ACCELERATING MULTIRESOLUTION SCHEMES WITH FULLY-ADAPTIVE-BLOCK REFINEMENT: APPLICATIONS TO REACTIVE FLOWS

A PREPRINT

Brandon Gusto
Department of Scientific Computing
Florida State University
Tallahassee, FL 32306
bgusto@fsu.edu

Tomasz Plewa
Department of Scientific Computing
Florida State University
Tallahassee, FL 32306
tplewa@fsu.edu

August 4, 2019

ABSTRACT

We present a generalization of Harten's original multiresolution scheme for simulating reactive flows on logically rectangular block-structured adaptive mesh refinement (AMR) grids in one and two dimensions. The scheme addresses a major shortcoming of tree-based AMR codes, which is the creation of blocks with a low filling factor; that is, many cells in such a block are resolved beyond the desired error tolerance, necessitating excessive computational resources. To overcome this issue, we introduce a multiresolution representation of the solution, not only to adapt the grid but also to adaptively compute fluxes and sources on each block. The scheme recycles regularity information obtained by the multiresolution grid adaptation in order to select flux and source calculations which may be accurately replaced by interpolation from the multiresolution basis. A block which employs this scheme is denoted as a fully adaptive block (FAB). The error introduced by this approximation is shown to be of the same order as the local truncation error of the reconstruction scheme. Thus the rate of convergence of the underlying spatial reconstruction scheme is preserved. Additionally with respect to parallel applications, the multiresolution transform and computation of fluxes and sources on FABs is asynchronous, requiring only one synchronization step which is equivalent to the filling of ghost cells for each block. The efficiency of the scheme is demonstrated using several one and two-dimensional problems.

***K*eywords** Multiresolution · Adaptive Mesh Refinement · Reactive Flows

1 Introduction

Highly energetic and reacting flows are often characterized by disparate spatial and temporal length scales. Features such as shockwaves and contact discontinuities, for instance, require vastly more resolution than do smooth regions of the flow. Naturally, intense effort has gone into the development of methods which employ some type of adaptive strategy. The purpose of an adaptive strategy is to accurately simulate such flows without expending excess computational resources in relatively smooth regions of the flow.

The most popular strategy to accurately capture regions of interest in fluid simulations is to introduce a non-uniform spatial grid, with higher resolution near discontinuities or sharp gradients. Methods which introduce a hierarchy of nested grid resolutions are generally described as adaptive mesh refinement (AMR) methods. AMR methods, first introduced in [**?**], usually rely on estimates of the local truncation error (LTE) of the numerical scheme to determine regions where refinement is necessary for solution accuracy. Some more simple strategies may refine based on the value of the gradient, or concentration of some species of interest. For a full review of the LTE estimators and refinement criterion, readers are referred to blank.

Regarding the implementation of AMR methods on large networks of parallel computers, certain engineering realities have neccessitated the reduction in granularity of the adaptive refinement. To use a single computational cell as the unit for refinement (i.e. cell-based refinement) introduces a number of costly compromises. Firstly, such an adapted grid requires the reconstruction method of choice to utilize nonuniform stencils, requiring increased computational resources. More significantly, the cell-based refinement requires costly data traversal. Traversing tree space requires on average $\mathcal{O}(n^d)$ (confirm this?) operations, where n is the number of cells per dimension, d. Thus most AMR codes make use of some type of block-based approach. Tree-based block-structured codes, where each block consists of a fixed number of cells, are a very popular choice. These types of approaches are implemented in a number of AMR libraries including Paramesh (cite), p4est (cite, and others. This approach allows for simple mesh management procedures, and scales well for very large numbers of processors in parallel. One clear drawback however is the gradedness of the tree, which necessitates that no branch can have an incomplete set of children. This typically leads to refinement of many blocks which would not be otherwise flagged by refinement indicators. A further complication imposed by most finite volume solvers is that there can not be a jump in refinement greater than one level. Together these consequences of AMR represent a non-trivial decrease in performance due to the fact that the mesh is not optimal (in some sense).

