduce the time step constraint and improve stability. For example, an eighth-order linear dissipation with the sixth-order centered non-compact and compact base schemes to approximate $F(U)_x$ (with the grid indices k and l for the y- and z-directions suppressed) is written as

$$\frac{\partial F}{\partial x} \approx D_{06}F_j + d(\Delta x)^7(D_+D_-)^4U_j, \tag{5}$$

$$\frac{\partial F}{\partial x} \approx C_{06}F_j + d(\Delta x)^7(D_+D_-)^4U_j, \tag{6}$$

where D_{06} is the standard sixth-order accurate centered difference operator, and D_+D_- is the standard second-order accurate centered approximation of the second derivative. The second terms in Eqs. (5) and (6) denoted by "AD8", if needed, are the eighth-order linear dissipation. The small parameter d of the AD8 term is a scaled value (e.g., spectral radius of $A(U)$) in the range of 0.00001–0.0005, depending on the flow problem, and has the sign which gives dissipation in the forward time direction. The D_{06} operator is modified at boundaries in a stable way by the so called summation-by-part (SBP) operators [25,24,41]. The linear numerical dissipation operator D_+D_- is modified at the boundaries to be semi-bounded [31]. The symbol C_{06} in Eq. (6) denotes the sixth-order centered compact operator. Similarly 8th-order and 10th-order central and compact base scheme operators with the corresponding 10th-order and 12th-order linear dissipation terms are denoted by "AD10" and "AD12" respectively.

Some comparison of the two base schemes has been reported in [45,46]. Previous studies [43,45,46] indicated that the two base schemes might require different amounts of linear dissipation (or linear filter) and nonlinear filter (to be discussed), depending on the test problem. Aside from improving numerical stability due to long time integration related spurious high frequency oscillations, the inclusion of non-zero AD8 can have a different effect on the locations where nonlinear filters are utilized than without AD8. For example, for the coefficient of AD8 with $d = 0$, the wavelet sensor would indicate that nonlinear filters are needed at locations of spurious high frequency oscillations as well as at discontinuity locations that experience Gibbs phenomena. However, for AD8 with $d \neq 0$, the linear dissipation would damp out some or all of the high frequency oscillation locations. However, in the actual case this is a very dynamic procedure and highly problem, base scheme and filter term dependent, especially when one is dealing with a chaotic-like flow. Ideally, AD8 should contain a proper flow sensor to indicate where linear dissipation is needed. For the MHD system, in order to maintain the divergence-free preserving property of the base scheme step in Cartesian grid, it is more desirable to apply a small amount of AD8 uniformly. It is noted that this combination of including non-zero AD8 linear dissipation in the base scheme and nonlinear filter might not be the optimal approach in general. Our numerical experiment gives just one simple-minded aspect of this study. In lieu of the non-zero AD8 base scheme approach, the adaptive multistep linear and nonlinear filters suggested in [45,46] with their own flow sensors might be an alternative. This is the topic of the next discussion.

2.2.2. *Multistep linear and nonlinear filters (suppression of high frequency oscillations and shock-capturing)*

Blending of different types of numerical dissipations – single step linear and nonlinear filter: In the early stages of our development, we proposed the blending of these two types of numerical dissipation into a single filter step after a complete full time step of the base scheme step (or after each stage of the temporal discretization if multistage temporal discretizations were employed). (See [41,43] and references cited therein.) Subsequent studies [43,45] showed that the blending of more than one type of filter in a single step might create numerical instability due to the frequent switching between filters. For the MHD system, the single step blending of more than one filter can interfere with the divergence-free preserving property as discussed above.

Multistep Filters: As discussed in [45,46], if instead, we apply the linear filter and nonlinear filter in separate steps, numerical stability is greatly improved. Moreover, the interfering with the divergence-free property is minimized. Our recent study indicates that a multistep filter, e.g., applying the nonlinear filter step after the high order linear filter step in sequence (or vice versa) is more effective and stable than the blending of different filters in a single step. Studies in Yee and Sjögreen [45] and the present paper indicate that if the compact base scheme Eq. (6) were used for complex shock interactions, the multistep filter is needed (a linear compact filter step and a nonlinear filter step).

The multistep filter or the single step filter can be applied (a) after each stage of a multistage temporal discretization (if such time discretization will be used), or (b) after the completion of each full time step of the multistage time discretization. Both options were implemented and tested on a wide variety of gas dynamics and MHD problems. Studies indicated that even if multistage Runge–Kutta methods are employed, there is no advantage in employing the filter step "after each Runge–Kutta stage" over the application of the filter step "after a full time step" of the Runge–Kutta method. On the other hand, option (b) is extremely efficient since only one Riemann solve per time step per dimension is required, independent of the time discretization. The next two sections discuss filter option (b) with filter option (a) similarly.

The following section gives a description of the nonlinear filter step. It is emphasized here that the order of applying the nonlinear filter and linear filter steps might have an effect on the final solution. Due to the dynamic filtering at each time step, the corresponding nonlinear filter and linear filter wavelet sensors are different, depending on the order in which they are applied. Before the description of the adaptive nonlinear filter step, we would like to discuss our procedure for solving the conservative system and the symmetrizable system if the scheme to be used requires the knowledge of the eigensystem.

Solving the conservative system using the symmetrizable eigenvectors

This class of filter schemes is suitable for solving both conservative and symmetrizable non-conservative systems. In solving the symmetrizable system, the base scheme and the filter step are applied to the non-conservative system Eq. (2) with a complete set of eigenvectors. However, for strong shocks, to ensure the correct shock strength and location, we prefer to solve the conservative MHD system. In solving the conservative system, the base scheme step presents no problem. The question is how to overcome the incomplete eigensystem issue if nonlinear filters involving Riemann solvers are required. In this case, as described in [42,43], we use eigenvectors of the symmetrizable form but with the degenerate eigenvalue replaced by an entropy correction (a small parameter ϵ that is scaled by the largest eigenvalue of $A(U)$) for the conservative form. For more than one space dimension, a multi-dimensional entropy correction [39] is used for each of the degenerate eigenvalues in each spatial direction. Our rationale for doing this is that both systems share the same eigenvalues and eigenvectors except for one. The incorrect eigenvector for the conservative form will be multiplied by an eigenvalue which is close to zero. Thus the effect of this "false" eigenvector will be small. (Note that in the present context, the use of an entropy correction is different from the standard entropy correction associated with expansion shocks in the Roe-type approximate solver in gas dynamics.) Another rationale is that solving the conservative system by the base scheme step has already ensured the correct shock speed and location of the solution. In turn, the flow sensor is sensing the resulting solution with the correct locations where shock-capturing dissipation is needed. The use of shock-capturing dissipation here is a post-processing step. It plays a different role than if one solves the conservative system by its full shock-capturing scheme counterpart (using the same false eigenvector).

2.3. Adaptive nonlinear filter step (discontinuity and high gradient capturing)

After the completion of a full time step of the divergence-free preserving base scheme step, the second step is to adaptively filter the solution by the product of a "wavelet sensor" and the "nonlinear dissipative portion of a high-resolution shock-capturing scheme" (involving the use of flux limiters). The final update step after e.g., the nonlinear filter step only can be written (with some grid indices suppressed and assume a single step filter for ease of illustration) as

$$U^{n+1}_{j,k,l} = U^*_{j,k,l} - \frac{\Delta t}{\Delta x}\left[H^{Nfx}_{j+1/2} - H^{Nfx}_{j-1/2}\right] - \frac{\Delta t}{\Delta y}\left[H^{Nfy}_{k+1/2} - H^{Nfy}_{k-1/2}\right] - \frac{\Delta t}{\Delta z}\left[H^{Nfz}_{l+1/2} - H^{Nfz}_{l-1/2}\right]. \tag{7}$$

Here, $H^{Nfx}_{j\pm1/2}$, $H^{Nfy}_{k\pm1/2}$ and $H^{Nfz}_{l\pm1/2}$ are the nonlinear filter numerical fluxes in the x, y and z-directions, respectively. If the dissipative portion of higher order Lax–Friedrichs or Nessyahu–Tadmor type of shock-capturing schemes is not employed, these nonlinear filters usually involve the use of field-by-field approximate Riemann solvers. If Roe's type of approximate Riemann solver is employed, for example, the x-filter numerical flux vector $H^{Nfx}_{j+1/2}$ is

$$H^{Nfx}_{j+1/2} = R_{j+1/2}\overline{H}_{j+1/2},$$

where $R_{j+1/2}$ is the matrix of right eigenvectors of the Jacobian of the non-conservative MHD flux vector $(A_{j+1/2} - N_{j+1/2})$ evaluated at, e.g., the Gallice average state [10] in terms of the U^* solution from the base scheme step Eq. (4). The subscript in $R_{j+1/2}$ indicates the average state evaluated in the x-direction of the eigenvectors in terms of U^*. See [10] or Appendix A of [43] for the average state formula for the 3D non-conservative system Eq. (2). The $\overline{H}_{j+1/2}$ (involving the use of wavelet sensors and flux limiters) are also evaluated from the same average state. Here, the dimension-by-dimension procedure of applying the approximate Riemann solver is adopted.

Denote the elements of the vector $\overline{H}_{j+1/2}$ by $\bar{h}^l_{j+1/2}$, $l = 1, 2, \ldots, 8$. The nonlinear portion of the filter $\bar{h}^l_{j+1/2}$, $l = 1, 2, \ldots, 8$, has the form

$$\bar{h}^l_{j+1/2} = \frac{1}{2}(s^N)^l_{j+1/2}(\phi^l_{j+1/2}). \tag{8}$$

Here $(s^N)^l_{j+1/2}$ is the sensor to activate the higher order nonlinear numerical dissipation $\phi^l_{j+1/2}$. For example, $(s^N)^l_{j+1/2}$ is designed to be zero or near zero in regions of smooth flow and near one in regions with discontinuities. $(s^N)^l_{j+1/2}$ varies from one grid point to another and is obtained from a wavelet analysis of the flow solution [33]. The wavelet sensor can be obtained from the characteristic variables for each wave or a single sensor for all eight waves, based on pressure and density. Both methods were implemented but for the numerical tests in this paper, the simpler non-characteristic sensor was employed.

The dissipative portion of the nonlinear filter $\phi^l_{j+1/2} = g^l_{j+1/2} - b^l_{j+1/2}$ is the dissipative portion of a high order high-resolution shock-capturing scheme for the local lth-characteristic wave. Here $g^l_{j+1/2}$ and $b^l_{j+1/2}$ are numerical fluxes of the uniformly high order shock-capturing scheme and a high order central scheme for the lth characteristic, respectively. It is noted that $b^l_{j+1/2}$ might not be unique since there is more than one way of obtaining $\phi^l_{j+1/2}$. The dissipative portion of TVD, MUSCL, and WENO schemes of orders five, seven, nine and eleven were considered.

For the numerical examples shown, three forms of nonlinear dissipation $\phi^l_{j+1/2}$ were considered, namely:

- Dissipative portion of the fifth-order WENO scheme (WENO5) [44]. It can be obtained e.g., in the x-direction by taking the full WENO5 scheme in the x-direction and subtracting $D_{06}F_j$ (or $C_{06}F_j$).
- Dissipative portion of the a second-order MUSCL scheme [40].
- Dissipative portion of the Harten–Yee TVD scheme [40,43].

This nonlinear filter if applied to the entire MHD system, will not preserve the divergence-free magnetic field condition in general, with the exception of certain smooth flows. This is due to the fact that the wavelet sensor turns off the nonlinear filter in regions of very smooth flow. For the computations in this paper and our previous work, the

"No filter on B" option is chosen. That is, the nonlinear filter step only applies to the first five equations of Eq. (1) or Eq. (2). Here the complete set of eigenvalues and eigenvectors of the full symmetrizable MHD system is used to evaluate the first five equations of Eq. (1) or Eq. (2). With the divergence-free spatial base scheme, the divergence-free property should be preserved for uniform grids. Extensive grid convergence comparison of the "no filter on B" with the "filter all of the MHD equations" (filter all) options were presented in [43]. Alternative approaches for obtaining divergence-free preserving shock-capturing filters follow in a similar vein as the constrained transport approach [6].

Note that if a high order linear filter step is employed prior to the nonlinear filter with the resulting solution denoted by U^{**} (right after the completion of a full time step of the base scheme step), it is understood that the numerical fluxes above are evaluated at U^{**} instead of U^{*}.

2.3.1. Flow sensor by multiresolution wavelet analysis of the computed flow data

The basic idea in obtaining the different flow sensors (e.g., $(s^N)^l_{j+1/2}$) by multiresolution wavelet analysis of the computed flow data can be found in Sjögreen and Yee [33] and Yee and Sjögreen [41]. Two types of multiresolution wavelets were considered. The mathematical procedures to obtain this type of flow sensor are very involved. However, the final algorithm is very simple. Interested readers are referred to the original papers for details. The two papers [13,32] are sources of background material for [33].

Wavelets were originally developed for feature extraction in image processing and for data compression. It is well known that the regularity of a function can be determined from its wavelet coefficients [4,20,17] far better than from its Fourier coefficients. By computing wavelet coefficients (with a suitable set of wavelet basis functions), we obtain very precise information about the regularity of the function in question. This information is obtained just by analyzing a given grid function. No information about the particular problem which is solved is used. Thus, wavelet detectors are general, problem independent, and rest on a solid mathematical foundation.

