A Very High-Order Finite Volume Technique for Convection-Diffusion Problems on Unstructured Grids

Filipe José Mendes Diogo
filipe.diogo@tecnico.ulisboa.pt

Instituto Superior Técnico, Universidade de Lisboa, Portugal

June 2019

Abstract

In this Thesis, a previously developed high-order finite volume scheme for unstructured grids is improved and extended to convection-diffusion problems. The method used is based on the weighted least-squares and reconstructs a high-order polynomial at each face center. A scaling technique is introduced when solving the least-squares problem resulting in a lower condition number and faster inversion time of the local least-squares matrix. The proposed algorithm is developed and studied for second, fourth, sixth and eighth orders schemes, showing a convergence order close to the theoretical one for several analytical solutions and grid topologies. A different stencil expansion algorithm, based on face neighbors, is tested, resulting in a higher accuracy while decreasing the required memory and solver run time (SRT). A new weight function is studied, having as main criteria the condition number of the global and local matrices, and also the convergence order of the mean error. Finally, a study on different meshes types (Cartesian, triangular, polyhedral, hybrid and irregular quadrilaterals) and scheme orders is done to compare the best and most efficient combinations for regular and irregular grids. This efficiency is compared by computing the memory and SRT required by each scheme versus the obtained accuracy.

Keywords: High-order Scheme, Finite Volume Method, Weighted Least-Squares, Unstructured Grids, Convection-Diffusion Equation.

1. Introduction

Almost all natural phenomena can be described and modeled via a set of mathematical equations. These analytical descriptions are one of the fundamental techniques used by scientists and engineers to understand and predict different physical processes. In the case of fluids, how they move and interact with objects, can be described by a set of partial differential equations (PDEs). Through these equations, it is possible to determine multiple parameters of a fluid flow such as temperature, pressure and velocity. The most common way of finding a solution for these equations is describing them into an algebraic form and solve them using computers. This is known as computational fluid dynamics (CFD). In the aerospace industry, CFD had a tremendous impact since it made possible to simulate the air flow around aircrafts without the need of wind tunnel testing, in every design iteration. This gave the aircraft manufacturers, the potential of decreasing the development time and cost associated to each product. Nonetheless, it is important to validate, whether the models and descriptions used are producing realistic results. To do so, complementary tests on wind tunnels should be made, but once the model is validated, the need for these tests decreases. The most used model for turbulent flows is konwn as RANS (Reynolds averaging Navier-Stokes), that allows engineers to use moderately refined grids and still produce coherent results. However, on sensitive regions where airflow separation occurs, this model is not accurate enough. Since it is still not feasible to use DNS (Direct numerical simulations) on these scales, LES (Large Eddy Simulation) emerges as a potential solution for highly accurate results. This method has the downside of demanding a very refined grid, not as much as DNS, but still consuming a lot of computational power. Fortunately, the computational power is increasing and LES is becoming an alternative option for most aircraft manufacturers. The commonly used second order discretizations can be replaced by higher-order schemes to help decrease the computational cost, while improving the accuracy [1]. Also by requiring less computational resources, the time needed for each simulation would

decrease. This would open the possibility to use better optimization tools, currently limited by the number of design variables (ND), since the number of simulations in an optimization problem varies approximately with $\mathcal{O}(\mathrm{ND}^2)$.

2. Background

In a comparison study made by Zing *et al.* [2], it was demonstrated that higher-order methods can achieve a better accuracy in comparison with second order schemes. Also, Visbal *et al.* [3], using a finite difference sixth order method, showed the superior accuracy and robustness of high-order schemes on deformed grids and boundary layer simulations.

