Journal of Multidisciplinary Engineering Science and Technology (JMEST)
ISSN: 2458-9403
Vol. 7 Issue 12, December - 2020

# KYAMOS Software
# Finite Volume TVD scheme for advection fluid simulations

**Antonis P. Papadakis*, Aimilios Ioannou and Wasif Almady**

KYAMOS LTD, 37 Polyneikis Street, Strovolos, 2047, Nicosia, Cyprus
*ceo@kyamosmultiphysics.com

***Abstract*—The Computer Aided Engineering industry utilizes computer simulations to analyze and predict mainly fluid flow phenomena. These simulations can be used both for research and commercial purposes, to comprehend physical phenomena, that will aid in the design of engineering systems. There are various technologies out there, that can be used to perform computing simulations, which are well-established. In this paper, firstly, the various new technologies that are expected to produce a disrupt in the market in the near future are discussed which are artificial intelligence, quantum computing and tensor processing units. Thereafter, the various methods to approximate partial differential equations are analyzed, and focus is given in the solution of convection-dominated problems, which is a major part of fluid simulation phenomena. We therefore discuss the advantages and disadvantages of the main traditional methods such as the finite difference, finite element and finite volume methods, as well as more innovative methods such as the Lattice Boltzmann method in mesoscopic range. In this paper, a very innovative Finite Volume-Total Variation Diminishing scheme is proposed, that has the capability to ideally predict the fluid flow of both diffused and shock wave phenomena, free from artificial numerical diffusion, spurious oscillations and guaranteeing flux conservation. The results demonstrate that the proposed method captures the waves in a nearly ideal way and guarantees accurate simulations, even when long time simulations are performed, excelling in accuracy. Since it is developed in a CUDA aware MPI environment, it is highly computationally efficient and it is expected to disrupt the market and establish KYAMOS software, as a competitive alternative in the Computer Aided Engineering industry.**

***Keywords—KYAMOS software, High Performance Computing, CUDA-GPU, Finite Volume, Total Variation Diminishing;***

## I. INTRODUCTION

In the solution of advection equations, there is always the problem of capturing the shock wave using 2nd order dynamics which results in the loss of monotonicity, artificial numerical diffusion and spurious oscillations. To avoid this, the research community has been looking for a numerical scheme which would not suffer from the above drawbacks and at the same time preserve conservation.

The finite volume falls within this category by ensuring conservation through a control volume formulation. In this paper, we propose a new method proven to work ideally in three-dimensional unstructured grids. The code uses the Total Variation-Diminishing technique (FV-TVD), a property of certain discretization schemes that is used to solve hyperbolic partial differential equations, usually in computational fluid dynamics to preserve monotonicity by ensuring that if the solution at current value is monotically increasing/decreasing, so does the next time step value. This monotonic behavior of TVD schemes ensures that non-physical oscillations are eliminated, hence are very attractive for solving engineering and scientific problems.

A theorem by Godunov proves that monotonicity is only guaranteed and preserved only in 1st order linear schemes. For the case of 2nd order linear schemes and higher, they perform well at smooth solutions, however they create spurious oscillation near shock waves. Hence, one extra step is necessary to avoid these spurious oscillations and this can be achieved by deploying high order flux/slope limiters at shock waves. Next, we present some Key Enabling Technologies that are expected to disrupt the CAE market in the near future.

## II. ARTIFICIAL INTELLIGENCE

Artificial Intelligence is a Key Enabling Technology which is expected to bring a change in the way human perceive and operate systems with impact in all areas of our lives. A subset of Artificial Intelligence, and specifically, deep learning can be utilized to predict the behavior of the fluid flows without solving the appropriate partial differential equations. To achieve this, one can utilize the well-known TensorFlow, feely available software from google under Apache2 license, written in language programme python to train a model to predict the fluid flow.

The data to be input to TensorFlow must be provided by a state of the art conventional and/or unconventional fluid flow solutions. TensorFlow operates more efficiently using GPUs to conduct the calculations, hence a cluster-computing GPU

InfiniBand structure likewise with the one developed by KYAMOS LTD, is ideal for such calculations.

In the literature, it has already been demonstrated that deep learning can be used in the simulation of partial differential equations with applications in the automotive industry [1]. The general idea is to create a list of random shapes and domain geometries which are variant in shape and size such that a diverse date set is created and then be simulated using state-of-the-art simulators. Then one needs to extract relevant data from the simulators and feed them into the Neural Network Implementation in TensorFlow. Information sources regarding CAE and Artificial Intelligence can be found in the following links [2], [3], [4], [5], [6] and [7].

According to [8], one can use neural networks to represent approximate solutions to partial differential equations modeling fluid phenomena and obtaining the optimal Neural Network model using collocation points on the interior and boundary of the domain and an appropriate optimization method.

Additionally, using CFD codes, one can generate lots of simulation data for different fluid properties and using the resulting dataset to construct inverse mappings, using Deep Learning or some other techniques, one can take a given CFD/experimental output and try to estimate the properties of the fluid. This is basically an approach one can use to try and tackle inverse problems in fluid mechanics; however, one needs to develop thousands of examples that will be input to TensorFlow, hence this process may need to be automated using a software that will generate randomly and automatically thousands of geometry shapes and domains that will be meshed and simulated automatically.

The thousands of output files which encompasses the simulated design i.e. geometry data and the flow fields will then be input to TensorFlow that will build the deep learning model using artificial neural network in a python library utilizing GPU technology. Hence, when a customer wishes to run a similar problem, the deep learning model library will be called upon and the result will be uploaded automatically to their personal computer to be visually processed. This will open a new field in CAE, since designers will be able to have real live insight during the design process, get detailed insight and optimize their designed systems. The users will also be able to verify the optimized suggested result through proper conventional and unconventional methods to ensure the validity of the results. Deep learning processes are initially dummy and use data samples to generate intelligence through trial and error and feedback, hence one can say/imply that there is no built-in intelligence in artificial intelligence.

## III. Quantum Computing

Quantum computing is a new way of conducting computer simulations using instead of bits, the so-called qubits. Qubits take into advantage that fact that in the microscopic world, the state of a subatomic particle can be a 0 or 1, or any of the intermediate states between a 0 and 1. This probability of the state of a particle to have multiple states simultaneously means it can be used concurrently to analyze mostly optimization problems, where multiple paths need to be investigated simultaneously for the optimum root or solution. Hence, quantum computers are very good in solving maze problems or finding the shortest distance on a map. In practice, quantum computers utilize highly sophisticated engineering devices to stabilize by freezing the state of the particle and ensuring that correct physical coupling between the various qubits exist at the same time and measuring these states accurately and efficiently. Unfortunately, this system is very unstable and hard to measure and results into uncertainties, hence statistics are used to ensure that correct results are given through multiple passes. The real breakthrough is expected to occur when scientists are able to provide supercomputers without extreme cooling, that will open the industry for on-premises computer simulations, however we expect this to occur, if so, in the next decade or so.