As of the 1990's, wavelets have served as basis functions that are finding use in analyzing and interpreting turbulence data from experiments. They also are used for analyzing the structure of turbulence from numerical data obtained from DNS or LES. See Farge [7] and Perrier et al. [26] for early work. There are several ways to introduce wavelets. One standard way is through the continuous wavelet transform and another is through multiresolution analysis, hereafter referred to as wavelet based multiresolution analysis. Mallet and collaborators [17–20] established important wavelet theory through multiresolution analysis. See references [36,35] for an introduction to the concept of multiresolution analysis. Wavelet based multiresolution analysis has been used for grid adaptation (Gerritsen and Olsson [11]), and to replace existing basis functions in constructing more accurate finite element methods. Here we utilize wavelet based multiresolution analysis to adaptively control the amount of numerical dissipation.

Our wavelet flow sensor estimates the Lipschitz exponent of a grid function f_j (e.g., the density and pressure). The Lipschitz exponent at a point x; with grid size h, is defined as the largest α satisfying

$$\sup_{h\neq 0}\frac{|f(x+h)-f(x)|}{h^{\alpha}}\leqslant C, \tag{9}$$

and this gives information about the regularity of the function f, where small α means poor regularity. For a C^1 wavelet function ψ with compact support, α can be estimated from the wavelet coefficients, defined as

$$w_{m,j}=\langle f,\psi_{m,j}\rangle=\int f(x)\psi_{m,j}(x)\mathrm{d}x, \tag{10}$$

where

$$\psi_{m,j}=2^m\psi\left(\frac{x-j}{2^m}\right) \tag{11}$$

is the wavelet function $\psi_{m,\,j}$ on scale m located at the point j in space. This definition gives a so called redundant wavelet, which gives (under a few technical assumptions on ψ) a non-orthogonal basis for L^2. Theorem 9.2.2 in [4] states that if ψ is C^1 and has compact support, and if the wavelet coefficients $\max_j|w_{m,j}|$ in a neighborhood of j_0 decay as $2^{m\alpha}$ as the scale is refined, then the grid function f_j has Lipschitz exponent α at j_0. In practical computations, we have a smallest scale determined by the grid size. We evaluate $w_{m,j}$ on this scale, m_0, and a few coarser scales, m_0+1, m_0+2, and estimate the Lipschitz exponent at the point j_0 by a least square fit to the line [33]

$$\max_{j\,\text{near}\,j_0}\log_2|w_{m,j}|=m\alpha+c. \tag{12}$$

Proper selection of the wavelet ψ is very important for an accurate detection of the flow features. The result in [20,19], which is used in [11], gives the condition that $\psi(x)$ should be the kth derivative of a smooth function $\eta(x)$ with the property

$$\eta(x)>0,\qquad \int\eta(x)\mathrm{d}x=1,\qquad \lim_{x\to\pm\infty}\eta^{(k)}(x)=0. \tag{13}$$

Then the result is valid for $0<\alpha<k$. A continuous function $f(x)$ has a Lipschitz exponent $\alpha>0$. A bounded discontinuity (shock) has $\alpha=0$, and a Dirac function (local oscillation) has $\alpha=-1$. Large values of k can be used in turbulent flow so that large vortices or vortex sheets can be detected. Although the theorem above does not hold for α negative, a useful upper bound on α can be obtained from the wavelet coefficient estimate. See our original paper on the wavelet flow sensor algorithm [33] based on the Lipschitz exponent of a chosen computed flow vector. The remainder of this section gives a summary of the three basic steps described in [33] for obtaining the wavelet flow sensors.

Step 1: Choose a wavelet type
- Redundant form of Harten's multiresolution form.
- Second-order or higher B-splines.
- Wavelets that can distinguish high frequency oscillations from turbulence.

Step 2: Choose flow variables to be sensed
- Density and pressure.
- Characteristic variables.
- Primitive or entropy variables.

Step 3: Flow sensors
- Apply wavelets to the flow variables to be sensed.
- Obtain the corresponding wavelet coefficient (involves 2–4 levels of nested difference operators).
- Obtain Lipschitz exponents (least square fit of the wavelet coefficients in domain of dependence).
- Determine the range of Lipschitz exponent values or the cutoff Lipschitz exponent value (or a smooth transition) for the appropriate type of numerical dissipation to be applied.
- Use cutoff Lipschitz exponents as "flow sensors" (filter with appropriate numerical dissipations).

For example, a Lipschitz exponent with a value near zero, -1, or wavelets with high order vanishing moments indicate the presence of a discontinuity, spurious local high frequency oscillations or large vortices/vortex sheets respectively. For example, the flow sensor $(s^N)^l_{j+1/2}$ to turn on the shock-capturing dissipation using the cut off procedure above is a vector (if applied dimension-by-dimension) consisting of "1's" and "0's".

The computer routines to compute the wavelet coefficient of the second-order B-spline and the redundant form of Harten's multiresolution wavelets and their corresponding Lipschitz exponents of a given grid function f_j (e.g., density or pressure, or characteristic variables) can be found in the Appendix.

2.4. Adaptive high order linear filter

This section discusses the non-compact and compact high order linear filter step used in conjunction with the sixth-order central and compact base schemes, respectively. Other compatible linear filters for orders other than the sixth-order base schemes follow the same vein. For ease of discussion, we assume that the linear filter is applied after the nonlinear filter step. The procedure is similar if the multistep filters are reversed. Denote the solution from the nonlinear filter step by U^{**} (i.e., replace U^{n+1} from the nonlinear filter step above by U^{**}).

The final update step after the linear filter step can be written (with some grid indices suppressed for ease of illustration) as

$$U^{n+1}_{j,k,l} = U^{**}_{j,k,l} - \frac{\Delta t}{\Delta x}\left[H^{Lfx}_{j+1/2} - H^{Lfx}_{j-1/2}\right] - \frac{\Delta t}{\Delta y}\left[H^{Lfy}_{k+1/2} - H^{Lfy}_{k-1/2}\right] - \frac{\Delta t}{\Delta z}\left[H^{Lfz}_{l+1/2} - H^{Lfz}_{l-1/2}\right]. \tag{14}$$

Here, $H^{Lfx}_{j\pm1/2}$, $H^{Lfy}_{k\pm1/2}$ and $H^{Lfz}_{l\pm1/2}$ are the linear filter numerical fluxes in the x, y and z-directions, respectively. If a spectral filter or compact filter is used, the linear filter numerical fluxes are the corresponding filter formulas. The following discusses the sixth-order non-compact linear filter.

2.4.1. Non-compact linear filter

If the linear filter step is applied to the local characteristic variables, the x-filter numerical flux vector $H^{Nfx}_{j+1/2}$ takes the form

$$H^{Lfx}_{j+1/2} = R_{j+1/2}\widehat{H}_{j+1/2},$$

where $R_{j+1/2}$ and $\widehat{H}_{j+1/2}$ are evaluated at the Gallice average states (or Roe's average state for perfect gas) based on U^{**}. Denote the elements of the vector $\widehat{H}_{j+1/2}$ by $\hat{h}^l_{j+1/2}$, $l = 1, 2, \ldots, 8$. They have the form

$$\hat{h}^l_{j+1/2} = -(s^L)^l_{j+1/2} d^l_{j+1/2}. \tag{15}$$

Here $(s^L)^l_{j+1/2}$ are sensors to activate the higher order linear filter. For example, $(s^L)^l_{j+1/2}$ is designed to be zero or near zero in regions of smooth flow and near one in regions with spurious high frequency oscillation (due to, e.g., long time integration of nonlinear systems). The functions $d^l_{j+1/2}$ are the dissipative portion of the respective linear filter for the local lth-characteristic wave in the x-direction. If a sixth-order central base scheme is used, $d^l_{j+1/2}$ has the same form as AD8 but utilized in a filter context. The eighth-order dissipative portion of the linear filter in terms of the local characteristic variables has the form

$$d^l_{j+1/2} = \frac{1}{2} d_f \Delta x^7 D_+(D_+D_-)^3 (w^l_{j+1/2} + w^l_{j-1/2}),$$

or

$$d^l_{j+1/2} = d_f \Delta x^7 D_+(D_+D_-)^3 w^l_j.$$

Here w^l_j is the local lth-characteristic variable in the x-direction evaluated at U^{**}. The term $w^l_{j+1/2}$ is the local lth-characteristic variable in the x-direction evaluated at the average state in terms of U^{**}. d_f is a small tuning parameter with a scaled range as the parameter d in Eq. (5). Note that the high order linear filter is not to be confused with the high order linear dissipation in the base scheme step Eq. (5).

An alternative to applying the linear filter in terms of the characteristic variables is to apply it in terms of the conservative variables. In this case,

$$H^{Lfx}_{j+1/2} = \widehat{H}^L_{j+1/2},$$

where $\widehat{H}^{L}_{j+1/2}$ now has the form

$$\widehat{H}^{L}_{j+1/2} = \frac{1}{2}s^{\text{wav}} d_f \Delta x^7 D_+ (D_+D_-)^3 (U_{j+1/2} + U_{j-1/2}),$$

or

$$\widehat{H}^{L}_{j+1/2} = s^{\text{wav}} d_f \Delta x^7 D_+ (D_+D_-)^3 U_j.$$

Here s^{wav} is the wavelet sensor based on the pressure and density at the grid point $j+1/2$.

All of the above linear filters if applied to the entire MHD system will not preserve the divergence-free magnetic field condition with the exception of certain smooth flows. The linear filter without the flow sensor will preserve the divergence-free magnetic field condition. Akin to the nonlinear filter, the "No Filter on B" can be used.

2.4.2. Compact linear filter

The general form of the compact linear filter applied to a one dimensional scalar grid function U_j is

$$\alpha_f \bar{U}_{j+1} + \bar{U}_j + \alpha_f \bar{U}_{j-1} = \sum_{p=0}^{q} \frac{a_p}{2}(U_{j+p} + U_{j-p}),$$
$$j = q+1, q+2, \ldots, N-q. \tag{16}$$

See [8] and references cited therein for details. U_j is the unfiltered (input) function and $\bar{U}_j$ denotes the filtered (output) function. These functions are defined on the grid $j = 1, 2, \ldots, N$. We use the eighth order filter, $F8$ in [8], which has $q = 4$ and

$$a_0 = \frac{93+70\alpha_f}{128}, \quad a_1 = \frac{7+18\alpha_f}{16}, \quad a_2 = \frac{-7+14\alpha_f}{32},$$
$$a_3 = \frac{1-2\alpha_f}{16}, \quad a_4 = \frac{-1+2\alpha_f}{128},$$

where we set $\alpha_f = 0.4$. The filter at the boundary point $j = 1$ is $\bar{U}_1 = U_1$. The boundary filters at the points $j = 2, 3, 4$ are the filters $F2$ with $\alpha_f = 0.49$, $F4$ with $\alpha_f = 0.49$, and $F6$ with $\alpha_f = 0.4$, respectively. These filters are given in [8] and are of the form Eq. (16). $F2$ has $p = 1$, order of accuracy 2, and coefficients

$$a_0 = \frac{1}{2} + \alpha_f, \quad a_1 = \frac{1}{2} + \alpha_f.$$

$F4$ has $p = 2$, order of accuracy 4, and coefficients

$$a_0 = \frac{5}{8} + \frac{3\alpha_f}{4}, \quad a_1 = \frac{1}{2} + \alpha_f, \quad a_2 = -\frac{1}{8} + \frac{\alpha_f}{4}.$$

$F6$ has $p = 3$, order of accuracy 6, and coefficients

$$a_0 = \frac{11}{16} + \frac{5\alpha_f}{8}, \quad a_1 = \frac{15}{32} + \frac{17\alpha_f}{16},$$
$$a_2 = -\frac{3}{16} + \frac{3\alpha_f}{8}, \quad a_3 = \frac{1}{32} - \frac{\alpha_f}{16}.$$

The filters at the upper boundary points $j = N-3$, $N-2$, $N-1$, N are symmetric with the lower boundary formulas.

For the three dimensional MHD system, Eq. (16) filters each solution component dimension by dimension. The order in which the dimensions are filtered will influence the result. Therefore, we change the filter dimension order after each time step in the order *xyz*, *yzx*, *zxy*, *xzy*, *zyx*, *yxz*, repeated cyclically.

