A MULTIRESOLUTION SCHEME FOR ADAPTIVE COMPUTATIONS ON BLOCK-STRUCTURED AMR MESHES: 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

September 1, 2019

ABSTRACT

We present a novel block-structured adaptive mesh refinement scheme featuring fully adaptive calculation of fluxes and source terms. The scheme addresses a major shortcoming of tree-based AMR codes: that the graded nature of the AMR mesh yields blocks whose cells are resolved beyond the desired error tolerance. To overcome this issue, we introduce a multiresolution representation of the solution not only for the purpose of grid adaptation but also to identify regions where fluxes and reaction-driven source terms can be interpolated from coarser levels. The error introduced by this approximation procedure 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 adaptive blocks 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 for problems in compressible flow and reaction-driven combustion.

***K*eywords** multiresolution · adaptive mesh refinement · reactive flows

1 Introduction

Energetic, reacting, and turbulent flows are characterized by disparate spatial and temporal length scales. In the compressible regime, flows are capable of producing shock waves, resulting in steep gradients and an extremely thin discontinuity. When fuels and oxidizers are present, compression of the fluid via shock wave interaction facilitates the chemical reaction process, which will itself become a driving force for shock waves due to the large release of energy, and increase in static pressure. The burning fronts wherein these reactions take place are also highly spatially localized. These features require a level of mesh resolution that would make the problems intractable if applied over the entire domain. Therefore, efficient simulation of such flows requires a fully adaptive strategy, which we present in the following paper.

Accurately resolving regions of interest in fluid dynamics simulations for real-world applications is typically not feasible without introducing a non-uniform spatial mesh. Methods which introduce a hierarchy of nested grids are generally described as adaptive mesh refinement (AMR) methods. First introduced in (berger1984), AMR methods typically rely on estimates of the local truncation error (LTE) to determine regions where refinement is necessary for solution accuracy. Some more simple strategies may refine based on the magnitudes of the solution gradients, or concentration of a quantity of interest. While many strategies are possible, there is not yet a significant amount of mathematical theory available to quantify the solution accuracy for AMR simulations. For a full review of the LTE estimators and refinement criterion, readers are referred to BLANK.

Alternate approaches to dynamic grid adaptation based on wavelet theory became popular after the seminal paper by Harten [**?**] was 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 (cite), followed by the inclusion of viscous terms in (cite), and then to source terms in the context of reactive flows in (cite). 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). list a bunch more papers.

The standard AMR methods have been the workhorse in computational fluid dynamics over the last several decades. The implementation of such AMR methods on large networks of parallel computers have neccessitated from a computational efficiency standpoint the reduction in granularity of the mesh 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 $\mathcal{O}(n^d)$ operations, where n is the number of cells per dimension, d. This operation is inherently sequential, and thus not conducive to massively parallel computations. Thus most AMR codes used for practical applications make use of some type of block-structured approach. Tree-based 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, p4est, and (list others). This approach allows for very 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 between adjacent blocks. Together these consequences of block-structured AMR represent a non-trivial decrease in performance.

To address the aforementioned issues with block-structured AMR, we propose a scheme which uses multiresolution-based indicators not only to adapt the grid, but also to identify flux and source term calculations which may be systemically replaced with interpolation. We show how this scheme increases computational performance with no additional overhead, since the multiresolution information is recycled after adaptation. Perhaps most importantly, we demonstrate that the scheme does not degrade the overall accuracy of the solver.

In section 2 of this paper, we describe the systems of conservations laws that we are interested in solving, as well as the underlying finite volume method and discretization. In section 3, we introduce the multiresolution decomposition of the discrete solution and the interpolation procedure in one spatial dimension. In section 4, the fully adaptive block-structured scheme is introduced, which combines grid adaptation with the interpolation procedure. In section 5 we present the numerical results from a number of one- and two-dimensional simulations.

2 Adaptive mesh refinement

3 Finite volume method for hyperbolic conservation laws

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

$$\begin{cases} \boldsymbol{u}_t + \nabla \cdot \boldsymbol{F} = \boldsymbol{s}(\boldsymbol{u}) \\ \boldsymbol{u}(\boldsymbol{x}, 0) = \boldsymbol{u}_0(\boldsymbol{x}), \end{cases} \tag{1}$$

where $\boldsymbol{u} \in \mathbb{R}^m, \boldsymbol{x} \in \mathbb{R}^n$, and $t > 0$. Here $\boldsymbol{u}(\boldsymbol{x}, t)$ represents a vector of conserved quantities, $\boldsymbol{F}$ is a tensor of fluxes, and $\boldsymbol{s}$ is a source term. These nonlinear, hyperbolic partial differential equations are known to produce discontinuities, requiring solutions to be found in a weak sense. In the present work, we work with the integral form of 1 via the finite volume method.