Alternate approaches to dynamic grid adaptation based on wavelet theory became popular after the seminal papers by Harten [[1],] were introduced. In this work, a multiresolution representation of the discrete solution on a uniform grid was used for adaptively computing the divergence of the flux within a finite volume framework. Rather than adapt the grid directly, the idea was to accelerate the computation of fluxes using the multiresolution information. In this approach, eligible fluxes in sufficiently smooth regions are interpolated from fluxes obtained at interfaces corresponding to coarser grid levels. The original scheme was applied solely to hyperbolic conservation laws in one spatial dimension, but was then expanded by Bihari et. al. to two-dimensional simulations in (citation), followed by the inclusion of viscous terms in (citation), and then to source terms in the context of reactive flows in (bihari). These works retained the original spirit of Harten's scheme, which was to evolve the solution on a uniform grid, but use multiresolution information to identify regions where flux (and/or source term) computations may be avoided.

Although Harten's original scheme was intended to be an alternative to spatially non-uniform grid adaptation, a series of papers have since reintroduced the concept of non-uniform grids within the MR framework. Thus the AMR approach was essentially redeveloped but with the refinement criterion defined by the MR representation rather than with the traditional metrics mentioned previously. The first fully adaptive scheme was presented by Cohen et. al. to study hyperbolic conservation laws in two dimensions in (cite). More recently, Rossinilli et. al. explored the use of wavelet-based refinement indicators for block-based adaptation.

In the following work, we present a novel block-based adaptive mesh refinement scheme using wavelet-based indicators, where Harten's scheme is used to essentially treat each block as a uniform grid. The indicators are used for two purposes: (1) refinement, and (2) adaptively calculating fluxes and source terms. We show how this scheme costs essentially nothing in additional computation (the MR information is recycled after adaptation), while not impacting the accuracy of the solver.

2 Conservation Laws and Finite Volume Discretization

In the present work we are interested in numerically solving conservation laws of the form

$$\begin{cases} u_t + f(u)_x = s(u) \\ u(x,0) = u_0(x), \end{cases} \tag{1}$$

where $x \in \Omega$, $t \in [t_0, t_{fin}]$, u represents a conserved quantity, $f(u)$ is the flux function, and $s(u)$ is a source term. For the sake of presentation, we let the scalar equation (1) stand in for the more complicated systems of conversation laws which will be the focus of our applications. In the finite volume formulation, the solution $u(x,t)$ is approximated by volume averages defined within each cell $I_i = \left[x_i - \frac{h}{2}, x_i + \frac{h}{2}\right]$ in the computational domain. The cell width h is determined by the number of cells, N, and the size of the domain $\Omega \in [a,b]$. The cell averages are given by

$$\overline{u}_i(t) = \frac{1}{h} \int_{x_{i-1/2}}^{x_{i+1/2}} u(\xi, t) d\xi, \tag{2}$$

where for convenience we use $i \pm 1/2$ as subscripts to indicate the left and right interfaces of the target cell (i.e. $x_{i+1/2} = x_i + \frac{h}{2}$). The governing equations are cast into the semi-discrete conservative form,

$$\frac{d\overline{u}_i(t)}{dt} = L(\boldsymbol{u}) = -\frac{1}{h}\left(\hat{f}_{i+1/2} - \hat{f}_{i-1/2}\right) + \overline{s}_i, \tag{3}$$

where $\overline{s} = \frac{1}{h}\int_{x_{i-1/2}}^{x_{i+1/2}} s(\xi)d\xi$, and the numerical flux is evaluated as

$$\hat{f}_{i\pm1/2} = \hat{f}(u^-_{i\pm1/2}, u^+_{i\pm1/2}). \tag{4}$$

The states $u^-_{i\pm1/2}$ and $u^+_{i\pm1/2}$ indicate the approximate value of the quantity u immediately to the left and right of the cell interface $x_{i\pm1/2}$. These values are reconstructed from a stencil of cell averages using weighted essentially non-oscillatory schemes of variable order.

2.1 Spatial Reconstruction and Calculation of Fluxes

In the vast majority of multiresolution finite volume schemes introduced in the previously mentioned works, the reconstruction methods of choice belong to the class of essentially non-oscillatory (ENO) or weighted ENO (WENO) schemes. For flows exhibiting strong shocks and contact discontinuities, the higher-order WENO schemes generally do not reap additional benefit (citation and also Tomek can provide input here?). In the present work, we use the piecewise parabolic method (PPM) of Woodward and Collela ref to reconstruct the field and provide left and right inputs to the Riemann solver at each cell interface.