2.5. Some attributes of the filtering and limiting approach

This subsection summarizes the rationale behind the multistep filter approach. There are four subtle attributes of our high order filter approach as compared with standard high order shock-capturing schemes. First, the filter approach utilizes high order conservative discretizations as base schemes with no involvement of flux limiters or Riemann solvers for each full time step discretization. Thus, no knowledge of the eigenstructure of the system is required. For example, we can always solve the conservative MHD system using our base scheme step even though it consists of an incomplete eigensystem set. After the completion of a full time step of the base scheme step, a post-processing filter step is employed. Only the filter step might involve the use of flux limiters and approximate Riemann solvers as stabilizing mechanisms to remove Gibbs phenomena related spurious oscillation resulting from the base scheme step at locations where needed. The flux limiters and approximate Riemann solvers, if needed, are not as crucial as in standard shock-capturing schemes in the sense of ensuring correct shock speeds and shock locations when one is dealing with e.g., the conservative MHD system containing an incomplete set of eigenvectors. Second, the physical viscosity, if present, is automatically taken into consideration by the base scheme step. The amount of filter numerical dissipation will be adjusted accordingly by the flow sensor in the presence of the physical viscosity. Third, the use of a wavelet decomposition of the computed flow data to determine the types and the location where numerical dissipation is needed is different from most existing numerical schemes in which the numerical dissipation is built into the discretization. In the presence of physical viscosity the more scales that are resolved by the base scheme, the less the filter is utilized, thereby gaining accuracy and computational time. In the limit when all scales are resolved, we are left with a "pure" non-dissipative centered (or very low dissipative) high order spatial scheme. If instead the inviscid part of the equations had been discretized by a scheme with an advanced numerical dissipation model, e.g., the TVD, ENO and WENO schemes, the expensive computation of the numerical dissipation would have been made everywhere in the computational domain, even in regions dominated by physical viscosity. Fourth, our sixth-order filter procedure in conjunction with the classical fourth-order Runge–Kutta method, in general, requires slightly more CPU time per time step (20%) than the standard second-order shock-capturing schemes. This is due to the fact that all of our filter schemes, regardless of the base scheme used, require only one Riemann solve per time step per direction (independent of the time discretizations of the base scheme step) as opposed to two Riemann solves per time step per direction by second-order shock-capturing schemes using a second-order Runge–Kutta method. Previous studies show that for a second-order base scheme filter method, improved accuracy was also realized in many multiscale shock/turbulence interactions. However, the improvement in accuracy is more pronounced if one uses the fourth-order or sixth-order base scheme which costs only slightly more CPU time (using the same second-order nonlinear filter). In fact, instead of nonlinear filtering, employing the flow sensor inside standard shock-capturing methods (for the dissipative portion) also results in improved accuracy. (See [1].)

3. Sample numerical results

Numerical experiments on the performance of compact and non-compact sixth-order spatial base schemes were conducted on the 2D test cases studied in [34,43,47]. Selected test cases are discussed here. Numerical examples concentrate on inviscid gas dynamics and ideal MHD systems. For the MHD results, all figures shown solve the conservative MHD system. In general, the computed results are slightly more accurate and stable than solving the symmetrizable system. For a comparison between conservative and symmetrizable systems, see [43] for details.

The wavelet filter schemes using the dissipative portion of WENO5, second-order MUSCL and Harten–Yee TVD schemes with sixth-order spatial central base scheme ($d = 0$ in Eq. (5)) for gas dynamics, the ideal and viscous non-ideal MHD flux derivatives and a fourth-order Runge–Kutta method are denoted by CEN66+WENO5, CEN66+MUSfi and CEN66+HYfi, respectively. The first number indicates the order of the base scheme for discretizing the inviscid flux derivatives. The second number indicates the order of the scheme for discretizing the viscous flux derivatives, if present. To adhere to the convention of previous work, even when dealing with inviscid flows, the same notation is used. Viscous flows are indicated with a non-zero Reynolds number. If the coefficient $d \neq 0$ in the base scheme step Eq. (5) for approximating the inviscid flux derivative, the symbol "AD8" is added as in CEN66+WENOfi+AD8. Computations using the sixth-order compact base scheme for the above filters are denoted by Comp66 as in Comp66+WENOfi and Comp66+WENOfi+AD8. Computations using the sixth-order compact base scheme for the above filters in conjunction with a second compact linear filter step are denoted by Comp66+Compfi as in Comp66+Compfi+WENOfi. Computations using the same temporal and spatial scheme for the viscous flux derivatives, and the standard fifth-order WENO scheme (using fourth-order Runge–Kutta temporal discretization) for the inviscid flux derivatives are denoted by WENO5. Computations using a second-order MUSCL and the Harten–Yee [43] TVD scheme for the inviscid MHD flux with the second-order central scheme for the viscous flux and a second-order Runge–Kutta method are denoted by MUSCL and HY, respectively.

It is noted that all the numerical results concentrate on the two base schemes for $d = 0$ or $d \neq 0$ in Eqs. (5) and (6). For the high order linear filters, results only show the effect of compact linear filters in conjunction with the compact base scheme. Studies on the different effects in applying the non-compact high order linear filter will be reported in a forthcoming paper.

The entropy fix parameter ϵ is in the range $(0, 0.25)$ [14,39] for the Harten–Yee and MUSCL, and the two nonlinear filter MUSfi and HYfi schemes (to avoid expansion shocks and the carbuncle phenomenon). The entropy value for the degenerate zero eigenvalue of the conservative system is in the range of 10^{-7}–10^{-10}. For simplicity, the cutoff wavelet Lipschitz exponent value β is 0.5 [33] for all the wavelet filter schemes. Other more appropriate range of Lipschitz exponent values to switch on the filter numerical dissipation will be reported in a forthcoming paper. See [33,40,41] or Appendix B of [43] for the definition of ϵ and β. Except for WENO5, the van Leer version of the van Albada limiter is used. For the second-order MUSCL scheme, the limiter is applied to the primitive variables. All methods employed Roe's approximate Riemann solver for the gas dyanmics cases and the Gallice approximate Riemann solver for the MHD cases using our method of solving the conservative MHD system [43]. The following illustrates the performance of the two base schemes Eqs. (5) and (6), and the three different nonlinear filters solving the conservative MHD system.

3.1. 1D shock–turbulence interactions

The first test case is the 1D compressible inviscid shock–turbulence interaction problem with initial data consisting of a shock propagating into an oscillatory density. The initial data are given by

$$(\rho_L, u_L, p_L) = (3.857143, 2.629369, 10.33333) \tag{17}$$

to the left of a shock located at $x = -4$, and

$$(\rho_R, u_R, p_R) = (1 + 0.2\sin(5x), 0, 1) \tag{18}$$

to the right of the shock where u is the velocity.

Figs. 1–4 show the comparison of MUSCL, WENO5, CEN66+WENOfi and Comp66+WENOfi using a 400-point uniform grid. The red curves are the computed solutions and the black curves are the reference solution by WENO5 using 4000 grid points. CEN66+WENOfi and Comp66+WENOfi exhibit similar accuracy and they are more accurate than MUSCL and WENO5. Computations using HY, CEN66+HYfi, Comp66+HYfi, CEN66+MUSfi and Comp66+MUSfi are not shown. The Colella and Woodward limiter is used for MUSCL, HY, and MUSfi and HYfi filters. For this particular test case, the HY solution is more accurate than MUSCL and more diffusive than WENO5. The accuracies of CEN66+MUSfi and Comp66+MUSfi are similar and are more diffusive than

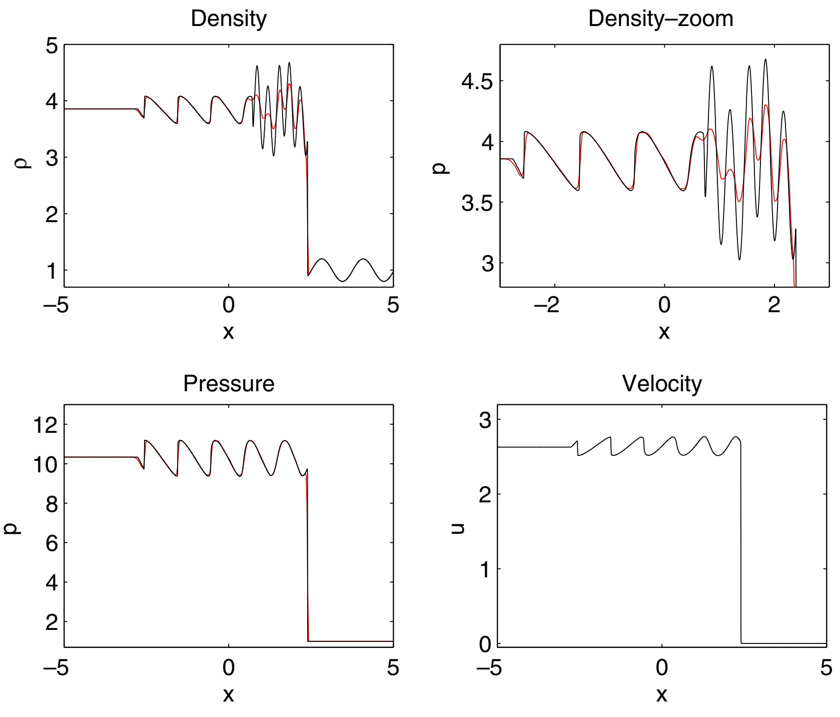


Fig. 1. 1D shock–turbulence problem: second-order MUSCL using 400 grid points. The solid black line is the reference solution.

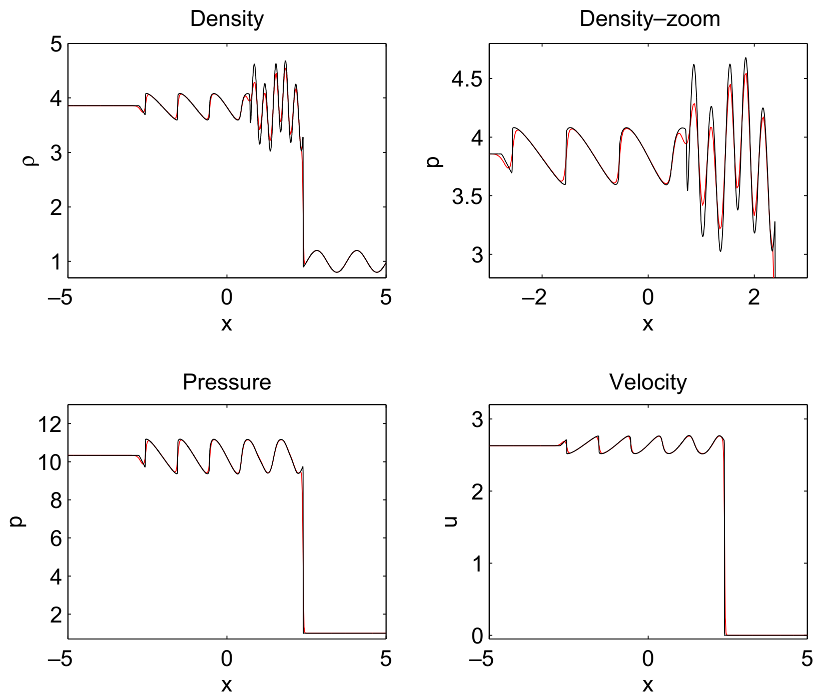


Fig. 2. 1D shock–turbulence problem: WENO5 using 400 grid points (red curve). The solid black curve is the reference solution. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

CEN66+HYfi and Comp66+HYfi. On the other hand, the accuracies of CEN66+HYfi and Comp66+HYfi are similar to CEN66+WENOfi and Comp66+WENOfi. Fig. 5 shows the computation by the 10th-order central base scheme using the WENOfi (CEN1010+WENOfi). There is only a slight gain in accuracy by the 10th-order base schemes at

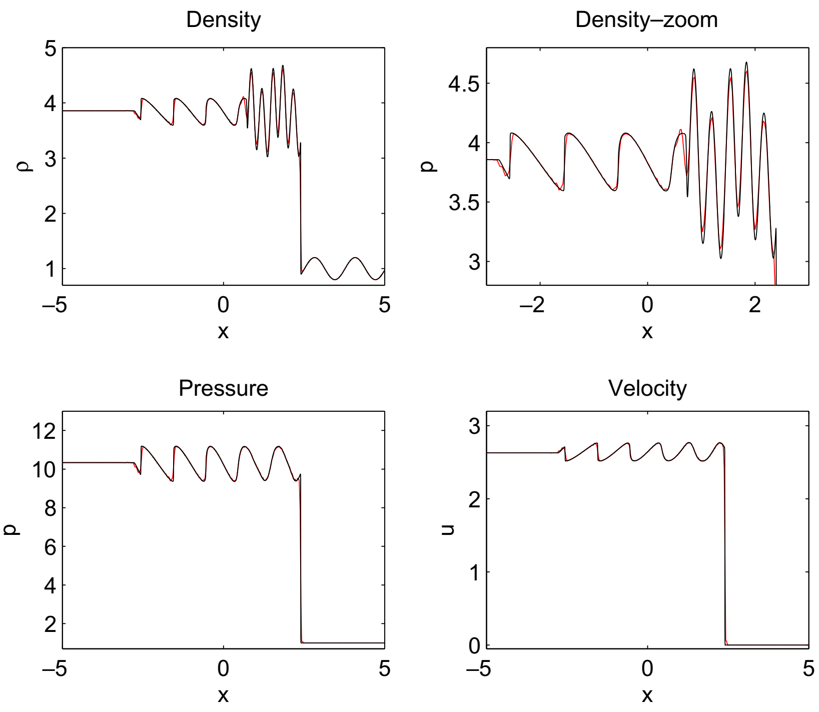


Fig. 3. 1D shock–turbulence problem: CEN66+WENOfi using 400 grid points (red curve). The solid black curve is the reference solution. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