We have recently seen a few quantum computers coming out, based on completely different architectures. Google in 2019 demonstrated quantum supremacy over supercomputers by utilizing a 54-qubit quantum computer called Sycamore that relied on superconducting metal loops architecture, Honeywell and IonQ have been developing quantum computing architectures based on trapped ions, Silicon Quantum Computing company has been developing quantum computers based on spin-based silicon qubits, and finally a Chinese company has been building photonic quantum computers, the so-called boson sampling machine.

One of the major drawbacks of quantum computers, it is the fact that they need to be extremely stable, hence they are usually situated in well shielded basements and most importantly, they need to be cooled down to nearly zero temperatures. Hence, we anticipate that this computing industry will thrive in the near future with the usage of cloud-computing services, rather than on premises simulations, hence, we believe that our company, which specializes on cloud-computing services, will be at a step forward from our main competitors, which utilize mainly on-premises computing.

Regarding the algorithms that can handle quantum computing well, they are just emerging now, however, they are expected to get increased attention, as quantum computers become more readily available and new algorithms exist that make their technology more optimized.

We expect that a breakthrough in quantum computing for computer aided simulations can only come not from quantum computing itself, but from a combination of Central Processing Unit, Graphical Processing Unit and Quantum Processing Unit, hence getting the best of all three technologies. This is similar to the currently known CPU-GPU systems, that we, as a company, believe and invest on.

Some have attempted already to apply algorithms to solve partial differential equations using quantum computing. Finally, we are also looking at breakthrough/groundbreaking technologies that could

potentially disrupt the market. Two such areas are the quantum computing and the presence of the TPUs from google.

Regarding quantum computing, it is mainly used for optimization problems such as the solution of a maze and utilizes the ability of an electron to be in a superposition state in combination with quantum entanglement. There are currently no applications at which quantum computing could be applied, however there is scare evidence of small breakthroughs such as [9, 10].

The humble opinion of the authors is that a breakthrough in CFD and other areas of computational science will probably originate from a combination of various technologies i.e. CPU, GPU and quantum computing. Due to the fact that quantum computers need a controlled environment such as minimal vibrations, extreme cooling, it is unlikely to see the release of quantum computers in the market for sale anytime soon, hence when available, they will be offered as cloud computing service (google does it). This is in accordance with the fact that GPUs can only work in the presence of CPUs.

## IV. Tensor Processing Units

Regarding the Tensor Processing Units (TPUs) released by google for Artificial Intelligence on the cloud, it is a technology that is specialized and designed for tensor calculations and has very low precision capabilities, inadequate for CAE simulations and does not apply to CFD directly. However, since one of the main asks of this company is to incorporate AI solutions for its customers, we have high interest in TPUs and their applications to CAE simulations, since it may indirectly influence CAE simulations by predicting instantly flow simulations. This is why we intend to utilize GPU cards for the moment with an eye opened at Tensor capable devices such as NVIDIA-V100 and TPUs from google.

## V. Computational Fluid Simulation Methods

### A. Finite Difference Scheme

This is a very well-known method in solving partial differential equations and it is highly popular, mainly due to the fact that it was the first one to come out, well known for its simplicity and ease to program. It uses a predefined mesh stencil to capture the geometry of the problem to be solved and utilizes Taylor series expansion to approximate the derivatives such as forward, backward and central differencing, both for time and space dependent derivatives. In many cases, the application of boundary conditions is implemented by using ghost cells, which are additional cells situated at the boundary of the domain.

Finite difference has proven reliable and robust and able to solve all three types of partial differential equations and it is widely used today, mainly in open source software. One of the major drawbacks of finite difference is that it cannot handle well non-uniform geometries and does not ensure monotonic, as well as conservative results. Hence, other methods have been implemented that deal with the weaknesses of the finite difference method, which are mainly incorporated into commercial expensive software. One method used is the finite element method, well known for its highly complicated mathematics since it uses weak and strong formulations of the problem and interpolation functions to approximate the solution.

### B. Finite Element Scheme

Finite element does very well in non-uniform geometries and there are linear, quadratic and higher order polynomial approximations for the shape functions. Also, there are nodal, as well as edge-based elements, which will do well, depending on the physics of the problem to be solved. In the solution of the Maxwell equations, edge elements are expected to be superior, since the electric field is continuous across the edges and it is able to capture this easily. Finite element is best in analyzing field simulations, where conservation is not so important and highly non-uniform geometries exist.

### C. Finite Volume Scheme

The finite volume method has gained very high popularity in the fluid dynamics industry, since it is based in principle on the concept of conservation. Conservation is applied across the control volume and the inward and outward fluxes are calculated, enforcing conservation in the overall solution domain. It can be either a cell centered or vertex centered scheme and it can be applied to meshes that consists of multiple types of elements. Most of the proprietary software in fluid simulations utilize finite volume to capture flows.

One of the main advantages of finite volume is that it is not so highly dependent on the mesh quality of the simulation and that it can capture shock waves accurately and efficiently, using various proposed discretization schemes. The most popular scheme utilized in the shock wave simulation is the upwind method, which in the finite volume context, conservation is conserved, but at the same time, monotonicity is achieved.

The authors have utilized such a method, and have been able to produce excellent shock wave propagation results, in which we present in this paper.

### D. Lattice Boltzmann Scheme

The Lattice Boltzmann method is a new method, recently being boosted by the advancements in GPUs by NVIDIA and their ability to be suited well for parallel applications. The LB is based on a mesoscopic analysis of the particles in a uniform regular mesh and has the ability to capture complex boundary conditions, easily and efficiently. It has advantages when used in turbulent flows and porous media, when compared to the other traditional methods in the macroscopic world. It encompasses a distribution function and collision-streaming operators.

According to the dimensions solved, there are a number of predefined stencils, with the more accurate, being the one with higher number of possible

directional velocities for the particles. It does extremely well in time-dependent diffusion problems such as heating in a slab bar. Usual stencils in 2D are D2Q4, D2W5, D2Q9 and in 3D, D3Q19 and D3Q27. The most difficult part is to find an accurate approximation for the collision operator. A thorough and detailed explanation of LB is depicted in a paper by Papadakis [11].