2.2 Time Integration

Once the system of ordinary differential equations (3) is constructed, the objective is to integrate them forward in time. In our implementation we use a second-order explicit TVD Runge-Kutta scheme to advance. This is summarized by

$$\tilde{\boldsymbol{u}}_1 = \boldsymbol{u}^n + c_1 \triangle t L(\boldsymbol{u}^n) \tag{5}$$
$$\tilde{\boldsymbol{u}}_2 = \tilde{\boldsymbol{u}}_1 + c_2 \triangle t L(\tilde{\boldsymbol{u}}_1) \tag{6}$$
$$\boldsymbol{u}^{n+1} = b_1\tilde{\boldsymbol{u}}_1 + b_2\tilde{\boldsymbol{u}}_2 \tag{7}$$

where $\boldsymbol{E}$ represents the nonlinear evolution operator, and codifies the temporal and spatial discretizations.

3 Harten's Multiresolution Scheme

We present here a brief review of Harten's multiresolution method for adaptive flux computations within the original context of uniform, one-dimensional grids. This is necessary to illustrate the extension of the scheme to block-structured AMR grids. The multiresolution approach is based on a hierarchy of nested discretizations on the domain of interest, Ω. The hierarchy consists of grids at a coarsest level $l = 1$ to a finest grid $l = L$, where any two adjacent levels differ by a refinement factor of two. The grids are defined by

$$\boldsymbol{\mathcal{G}}_l = \left\{x_i^l\right\}_{i=0}^{N_l}, \quad x_i^l = i \cdot h_l, \;\; h_l = 2^{L-l} \cdot h_L, \;\; N_l = N_L/2^{L-l}, \tag{8}$$

where h_l and N_l denote the cell width and number of cells, respectively, on level l. We denote the finest grid by $\boldsymbol{\mathcal{G}}_L$. denote the index space of cells on each level of the hierarchy by $\boldsymbol{\mathcal{I}}^l = \{1, \ldots, N_l\}$.

Encoding

Given a vector of discrete cell averages at the finest resolution, $\boldsymbol{u}^L$, the multiresolution representation is obtained by performing a set of operations on levels $l = L-1, L-2, \ldots, 1$. The operations for an arbitrary level l are as follows:

Split: The cells at grid level $l+1$ are split into even and odd sets, $\boldsymbol{\mathcal{I}}^{l+1}_{even}$ and $\boldsymbol{\mathcal{I}}^{l+1}_{odd}$, based on their indices.

Project: The cells at level $l+1$ are projected by means of averaging, onto the coarser grid level l. The projection is defined by a linear operator which performs the mapping $\boldsymbol{P}^l_{l+1} : \boldsymbol{u}^{l+1} \mapsto \boldsymbol{u}^l$.

Predict: Cell averages with indices $i \in \boldsymbol{\mathcal{I}}^{l+1}_{odd}$ are predicted by an average-interpolating polynomial constructed on level l. Thus, the prediction operator performs the mapping $\boldsymbol{P}^{l+1}_l : \boldsymbol{u}^l \mapsto \tilde{\boldsymbol{u}}^{l+1}$.

The projection preserves averages at the finer level, by design, and in the context of uniform grids is the simple averaging of parent cells,

$$u_i^l = \left(\boldsymbol{P}^l_{l+1}\boldsymbol{u}^{l+1}\right)_i = \frac{1}{2}(u^{l+1}_{2i-1} + u^{l+1}_{2i}), \quad \forall i \in \boldsymbol{\mathcal{I}}^l. \tag{9}$$

The prediction operator is defined by a unique average-interpolating polynomial to predict, based on a stencil of cell averages at grid level l, the odd cell averages at the finer level $l+1$. The prediction mapping is performed using the following centered interpolants,

$$\tilde{u}_{2i}^{l+1} = \left(\boldsymbol{P}_l^{l+1}\boldsymbol{u}^l\right)_i = u_i^l - \sum_{p=1}^{s} \gamma_p \left(u_{i-p}^l - u_{i+p}^l\right) \quad \forall i \in \boldsymbol{\mathcal{I}}^l \tag{10}$$

The coefficients γ_i are supplied in Table (3). The order of accuracy of each interpolant is $r = 2s+1$.

order	γ_1	γ_2	γ_3
3	1/8	0	0
5	-22/128	3/128	0
7	0	0	0

Table 1: For the derivation of these coefficients, the reader is referred to Appendix (ref).