In the finite volume (FV) field, high-order techniques had as one of its precursor the work of Barth and Frederickson in 1990 [4], with a quadratic reconstruction of the Euler equations. Since then, high-order finite volume methods have been developed by Ollivier-Gooch *et al.* [5, 6, 7], with schemes up to fourth order, Nogueira *et al.* [8, 9], with studies of high-order laminar flow around a cylinder and the sixth order schemes on unstructured grids developed by Clain *et al.* [10, 11, 12, 13]. The moving least-squares (MLS) introduction in FV was proposed by Cueto-Felgueroso *et al.* [14, 15, 16, 17], and has also influenced the works Nogueira *et al.* [8, 9], previously mentioned. This technique uses a weight function, in the least-squares problem, to obtain with greater accuracy the original function and its derivatives. The weight function used is directly related to the distance between the cell sample and the face reconstruction location.

The study made by Nogueira *et al.* [18], compared the finite volume method and the Discontinuous Galerking (DG) one, when solving compressible flow. It showed the finite volume advantages in terms of degrees of freedom (Dof), concluding that for the Navier-Stokes equations, the finite volume method requires four times less Dof for the same accuracy level.

Following the work of the previous authors, Vasconcelos *et al.* [19] developed a stencil generation algorithm based on vertex neighbor cells, able to deal with unstructured meshes. In his work, several high-order finite volume diffusive schemes are presented (up to eighth order), reconstructing the polynomial at the cell faces. The proposed method is also shown to work correctly on highly anisotropic meshes used on boundary layer grids.

Some authors have shown the importance of using high-order schemes combined with LES models as a mean to use coarser grids while maintaining the same accuracy. Neto *et al.* [20] has compared second and fourth order schemes for a compressible gaseous reactive mixture using Navier-Stokes equations. Also did Kok *et al.* [21] for a flow on a cavity, concluding that the fourth order scheme could have a grid twice as large to reach the same accuracy as second order.

3. Implementation

A convection-diffusion equation is composed by a diffusion term, a convective term and an optional source term. This can be seen on equation 1, where ϕ is the variable being studied (for example: velocity, temperature, etc), $\mathbf{V}$ stands for the convective velocity, Γ is the diffusion coefficient and φ_ϕ is the source term. No time derivatives are written, since the system is considered to be in steady state.

$$\bigtriangledown.(\mathbf{V}\phi) - \bigtriangledown.(\Gamma \bigtriangledown \phi) = \varphi_\phi \tag{1}$$

By integrating the same equation over a control volume (CV), equation 2 is obtained. This is considered the first step required to apply the finite volume method (FVM).

$$\int_{CV} \bigtriangledown.(\mathbf{V}\phi)dV - \int_{CV} \bigtriangledown.(\Gamma \bigtriangledown \phi)dV = \int_{CV} \varphi_\phi dV \tag{2}$$

The Gauss Divergence Theorem states that the outward flux property in a close surface is equal to the volume integral of its divergence. In this way, the CV integral of the previous equation can be rearranged into a control surface (CS) integral, with $\mathbf{n}$ being the outward face normal [22].

$$\int_{CS} (\phi\mathbf{V}).\mathbf{n}dS - \int_{CS} \Gamma(\bigtriangledown\phi).\mathbf{n}dS = \int_{CV} \varphi_\phi dV \tag{3}$$

Taking into account equation 3 and adapting it to a two-dimensional plane, it is necessary to consider that the variable values change along the cell face. So, in order to have a high-order integration at each face, a high-order 1D Gauss Quadrature is adopted. The generalized equation for a two-dimensional case can be seen in equation 4, where $\mathcal{F}(I)$ is a set of faces that compose cell I, $\mathcal{G}(f)$ is the group of Gauss points of face f, w_{G_g} are the Gauss weights and $\mathbf{S_f}$ is the face area multiplied by the face normal vector. Since this approach reconstructs a high-order polynomial at the cell faces, it is designated by face least-squares (FLS). Contrary to what is used for cell-centered reconstructions, this method does not require an interpolation to obtain the face value or fluxes.