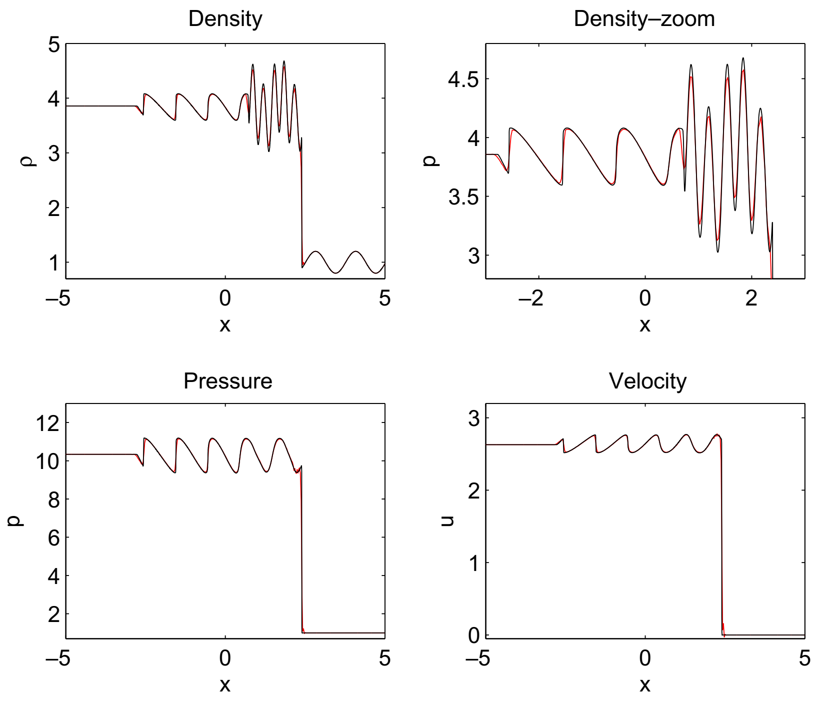


Fig. 4. 1D shock–turbulence problem: Comp66+WENOfi using 400 grid points (red curve). The solid black curve is the reference solution. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

the peaks of the oscillatory solutions. The result is similar for Comp1010+WENOfi. The very small oscillations are characteristic of these high order base schemes. If a minmod or the van Albada limiter is use in conjunction with MUSfi or HYfi, these small oscillations can be reduced. For Comp66 and Comp1010 base schemes with the same

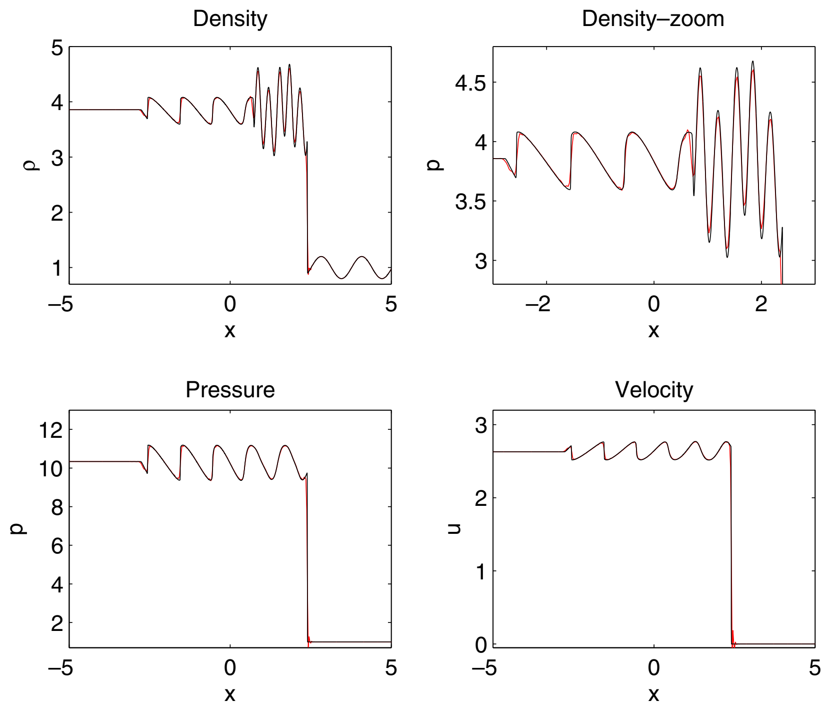


Fig. 5. 1D shock–turbulence problem: CEN1010+WENOfi using 400 grid points (red curve). The solid black curve is the reference solution. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

filter, these small oscillations can be reduced by including the compact filter step. Again, their resolutions are similar. With our filter approach, it appears that there is no visible gain in accuracy using the high order compact base scheme over the central base scheme of the same order but at the same time with the shortcoming of requiring a larger CPU time.

3.2. Magnetized Kelvin–Helmholtz instability

For the magnetized Kelvin–Helmholtz problem, the initial data are

$$\begin{aligned}
&\rho = 1,\\
&u(x,y) = 5(\tanh 20(y+0.5) - \tanh 20(y-0.5) - 1),\\
&v(x,y) = 0.25\sin 2\pi x(\mathrm{e}^{-100(y+0.5)^2} - \mathrm{e}^{-100(y-0.5)^2}),\\
&w(x,y) = 0,\\
&p = 50,\\
&B_x = 1,\\
&B_y = 0,\\
&B_z = 0.
\end{aligned}$$

We use the same $\gamma = 1.4$ as in [5], which is non-standard for plasmas. The boundaries are periodic both in the x- and y-directions with the computational domain $0 < x < 1, -1 < y < 1$. The computations stop at an evolution time $T = 0.5$ when the solution is still smooth enough that it can be solved by the base scheme alone in conjunction with a small amount of linear dissipation in Eq. (5). For the compact base scheme Eq. (6) with $d = 0$, a compact linear filter is needed. All methods considered use uniform grid spacing.

Computations by Comp66+Compfi using five grids 51×101, 101×201, 201×401, 401×801 and 801×1601 are compared. The accuracy Comp66+AD8 with $d = 0.0005$ is similar. The same computations using the sixth-order central scheme Eq. (5) with $d = 0.0005$, denoted by CEN66+AD8, were conducted. As a reference solution, computations using the eighth-order central scheme with the 10th-order linear dissipation and a dissipation coefficient of 0.0005 as the base scheme, denoted by CEN88+AD10, for the same six grids were used. There is no visible difference in accuracy between Comp66+Compfi and CEN66+AD8. Similar accuracy was obtained using either Eq. (5) or Eq. (6) as the base scheme in conjunction with any of the nonlinear filters discussed above. Fig. 6 shows a comparison.

3.3. Orszag–Tang vortex problem

The 2D compressible version of the Orszag–Tang vortex problem [3] consisting of periodic boundary conditions and smooth initial data is,

$$(\rho,u,v,w,p,B_x,B_y,B_z) = (25/9,-\sin y,\sin x,0,5/3,-\sin y,\sin 2x,0).$$

The computational domain is $0 < x < 2\pi$, $0 < y < 2\pi$ and the computation stops at time $T = 3.14$ ($\approx\pi$), when compli-

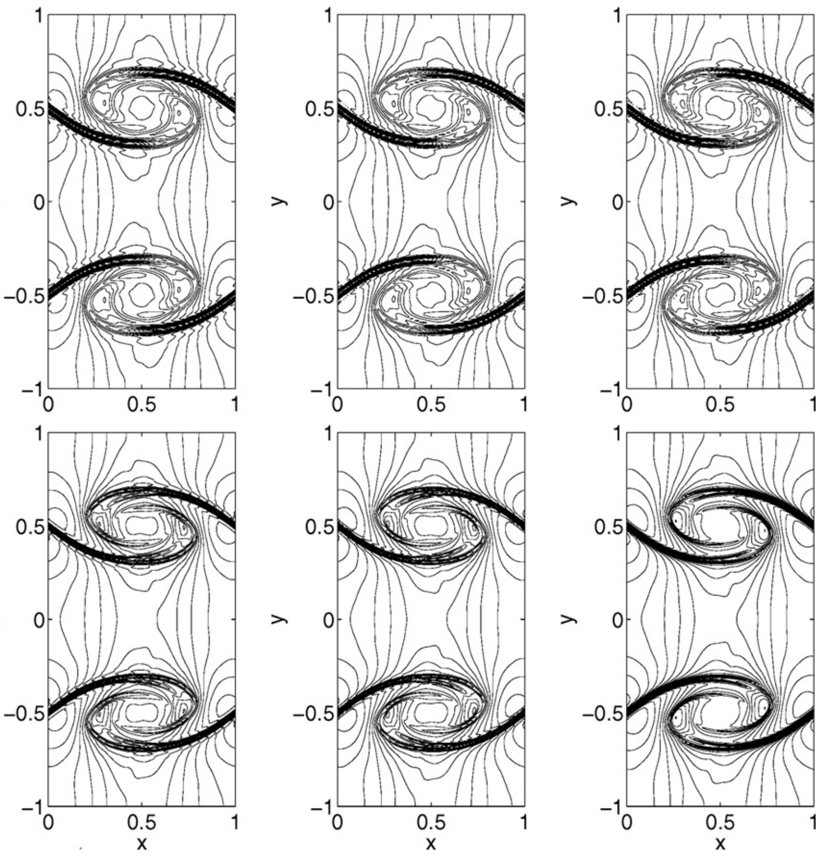


Fig. 6. Density contours of the Kelvin–Helmholtz problem at $T = 0.5$ using 101×201 (top) and 201×401 (bottom) grids by Comp66+Compfi (left) and CEN66+AD8 (middle) compared with reference solution by CEN88+AD10 (right) using 101×201 and 801×1601 grids.

cated structure and discontinuities have formed and interacted. The comparison among the three filter schemes (no filter on **B** option), WENO5, MUSCL and Harten–Yee (HY) using six uniform 51×51, 101×101, 201×201, 401×401, 801×801 and 1601×1601 grids for ideal and non-ideal MHD were conducted in [44]. Grid convergence was obtained by all six methods (WENO5, MUSCL, HY, CEN66+WENOfi, CEN66+MUSfi and CEN66+HYfi) using the 801×801 grid. Computations based on a 1601×1601 grid were used as the reference solutions. For 51×51 through 401×401 grids, small structures are better captured by the three filter methods than by WENO5, MUSCL or Harten–Yee. In addition, for the inviscid case, the three filter methods are more stable than the other three methods in the sense that a larger CFL number can be used. WENO5 and MUSCL show a slight small oscillation. These oscillations can be suppressed by applying the limiter to the characteristic variables in the MUSCL scheme (figures not shown).

For the viscous case with a Reynolds number of 1000 and a conductivity coefficient of 100, the flow structure is less complicated than the inviscid case. All computations use a CFL number of 0.6. For coarse grids, again small structures are better captured by the three filter methods than by WENO5, MUSCL or Harten–Yee. In other words, in order to exhibit similar accuracy as the three filter methods, the three standard shock-capturings methods require a finer grid. For both the inviscid and viscous computations, all three filter methods using the "no filter on B" option are divergence-free preserving, whereas the "filter all" option as well as standard WENO5, MUSCL and HY without divergence cleaning are not divergence free. Their $\nabla \cdot \mathbf{B}$ numerical error at $T = 3.14$ increases as the grid is refined. See [43] for some illustrations.

The compact base scheme in conjunction with the compact linear filter also becomes highly oscillatory. The left and middle columns of Fig. 7 show the computations by, respectively, (a) a two-step filter, Comp66+Compfi+WENOfi, and (b) a one-step filter, Comp66+WENOfi. The right column of Fig. 7 shows the same computation using CEN66+WENOfi. The small spurious oscillations by CEN66+WENOfi using the 101×101 grid can be suppressed by adding the AD8 term As the grid is refined (201×201 or larger), these small spurious oscillations vanished by CEN66+WENOfi alone without the aid of AD8. For finer grids (201×201 or larger), the numerical solutions exhibit spurious oscillations by Comp66+WENOfi but not by CEN66+WENOfi.