Once the prediction is made, the difference information is obtained by computing the detail coefficient as

$$d_i^l = u_{2i}^{l+1} - \tilde{u}_{2i}^{l+1}, \quad \forall i \in \boldsymbol{\mathcal{I}}^l. \tag{11}$$

for each cell. The encoding procedure is complete when the detail coefficients have been computed on all levels $l = 1, \ldots, L-1$. The entire procedure can be succinctly written in terms of a matrix-vector operation to yield the multiresolution representation of the data,

$$\boldsymbol{u}_M^L = \boldsymbol{M}\boldsymbol{u}^L = \left(\boldsymbol{d}^{L-1}, \boldsymbol{d}^{L-2}, \ldots, \boldsymbol{d}^1, \boldsymbol{u}^1\right)^T. \tag{12}$$

Here the multiresolution operator $\boldsymbol{M}$ contains the prediction operation and subsequent detail coefficient calculation for each cell on each level of the hierarchy.

Truncation

Once the difference information has been computed, compression of the fine-grid cell averages $\boldsymbol{u}^L$ is obtained by truncating coefficients whose magnitude is below a prescribed level-dependent threshold, ε_l. In [1] the following threshold is proposed

$$\varepsilon_l = \varepsilon_L / 2^{L-l}, \tag{13}$$

where ε_L is the tolerance prescribed for the finest level, naturally. The detail coefficients are truncated according to

$$\tilde{d}_i^l = \begin{cases} d_i^l, \text{ if } |d_i^l| > \varepsilon_l \\ 0, \text{ if } |d_i^l| \leq \varepsilon_l, \end{cases} \tag{14}$$

producing the following approximate representation,

$$\tilde{\boldsymbol{u}}_M^L = T_\varepsilon(\boldsymbol{u}_M^L) = \left(\tilde{\boldsymbol{d}}^{L-1}, \tilde{\boldsymbol{d}}^{L-2}, \ldots, \tilde{\boldsymbol{d}}^1, \boldsymbol{u}^1\right)^T. \tag{15}$$

Decoding

The fine-grid solution is reconstructed by computing from levels $l = 1, \ldots, L-1$ the following

$$u_{2i}^{l+1} = d_i^l + \tilde{u}_{2i}^{l+1} \tag{16}$$

$$u_{2i-1}^{l+1} = 2u_i^l - u_{2i}^{l+1}, \tag{17}$$

for each cell on the level. If the detail coefficients are truncated below a nonzero ε, the discrete representation becomes

$$\boldsymbol{u}_\varepsilon^L \approx \boldsymbol{M}^{-1}\tilde{\boldsymbol{u}}_M^L. \tag{18}$$

order	β_1	β_2	β_3	β_4
3	9/16	-1/16	0	0
5	75/128	-25/256	3/256	0
7	1225/2048	-245/2048	49/2048	-5/2048

Table 2:

Adaptive Calculation of Fluxes and Sources

Once the detail coefficients have been obtained, the MR scheme proceeds by setting a threshold ε and truncating coefficients which have an absolute value below the threshold. Lastly, the inverse transform then starts from grid $l = L$ and at each interface either computes fluxes using the fine-grid scheme, or interpolates them using the MR basis. The fluxes are interpolated by

$$\hat{f}^{l-1}_{2i+1} \approx \sum_{p=1}^{s+1} \beta_p \left(\hat{f}^{l}_{i-p+1} + \hat{f}^{l}_{i+p} \right), \tag{19}$$

where the interpolants are of degree $2s + 1$. The coefficients for various degrees of polynomial interpolants are shown in Table (ref). The process repeats until all fluxes are either computed or interpolated on the fine grid $l = 0$.

To reduce the cost of calculating source terms for reactive flows, we employ the same multiresolution procedure. For each cell which is not in the mask, we compute its children using the same operator as in **??**. We compute

$$\bar{s}^{l+1}_i \approx \left(\boldsymbol{P}^{l+1}_l \boldsymbol{s}^l \right)_i, \quad \forall i \in \boldsymbol{\mathcal{I}}^{l+1}. \tag{20}$$

Error Analysis

4 Fully Adaptive Block Scheme

Block-Structured Mesh Adaptation

We define a mask $\boldsymbol{\mathcal{M}}(\varepsilon)$.