$$\sum_{f\in\mathcal{F}(I)} \sum_{g\in\mathcal{G}(f)} w_{G_g}(\phi_g\mathbf{V} - \Gamma\nabla\phi_g)\cdot\mathbf{S_f} = \int_{CV} \varphi_\phi dV, \tag{4}$$

The value of ϕ_g is obtained by using the Taylor expansion written in equation 5, where $\mathbf{d}_f(\mathbf{x},\mathbf{y})$ is $[1,\ (x-x_f),\ (y-y_f),\ \cdots]$, and replacing x and y by the Gauss point coordinates. In this case, and for

simplicity, all terms that multiply by the distance can be included into the coefficients $\mathbf{c}_f$. The accuracy of the numerical scheme will be directly related to the polynomial degree by $p+1$, where p is the polynomial order. The polynomial coefficients will be estimated for each face center (x_f, y_f) and only afterwards the value of ϕ_g is computed. To obtain $\nabla\phi_g$, it is necessary to derive equation 5 relative to the desired coordinate and replace, as before, x and y by the Gauss point coordinates.

$$\phi_f(\mathbf{x},\mathbf{y}) = \mathbf{d}_f(\mathbf{x},\mathbf{y})\,\mathbf{c}_f. \tag{5}$$

Since the algorithm needs to be prepared to work with unstructured grids, it is not feasible to compute directly the coefficients $\mathbf{c}_f$. Furthermore, the choice of the appropriate cells to use would be a complex task, since an unbalanced stencil could affect the results. For this reason, a higher number of cells than polynomial coefficients is used, guaranteeing that the stencil is balanced and its stencil generation algorithm is simpler. To deal with this additional number of cells, the coefficients are estimated using the weighted least-squares algorithm.

The generic formula of weighted least-squares (WLS) in matrix form can be seen on equation 6. The cells values ($\boldsymbol{\phi}_s$) for this work are also unknown, since it is being proposed an implicit solution of the global system. The weight matrix ($\mathbf{W_{LS}}$) is a square matrix with its diagonal fill with the respective weights of each stencil point. The weight function is inversely proportional to the distance from the stencil point to the face where the polynomial is being reconstructed, powered by a shape factor k. Since the purpose of this technique is to find the coefficients values, in equation 7, these are isolated on the left hand side of the equation, where $\mathbf{P_I}$ (pseudo-inverse matrix) is equal to $\left(\mathbf{D}_f^T\mathbf{W_{LS}}_f\mathbf{D}_f\right)^{-1}\mathbf{D}_f^T\mathbf{W_{LS}}_f$.

$$\mathbf{D}_f^T\mathbf{W_{LS}}_f\mathbf{D}_f\mathbf{c}_f = \mathbf{D}_f^T\mathbf{W_{LS}}_f\boldsymbol{\phi}_s. \tag{6}$$

$$\mathbf{c}_f = \mathbf{P_I}\boldsymbol{\phi}_s \tag{7}$$

By combining equation 5 with 7, it is possible to obtain the value of ϕ_f as a function of the cell values (ϕ_s) used in the stencil(equation 8).

$$\phi_f(\mathbf{x},\mathbf{y}) = \mathbf{d}_f(\mathbf{x},\mathbf{y})\,\mathbf{P_I}\boldsymbol{\phi}_s. \tag{8}$$

Since ϕ_f is directly related with a convection term of the global equation, the previous equation can be modified to $\phi_f(x,y) = \mathbf{T}_f^C(x,y)\,\boldsymbol{\phi}_s$, where $\mathbf{T}_f^C$ is a row vector with number of columns equal to the number of stencil samples.

To compute the gradient of the property ($\nabla\phi_f$), equation 8 is derived obtaining:

$$\nabla\phi_f(\mathbf{x},\mathbf{y}) = \left(\nabla\mathbf{d}_f(\mathbf{x},\mathbf{y})\right)\mathbf{P}_I\boldsymbol{\phi}_s \tag{9}$$

In a two-dimensional case, $\nabla\mathbf{d}_f$ has components in the X and Y directions and, for that reason, it is a matrix with two rows and a number of columns equal to the number of coefficients. Computing the gradient is, in finite volume, associated with a diffusion term. So, the last equation can be written as $\nabla\phi_f(\mathbf{x}) = \mathbf{T}_f^D\boldsymbol{\phi}_s$.