These spurious oscillations become more pronounced as the grid is refined. Unlike the central base scheme, it appears that the compact base scheme with the nonlinear filter alone is not able to suppress the spurious oscillation

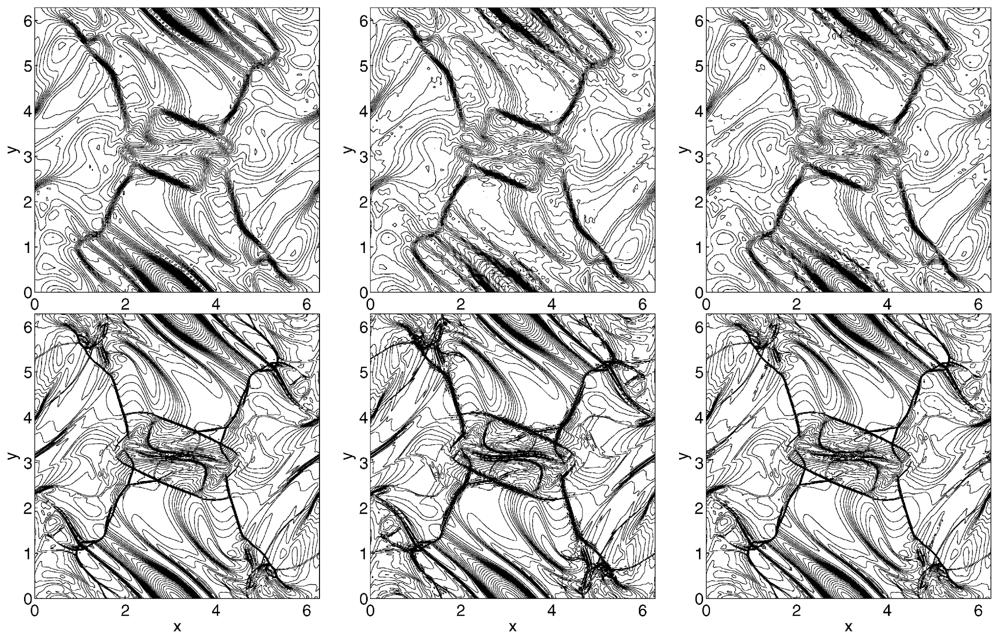


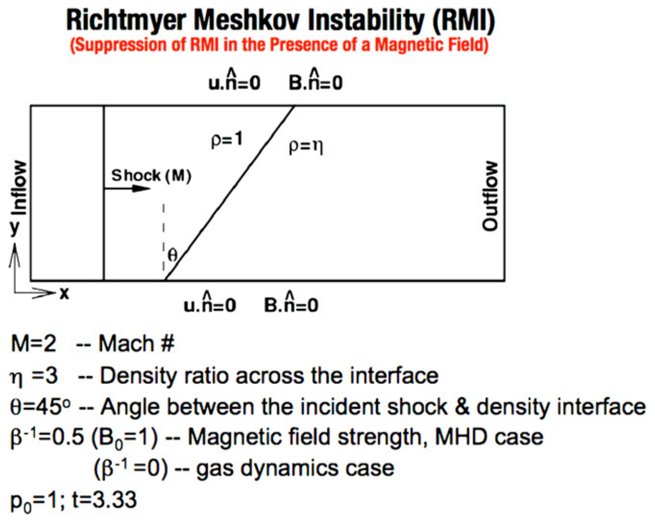
Fig. 7. Density contours of the Orszag–Tang problem at $T = 3.14$ using 101×101 (top) and 401×401 (bottom) grids by Comp66+Compfi+WENOfi (left), Comp66+WENOfi (middle) and CEN66+WENOfi (right).

completely as the grid is refined. It needs the combination of the compact linear filter and nonlinear filter (or adding AD8 as part of the compact base scheme step) to suppress the spurious oscillations. If we add $d = 0.0005$ in (b) above as part of the base scheme step, there is no visible difference in accuracy among the three methods for grids 201×201 or larger (i.e., comparing Comp66+Compfi+WENOfi, Comp66+AD8+WENOfi and CEN66+WENOfi). See [43] for the reference solution. The same comparison was performed on other test cases studied in [34,43,47] with and without physical viscosity and resistivity, and the same conclusion was arrived at as in the aforementioned two test cases. Our recent gas dynamics and MHD studies (see also the next example) arrive at the same conclusion drawn for the gas dynamics case in [40] on the behavior of compact schemes for problems containing multiscale shock interaction for the gas dynamics case.

3.4. Richtmyer–Meshkov instability (RMI) [34,47]

This study illustrates many aspects of the interplay between multiscale and multiphysics flows with numerical simulations, e.g., the suppression of the RMI in the presence of a magnetic field, and the "failure of grid refinement" for unsteady chaotic-like inviscid flows. We use the expression "achieving grid convergence" or "a mesh resolved solution" of a numerical method for a chosen mathematical model to mean that the computed solution converges to a discrete solution having the same global structure as well as same key fine scale structures when compare with well-tested commonly used scheme under grid refinement. The chosen model is assumed to have no known solution. We also use the term "failure of grid refinement" to mean:

"For a chosen model, one cannot obtain a grid convergence solution by well-tested commonly used numerical schemes with fine grid refinement. Their solution structures are different from method to method and from one grid to another and yet each scheme does not diverge during the entire time evolutionary process and grid sequence process. Aside from having very different fine scale structures, the global structures by each method do not indicate a trend of convergence to the same global feature as the grid is refined."


Richtmyer Meshkov Instability (RMI)
(Suppression of RMI in the Presence of a Magnetic Field)
u.n̂=0 B.n̂=0
ρ=1
ρ=η
Shock (M)
θ
y Inflow
Outflow
y
x
u.n̂=0 B.n̂=0
M=2 -- Mach #
η =3 -- Density ratio across the interface
θ=45° -- Angle between the incident shock & density interface
β⁻¹=0.5 (B0=1) -- Magnetic field strength, MHD case
(β⁻¹ =0) -- gas dynamics case
p0=1; t=3.33
Fig. 8. Problem definition.

RMI occurs when an incident shock accelerates an interface between two fluids of different densities. This interfacial instability was theoretically predicted by Richtmyer [29] and experimentally observed by Meshkov [21]. For the present study, the RMI problem investigated by Samtaney [30] and Wheatley et al. [38] as indicated in Fig. 8 has been chosen. The mathematical models are the 2D Euler gas dynamics equations and the ideal MHD equations. The computational domain is $-2 < x < 6$ and $0 < y < 1$. Viscous effects are considered in [47]. A planar shock at $x = -0.2$ is moving (left to right) toward the density interface with an incline angle of θ with the lower end initialized at $x = 0$. The density ratio across the interface is denoted by η, and the nondimensional strength of the magnetic field $\beta = 2p_0/B_0^2$, where the initial pressure in the preshocked region is $p_0 = 1$, and B_0 is the initial magnetic field. The initial magnetic field is uniform in the (x,y) plane and perpendicular to the incident shock front. Numerical results shown below are for $M = 2, \theta = 45°$, $\eta = 3$, $\beta^{-1} = 0$ (Euler gas dynamics) and $\beta^{-1} = 0.5$ (magnetic field present). The computation stops at an evolution time $t = 3.33$. For this set of parameters and all studied numerical schemes, instability occurs for the gas dynamics case but not for the MHD case for the entire time evolution. Our numerical results exhibit behavior similar to the study of Samtaney.

Computations by Comp66+Compfi+WENOfi using a 801×101 grid are shown in Fig. 9 (left) for the inviscid gas dynamics (GD) and the ideal MHD equations. The same computation using CEN66+WENOfi (WAV66+WENOfi) is shown in Fig. 9 (right). For the MHD case, both solutions have been converged when compared with the reference solution by CEN66+WENOfi and CEN66+HYfi using a $6401 \times$ 801 grid. For the Euler gas dynamics case, however, their resulting solutions are different and it is difficult to judge the accuracy among methods. To show their behavior, Fig. 10 shows the same comparison of gas dynamics computations using a 1601×201 grid. A finer grid refinement is needed for the gas dynamics case to evaluate the situation. Before the gas dynamics grid refinement study, we first want

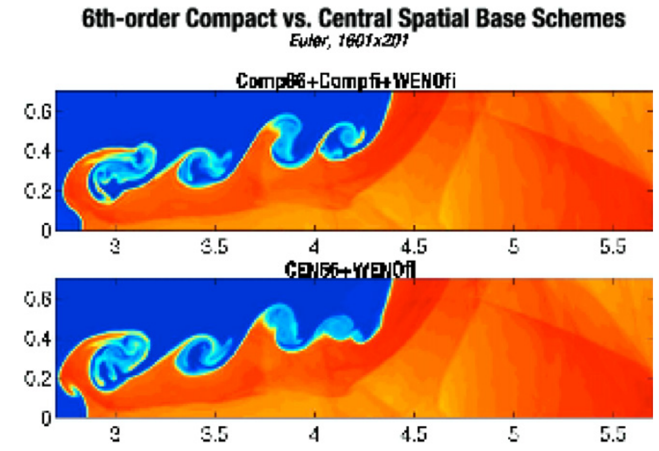


Fig. 10. Euler RMI problem. Comparison between Comp66+Compfi+WENOfi and CEN66+WENOfi using a (1601×201) grid at $t = 3.33$ for the gas dynamics case.

to show the MHD computation using half of the magnetic strength as shown previously. Fig. 11 shows the same comparison for the MHD RMI computations for half of the magnetic strength shown in Fig. 9. Again both solutions have been converged using a 801×101 grid. The bottom row of Fig. 11 shows the reference solution by CEN66+WENOfi and CEN66+HYfi using a 6401×801 grid. Computations using Comp66+WENOfi (i.e., without the linear compact filter step) or Comp66+Compfii (i.e., without the nonlinear WENOfi filter step) indicate spurious oscillations around shock regions.

Fig. 12 shows the inviscid gas dynamics comparison among a second-order MUSCL, CEN66+MUSfi, and CEN66+WENOfi and for four grids (801×101, 1601×201, 3201×401, 6401×801). Not shown is the same computation using WENO5, HY and CEN66+HYfi. The standard shock-capturing scheme MUSCL requires nearly 3 times finer grid size per spatial direction than CEN66+MUSfi, CEN66+HYfi and CEN66+WENOfi in order to exhibit similar complicated (yet different) eddy structures. The eddy structures are different among

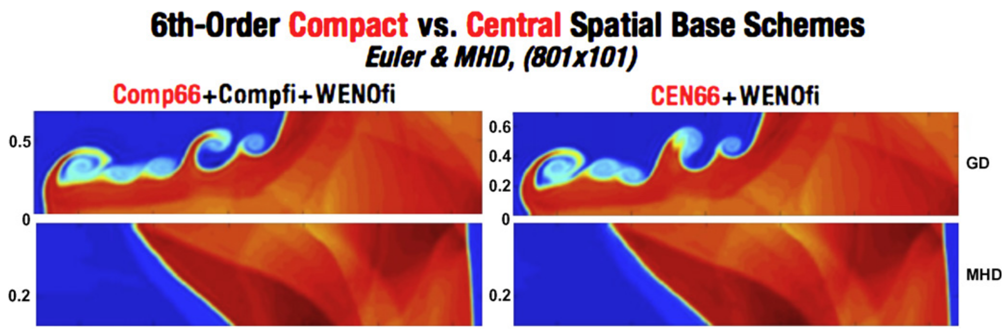


Fig. 9. RMI problem. Comparison between Euler gas dynamics and MHD for Comp66+Compfi+WENOfi and CEN66+WENOfi using a (801×101) grid at $t = 3.33$. MHD solutions shown are mirror images of the original computations.

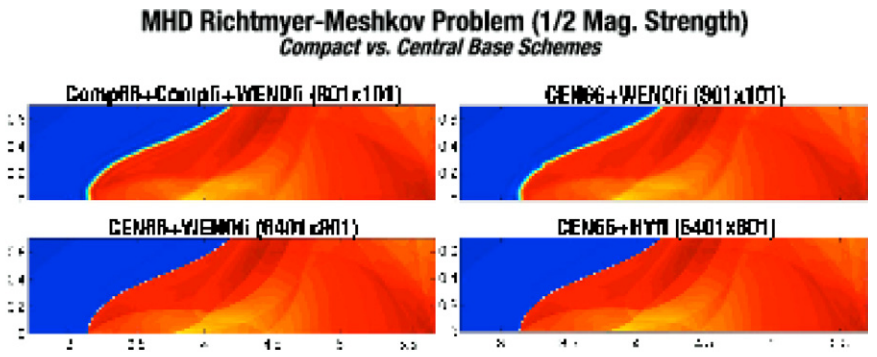


Fig. 11. MHD RMI problem. Comparison between Comp66+Compfi+WENOfi (top left) and CEN66+WENOfi (top right) using a (801 × 101) grid at $t = 3.33$ for half of the magnetic strength of the previous case. The reference solutions (bottom row) are computed by CEN66+WENOfi and CEN66+HYfi using a 6401 × 801 grid.

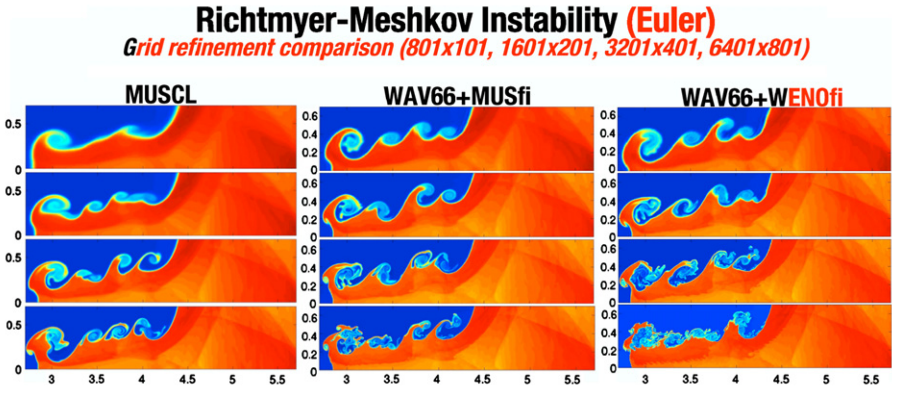


Fig. 12. RMI problem. Grid refinement study of the second-order MUSCL (left), CEN66+MUSfi (WAV66+MUSfi, middle) and CEN66+WENOfi (WAV66+WENOfi, right) at $t = 3.33$ using (801 × 101), (1601 × 201), (3201 × 401) and (6401 × 801) grids.

the three filter methods and they are very different from the Samtaney adaptive mesh refinement (AMR) simulation with an equivalent uniform grid of 16,384 × 2048. Due to the fact that the global structures of the gas dynamics case change from method to method and from grid size to grid size on the considered fine grid sequence, the different behavior of all studied methods prompted us to investigate the effect of different numerical dissipation coefficients on the eddy structures of the computed solutions.