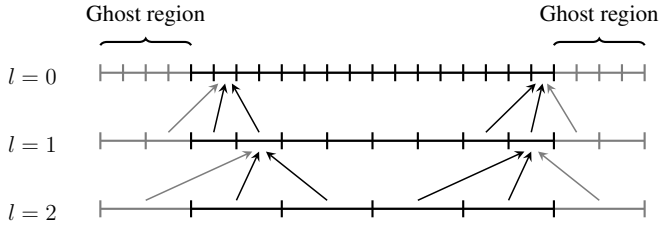

Figure 1: A block of consisting of $N^0 = 16$ cells is shown. Four ghost cells are included on each end of the block, allowing the multiresolution decomposition to descend two levels (to grid level $l = 2$). Interpolation stencils for the computation of detail coefficients at levels $l = 1, l = 2$ are shown, indicating the need for ghost cells.

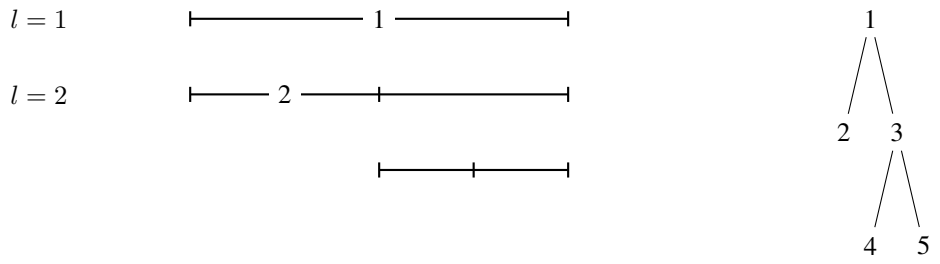

Figure 2:

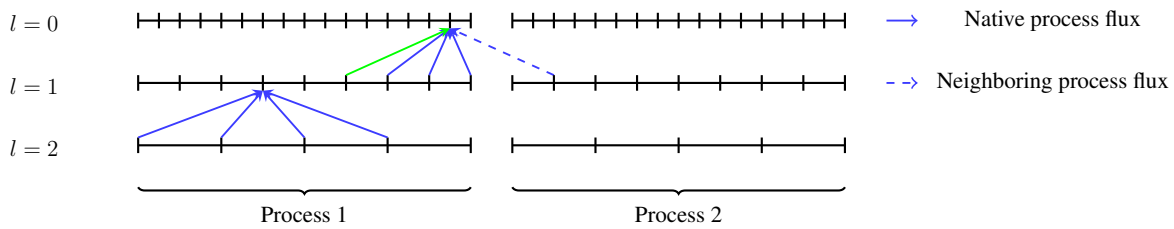

Figure 3: Two examples of flux interpolation on a hierarchy of grids on Process 1: one procedure requires flux data from the adjacent process, the other does not.

Buffer Region

Load Balancing

5 Numerical Results

Convergence Analysis for Smooth Advection Problem

The

$$u_t + f(u)_x = 0 \tag{21}$$

where $u = (\rho, \rho u, \rho w, E)^T$ is the state vector, the flux vector is

$$f = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ u(E + p) \end{pmatrix}, \tag{22}$$

and the total energy per unit volume is given by

$$E = \rho \left(\frac{1}{2} u^2 + e \right),$$

where e is the internal energy.

	$\varepsilon = 0.0$				$\varepsilon = 10^{-12}$			
grid cells	L_1 error	order	L_∞ error	order	L_1 error	order	L_∞ error	order
16								
32								
64								
128								
256								
	$\varepsilon = 10^{-6}$				$\varepsilon = 10^{-4}$			
grid cells	L_1 error	order	L_∞ error	order	L_1 error	order	L_∞ error	order
16								
32								
64								
128								
256								

Interacting Blast Waves

Nuclear Burning

Mach Reflection

Using the inviscid flow assumption, the dynamics of compressible fluids are modeled using the reactive Euler equations add domain notation

$$u_t + f(u)_x + g(u)_y = s(u), \tag{23}$$

where $u = (\rho, \rho u, \rho v, \rho w, E)^T$ is the state vector, the flux vectors are given by

$$f = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ u(E+p) \end{pmatrix}, \quad g = \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ v(E+p) \end{pmatrix}, \tag{24}$$

and $s(u)$ represents sources. The total energy per unit volume is given by

$$E = \rho \left(\frac{1}{2}\mathbf{V}^2 + e \right),$$

where e is the internal energy and the kinetic energy contribution is

$$\frac{1}{2}\mathbf{V}^2 = \frac{1}{2}\mathbf{V} \cdot \mathbf{V} = \frac{1}{2}\left(u^2 + v^2\right).$$

The system of nonlinear equations is closed by an equation of state which is in general not derived from that of an ideal gas.