Bringing together the left hand side of equation 4 with the recently obtained equation 8, for convection, and 9, for diffusion, the final equation 10 is attained and will be used to fill in the system $\mathbf{A\Phi} = \mathbf{b}$. The convection ($\mathbf{T}_f^C$) and diffusion ($\mathbf{T}_f^D$) matrices need to be updated with the Gauss points locations ($\mathbf{X}_g$), since the polynomials are reconstructed at the faces centroids.

$$\sum_{f\in\mathcal{F}(P)}\sum_{g\in\mathcal{G}(f)} w_{G_g}(\mathbf{T}_f^C(\mathbf{X}_g)\boldsymbol{\phi}_s\mathbf{V} - \Gamma\mathbf{T}_f^D(\mathbf{X}_g)\boldsymbol{\phi}_s)\cdot\mathbf{S_f}, \tag{10}$$

4. Source term

The right hand side of equation 4 has to be integrated using gauss quadrature (GQ). To make the 2D algorithm simpler and since it has to deal with unstructured grids, all cells shapes will be divided into smaller triangles.

The integral of the source term can then be rearranged as in equation 11, where $\mathcal{T}(P)$ is the group of triangles that compose a cell, S_{Δ_i} is the area of triangle i and $\mathcal{G}_i$ are the gauss points of that same triangle.

$$\int_{CV}\varphi_\phi dV = \sum_{i\in\mathcal{T}(P)}\left(S_{\Delta_i}\sum_{j\in\mathcal{G}_i(\Delta_i)} w_{G_j}\varphi_\phi(\mathbf{x}_j)\right), \tag{11}$$

The coordinates are modified according with equation 12, where x_1, y_1, x_2, y_2, x_3 and y_3 are the triangle vertices and ζ_{1_j}, ζ_{2_j} and ζ_{3_j} are the simplex coordinates taken from [23] and [24].

$$\mathbf{x}_j = \boldsymbol{\zeta}\mathbf{x}_v = \begin{bmatrix}\zeta_{1_j} & \zeta_{2_j} & \zeta_{3_j}\end{bmatrix}\begin{bmatrix}x_1 & y_1\\ x_2 & y_2\\ x_3 & y_3\end{bmatrix} \tag{12}$$

5. Stencil Generation

The stencil is a set of samples or points that are used to reconstruct the polynomial at a given face. The main goal is to generate a stencil as compact as possible, but still have sufficient samples for each polynomial reconstruction. In order to reduce the size of the stencils and the fitting error, a new approach to generate the stencils was tested. Instead of using vertex neighbors, it uses a face neighbors expansion for fourth order schemes or higher. This means that the base stencil (associated with second

order scheme) is always generated with all the cells that have at least one vertex of the face.

During the development and testing of these new stencils, it was understood that there is a minimum number of samples for the WLS to work properly. So, in order to make the algorithm robust to every grid type, a minimum number of stencil points was defined for each polynomial order considered. Those numbers were chosen based on smallest number of points that the tested grids could have, but still producing coherent results. Therefore, for the second order scheme the minimum number is 4, for the fourth is 16, for sixth is 30 and lastly for the eighth order scheme is 48.

5.1. Interior faces

Assuming as interior faces, all the faces on which their respective stencil does not intersect any of the boundaries. The stencil will have a structure like the one shown on figure 1, generated around each face and based on face neighbor cells. In that figure, the level 1 is for a second order scheme, the level 2 is for fourth, level 3 is for sixth and level 4 is for eighth.