The effect of high order linear dissipation added to the base scheme in conjunction with nonlinear filters for the RMI is reported in [47]. For completeness, Fig. 13 shows results from [47] on the effect of linear dissipation (AD8) added to the base scheme in conjunction with two different filters CEN66+AD8+WENOfi and CEN66+AD8+HYfi for four linear dissipation coefficients of AD8 $(0, 0.0005, 0.001, 0.002)$ in Eq. (5) using a (6401 × 801) grid. The top sub-figures show the computations using only a nonlinear filter without AD8 on the base scheme. The rest of the sub-figures are computations using three different non-zero AD8 coefficients. With such a fine grid, the eddy structures are very different. It appears that the considered model exhibits chaotic-like behavior. Traditionally, when dealing with non-chaotic turbulent type models, grid refinement can serve as a measure of the accuracy and convergence property of the numerical methods. However, due to the chaotic-like nature of the present Euler MRI model, the small amount of high order linear dissipation present on the spatial base scheme actually alters the type of governing equation that we are solving. In effect, we are solving the Navier–Stokes equations with a linear viscosity term. This in conjunction with the adaptive nonlinear filter (i.e., shock-capturing dissipations were employed at locations that are dictated by the wavelet flow sensor), results in a complex interplay of different types and amount of numerical dissipation which can further alter the chosen governing equations that we are solving in a nonlinear way. This manifest different chaotic-like pattern of the flow. The study can serve as a good example of failure of grid refinement for unsteady chaotic-like inviscid flow. As the grid is refined (in conjunction with different amounts and types of numerical dissipations contained in each scheme), smaller and smaller eddies are formed which combine to affect global flow through the energy cascade effect.

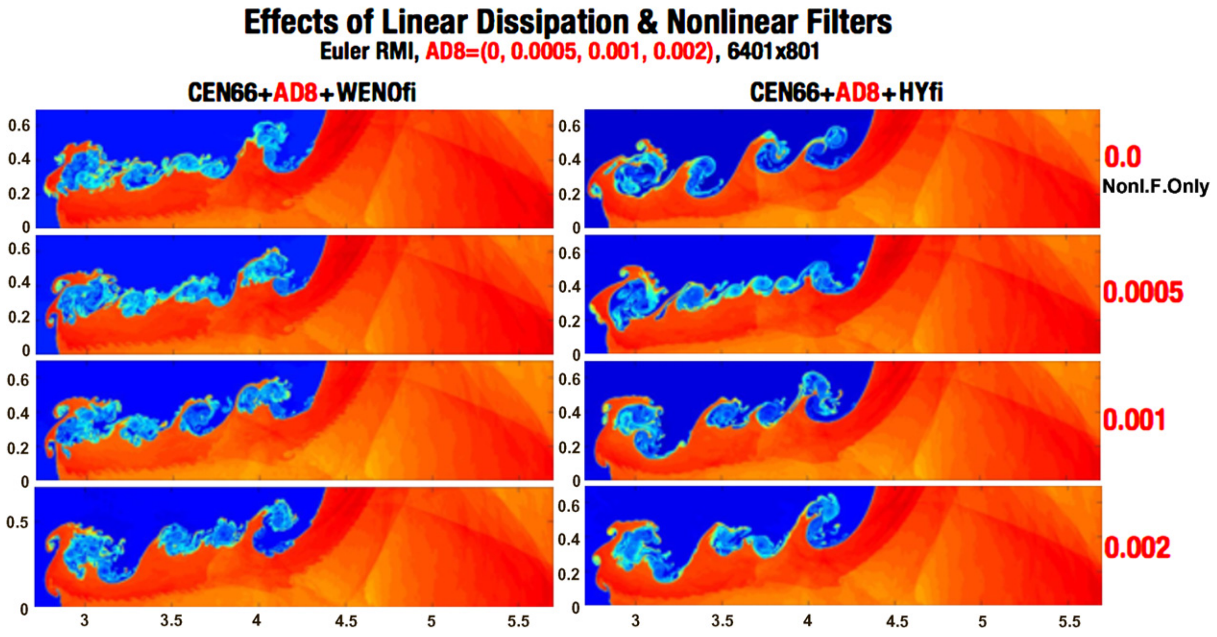


Fig. 13. RMI problem. Effect of linear dissipation (AD8) and nonlinear filter by two different filters and AD8 coefficients of $(0, 0.0005, 0.001, 0.002)$ using a (6401×801) grid.

The failure of grid refinement prompted us to investigate the same RMI problem in the presence of physical viscosity. The aim is to obtain a rough estimate of the Reynolds number when the viscous RMI ceases to exist. For Navier–Stokes computations with Reynolds numbers higher than 10,000, the same failure of grid refinement was encountered by all studied methods. In the presence of physical viscosity and for Reynolds number equal to or below 10,000, grid refinement has been achieved by all studied methods. To achieve similar resolution, MUSCL and WENO5 required more than double the grid points in each spatial direction than that of filter schemes and yet the CPU time per grid point and time step with the same grid for most of the studied methods is comparable. These results, including physical viscosity effects, are reported in [34,47].

4. Concluding remarks

Our adaptive multistep high order filter schemes employing multiresolution wavelet analysis of computed flow data to control the types and amount of numerical dissipations have been extended to include high order compact base scheme operators. This is a follow on study initiated in [45] with numerical experiments on a variety of gas dynamics and MHD multiscale test cases. Akin to the high order central spatial base scheme, the combination of the compact base scheme with multistep nonlinear and compact linear filters can capture multiscale shock interactions far better than their standard shock-capturing counterparts. However, among the various test cases, we arrive at the same conclusion drawn in [40,45] on the behavior of compact spatial schemes for problems containing multiscale shock interaction. High order compact schemes are methods of choice for many incompressible and low speed turbulent/acoustic flows due to their advantage of requiring very low number of grid points per wavelength. In the presence of multiscale shock interactions and under our filter framework, however, this desired property of high order compact base schemes seems to have diminished in both the gas dynamic and MHD test cases that we have studied (compared with the same order of accuracy of non-compact central base schemes). Also the compact spatial base scheme requires more CPU time per time step and it is less compatible with parallel computations than the central spatial base scheme. Consequently, the compact spatial base scheme requires added CPU time in a parallel computer framework.

Acknowledgements

The financial support from the NASA Fundamental Aeronautics Program for the first author is gratefully acknowledged. The authors wish to thank Datta Gaitonde for providing his compact filter boundary scheme code as a guide for implementing the linear compact filter into our MPI code.