A Derivation of Prediction Operator in One-Dimension

We are interested in obtaining the difference between approximation spaces at varying levels of resolution. We are given cell-averaged values as input data to our wavelet transform. This data is fed to the scheme at some arbitrary maximum resolution level J, and the wavelet transform produces details coefficients at each lower level until the coarsest level, $j = 0$, is reached. The coefficients in this case are interchangeable with the cell-averages and are denoted by u_k^j, where the level of resolution is denoted by j, and the spatial index is denoted by k. We consider an interpolating polynomial $p(x)$ such that

$$u_{k-1}^j = \int_{x_{k-1}^j}^{x_k^j} p(x)dx \tag{25}$$

$$u_k^j = \int_{x_k^j}^{x_{k+1}^j} p(x)dx \tag{26}$$

$$u_{k+1}^j = \int_{x_{k+1}^j}^{x_{k+2}^j} p(x)dx. \tag{27}$$

The polynomial $p(x)$ should then predict the finer cell-averages of cell u_k^j as

$$\tilde{u}_{2k}^{j+1} = 2\int_{x_k^j}^{x_{k+1/2}^j} p(x)dx \tag{28}$$

$$\tilde{u}_{2k+1}^{j+1} = 2\int_{x_{k+1/2}^j}^{x_{k+1}^j} p(x)dx \tag{29}$$

At present, it may not be clear how to implement such a scheme on a computer. However this interpolation procedure can be cast in a more suitable form by introducing another polynomial, the integral of $p(x)$:

$$P(x) = \int_0^x p(y)dy. \tag{30}$$

Now the problem is to interpolate the following data

$$0 = P(x^j_{k-1}) \tag{31}$$
$$u^j_{k-1} = P(x^j_k) \tag{32}$$
$$u^j_{k-1} + u^j_k = P(x^j_{k+1}) \tag{33}$$
$$u^j_{k-1} + u^j_k + u^j_{k+1} = P(x^j_{k+2}). \tag{34}$$

This can easily be done using Lagrange polynomials. Then the predictions are given in terms of $P(x)$ by

$$\tilde{u}^{j+1}_{2k} = 2\left(P(x^j_{k+1/2}) - P(x^j_k)\right) \tag{35}$$
$$\tilde{u}^{j+1}_{2k+1} = 2\left(P(x^j_{k+1}) - P(x^j_{k+1/2})\right). \tag{36}$$

This interpolating polynomial is cast in the Lagrange form,

$$P(x) = \sum_{i=0}^{n} y_i l_i(x), \tag{37}$$

where y_i are the functional data, and $l_i(x)$ are the Lagrange polynomials. For $n = 3$ these are given by

$$l_0(x) = \frac{x - x_1}{x_0 - x_1}\frac{x - x_2}{x_0 - x_2}\frac{x - x_3}{x_0 - x_3} \tag{38}$$
$$l_1(x) = \frac{x - x_0}{x_1 - x_0}\frac{x - x_2}{x_1 - x_2}\frac{x - x_3}{x_1 - x_3} \tag{39}$$
$$l_2(x) = \frac{x - x_0}{x_2 - x_0}\frac{x - x_1}{x_2 - x_1}\frac{x - x_3}{x_2 - x_3} \tag{40}$$
$$l_3(x) = \frac{x - x_0}{x_3 - x_0}\frac{x - x_1}{x_3 - x_1}\frac{x - x_2}{x_3 - x_2}, \tag{41}$$

and the final interpolating polynomial is

$$P(x) = (0)l_0(x) + (u^j_{k-1})l_1(x) + (u^j_{k-1} + u^j_k)l_2(x) + (u^j_{k-1} + u^j_k + u^j_{k+1})l_3(x). \tag{42}$$

Several evaluations are necessary in order to obtain the predictions. Using intervals of equal length, these values are

$$P(x^j_k) = u^j_{k-1} \tag{43}$$
$$P(x^j_{k+1/2}) = \frac{17}{16}u^j_{k-1} + \frac{1}{2}u^j_k - \frac{1}{16}u^j_{k+1} \tag{44}$$
$$P(x^j_{k+1}) = u^j_{k-1} + u^j_k. \tag{45}$$