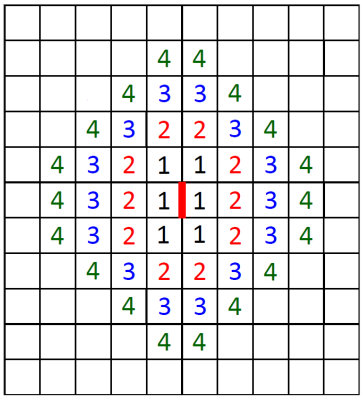


Figure 1: Face neighbors stencils for all orders, on a Cartesian grid.

5.2. Boundaries

In case there is a cell from the stencil, with one of its faces being a boundary face, the algorithm will automatically add that face to the stencil. However, depending on the scheme order required, adding the boundary faces as stencil samples may not be enough. In order to overcome this situation, a parameter was developed to verify the stencil size and on which direction it should be expanded, when required. This parameter, designated dimensionless length of the stencil, is given by equation 13, where it is computed the stencil length in the X direction ($L_x^s = x_{\max}^s - x_{\min}^s$), divided by the mean reference length of the stencil cells. This last value is given by equation 14, where L_x^i is the length of cell i in X direction, n_{cells}^s is the number of cell in the stencil and $\mathcal{N}_c(f)$ is the stencil set, corresponding to the face f. The factor 1.6 is added considering two main factors, the first is that to compute the stencil length the cell centroids are used, so a factor of one had to be considered. The second is that, since the border faces can enter the stencil samples, it is important to add 0.5 to account for that. The additional 0.1 is a tolerance factor. The same procedure is done to compute the dimensionless length of the stencil in the Y direction (n_y^s).

$$n_x^s = \frac{L_x^s}{L_{x_{ref}}^s} + 1.6, \tag{13}$$

$$L_{x_{ref}}^s = \frac{1}{n_{cells}^s} \sum_{i \in \mathcal{N}_c(f)} L_x^i, \tag{14}$$

Figure 2 shows an example for an eighth order scheme stencil, on a Cartesian grid. In this case, the algorithm expands the stencil up to level four, checks the dimensionless length in the vertical direction and verifies that is greater than 8, so there is no need for vertical expansion and the vertical limits are defined. Then the dimensionless horizontal length is checked and the value obtained is $n_x^s = 6.1$ that is rounded to 6. To reach a $n_x^s = 8$, two additional expansions are needed, adding levels 5 and 6 that don't exceed the vertical limits of the stencil. The final value of n_x^s is $8.1 \approx 8$, ending the stencil generation algorithm.

Figure 2: Boundary stencil for an eighth order scheme, on Cartesian grid.

6. Local matrix condition number improvement

The condition number of each matrix gives an measure on how perturbations or round off errors can affect the system accuracy. A higher ratio means that the inversion process is subjected to more errors and an increased computational time, depending on the selected algorithm.

To address this situation a scaling technique inspired in Gooch *et al.* [7] is adopted. Such procedure is also used on machine learning algorithms and consists in normalizing each column of matrix

$\mathbf{D}_f$. Each column is divided by its maximum absolute value, making all column values ranging between -1 and 1. To return to their proper values, after the obtaining $\mathbf{P_I}$, each row is divided by the maximum value of each $\mathbf{D}_f$ column. The theoretical results of the pseudo inverse matrix should be the same, however due to round off errors, better results can be obtained when using this technique.

Considering only the performance gains, in table 1, are the obtained mean condition numbers of matrix $\mathbf{D}_f^T\mathbf{W_{LS}}_f\mathbf{D}_f$ in a Cartesian grid with vertex neighbor stencils, using the scaling technique ($cond(T_S)$) and the regular procedure ($cond(T)$). As it is possible to see, both techniques present high condition numbers, however there is a significant reduction when using scaling. Another fact is that, on the same scheme, without using scaling the condition number increases as the reference length (h_{ref}) diminishes, but by using this technique the condition number tends to be more stable, even decreasing slowly, with mesh refinement. Similar results were also obtained for the remaining second, fourth and sixth order schemes.