## VI. Finite Volume – Total Variation Diminishing Mathematical Formulation

The partial differential equation for a time-dependent, convection diffusion dominated problem with a source term is given as follows:

$$\frac{\partial \varphi}{\partial t} + \nabla.(\varphi u) - \nabla.(D\nabla\varphi) + k\varphi = S_\varphi \qquad (1)$$

where u=u(x) is the velocity vector, φ is the dependent scalar variable or conserved quantity, t is the time, D is the positive diffusion coefficient, *k* is the reaction coefficient and $S_\varphi = S_\varphi$(x,t) is the prescribed source term for generating the φ property, where $t \in (O,T)$ and x is the three-dimensional space and T is the total time.

The computational domain is discretized in the context of finite volume in a group of non-overlapping tetrahedral control volumes $V_i \in V$, with i = 1,….N, which completely cover the whole three-dimensional domain as follows:

$$\cup_{i=1}^{N} V_i, \text{ and } V_i \neq \emptyset \text{ ,and } V_i \cap V_j = \emptyset \text{ if } i \neq j. \qquad (2)$$

If one conducts integration over the control volume V and in the time interval from $t^n$ to $t^{n+1}$, the time-dependent convection-diffusion equation with a source term will result into the following equation:

$$\int_{V_i}\int_{t^n}^{t^{n+1}} \left(\frac{\partial \varphi}{\partial t} + \nabla.(\varphi u - D\nabla\varphi) + k\varphi - S_\varphi\right) dtdx = 0 \qquad (3)$$

Then, we conduct a time integration on the first term and the divergence theorem on the second term to get:

$$\begin{gathered}\int_{V_i} \varphi(\vec{x}, t^{n+1})dx = \int_{V_i} \varphi(\vec{x}, t^n)d\vec{x} - \\ \int_{t^n}^{t^{n+1}} \int_{\partial V_i} \vec{n}_i(v).\left(\vec{v}(v)\varphi(v,t) - D\nabla\varphi(v,t)\right)dvdt - \int_{V_i}\int_{t^n}^{t^{n+1}} (k\varphi(\vec{x},t)d\vec{x}dt) + \\ \int_{V_i}\int_{t^n}^{t^{n+1}} \left(S_\varphi(\vec{x},t)d\vec{x}dt\right)\end{gathered} \qquad (4)$$

where $\vec{n}_i(v)$ is the unit normal vector of $dV_i$.

Using finite volume techniques, the cell average is approximated over $V_i$ at time $t^n$ and $t^{n+1}$ as follows:

$$\begin{aligned}\varphi_i^{n+1} &= \frac{1}{|V_i|}\int_{V_i} \varphi(\vec{x}, t^{n+1})dx \\ \varphi_i^{n} &= \frac{1}{|V_i|}\int_{V_i} \varphi(\vec{x}, t^{n})dx\end{aligned} \qquad (5)$$

Now substituting the above equations into the other equations gives:

$$\begin{gathered}|V_i|\varphi_i^{n+1} = \\ |V_i|\varphi_i^{n} - \int_{t^n}^{t^{n+1}} \int_{\partial \boldsymbol{V_i}} \vec{\boldsymbol{n}}_{\boldsymbol{i}}(v).\left(\vec{\boldsymbol{v}}(v)\varphi(v,t) - D\nabla\varphi(v,t)\right)dvdt - \int_{V_i}\int_{t^n}^{t^{n+1}} (k\varphi(\vec{\boldsymbol{x}},t)d\vec{\boldsymbol{x}}dt) + \\ \int_{V_{1i}}\int_{t^n}^{t^{n+1}} \left(S_\varphi(\vec{\boldsymbol{x}},t)d\vec{\boldsymbol{x}}dt\right)\end{gathered} \qquad (6)$$

Next, we isolate the convection and diffusion terms and apply the surface integral over the control volume. In the case of a tetrahedral finite volume mesh, the flux integral over a control volume is approximated by the summation of fluxes that passes through the four cell faces that define the control volume of the tetrahedral element.

$$\begin{gathered}\int_{t^n}^{t^{n+1}} \int_{\partial \boldsymbol{V_i}} \vec{\boldsymbol{n}}_{\boldsymbol{i}}(v).\left(\vec{\boldsymbol{v}}(v)\varphi(v,t) - D\nabla\varphi(v,t)\right)dvdt = \\ \Delta t \sum_{\boldsymbol{j=1}}^{\boldsymbol{4}} \left|A_{i,j}\right|\left|\vec{\boldsymbol{n}}_{\boldsymbol{i,j}}\right| \left[\left(\boldsymbol{v}_{\boldsymbol{i,j}}\right)\varphi_{i,j,f}(t^n) - D\nabla\varphi_{i,j,f}(t^n)\right]\end{gathered} \qquad (7)$$

where $A_{i,j}$ is the surface of boundary $\partial V_i$ between the two control volumes $V_i$ and $V_j$.

$$\partial V_i = \cup_{j=1}^{4} A_{i,j} \text{ and } A_{i,j} = \partial A_i \cap \partial A_j \qquad (8)$$

Regarding the reaction term, one has:

$$\int_{V_i}\int_{t^n}^{t^{n+1}} (k\varphi(\vec{x},t)d\vec{x}dt) = \Delta t|V_i|k\varphi_i^n \qquad (9)$$

and:

$$\int_{V_{1i}}\int_{t^n}^{t^{n+1}} \left(S_\varphi(\vec{x},t)d\vec{x}dt\right) = \Delta t|V_i|S_\varphi(x,t^n) \qquad (10)$$

Now substituting equations (7, 9, 10666677777) into equation ( 6), one gets the finite volume formulation as follows:

$$\begin{gathered}\varphi_i^{n+1} = \\ \varphi_i^n - \frac{\Delta t}{|V_i|}\sum_{j=1}^{4}\left|A_{i,j}\right|\left|\vec{n}_{i,j}\right| \left[\left(v_{i,j}\right)\varphi_{i,j,f}(t^n) - D\nabla\varphi_{i,j,f}(t^n)\right] - \Delta t(k\varphi_i^n - S_\varphi(x,t^n))\end{gathered} \qquad (11)$$

One needs to find a way to calculate the perpendicular face values at the mesh faces. Since TVD can only be ensured in 1st order, one needs to utilize a high order accurate interpolation scheme, hence we have chosen to use a 2nd order accurate interpolation scheme based on central and single sided upwind differencing.