In the finite volume formulation, the spatial domain is discretized into control volumes V_i, inside which the solution is approximated by volume averages at time t as

$$\overline{\boldsymbol{u}}_i(t) = \frac{1}{|V_i|}\int_{V_i} \boldsymbol{u}(\boldsymbol{x},t)d\boldsymbol{x}. \tag{2}$$

We approximate also the source terms in a cell-averaged sense, and denote them as $\overline{\boldsymbol{s}}_i$. Finally the fluxes at each interface are evaluated numerically as

$$\hat{\boldsymbol{f}} = \hat{\boldsymbol{f}}(\boldsymbol{u}^-, \boldsymbol{u}^+), \tag{3}$$

where $\boldsymbol{u}^-$ and $\boldsymbol{u}^+$ represent states to the left and right of the interface, respectively.

3.1 Spatial reconstruction and calculation of fluxes

The correct calculation of numerical flux is the most crucial aspect of any solver for compressible flows. It is also the most time-consuming portion of such a program, typically.

In the vast majority of multiresolution 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. These are schemes which determine the smoothest polynomial to use for interpolation from a set of candidate stencils. In the WENO setting, the interpolant is formed from a convex combination of interpolants generated by the candidate stencils. These schemes are easily used to generate high-order accurate polynomials with minimal oscillations, even in the presence of discontinuities. However, for flows exhibiting strong shocks and contact discontinuities, they generally do not reap additional benefit, especially considering their high computational cost. In the present work, we use the piecewise parabolic method (PPM) of Woodward and Collela (cite) to reconstruct the field and provide left and right inputs to the Riemann solver at each cell interface.

4 Adaptive multiresolution scheme on a uniform grid

We present here a brief review of the multiresolution concepts, as first introduced by Harten cite within the context of conservative finite volume schemes. The scope here is restricted to one-dimensional uniform grids, as the operations required for the fully adaptive block scheme are nearly identical.

4.1 Reference scheme on one-dimensional uniform grid

We illustrate the multiresolution interpolatory procedure of Harten using a scalar version of 1 on a one-dimensional uniform grid, for simplicity. We begin with the discretized scheme

$$u_i^{n+1} = u_i^n + -\frac{1}{\Delta x}\left(\hat{f}_{i+1/2} - \hat{f}_{i-1/2}\right) + \overline{s}_i, \tag{4}$$

where the subscripts $i-1/2$, $i+1/2$ denote the left and right interfaces of the target cell, respectively.

4.2 Multiresolution representation of data

The multiresolution approach decomposes data by introducing a hierarchy of nested discretizations, $\boldsymbol{\mathcal{G}}_l$. The objective in introducing these grids is to represent the fine-grid data as a sum of values at a caorse level plus a series of differences at finer levels of resolution. The result is a decomposition which provides frequency and scale information of the underlying data. The grids are defined by

$$\boldsymbol{\mathcal{G}}_l = \left\{x_i^l\right\}_{i=0}^{N_l}, \quad x_i^l = i\cdot\Delta x_l,\ \ \Delta x_l = 2^{L-l}\cdot\Delta x_L,\ \ N_l = N_L/2^{L-l}, \tag{5}$$

where Δx_l and N_l denote the cell width and number of cells, respectively, on level l. The index space of cells on each level of the hierarchy is denoted by $\boldsymbol{\mathcal{I}}_l = \{1,\ldots,N_l\}$. Given cell averages at the finest resolution, $\overline{\boldsymbol{u}}^L$, the encoding process proceeds by performing the following set of mappings on levels $l = L-1, L-2, \ldots, 1$:

> *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 $\boldsymbol{P}_{l+1}^l : \overline{\boldsymbol{u}}^{l+1} \mapsto \overline{\boldsymbol{u}}^l$.
>
> *Predict*: Cell averages at level $l+1$ are predicted by an average-interpolating polynomial constructed of cells on level l. The prediction operator performs the mapping $\boldsymbol{P}_l^{l+1} : \overline{\boldsymbol{u}}^l \mapsto \tilde{\boldsymbol{u}}^{l+1}$.