Table 1: Comparison of the average condition number of matrix ($\mathbf{D}_f^T\mathbf{W_{LS}}_f\mathbf{D}_f$) using the scaling technique ($cond(T_S)$) and the regular procedure ($cond(T)$), on a Cartesian grid.

Cratesian grid		FLS8	
n_{cells}	h_{ref}	$cond(T)$	$cond(T_S)$
400	5.00E-02	4.63E+20	6.47E+14
1600	2.50E-02	2.67E+24	3.29E+14
6400	1.25E-02	2.26E+28	1.66E+14

7. Weight function

Due to the more compact stencils, it was considered necessary to re-test if the weight function used on the vertex neighbors stencils and studied by Vasconcelos [25] was still the optimal one. Two weight functions were studied, however based on the local and global matrices condition numbers and in the norm-1 error, it was concluded that the function $w_{LS_1}(\mathbf{x}_P) = \frac{1}{(d_{fP})^k}$, proposed by Vasconcelos, was still the one providing the best results. In this function, $d_{fP} = \sqrt{(x_P - x_f)^2 + (y_P - y_f)^2}$ is the distance between the face centroid $\boldsymbol{f}$ and the cell centroid $\boldsymbol{P}$ and the variable k is the shape factor.

The shape factor values proposed by Vasconcelos were no longer the optimal ones for this new stencil configuration. A study considering only w_{LS_1} and a varying k lead to the results in figure 3, where it is shown the norm-1 error for three different grid types. It is seen a stabilization of the error close to $k = 6$. From previous tests, it was observed that the local matrix condition number can increase up to 74 times, when the shape factor changes from 6 to 8, on a Cartesian grid. Due to these higher condition numbers and relatively low gains in the norm-1 error (at most 6.3%), it was opted to use $k = 6$ as the shape factor for the eighth order scheme. Since similar results were also obtained for the other orders, all schemes will use w_{LS_1} with k=6.

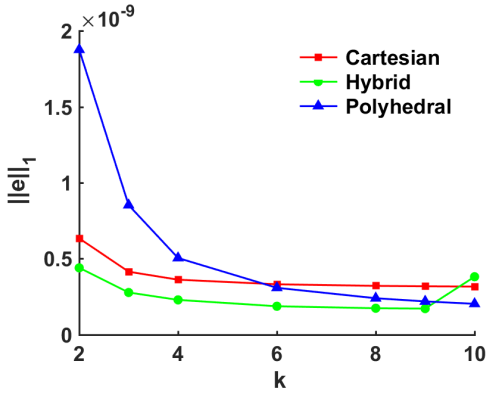


Figure 3: The $\|e\|_1$ for eighth order scheme on three different grid types, using the weight function w_{LS_1}.

8. Results

8.1. Grid type comparison

In this section, the accuracy of each grid type is compared, by using a convection-diffusion solution (equation 15) with a convective coefficients of (10,10) and a diffusive coefficient of 1. To help the comparison, the grids are divided into two categories: regular and irregular. The obtained results for the regular triangular grid are shown in figure 4, while the results for the irregular, quadrilaterals and polyhedrons, are presented in figure 5.

$$\psi_i = \sum_{i=1}^{4} \exp\left(-120\left((x - x_i)^2 + (y - y_i)^2\right)\right), \tag{15}$$

In order to quantify, these differences in terms of ratios, for each grid type, the best fit of coefficients a and b in the logarithmic equation $log(\|e\|_1) = a \times log(h_{ref}) + b$ was computed. From these results, it was estimated the norm-1 error for a $h_{ref} = 0.01$. The norm-1 error was then divided by the same error type estimated for the polyhedral grid, obtaining the norm-1 error ratios. The ratios, for regular grid types, can be seen in table 2 and show that the polyhedral grid is the best for second, sixth and eighth orders, while for the fourth order scheme the Cartesian grid is the most accurate. As for irregular meshes, the best option for the second and fourth order schemes are the irregular quadrilaterals, while for the sixth and eighth orders are the irregular polyhedrons. The ratios of the irregular

grids can be seen in table 3.