## VII. Standard Flux Limiters and Sweby Diagrams

The Sweby diagram is a plot of the Flux limit value (FL) versus a ratio variable r as shown in Fig. 1 and defined by:

$$r = \frac{\varphi_C - \varphi_U}{\varphi_D - \varphi_C} \quad (12)$$

where $\varphi_D$ stands for the dependent variable downstream, $\varphi_C$ stands for the dependent variable at the center and $\varphi_U$ stands for the dependent variable upstream.

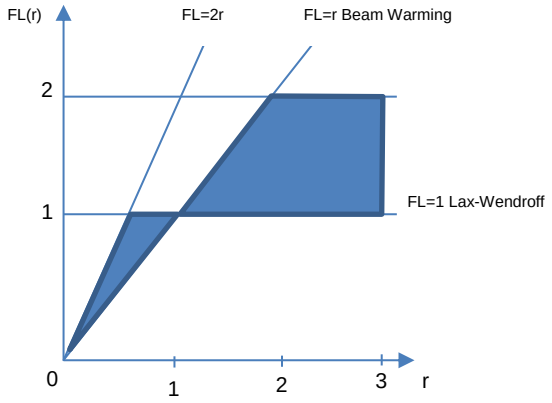


Fig.1 Sweby diagram depicting Bean Warming and Lax-Wendroff limiters.

The Sweby diagram of the minmod flux limiter is shown in Fig.2. The minmod limiter is defined as follows:

$$FL(r) = \max(0, \min(2, r)) \quad (13)$$

Regarding the minmod limiter, it is considered to be dissipative, as well as a compressive limiter, however it performs fairly well and it is very simple to implement. However, the compressive Min-mod limiter may suffer from its slightly poor convergence [12]. Since minmod flux limiter applies the maximum possible
limiting allowed within the second order TVD region and being rather dissipative, it smears out discontinuities.

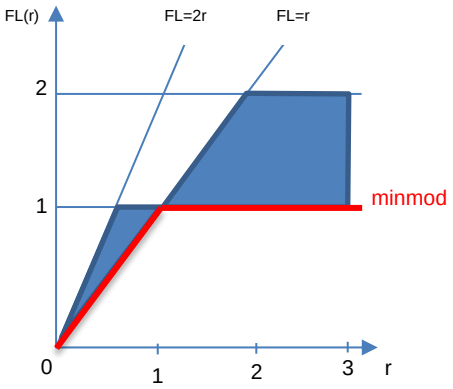


Fig.2 Sweby diagram depicting the minmod limiter.

The Sweby diagram of the Superbee flux limiter is shown in Fig.3. The Superbee limiter is defined as follows:

$$FL(r) = \max(0, \min(1,2r), \min(2, r)) \quad (14)$$

Regarding the Superbee limiter, it applies the minimum limiting and maximum steepening possible to remain the TVD, therefore it suffers from excessive sharpening of slopes and has a general tendency to flatten the peaks and steepen the slopes. However, in our case, when used in conjunction with the modified face interpolated scheme, it produces excellent results, both for the peaks and smooth slopes.

The Van Leer limiter is defined as follows:

$$FL(r) = \frac{\mathrm{r} + |\mathrm{r}|}{1 + |r|} \quad (15)$$

The Sweby diagram of the Van Leer flux limiter is shown in Fig.4. Regarding the Van Leer limiter, it offers smooth variation and hence no abrupt changes and it is symmetric. Is clips the smooth peaks and it is unable to capture the steep gradients from shock waves, even though it captures well the peak values.

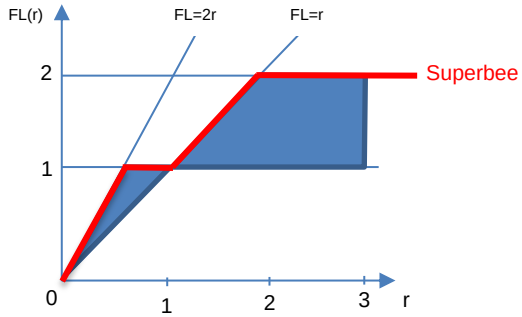


Fig.3 Sweby diagram depicting the Superbee limiter.

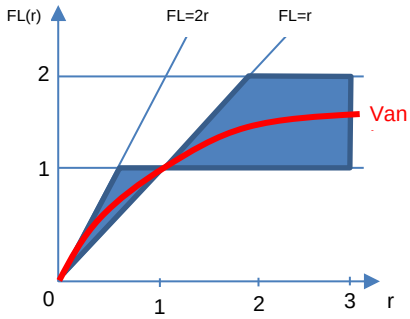


Fig.4 Sweby diagram depicting the Van Leer limiter.

The Sweby diagram of the MUSCL flux limiter is shown in Fig.5. The MUSCL limiter is defined as follows:

$$FL(r) = \max(0, min(2r, (r + 1)/2, 2)) \quad (16)$$

Regarding the MUSCL limiter, it is able to capture smooth slopes very well, however clips the peaks and it is unable to capture sharp edges.

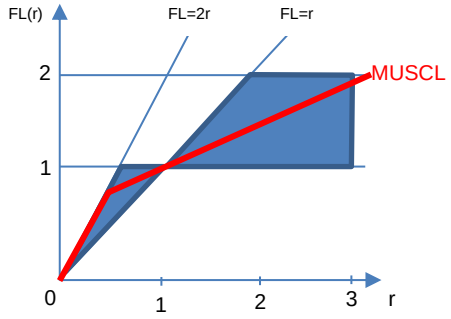


Fig.5 Sweby diagram depicting the MUSCL limiter.

The Sweby limiter is defined as follows:

$$FL(r) = \max(0, min(\beta r, 1), min(r, \beta))) \quad (17)$$

The Sweby diagram of the Sweby flux limiter is shown in Fig. 6. Regarding the Sweby limiter, in smoothed regions, it is able to capture well the slopes, however it clips the peak values. In the presence of shocks, the Sweby limiter, it is able to capture fairly well the slopes and the peak value, however not as accurately as the Superbee flux limiter.

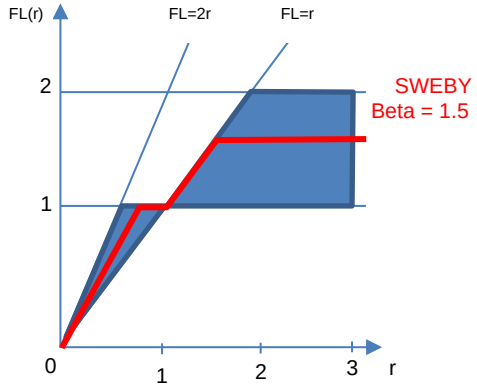


Fig.6 Sweby diagram depicting the Sweby limiter.