Then the predictions of the cell-averages at the higher level of resolution are finally given by

$$\tilde{u}^{j+1}_{2k} = u^j_k + \frac{1}{8}\left(u^j_{k-1} - u^j_{k+1}\right) \tag{46}$$
$$\tilde{u}^{j+1}_{2k+1} = u^j_k - \frac{1}{8}\left(u^j_{k-1} - u^j_{k+1}\right). \tag{47}$$

This procedure could easily be extended to non-uniformly spaced intervals, giving different weights. Note that only the odd indices are counted because in the multiresolution scheme the data is initially split into even and odd signals. All data at level j are just considered to be a copy of the even-index data at level $j + 1$, whereas the odd-indexed data at level $j + 1$ is what is predicted by even-indexed data at level $j + 1$. Also important are the interpolants near the boundaries of the domain. Given below are the left and right predictions, respectively:

$$\tilde{u}^{j+1}_{2k+1} = \frac{5}{8}u^j_k + \frac{1}{2}u^j_{k+1} - \frac{1}{8}u^j_{k+2} \tag{48}$$
$$\tilde{u}^{j+1}_{2k+1} = \frac{1}{8}u^j_{k-2} - \frac{1}{2}u^j_{k-1} + \frac{11}{8}u^j_k. \tag{49}$$

B WENO Reconstruction

The class of WENO schemes are designed to provide the smoothest polynomial for reconstructing the solution field. In the following work, we utilize the recent hierarchical WENO schemes of Zhu et. al., [**?**]. We briefly review the fundamentals of the approach in this section.

To provide left and right states as input to the Rieman solver, the final WENO polynomial is evaluated at cell interfaces, $u(x_{i\pm1/2})$. Given a target cell I_i, we first construct polynomials $q_s(x)$ of degree $2s$ based on the stencils $\mathcal{T}_s = \{I_{i-s}, \ldots, I_{i+s}\}$, which preserve respective cell averages. The polynomial $q(x)$ is evaluated at interfaces $x_{i-1/2}$ and $x_{i+1/2}$ to provide the states $u^R_{i-1/2}$ and $u^L_{i+1/2}$, respectively. The result of the polynomials evaluated at the interface $x_{i+1/2}$ are

$$q_s(x_{i+1/2}) = \sum_{j\in\mathcal{T}_s} \alpha_{s,j} u_j, \tag{50}$$

where the weights $\alpha_{s,j}$ are provided in Table(C). Evaluating the polynomial at the interface $x_{i-1/2}$ is mirror-symmetric about the target cell.

The next step is to construct polynomials $p_s(x)$ using a convex combination of the $q_s(x)$. We have

$$p_s(x) = \frac{1}{\gamma_{s,s}} q_s(x) - \sum_{l=1}^{s-1} \frac{\gamma_{l,s}}{\gamma_{s,s}} p_l(x), \tag{51}$$

where the linear weights γ are supplied in Table (ref).

Smoothness indicators are used to determine the near-optimal weight to ascribe to each of the candidate polynomials, $p_s(x)$. We use the same indicator and nonlinear weights as in ([**?**],others). The final polynomial reconstruction is

$$w_s(x) = \sum_{l=1}^{s} \omega_{l,s} p_l(x). \tag{52}$$

order	α_{i-4}	α_{i-3}	α_{i-2}	α_{i-1}	α_i	α_{i+1}	α_{i+2}	α_{i+3}	α_{i+4}
3	0	0	0	-1/6	5/6	1/3	0	0	0
5	0	0	1/30	-13/60	47/60	9/20	-1/20	0	0
7	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0

Table 3:

References

[1] Harten, A., Adaptive Multiresolution Schemes for Shock Computations, Journal of Computational Physics, Volume 115, pg 319-338 1994.

[2] Woodward, P. and Colella, P., The Numerical Simulation of Two-Dimensional Fluid Flow with Strong Shocks, Journal of Computational Physics, Volume 54, pg 115-173 1984.

[3] Bihari, B.L., and Schwendeman, D., Multiresolution Schemes for the Reactive Euler Equations, Journal of Computational Physics, Volume 154, pg 197-230 1999.

[4] Shu, Chi-Wang, High Order Weighted Essentially Nonoscillatory Schemes for Convection Dominated Problems, SIAM Review, Volume 51, pg 82-126.

[5] Zhu, and Shu, Chi-Wang, A new type of multi-resolution WENO schemes with increasingly higher order of accuracy, Unknown, Unknown, 2018.