The projection operation, which preserves averages at the finer level by design, is the simple weighted average of parent cells. In one dimension, this average is given by

$$\overline{u}_i^l = \left(\boldsymbol{P}_{l+1}^l \overline{\boldsymbol{u}}^{l+1}\right)_i = \frac{1}{2}(\overline{u}_{2i-1}^{l+1} + \overline{u}_{2i}^{l+1}), \quad \forall i \in \boldsymbol{\mathcal{I}}_l. \tag{6}$$

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 with $m \in \{0, 1\}$,

$$\tilde{u}_{2i-m}^{l+1} = \left(\boldsymbol{P}_l^{l+1} \overline{\boldsymbol{u}}^l\right)_i = u_i^l - (-1)^m \sum_{p=1}^{s} \gamma_p \left(u_{i+p}^l - u_{i-p}^l\right), \quad \forall i \in \boldsymbol{\mathcal{I}}_l \tag{7}$$

The coefficients γ_i are supplied in Table (**??**). The order of accuracy of each interpolant is $r = 2s + 1$. Once the prediction is made, the difference information is obtained by computing detail coefficients as

$$d_i^l = u_{2i}^{l+1} - \tilde{u}_{2i}^{l+1}, \quad \forall i \in \boldsymbol{\mathcal{I}}_l. \tag{8}$$

The detail coefficients are a measure of the local regularity of the underlying function, and decay proportionally to that regularity as the resolution increases. The encoding procedure is complete when the detail coefficients have been computed on all levels $l = L-1, \ldots, 1$. This linear transformation can also be written more succinctly as

$$\boldsymbol{u}_M^L = \boldsymbol{M}\boldsymbol{u}^L = \left(\boldsymbol{d}^{L-1}, \boldsymbol{d}^{L-2}, \ldots, \boldsymbol{d}^1, \boldsymbol{u}^1\right)^T, \tag{9}$$

where M denotes the linear transform operator, and $\boldsymbol{d}^l = \left\{d_i^l\right\}_{i=1}^{N_l}$. This expression forms the multiresolution representation of the fine-grid data.

4.3 Truncation

The purpose of performing such a multiresolution transformation of the data is to achieve compression. 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 [**?**] the following threshold is proposed

$$\varepsilon_l = \varepsilon_L / 2^{L-l}, \tag{10}$$

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

$$\tilde{d}_i^l = \begin{cases} d_i^l, \text{ if } |d_i^l| > \varepsilon_l \\ 0, \text{ if } |d_i^l| \leq \varepsilon_l. \end{cases} \tag{11}$$

The resulting multiresolution decomposition is given by,

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

where $\tilde{\boldsymbol{d}}^l = \left\{\tilde{d}_i^l\right\}_{i=1}^{N_l}$ now represent the array of truncated detail coefficients at the given level. We may reconstruct the fine-grid data using this truncated representation as

$$\hat{\boldsymbol{u}}^L = \boldsymbol{M}^{-1}\boldsymbol{T}_\varepsilon(\boldsymbol{u}_M^L), \tag{13}$$

simply applying the inverse operation of **??**. The error associated with this truncatated representation is

$$||\overline{\boldsymbol{u}}^L - \hat{\boldsymbol{u}}^L|| \leq C\varepsilon, \tag{14}$$

measured in the L_1 and L_∞ norms.

4.4 Adaptive calculation of fluxes

In [**?**], Harten shows how the numerical flux, **??**, may be replaced by interpolation of $\hat{\boldsymbol{f}}^l$ in smooth regions. We introduce here an array of fluxes which each correspond to the interfaces on a given grid, $\boldsymbol{\mathcal{G}}_l$, $\hat{\boldsymbol{f}}^l = \left\{\hat{f}_i^l\right\}_{i=1}^{N_l+1}$. Given the fluxes on some arbitrary level l, the odd-indexed fluxes on level $l+1$ may be interpolated in a point-wise sense as

$$\hat{f}_{2i+1}^{l+1} \approx \sum_{p=1}^{s+1} \beta_p \left(\hat{f}_{i-p+1}^l + \hat{f}_{i+p}^l\right). \tag{15}$$

The interpolants are of order $2s+2$. The coefficients for various degrees of polynomial interpolants are shown in Table **??** Even-indexed fluxes are simply copied from level l as

$$\hat{f}_{2i}^{l+1} = \hat{f}_i^l, \quad \forall i \in \boldsymbol{\mathcal{I}}_l \tag{16}$$

4.5 Adaptive calculation of source terms

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

$$\bar{s}_i^{l+1} \approx \left(\boldsymbol{P}_l^{l+1}\boldsymbol{s}^l\right)_i, \quad \forall i \in \boldsymbol{\mathcal{I}}^{l+1}. \tag{17}$$

5 Fully adaptive block-structured multiresolution scheme

6 Numerical Results

Interacting blast waves

The adaptive multiresolution scheme is tested for the problem of two interacting blast waves (cite collela woodward). We solve the Euler equations in one spatial dimension,