## VIII. Modified Flux Limiters and Sweby Diagrams

In this paper, the work of Hou et al. [13] is extended in three-dimensional coordinates in tetrahedral meshes and tested for its ability to capture the shocks and slow varying slopes in two-dimensions. The innovation of this scheme emanates from the fact that instead of limiting the centroid values between volume elements, it uses only the projections to the normal of the faces in the context of a 2nd order scheme for accuracy. This is ideal since fluxes being transported across adjacent volume elements through common sharing faces only exchange flux in the normal to the face direction, hence a centroid based approach would be expected to produce non-accurate results, especially in the case of non-good quality meshes, where the centroid lines and normal to the faces lines are far apart. Hence, this face developed algorithm depicts advanced superiority on unstructured bad quality grids and it is also very fast since it uses only the predetermined projected distances along the normal boundary faces which are kept constant along the simulations, hence reduced times are possible. Due to this process, there is no expensive interpolation necessary during the simulation process. Additionally, the mesh size differences, as well as the face positions are also taken into consideration and has superior performance in both sharp and smooth fronts when compared to Darwish et al. [14], Li et al. [15] and Hou et al. [16].

Darwish and Li calculate the face value between center and downwind element $\varphi_{cd}$ with the method proposed by Bruner et al. [17], which uses the ratio of the central difference of u to the one-sided (or upwind) difference of u to calculate $r_{C,D}$:

$$r_{C,D} = \frac{(\varphi_c - \varphi_U)}{(\varphi_D - \varphi_c)} \quad (18)$$

$$\varphi(C,D) = \varphi_c + \frac{1}{2} FL(r_{C,D})(\varphi_D - \varphi_c) \quad (19)$$

Hou et al. [16] utilize an alternative approach by utilizing weighted factors as follows:

$$r_{C,D} = \frac{(\varphi_c - \varphi_U)}{(\varphi_D - \varphi_c)} \quad (20)$$

$$\varphi(C,D) = \varphi_c + \frac{1}{RCD} FL(r_{C,D})(\varphi_D - \varphi_c) \quad (21)$$

$$RCD = \frac{(D_{CF} - D_{FD})}{D_{CF}} \quad (22)$$

where $D_{UC}$ and $D_{CD}$ depict the distances between upwind and center point and centroid and downwind point, respectively.

Most recent work by Hou et al. [13], which is the work adopted from the authors of this paper, is to use a modified better version of the above as follows:

$$r_{C,D} = \frac{(\varphi_c - \varphi_c)/D_{UC}}{(\varphi_D - \varphi_c)/D_{CD}} \quad (23)$$

$$\varphi(C,D) = \varphi_c + \frac{1}{RCD} FL(r_{C,D})(\varphi_D - \varphi_c) \quad (24)$$

In order to avoid excessive interpolation in an unstructured mesh, instead of using the values at the exact U, D, C points, we use the central element values for simplicity.

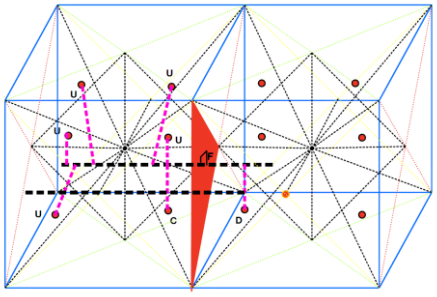


Fig.7 Mesh diagram depicting the centered, downwind and upwind nodes.

In this paper, we investigate the performance of a modified SUPERBEE limiter, used by Hou et al. as follows:

$$FL(r) = \max(0, \min(Rr, 1), \min(r, R)) \qquad (25)$$

which results in a Sweby diagram as shown in Fig. 8.

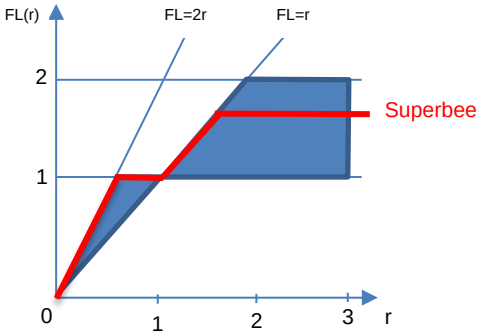


Fig.8 Sweby diagram depicting the Superbee modified limiter.

Instead of extending the centroid, we use the distances with reference to the normal line to the boundary face, hence making the algorithm more accurate, since finite volume TVD deals occurs in this direction and not along centroid directed lines.

One needs to mention that in the case the dot product between the velocity from the central value and the normal to the surface of the control volume is negative, then one needs to reconsider which are the upwind, downwind and center points, since now the center point is transformed to downwind, the upwind to center point and one needs to look and find the upwind point (multiple candidates exist) for the new center point and ex-upwind point. This transformation of the points in the case of negative dot product between the velocity and the outward normal to the boundary face, can be predetermined beforehand in the beginning, hence not adding additional computational cost during the time looping process.

The current time-dependent, convection-diffusion with source term equation solver has been developed in 3-dimensional tetrahedral mesh structures in a distributive computing environment. To minimize inter-communication, the usage of ghost cells is further complicated by also the additional need to keep track of the transformation of points in the case that the dot product is negative.

Near the boundaries, where there are possibilities that a downwind or an upwind point does not exist, and this occurs usually at the physical boundaries of the overall domain, the modified Superbee limiter is not used and the centroid value of the center element is used.

## IX. Finite Volume – TVD Simulations

In the finite volume formulation, optimization testing is performed in two-dimensions, rather than in three-dimensions, since analysis is much easier and faster to be performed. It is expected that the same accuracy and flux limiting values will hold also in the three-dimensional case as well. Hence, we test our solver in the case of a two-dimensional geometry and specifically through a square box of size 200 m x 200 m with three meshes created in NETGEN software. The square wave pulse has a starting point of (20 m, 20 m) and spans a length and width of 50 m, and a height value of 1, whereas the rest of the mesh, a height value of 0.3. The wave propagates with a velocity of 1 $ms^{-1}$ either in the x-direction only, y-direction only or in both x and y-directions.