Appendix

This includes the computer subroutines to compute the redundant form of Harten's multiresolution wavelets and the second-order B-spline wavelets and their corresponding Lipschitz exponents of a given grid function f_j (e.g., density or pressure, characteristic variables, or entropy variables). For the Harten and B-spline multiresolution wavelet formula, see [33].

```
c-------------------------------------------------------------------
      subroutine WASWITCHMR( idim, jdim, q, ind, dir, sw, case, waeps,
     *                     waezero )

*********************************************************************
***
***  Flow sensor  based on multi-resolution analysis (wavelets).
***  The routine performs a one dimensional sweep along one grid line,
***  and detects non-smoothness in the flow field.
***
***  This routine based on the ideas of multi resolution
***  by Ami Harten. Here it is made into a redundant(continuous) wavelet.
***  A simple interpolation operator is used at each
***  point, and the difference between the actual and interpolated
***  values is a wavelet coefficient.
***
***  The common block /RMN/ is included, because it is needed for case=2
***
***  Input : idim, jdim  - Dimensions of the flow field.
***          q           - The flow field
***          ind         - Index of the sweep grid line.
***          dir         - Sweep direction, 1 I-direction, q(i,ind),i=1,idim
***                                          2 J-direction. q(ind,j),j=1,jdim
***          case   - 1-Monitor pressure and density to detect
***                            non-smoothness. Same switch in all components.
***                        2-Use coefficients alpha $(=R^{-1}Delta u)$ to
***                            determine smoothness field by field.
***                        3- Use Roe state variables $p_{i+1/2},rho_{i+1/2}$.
***                        4- Use Entropy variables.
***          waeps       - Truncation level. If wavelet coefficient smaller
***                        than waeps, consider solution as smooth.
***          waezero     - If the computed Lipschitz exponent is smaller
***                        than waezero, we consider the solution as
***                        non-smooth. Values of around 0.5 is usually ok.
***
***  Output: sw          - Array of flow sensor switch parameters.
***
*********************************************************************

      integer maxi

      PARAMETER (MAXI = 644)
      COMMON /RMN/ WM(MAXI,4),ALAM(MAXI,4),AL(MAXI,4),GT(MAXI,4)
      real wm, alam, al, gt

      integer idim, jdim, case, nc, c, i, j
      integer ulim, ind, dir
      real q(idim,jdim,5), sw(maxi,4), ccof(maxi,4)
      real dcof(maxi,4,3), val(maxi,4), ccof2(maxi,4)
      real lipexp, waeps, waezero, ilog2, mx
      real gamma, entalpha, entbeta, entfrac2, fpow
      real pstar, pstarop, pressp, fracp
      real gami, igami
      real tro, uu, vv, hh, entalpha, entbeta, entfrac2, fracp
      real fpow, rrp, up, vp, pressp, ents, pstar, pstarop

      gamma = 1.4
      gami  = gamma-1
      igami = 1/gami

      if( dir.eq.1 )then
         ulim = idim
```

```
 else
    ulim = jdim
 endif

 if( case.eq.1 )then
    nc = 2
    if( dir.eq.1 )then
       do i=1,idim
          val(i,1) = q(i,ind,1)
          val(i,2) = gami*(q(i,ind,4)-0.5*(q(i,ind,2)**2+
*                    q(i,ind,3)**2)/q(i,ind,1) )
       enddo
    else
       do j=1,jdim
          val(j,1) = q(ind,j,1)
          val(j,2) = gami*(q(ind,j,4)-0.5*(q(ind,j,2)**2+
*                          q(ind,j,3)**2 )/q(ind,j,1) )
       enddo
    endif
 elseif( case.eq.2 )then
    nc = 4
    do j=1,ulim
       val(j,1) = al(j,1)
       val(j,2) = al(j,2)
       val(j,3) = al(j,3)
       val(j,4) = al(j,4)
    enddo
 elseif( case.eq.3 )then
    nc = 2
    do j=1,ulim-1
       val(j,1) = wm(j,1)*wm(j+1,1)
       tro=1.0/(wm(j,1)+wm(j+1,1))
       uu = (wm(j,2)+wm(j+1,2))*tro
       vv = (wm(j,3)+wm(j+1,3))*tro
       hh = (wm(j,4)+wm(j+1,4))*tro
       val(j,2) = val(j,1)*gamma*igami*( hh-0.5*(uu*uu+vv*vv) )
    enddo

    j = ulim
    val(j,1) = wm(j,1)
    tro      = 1.0/wm(j,1)
    uu = wm(j,2)*tro
    vv = wm(j,3)*tro
    hh = wm(j,4)*tro
    val(j,2) = val(j,1)*gamma*igami*( hh-0.5*(uu*uu+vv*vv) )

 elseif( case.eq.4 )then
    nc       = 4
    entalpha = -5.      ! pick entalpha < -gamma (best if < -1.8)
    entbeta  = (entalpha + gamma)/(1. - gamma)
    entfrac2 = 1./(1.+entbeta)
    fpow    = 1./(entalpha+gamma)
    if( dir.eq.1 )then
       do i=1,ulim
          rrp = 1./q(i,ind,1)
          up = q(i,ind,2)*rrp
          vp = q(i,ind,3)*rrp
          pressp = gami*(q(i,ind,4)-0.5*q(i,ind,1)*(up*up+vp*vp))

          ents    = pressp*((q(i,ind,1))**(-gamma))
          pstar   = -((ents)**fpow)     ! K=entbeta
          pstarop = pstar/pressp
          fracp   = entfrac2*pstarop

          val(i,1) = (q(i,ind,4) +
     *                   (entalpha-1.)*pressp/(gamma-1.))*fracp
               val(i,2) = -q(i,ind,2)*fracp
               val(i,3) = -q(i,ind,3)*fracp
               val(i,4) =  q(i,ind,1)*fracp
            enddo
         else
            do j=1,ulim
               rrp = 1./q(ind,j,1)
               up = q(ind,j,2)*rrp
               vp = q(ind,j,3)*rrp
               pressp = gami*(q(ind,j,4)-0.5*q(ind,j,1)*(up*up+vp*vp))

               ents    = pressp*((q(ind,j,1))**(-gamma))
               pstar   = -((ents)**fpow)     ! K=entbeta
               pstarop = pstar/pressp
               fracp   = entfrac2*pstarop

               val(j,1) = (q(ind,j,4) +
     *                   (entalpha-1.)*pressp/(gamma-1.))*fracp
               val(j,2) = -q(ind,j,2)*fracp
               val(j,3) = -q(ind,j,3)*fracp
               val(j,4) =  q(ind,j,1)*fracp
            enddo
         endif
      endif

*** Compute wavelet coefficients: dcof. Scaling function coeffs: ccof.
      do c=1,nc
         j = 1
         ccof(j,c) = 0.5*(3*val(j,c)-2*val(j+1,c)+val(j+2,c))
         dcof(j,c,1) = ABS(val(j,c)-ccof(j,c))
         do j=2,ulim-1
            ccof(j,c) = 0.5*(val(j-1,c)+val(j+1,c))
            dcof(j,c,1) = ABS(val(j,c)-ccof(j,c))
         enddo
         j = ulim
         ccof(j,c) = 0.5*(3*val(j,c)-2*val(j-1,c)+val(j-2,c))
         dcof(j,c,1) = ABS(val(j,c)-ccof(j,c))

         j=1
         ccof2(j,c) = 0.5*(3*ccof(j,c)-2*ccof(j+2,c)+ccof(j+4,c))
         dcof(j,c,2) = ABS(ccof2(j,c)-ccof(j,c))
         j=2
         ccof2(j,c) = 0.5*(3*ccof(j,c)-2*ccof(j+2,c)+ccof(j+4,c))
         dcof(j,c,2) = ABS(ccof2(j,c)-ccof(j,c))
         do j=3,ulim-2
            ccof2(j,c) =  0.5*(ccof(j+2,c)+ccof(j-2,c))
            dcof(j,c,2) = ABS(ccof(j,c) - ccof2(j,c))
         enddo
         j=ulim-1
         ccof2(j,c) = 0.5*(3*ccof(j,c)-2*ccof(j-2,c)+ccof(j-4,c))
         dcof(j,c,2) = ABS(ccof2(j,c)-ccof(j,c))
         j=ulim
         ccof2(j,c) = 0.5*(3*ccof(j,c)-2*ccof(j-2,c)+ccof(j-4,c))
         dcof(j,c,2) = ABS(ccof2(j,c)-ccof(j,c))


         do j=1,4
            ccof(j,c) = 0.5*(3*ccof2(j,c)-2*ccof2(j+4,c)+ccof2(j+8,c))
            dcof(j,c,3) = ABS(ccof2(j,c)-ccof(j,c))
         enddo
         do j=5,ulim-4
            ccof(j,c) =  0.5*(ccof2(j+4,c)+ccof2(j-4,c))
            dcof(j,c,3) = ABS(ccof(j,c) - ccof2(j,c))
         enddo
         do j=ulim-3,ulim
            ccof(j,c) = 0.5*(3*ccof2(j,c)-2*ccof2(j-4,c)+ccof2(j-8,c))
            dcof(j,c,3) = ABS(ccof2(j,c)-ccof(j,c))
         enddo
      enddo

*** max over cone of influence, and estimate lipschitz exponent.
*** Unrolled for better vector performance.
      ilog2 = 1.0/LOG(2.0)

      do c=1,nc

         i = 1
         ccof(i,1) = MAX( dcof(i,c,1), dcof(i+1,c,1) )
         do i=2,ulim-1
            ccof(i,1) = MAX( dcof(i-1,c,1), dcof(i,c,1), dcof(i+1,c,1))
         enddo
         i = ulim
         ccof(i,1) = MAX( dcof(i,c,1), dcof(i-1,c,1) )

         i = 1
         ccof(i,2) = MAX( dcof(i,c,2), dcof(i+1,c,2), dcof(i+2,c,2) )
         i = 2
         ccof(i,2) = MAX( dcof(i-1,c,2), dcof(i,c,2), dcof(i+1,c,2),
     *                    dcof(i+2,c,2) )
         do i=3,ulim-2
            ccof(i,2) = MAX( dcof(i-2,c,2), dcof(i-1,c,2), dcof(i,c,2),
     *                       dcof(i+1,c,2), dcof(i+2,c,2) )
         enddo
         i = ulim
         ccof(i,2) = MAX( dcof(i,c,2), dcof(i-1,c,2), dcof(i-2,c,2) )
         i = ulim-1
         ccof(i,2) = MAX( dcof(i+1,c,2), dcof(i,c,2), dcof(i-1,c,2),
     *                    dcof(i-2,c,2) )


         i = 1
         ccof(i,3) = MAX( dcof(i,c,3),   dcof(i+1,c,3), dcof(i+2,c,3),
     *                    dcof(i+3,c,3), dcof(i+4,c,3) )
         i = 2
         ccof(i,3) = MAX( dcof(i-1,c,3), dcof(i,c,3),   dcof(i+1,c,3),
     *                    dcof(i+2,c,3), dcof(i+3,c,3), dcof(i+4,c,3) )
         i = 3
         ccof(i,3) = MAX( dcof(i-2,c,3), dcof(i-1,c,3), dcof(i,c,3),
     *                    dcof(i+1,c,3), dcof(i+2,c,3), dcof(i+3,c,3),
     *                    dcof(i+4,c,3) )
         i = 4
         ccof(i,3) = MAX( dcof(i-3,c,3), dcof(i-2,c,3), dcof(i-1,c,3),
     *                    dcof(i,c,3),   dcof(i+1,c,3), dcof(i+2,c,3),
     *                    dcof(i+3,c,3), dcof(i+4,c,3) )

         do i=5,ulim-4
            ccof(i,3)=MAX( dcof(i-4,c,3), dcof(i-3,c,3), dcof(i-2,c,3),
     *                     dcof(i-1,c,3), dcof(i,c,3),   dcof(i+1,c,3),
     *                     dcof(i+2,c,3), dcof(i+3,c,3), dcof(i+4,c,3) )
         enddo
         i = ulim
         ccof(i,3) = MAX( dcof(i,c,3),   dcof(i-1,c,3), dcof(i-2,c,3),
     *                    dcof(i-3,c,3), dcof(i-4,c,3) )
         i = ulim-1
         ccof(i,3) = MAX( dcof(i+1,c,3), dcof(i,c,3),   dcof(i-1,c,3),
     *                    dcof(i-2,c,3), dcof(i-3,c,3), dcof(i-4,c,3) )
         i = ulim-2
         ccof(i,3) = MAX( dcof(i+2,c,3), dcof(i+1,c,3), dcof(i,c,3),
     *                    dcof(i-1,c,3), dcof(i-2,c,3), dcof(i-3,c,3),
     *                     dcof(i-4,c,3) )
         i = ulim-3
         ccof(i,3) = MAX( dcof(i+3,c,3), dcof(i+2,c,3), dcof(i+1,c,3),
     *                     dcof(i,c,3),   dcof(i-1,c,3), dcof(i-2,c,3),
     *                     dcof(i-3,c,3), dcof(i-4,c,3) )

         do i=1,ulim
            sw(i,c) = 1
            if( ccof(i,1).lt.waeps )then
               lipexp = 1
            elseif( ccof(i,3).lt.waeps )then
               lipexp = 0
            else
               lipexp = 0.5*ilog2*log(ccof(i,3)/ccof(i,1))
            endif
c            sw(i,c) = MAX( 0.0, (waezero-lipexp)/waezero )
            if( lipexp.gt.waezero )then
               sw(i,c) = 0
            endif
         enddo
      enddo

      if( nc.eq.2 )then
         do i=1,ulim
             mx = MAX( sw(i,1), sw(i,2) )
             sw(i,1) = mx
             sw(i,2) = mx
             sw(i,3) = mx
             sw(i,4) = mx
         enddo
      elseif( nc.lt.4 )then
       write(*,*) 'WASWITCHMR: Warning, incomplete number of components'
      endif

      return
      end




c-----------------------------------------------------------------------
      subroutine WASWITCHGE( idim, jdim, q, ind, dir, sw, case, waeps,
     *                       waezero )

***********************************************************************
***
***  Flow sensor based on multi-resolution analysis (wavelets).
***  The routine performs a one dimensional sweep along one grid line,
***  and detects non-smoothness in the flow field.
***  This routine based on the spline wavelets,
***  described by Mallat, and later used by Gerritsen for shock detection.
***
***  The common block /RMN/ is included, because it is needed for case=2
***
***  Input : idim, jdim  - Dimensions of the flow field.
***          q           - The flow field
***          ind         - Index of the sweep grid line.
***          dir         - Sweep direction, 1 I-direction, q(i,ind),i=1,idim
***                                         2 J-direction. q(ind,j),j=1,jdim
***          case   - 1-Monitor pressure and density to detect
***                        non-smoothness. Same switch in all components.
***                      2-Use coefficients alpha $(=R^{-1}Delta u)$ to
***                        determine smoothness field by field.
***                      3- Use Roe state variables $p_{i+1/2},rho_{i+1/2}$.
***                      4- Use Entropy variables.
***          waeps       - Truncation level. If wavelet coefficient smaller
***                        than waeps, consider solution as smooth.
***          waezero     - If the computed Lipschitz exponent is smaller
***                        than waezero, we consider the solution as
***                        non-smooth. Values of around 0.5 is usually ok.
***
***  Output: sw          - Array of flow sensor switch parameters.
***
************************************************************************

      integer maxi
      PARAMETER (MAXI = 644)
      COMMON /RMN/ WM(MAXI,4),ALAM(MAXI,4),AL(MAXI,4),GT(MAXI,4)
      real wm, alam, al, gt

      integer idim, jdim, case, nc, c, i, j
      integer ulim, ind, dir
      real q(idim,jdim,5), sw(maxi,4), ccof(maxi,4)
      real dcof(maxi,4,3), val(maxi,4), ccof2(maxi,4)
      real lipexp, waeps, waezero, ilog2, mx, gamma, gami, igami
      real tro, uu, vv, hh, entalpha, entbeta, entfrac2, fracp
      real fpow, rrp, up, vp, pressp, ents, pstar, pstarop

*** Insert lines from previous subroutines WASWITCHMR

      gama=1.4
      .
      .
      .
      the line before "Compute wavelet coefficients: ......"


*** Compute wavelet coefficients: dcof. Scaling function coeffs: ccof.
      do c=1,nc

*** first level
         j = 1
         ccof(j,c)   = 0.125*( 7*val(j,c)   - 3*val(j+1,c)+
     *                         5*val(j+2,c) -   val(j+3,c)  )
         dcof(j,c,1) = 0.5*ABS( val(j,c)-val(j+1,c) )
         do j=2,ulim-2
            ccof(j,c) = 0.125*( val(j-1,c) + 3*val(j,c) + 3*val(j+1,c) +
     *                          val(j+2,c) )
            dcof(j,c,1) = 0.5*ABS( val(j-1,c)-val(j,c) )
         enddo
         j = ulim-1
         ccof(j,c)   = 0.125*(  val(j+1,c) + 3*val(j,c) +
     *                        3*val(j-1,c) +   val(j-2,c) )
         dcof(j,c,1) = 0.5*ABS( val(j,c)-val(j-1,c) )

         j = ulim
         ccof(j,c)   = 0.125*( 7*val(j,c)   - 3*val(j-1,c) +
     *                         5*val(j-2,c) -   val(j-3,c)   )
         dcof(j,c,1) = 0.5*ABS( val(j,c)-val(j-1,c) )


*** second level
         j=1
         ccof2(j,c) =  0.125*( 7*ccof(j,c)   - 3*ccof(j+2,c) +
     *                         5*ccof(j+4,c) -   ccof(j+6,c)   )
         dcof(j,c,2) = 0.5*ABS( ccof(j+2,c)-ccof(j,c) )

         j=2
         ccof2(j,c) = 0.125*( 7*ccof(j,c)   - 3*ccof(j+2,c) +
     *                        5*ccof(j+4,c) -   ccof(j+6,c)   )
```

```
          dcof(j,c,2) = 0.5*ABS( ccof(j+2,c)-ccof(j,c) )

          do j=3,ulim-4
             ccof2(j,c) =  0.125*( ccof(j-2,c)+3*ccof(j,c)+
     *                             3*ccof(j+2,c)+ccof(j+4,c) )
             dcof(j,c,2) = 0.5*ABS(ccof(j-2,c) - ccof(j,c))
          enddo

          j=ulim-3
          ccof2(j,c) = 0.125*(  ccof(j+2,c) + 3*ccof(j,c) +
     *                        3*ccof(j-2,c) +   ccof(j-4,c) )
          dcof(j,c,2) = 0.5*ABS( ccof(j-2,c)-ccof(j,c) )

          j=ulim-2
          ccof2(j,c) = 0.125*(  ccof(j+2,c) + 3*ccof(j,c) +
     *                        3*ccof(j-2,c) +   ccof(j-4,c) )
          dcof(j,c,2) = 0.5*ABS( ccof(j-2,c)-ccof(j,c) )

          j=ulim-1
          ccof2(j,c) = 0.125*( 7*ccof(j,c)   - 3*ccof(j-2,c) +
     *                         5*ccof(j-4,c) -   ccof(j-6,c)   )
          dcof(j,c,2) = 0.5*ABS( ccof(j-2,c)-ccof(j,c) )

          j=ulim
          ccof2(j,c) =  0.125*( 7*ccof(j,c)   - 3*ccof(j-2,c) +
     *                          5*ccof(j-4,c) -   ccof(j-6,c)   )
          dcof(j,c,2) = 0.5*ABS( ccof(j-2,c)-ccof(j,c) )



*** Third level
          do j=1,4
              ccof(j,c) = 0.125*( 7*ccof2(j,c) - 3*ccof2(j+4,c) +
     *                            5*ccof2(j+8,c) - ccof2(j+12,c) )
              dcof(j,c,3) = 0.5*ABS( ccof2(j+4,c)-ccof2(j,c) )
          enddo

          do j=5,ulim-8
             ccof(j,c) =  0.125*( ccof2(j+8,c) + 3*ccof2(j+4,c)+
     *                            3*ccof2(j,c) + ccof2(j-4,c)    )
             dcof(j,c,3) = 0.5*ABS( ccof2(j-4,c) - ccof2(j,c) )
          enddo

          do j=ulim-7,ulim-4
             ccof(j,c) = 0.125*( ccof2(j+4,c) + 3*ccof2(j,c) +
     *                         3*ccof2(j-4,c) +   ccof2(j-8,c)   )
             dcof(j,c,3) = 0.5*ABS(ccof2(j,c)-ccof2(j-4,c))
          enddo
          do j=ulim-3,ulim
             ccof(j,c) = 0.125*( 7*ccof2(j,c)  - 3*ccof2(j-4,c) +
     *                           5*ccof2(j-8,c) -  ccof2(j-12,c)   )
             dcof(j,c,3) = 0.5*ABS(ccof2(j,c)-ccof2(j-4,c))
          enddo

       enddo

*** max over cone of influence, and estimate lipschitz exponent.
*** Unrolled for better vector performance.
       ilog2 = 1.0/LOG(2.0)

       do c=1,nc

          i = 1
          ccof(i,1) = MAX( dcof(i,c,1), dcof(i+1,c,1) )
          do i=2,ulim-1
```

```
         ccof(i,1) = MAX( dcof(i-1,c,1), dcof(i,c,1), dcof(i+1,c,1))
      enddo
      i = ulim
      ccof(i,1) = MAX( dcof(i,c,1), dcof(i-1,c,1) )

      i = 1
      ccof(i,2) = MAX( dcof(i,c,2), dcof(i+1,c,2), dcof(i+2,c,2) )
      i = 2
      ccof(i,2) = MAX( dcof(i-1,c,2), dcof(i,c,2), dcof(i+1,c,2),
     *                 dcof(i+2,c,2) )
      do i=3,ulim-2
         ccof(i,2) = MAX( dcof(i-2,c,2), dcof(i-1,c,2), dcof(i,c,2),
     *                   dcof(i+1,c,2), dcof(i+2,c,2) )
      enddo
      i = ulim
      ccof(i,2) = MAX( dcof(i,c,2), dcof(i-1,c,2), dcof(i-2,c,2) )
      i = ulim-1
      ccof(i,2) = MAX( dcof(i+1,c,2), dcof(i,c,2), dcof(i-1,c,2),
     *                 dcof(i-2,c,2) )


      i = 1
      ccof(i,3) = MAX( dcof(i,c,3),   dcof(i+1,c,3), dcof(i+2,c,3),
     *                 dcof(i+3,c,3), dcof(i+4,c,3) )
      i = 2
      ccof(i,3) = MAX( dcof(i-1,c,3), dcof(i,c,3),   dcof(i+1,c,3),
     *                 dcof(i+2,c,3), dcof(i+3,c,3), dcof(i+4,c,3) )
      i = 3
      ccof(i,3) = MAX( dcof(i-2,c,3), dcof(i-1,c,3), dcof(i,c,3),
     *                 dcof(i+1,c,3), dcof(i+2,c,3), dcof(i+3,c,3),
     *                 dcof(i+4,c,3) )
      i = 4
      ccof(i,3) = MAX( dcof(i-3,c,3), dcof(i-2,c,3), dcof(i-1,c,3),
     *                 dcof(i,c,3),   dcof(i+1,c,3), dcof(i+2,c,3),
     *                 dcof(i+3,c,3), dcof(i+4,c,3) )

      do i=5,ulim-4
         ccof(i,3)=MAX( dcof(i-4,c,3), dcof(i-3,c,3), dcof(i-2,c,3),
     *                  dcof(i-1,c,3), dcof(i,c,3),   dcof(i+1,c,3),
     *                  dcof(i+2,c,3), dcof(i+3,c,3), dcof(i+4,c,3) )
      enddo
      i = ulim
      ccof(i,3) = MAX( dcof(i,c,3),   dcof(i-1,c,3), dcof(i-2,c,3),
     *                 dcof(i-3,c,3), dcof(i-4,c,3) )
      i = ulim-1
      ccof(i,3) = MAX( dcof(i+1,c,3), dcof(i,c,3),   dcof(i-1,c,3),
     *                 dcof(i-2,c,3), dcof(i-3,c,3), dcof(i-4,c,3) )
      i = ulim-2
      ccof(i,3) = MAX( dcof(i+2,c,3), dcof(i+1,c,3), dcof(i,c,3),
     *                 dcof(i-1,c,3), dcof(i-2,c,3), dcof(i-3,c,3),
     *                 dcof(i-4,c,3) )
      i = ulim-3
      ccof(i,3) = MAX( dcof(i+3,c,3), dcof(i+2,c,3), dcof(i+1,c,3),
     *                 dcof(i,c,3),   dcof(i-1,c,3), dcof(i-2,c,3),
     *                 dcof(i-3,c,3), dcof(i-4,c,3) )

      do i=1,ulim
         sw(i,c) = 1
         if( ccof(i,1).lt.waeps )then
            lipexp = 1
         elseif( ccof(i,3).lt.waeps )then
            lipexp = 0
         else
            lipexp = 0.5*ilog2*log(ccof(i,3)/ccof(i,1))
         endif
```

```
c             sw(i,c) = MAX( 0.0, (waezero-lipexp)/waezero )
            if( lipexp.gt.waezero )then
               sw(i,c) = 0
            endif
         enddo
      enddo

      if( nc.eq.2 )then
         do i=1,ulim
             mx = MAX( sw(i,1), sw(i,2) )
             sw(i,1) = mx
             sw(i,2) = mx
             sw(i,3) = mx
             sw(i,4) = mx
         enddo
      elseif( nc.lt.4 )then
       write(*,*) 'WASWITCHGE: Warning, incomplete number of components'
      endif

      return
      end
```