Mach reflection

Nuclear burning

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

$$u_{k-1}^j = \int_{x_{k-1}^j}^{x_k^j} p(x)dx \tag{18}$$

$$u_k^j = \int_{x_k^j}^{x_{k+1}^j} p(x)dx \tag{19}$$

$$u_{k+1}^j = \int_{x_{k+1}^j}^{x_{k+2}^j} p(x)dx. \tag{20}$$

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

$$\tilde{u}_{2k}^{j+1} = 2\int_{x_k^j}^{x_{k+1/2}^j} p(x)dx \tag{21}$$

$$\tilde{u}_{2k+1}^{j+1} = 2\int_{x_{k+1/2}^j}^{x_{k+1}^j} p(x)dx \tag{22}$$

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 $p(x)$:

$$P(x) = \int_0^x p(y)dy. \tag{23}$$

Now the problem is to interpolate the following data

$$0 = P(x_{k-1}^j) \tag{24}$$

$$u_{k-1}^j = P(x_k^j) \tag{25}$$

$$u_{k-1}^j + u_k^j = P(x_{k+1}^j) \tag{26}$$

$$u_{k-1}^j + u_k^j + u_{k+1}^j = P(x_{k+2}^j). \tag{27}$$

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

Table 1: Coefficients for centered, conservative interpolants of orders 3-7.

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

$$\tilde{u}_{2k}^{j+1} = 2\left(P(x_{k+1/2}^j) - P(x_k^j)\right) \tag{28}$$

$$\tilde{u}_{2k+1}^{j+1} = 2\left(P(x_{k+1}^j) - P(x_{k+1/2}^j)\right). \tag{29}$$

This interpolating polynomial is cast in the Lagrange form,

$$P(x) = \sum_{i=0}^{n} y_i l_i(x), \tag{30}$$

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

$$l_0(x) = \frac{x-x_1}{x_0-x_1}\frac{x-x_2}{x_0-x_2}\frac{x-x_3}{x_0-x_3} \tag{31}$$

$$l_1(x) = \frac{x-x_0}{x_1-x_0}\frac{x-x_2}{x_1-x_2}\frac{x-x_3}{x_1-x_3} \tag{32}$$

$$l_2(x) = \frac{x-x_0}{x_2-x_0}\frac{x-x_1}{x_2-x_1}\frac{x-x_3}{x_2-x_3} \tag{33}$$

$$l_3(x) = \frac{x-x_0}{x_3-x_0}\frac{x-x_1}{x_3-x_1}\frac{x-x_2}{x_3-x_2}, \tag{34}$$

and the final interpolating polynomial is

$$P(x) = (0)l_0(x) + (u_{k-1}^j)l_1(x) + (u_{k-1}^j + u_k^j)l_2(x) + (u_{k-1}^j + u_k^j + u_{k+1}^j)l_3(x). \tag{35}$$

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

$$P(x_k^j) = u_{k-1}^j \tag{36}$$

$$P(x_{k+1/2}^j) = \frac{17}{16}u_{k-1}^j + \frac{1}{2}u_k^j - \frac{1}{16}u_{k+1}^j \tag{37}$$

$$P(x_{k+1}^j) = u_{k-1}^j + u_k^j. \tag{38}$$

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

$$\tilde{u}_{2k}^{j+1} = u_k^j + \frac{1}{8}\left(u_{k-1}^j - u_{k+1}^j\right) \tag{39}$$

$$\tilde{u}_{2k+1}^{j+1} = u_k^j - \frac{1}{8}\left(u_{k-1}^j - u_{k+1}^j\right). \tag{40}$$

This procedure could easily be extended to non-uniformly spaced intervals, giving different weights.

$$\tilde{u}_{2k+1}^{j+1} = \frac{5}{8}u_k^j + \frac{1}{2}u_{k+1}^j - \frac{1}{8}u_{k+2}^j \tag{41}$$

$$\tilde{u}_{2k+1}^{j+1} = \frac{1}{8}u_{k-2}^j - \frac{1}{2}u_{k-1}^j + \frac{11}{8}u_k^j. \tag{42}$$

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.

order	β_1	β_2	β_3	β_4
4	9/16	-1/16	0	0
6	75/128	-25/256	3/256	0
8	1225/2048	-245/2048	49/2048	-5/2048

Table 2: Coefficients for centered, point-wise interpolation of orders 4-8

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

$$q_s(x_{i+1/2}) = \sum_{j\in\mathcal{T}_s} \alpha_{s,j} u_j, \tag{43}$$

where the weights $\alpha_{s,j}$ are provided in Table(**??**). 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

$$p_s(x) = \frac{1}{\gamma_{s,s}} q_s(x) - \sum_{l=1}^{s-1} \frac{\gamma_{l,s}}{\gamma_{s,s}} p_l(x), \tag{44}$$

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

$$w_s(x) = \sum_{l=1}^{s} \omega_{l,s} p_l(x). \tag{45}$$

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.