We have tested the algorithm under various conditions. One case was to push the limit and see when results started being unstable. We have computed that approximately at a time step of 0.1 s the results were stable and accurate. Provided that we used speeds of 1 $ms^{-1}$, then the distance travelled every time step was 0.1 s, hence indicating that for a tolerance of approximately 0.5 m, the rule of thumb is that the time step needs to be approximately ~ 0.5/0.1 = 5 times less than the mesh tolerance to produce stable results. In all simulations, the wave was propagating and traversing across the whole mesh. It was found that the modified Van Leer scheme became unstable in the Mesh3 case, which we believe is a combination of the Van Leer scheme stability and of the mesh quality, whereas the modified Superbee scheme exhibited stable behavior at all times. Table No1. shows the data information for the three different meshes used such as node and element numbers, tolerance of the mesh and minimum element quality. Additionally, we depict the Mean Percentage Error (MPE) for the propagation of a square wave pulse for 100 s, which is calculated as follows:

$$\text{MPE} = \frac{100}{n} \sum_{i<n} \frac{A_n - E_n}{A_n} \qquad (26)$$

where $A_n$ is the analytical solution and $E_n$ is the estimated solution and n is the number of mesh elements.

The very low values of the MPE error show that our results are in excellent agreement with the

analytical solution after 10,000-time steps and the shock can be captured, using the modified Superbee flux limiter. Furthermore, one can deduce that the solution converges, as finer meshes are used, which is expected according to the finite element mathematical formulation.

Table No1: Meshes used in the simulation of finite volume-total variation diminishing technique

| | Nodes | Elements | Tolerance | Minimum element quality | Mean Percentage Error (MPE %) Square Pulse) |
|---|---|---|---|---|---|
| Mesh No1 | 46739 | 92672 | 0.746 | 0.808 | 0.024 |
| Mesh No2 | 72532 | 144058 | 0.559 | 0.794 | 0.021 |
| Mesh No3 | 109725 | 218216 | 0.488 | 0.800 | 0.020 |

Since we thrive for the best, we tried to improve the simulation results even further, hence thereafter, we used the Mean Percentage Absolute Error (MPAE), which is calculated as the sum of the absolute value of the difference between analytical and estimated value divided by analytical value as follows:

$$\text{MPAE} = \frac{100}{n}\sum_{i<n}\left|\frac{A_n - E_n}{A_n}\right| \qquad (27)$$

to give as a more accurate sense of the error in the results.

In order to test also the behavior of our solver in diffused wave pulses, we conducted a test of a Gaussian pulse propagated in time along the x-direction for 100 s. This was performed in steps of Δt = 0.01 s and 10,000 steps. The simulation was performed for the three different meshes extracted from NETGEN software. The Gaussian pulse has the following characteristic equation:

$$GP(x,y) = a\, e^{\frac{-(b^2*((xinit-x)^2+(yinit-y)^2)}{2c^2}} \qquad (28)$$

where a = 1, b = 0.4 , c = 50, xinit = 46 m and yinit = 46 m.

Fig. 9 below shows the two-dimensional distribution of the propagated Gaussian pulse in the x-direction for a duration of 100 s and shows the ability of the modified Superbee limiter to capture and maintain the shape of the pulse through long calculations.

Fig. 10 shows the one-dimensional comparison between analytical and actual solution for the propagation of a Gaussian pulse for a duration of 100 s at 10,000 steps of Δt = 0.01 s using the modified Superbee limiter. We observe that in this case, there is some flattening at the top which emanates from the tendency of the modified Superbee limiter to sharpen and square edges, as well as minor differences at the bottom of the Gaussian pulse again from the sharpening tendency of the limiter.

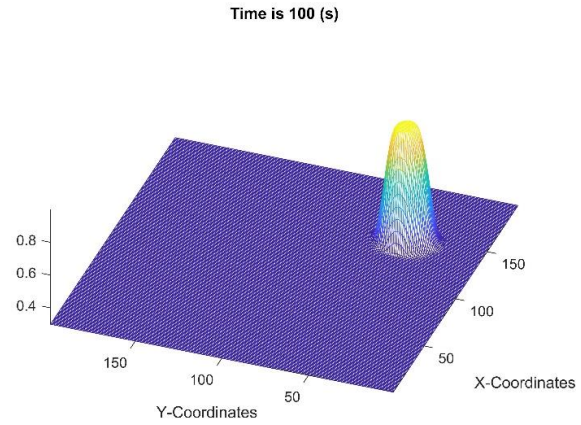


Fig. 9. Two-dimensional plot of a Gaussian plot propagated using the FV-TVD scheme for 100 s in the x-direction using the modified Superbee limiter.

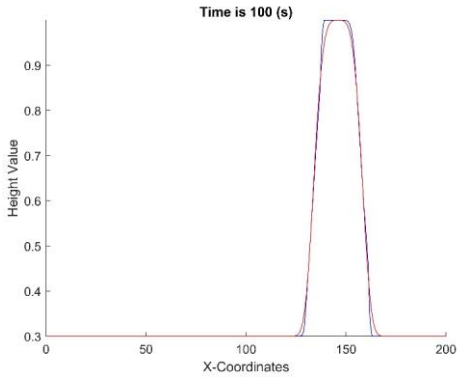


Fig. 10. One-dimensional plot depicting the comparison between analytical (red solid line) and actual solution (blue solid line) of a Gaussian plot propagated using the FV-TVD scheme for 100 s in the x-direction.

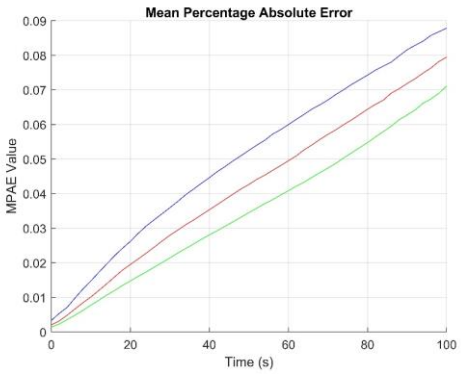


Fig. 11. The Mean Percentage Absolute Error value - MPAE (%) is plotted vs time for a Gaussian pulse travelling in the x-direction for

100 s using the modified Superbee limiter for 3 different meshes – blue (MeshNo1), red (MeshNo2) and green (MeshNo3).

Fig. 11 shows the MPAE for the three different meshes regarding the propagation of a Gaussian pulse in the x- direction for 100 s. It is shown that the error in all three cases, grows linearly with time as the Gaussian pulse traverses through the domain. The finer the mesh, the initial offset is closer to 0, and more or less, all three cases follow a linear increase, each one with a different gradient, with the finer mesh having the smallest slope. The results are as expected, since the solution seems to converge as a finer mesh is used.