References

[1] Barone MF, Roy CJ. Evaluation of detached eddy simulation for turbulent wake applications, AIAA Paper 2005-0504. In: 43rd aerospace science meeting and exhibit; 2005.

[2] Brio M, Wu CC. An upwind differencing scheme for the equations of ideal magnetohydrodynamics. J Comput Phys 1988;75:400–22.

[3] Dai W, Woodward PR. A simple finite difference scheme for multidimensional magnetohydrodynamical equations. J Comput Phys 1998;142:331–69.

[4] Daubechies I. Ten lectures on wavelets. CBMS-NSF regional conference series in applied mathematics, vol. 61. SIAM; 1992.

[5] Dedner A, Kemm F, Kröner F, Munz C-D, Schnitzer T, Wesenberg M. Hyperbolic divergence cleaning for the MHD equations. J Comput Phys 2002;175:645–73.

[6] Evans CR, Hawley JF. Simulation of magnetohydrodynamic flows: a constrained transport method. Astrophys J 1988;332:659–77.

[7] Farge M. Wavelet transforms and their applications to turbulence. Ann Rev Fluid Mech 1992;24:395.

[8] Gaitonde DV, Visbal MR. Further development of a Navier–Stokes solution procedure based on higher-order formulas, AIAA Paper 99-0557; 1999.

[9] Gaitonde DV. Development of a solver for 3D non-ideal magnetogasdynamics, AIAA Paper 99-3610; 1999.

[10] Gallice G. Systéme D'Euler–Poisson, Magnétohydrodynamique et Schemas de Roe, PhD Thesis, L'Université Bordeaux I; 1997.

[11] Gerritsen M, Olsson P. Designing an efficient solution strategy for fluid flows I. J Comput Phys 1996;129:245–62.

[12] Godunov SK. Symmetric form of the equations of magnetohydrodynamics. Numer Methods Mech Continuum Medium 1972;13(1): 26–34.

[13] Harten A. Multiresolution algorithms for the numerical solution of hyperbolic conservation laws. Commun Pure Appl Math 1995;48: 1305–42.

[14] Harten A, Hyman J. Self-adjusting grid methods for one-dimensional hyperbolic conservation laws. J Comput Phys 1983;50:235.

[15] Jiang G-S, Shu C-W. Efficient implementation of weighted ENO schemes. J Comput Phys 1996;126:202.

[16] Maday Y, Patera AT. Spectral element methods for the Navier–Stokes equations. In: Noor AK, editor. State of the art surveys in computational mechanics. New York: ASME; 1989. p. 71–143.

[17] Mallat SG. A wavelet tour of signal processing. Second ed. San Diego: Academic Press; 1999.

[18] Mallat SG. A theory for multiresolution signal decomposition: the wavelet representation. IEEE Trans Pattern Anal Mach Intell 1989;11:674.

[19] Mallat SG, Hwang WL. Singularity detection and processing with wavelets. IEEE Trans Information Theory 1992;38:617.

[20] Mallat SG, Zhong S. Characterization of signals from multiscale edges. IEEE Trans Pattern Anal Mach Intell 1992;14:710–32.

[21] Meshkov YY. Instability of a shock wave accelerated interface between two gases, NASA Tech Trans NASA TT F-13074; 1970.

[22] Müller B, Yee HC. Entropy splitting for high order numerical simulation of vortex sound at low Mach Numbers. In: Proceedings of the fifth international conference on spectral and high order methods; 2001.

[23] Nessyahu H, Tadmor E. Non-oscillatory central differencing for hyperbolic conservation laws. J Comput Phys 1990;87:408–63.

[24] Nordstrom J, Carpenter MH. Boundary and interface conditions for high-order finite-difference schemes applied to the Euler and Navier–Stokes equations. J Comput Phys 1999;148:621–45.

[25] Olsson P. Summation by parts, projections and stability, I. Math Comput 1995;64:1035–65.

[26] Perrier V, Philipovitch T, Basdevant C. Wavelet spectra compared to Fourier spectra. Paris: Publication of ENS; 1999.

[27] Powell KG. An approximate Riemann solver for magnetohydrodynamics (that works in more than one dimension). ICASE-Report 94-24, NASA Langley Research Center, April 1994.

[28] Qiu J, Khoo BC, Shu C-W. A numerical study for the performance of the Runge–Kutta discontinuous Galerkin method based on different numerical fluxes. J Comput Phys 2006;212:540–65.

[29] Richtmyer RD. Taylor instability in shock acceleration of compressible fluids. Commun Pure Appl Math 1960;13:297.

[30] Samtaney R. Suppression of the Richtmyer–Meshkov instability in the presence of a magnetic field. Phys Fluids 2003;15:L53–6.

[31] Sjögreen B. High order centered difference methods for the compressible Navier–Stokes equations. J Comput Phys 1995;117: 67–78.

[32] Sjögreen B. Numerical experiments with the multiresolution scheme for the compressible Euler equations. J Comput Phys 1995;117:251.

[33] Sjögreen B, Yee HC. Multiresolution wavelet based adaptive numerical dissipation control for shock-turbulence computation. RIACS Technical Report TR01.01, NASA Ames Research Center (October 2000) [also J Sci Comput 2004;20:211–55].

[34] Sjögreen B, Yee HC. Performance of high order filter methods for a Richtmyer–Meshkov instability. In: Proceedings of the ICCFD4 conference; 2006.

[35] Strang G. Wavelet transforms versus Fourier Transforms. Bull Am Math Soc (NS) 1993;28:288.

[36] Strang G. Wavelet and dilation equations: a brief introduction. SIAM Rev 1989;31:614.

[37] Vinokur M, Yee HC. Extension of efficient low dissipative high order schemes for 3D curvilinear moving grids. NASA TM 209598, June 2000 [also Caughey DA, Hafez M, editors. Frontiers of computational fluid dynamics, World Scientific; 2002. p. 129–64].

[38] Wheatley V, Pullin DI, Samtaney R. Regular shock refraction at an oblique planar density interface in magnetohydrodynamics. J Fluid Mech 2005;522:179–214.

[39] Yee HC. A class of high-resolution explicit and implicit shock-capturing methods. VKI Lecture Series 1989–04 [NASA TM-101088, February 1989].

[40] Yee HC, Sandham ND, Djomehri MJ. Low dissipative high order shock-capturing methods using characteristic-based filters. J Comput Phys 1999;150:199–238.

[41] Yee HC, Sjögreen B. Designing adaptive low dissipative high order schemes for long-time integrations. In: Drikakis D, Geurts B, editors. Turbulent flow computation. Kluwer Academic Publishers; 2002 [also RIACS Technical Report TR01-28, December 2001].

[42] Yee HC, Sjögreen B. Divergence free high order filter methods for the compressible MHD equations. In: Proceedings of the international conference on high performance scientific computing; 2003.

[43] Yee HC, Sjögreen B. Efficient low dissipative high order scheme for multiscale MHD flows, II: minimization of Div(B) numerical error. RIACS Technical Report TR03.10, July, 2003. NASA Ames Research Center [J Sci Comput 2005, doi:10.1007/s10915-005-9004-5 2006;29:115–64].

[44] Yee HC, Sjögreen B. Nonlinear filtering and limiting in high order methods for ideal and non-ideal MHD. Proceedings of the ICOSAHOM. J Sci Comput 2006:507–21.

[45] Yee HC, Sjögreen B. Nonlinear filtering in compact high order schemes. Proceedings of the 19th ICNSP and 7th APPTC Conference. J Plasma Phys 2006;72:833–6.

[46] Yee HC, Sjögreen B. development of low dissipative high order filter schemes for multiscale Navier–Stokes/MHD systems. In: Proceedings of the calSpace/UCR ASTRONUM conference; 2006. J. Comput. Physics, 2007, in press, http://dx.doi.org/10.1016/j.jcp.2007.01.012.

[47] Yee HC, Sjögreen B. Simulation of Richtmyer–Meshkov instability by sixth-order filter methods. In: Proceedings of the 17th international shock interaction symposium. Shock Waves J, in press, doi:10.1007/s00193-007-0104-z.