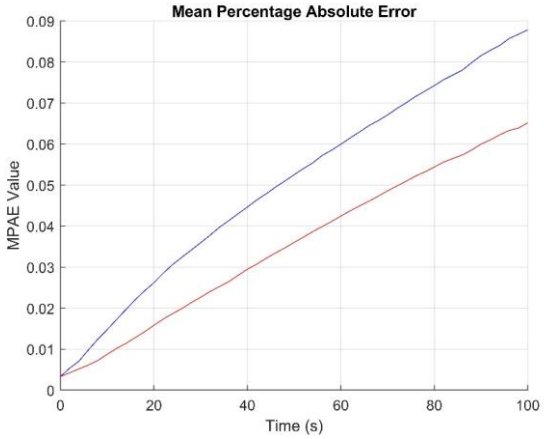


Fig. 12. The Mean Percentage Absolute Error value - MPAE (%) is plotted vs time for a Gaussian pulse travelling in the x-direction for 100 s using the modified Superbee and the modified Van Leer limiter for MeshNo1.

Fig. 12 shows the MPAE comparison between the modified Superbee and Van Leer schemes in the propagation of a Gaussian pulse for a duration of 100 s in the x-direction for the same mesh and shows that there is a significant difference between the results in favor of Van Leer modified scheme. However, it has been observed that the Van-Leer scheme may also cause random instabilities and cannot be used exclusively as a modified flux limiter. From the above results, it is obvious that the modified Superbee performs much better at shock wave like phenomena, whereas the Van-Leer performs better at diffused wave like phenomena, as expected from the nature of the schemes.

Next, we tested something innovative, never been done before to our knowledge, the combination of modified Van-Leers and Superbee limiters to try and get the best of both worlds. The criterion when choosing to use the modified Van-Leer and the modified Superbee was the $\varphi(C,D)$ value of the modified Superbee scheme. Two limits were set for the simulations. The lower limit which was 0.1, meant that below this value, the modified Superbee would be used at all times to avoid any instabilities caused by the modified Van Leer flux limiter. Then we tested to find a lower modified Superbee limit, where the modified Superbee would kick in, to ensure to capture shock waves. In the range between lower and upper limit, the modified Van Leer was used that produced diffused results.

Fig. 13 shows a direct comparison between the analytical and actual solution of the propagated Gaussian pulse and show to be in excellent agreement, provided that the wave has already propagated 10,000 steps. The criteria used was using the mixed modified Superbee and the modified Van Leer limiter for MeshNo3 using 0.1 and 1.9 limits vs the Van Leer scheme. It was shown that the MPAE error halved, which gave as confidence that our approach was in the right direction. Thereafter, we have run a series of tests and plotted dynamically the modified Superbee $\varphi(C,D)$ of the wave and identified the limits of 0.1 and 0.8 as ideal. We have seen that the MPAE error was not changing significantly when compared to the 0.1 and 1.9 limits, however the capturing of the shape of the curve was significantly improved.

Fig. 14 shows the two-dimensional plot for the propagation of a square wave pulse in the x-direction for a duration of 100 s at 10,000 steps of Δt = 0.01 s using the combined modified Superbee and Van Leer flux limiter using 0.1 and 0.8 limits. We observe that the proposed method is nearly ideal, preserving the shape of the square wave pulse, even after 10,000-time steps, producing nearly ideal results.

Fig. 15 shows the one-dimensional plot depicting the comparison between analytical and actual solution of a square wave pulse propagated using the FV-TVD scheme for 100 s in the x-direction using the mixed modified Superbee and the modified Van Leer limiter for MeshNo3 using 0.1 and 0.8 limits. This graph shows clearly the ability of the proposed algorithm to capture shock waves in a nearly ideal way.

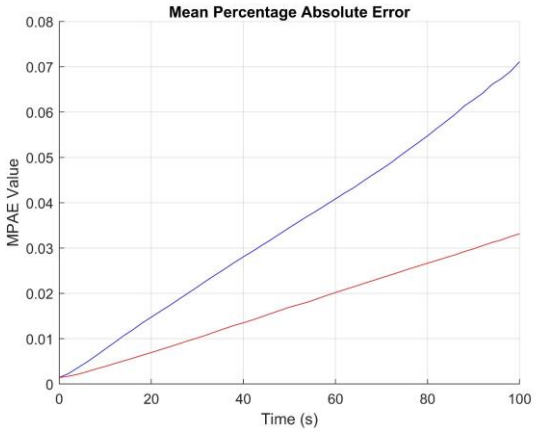


Fig. 13. The Mean Percentage Absolute Error value - MPAE (%) is plotted vs time for a Gaussian pulse travelling in the x-direction for 100 s using the mixed modified Superbee and the modified Van Leer limiter for MeshNo3 using 0.1 and 1.9 limits (red line) vs Van Leer (blue) scheme.

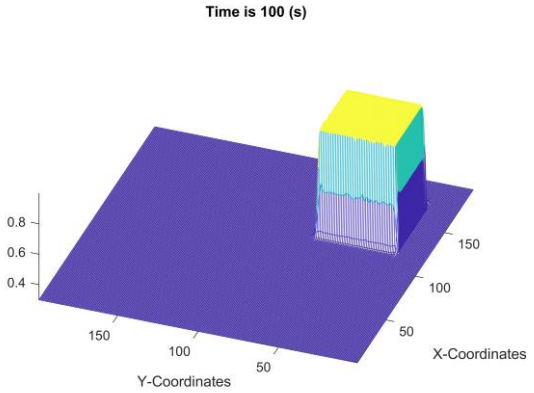


Fig. 14. Two-dimensional plot of a square wave plot propagated in the x-direction using the FV-TVD scheme for 100 s using the mixed modified Superbee and the modified Van Leer limiter for MeshNo3 using 0.1 and 0.8 limits.

.

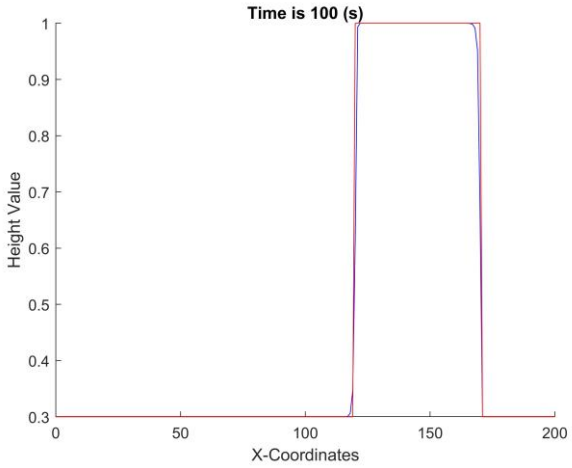


Fig. 15. One-dimensional plot depicting the comparison between analytical (red solid line) and actual solution (blue solid line) of a square wave pulse propagated using the FV-TVD scheme for 100 s in the x-direction using the mixed modified Superbee and the modified Van Leer limiter for MeshNo3 using 0.1 and 0.8 limits.

Fig. 16 shows the one-dimensional plot depicting the comparison between analytical and actual solution of the Gaussian wave pulse propagated using the FV-TVD scheme for 100 s in the x-direction using the mixed modified Superbee and the modified Van Leer limiter for MeshNo3 using 0.1 and 0.8 limits. This graph shows clearly the ability of the proposed algorithm to capture diffused waves in a nearly ideal way. The fact that we are able to achieve a universal scheme with flux limiting values of 0.1 and 0.8 that can guarantee accurate results, both in diffused and shock phenomena, paves the way for the basis of very accurate Euler, Navier-Stokes and turbulent solvers, that could potentially establish KYAMOS software as market leader in the CAE market in the long run.

To conclude, we anticipate that if one uses even finer meshes, the algorithm will eventually converge and produce results that will be indistinguishable to the naked eye.

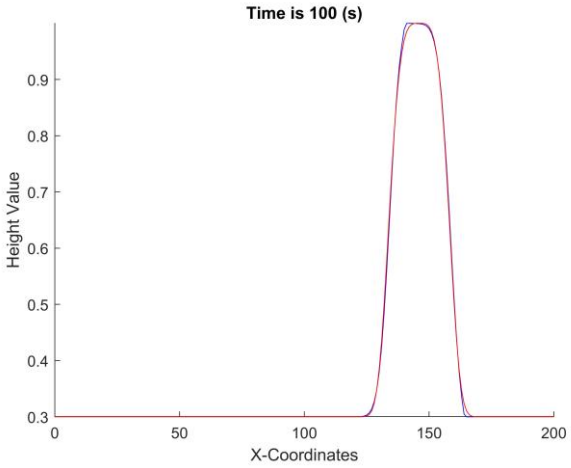


Fig. 16. One-dimensional plot depicting the comparison between analytical (red solid line) and actual solution (blue solid line) of a Gaussian wave propagated using the FV-TVD scheme for 100 s in the x-direction using the mixed modified Superbee and the modified Van Leer limiter for MeshNo3 using 0.1 and 0.8 limits.

## X. CONCLUSIONS

KYAMOS software aims to realize the formulation, development, validation, and optimization of FV-TVD solvers that can be applied to engineering problems by utilizing high performance computing through cloud-based distributed GPUs and state-of-the-art mathematical algorithms. One of the most challenging problems in computational fluid simulations is the ability of an algorithm to be able to capture a flow accurately and efficiently, without artificial numerical diffusion, loss of monotonicity and spurious oscillations. It is clearly shown that our suggested scheme overall behaves extremely well in analyzing both diffused and shock phenomena of the advective term and is expected to be a game changing solver for the simulation of multiphysics engineering phenomena.

## ACKNOWLEDGMENT

This work was co-funded by the European Regional Development Fund and the Republic of Cyprus through the Research and Innovation Foundation (Project: START-UPS/0618/0058).

## REFERENCES

[1] UberCloud. (2019, 09-12-2020). *When CAE Meets AI: Deep Learning For CFD Simulations*. Available: https://blog.theubercloud.com/when-cae-meets-ai-deep-learning-for-cfd-simulations

[2] B. Albright. (2019, 09-12-2020). *Deep Learning and Design Engineering*. Available: https://www.digitalengineering247.com/article/deep-learning-and-design-engineering

[3] J. N. Kutz, "Deep learning in fluid dynamics," *Journal of Fluid Mechanics,* vol. 814, pp. 1-4, 2017.

[4] J. Ling, M. F. Barone, W. Davis, K. Chowdhary, and J. Fike, "Development of

Machine Learning Models for Turbulent Wall Pressure Fluctuations," in *55th AIAA Aerospace Sciences Meeting*(AIAA SciTech Forum: American Institute of Aeronautics and Astronautics, 2017.

[5] K. O. Lye, S. Mishra, and D. Ray, "Deep learning observables in computational fluid dynamics," *Journal of Computational Physics,* vol. 410, p. 109339, 2020.

[6] I. Sadrehaghighi, *Artificial Intelligence (AI) and Deep Learning For CFD*. 2020.

[7] B. N. Hanna, N. T. Dinh, R. Youngblood, and I. Bolotnov, "Coarse-Grid Computational Fluid Dynamics (CG-CFD) Error Prediction using Machine Learning," *arXiv: Fluid Dynamics,* 2017.

[8] (09-12-2020). *What-are-the-hot-topics-in-Fluid-Dynamics-involving-Machine-Learning*. Available: https://www.quora.com/What-are-the-hot-topics-in-Fluid-Dynamics-involving-Machine-Learning

[9] Y. Cao, A. Daskin, S. Frankel, and S. Kais, "Quantum circuit design for solving linear systems of equations," *Molecular Physics,* vol. 110, no. 15-16, pp. 1675-1680, 2012.

[10] R. Steijl, "Quantum algorithms for fluid simulations," 2019.

[11] A. P. Papadakis, "KYAMOS Multiphysics Software Lattice Boltzmann Solver."

[12] G. Li, D. Bhatia, and J. Wang, "Compressive properties of Min-mod-type limiters in modelling shockwave-containing flows," *Journal of the Brazilian Society of Mechanical Sciences and Engineering,* vol. 42, no. 6, p. 290, 2020/05/12 2020.

[13] J. Hou, F. Simons, and R. Hinkelmann, "A new TVD method for advection simulation on 2D unstructured grids," *International Journal for Numerical Methods in Fluids,* vol. 71, no. 10, pp. 1260-1281, 2013.

[14] M. Darwish and F. Moukalled, "TVD schemes for unstructured grids," *International Journal of heat and mass transfer,* vol. 46, no. 4, pp. 599-611, 2003.

[15] L.-x. Li, H.-s. Liao, and L.-j. Qi, "An improved r-factor algorithm for TVD schemes," *International Journal of Heat and Mass Transfer,* vol. 51, no. 3-4, pp. 610-617, 2008.

[16] J. Hou, F. Simons, and R. Hinkelmann, "Improved total variation diminishing schemes for advection simulation on arbitrary grids," *International Journal for Numerical Methods in Fluids,* vol. 70, no. 3, pp. 359-382, 2012.

[17] C. Bruner, R. Walters, C. Bruner, and R. Walters, "Parallelization of the Euler equations on unstructured grids," in *13th Computational Fluid Dynamics Conference*, 1996, p. 1894.