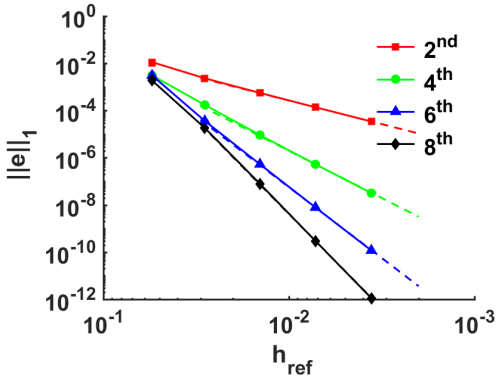
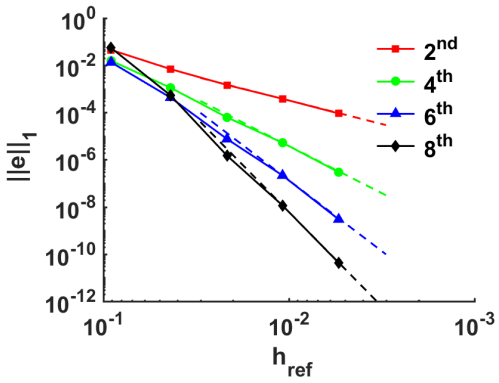
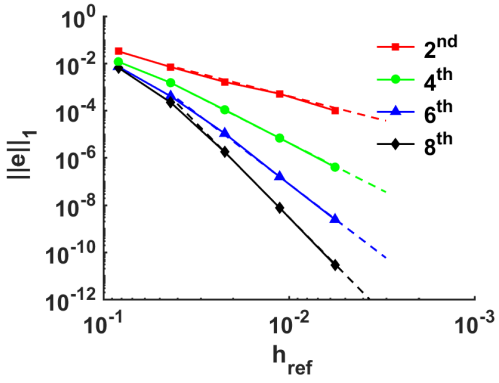


Figure 4: The $\|e\|_1$ for all schemes, on a regular triangular grid, where the dashed lines have the exact theoretical order.



(a) $\|e\|_1$ for the irregular quadrilaterals grid.



(b) $\|e\|_1$ for the irregular polyhedral grid.

Figure 5: The $\|e\|_1$ for all schemes, where the dashed lines have the exact teoretical order.

Table 2: The $\|e\|_1$ ratios estimated for an $h_{ref} = 0.01$, using as reference the solution of the regular polyhedral grid.

Grid type	FLS2	FLS4	FLS6	FLS8
Cartesian	1,12	0,80	1,69	2,14
Triangular	1,04	0,82	2,56	6,76
Polyhedral	1,00	1,00	1,00	1,00

Table 3: The $\|e\|_1$ ratios estimated for an $h_{ref} = 0.01$, using as reference the solution of the irregular polyhedral grid.

Grid type	FLS2	FLS4	FLS6	FLS8
Quadrilateral	0,92	0,82	1,39	1,75
Triangular	1,11	1,33	3,35	6,28
Polyhedral	1,00	1,00	1,00	1,00
Hybrid	1,70	2,51	7,35	13,03

8.2. Number of non zeros

The number of non-zero elements (NNZ) of the global matrix is directly related to the memory used to solve the problem. So, for the same accuracy, if less memory is occupied, the more efficient the scheme is. This parameter is generally affected by the number of faces from a cell, by the stencil size and the number of cells inside the domain.

To better compare the results a fitting was made to the data, as previously. The obtained fitting functions allow to estimate the number of non zeros that each scheme and grid would required for a given norm-1 error. Table 4 presents the expected occupied memory for a norm-1 error of 1E-05 using regular grids, while table 5 shows that same parameter for the irregular grid types.

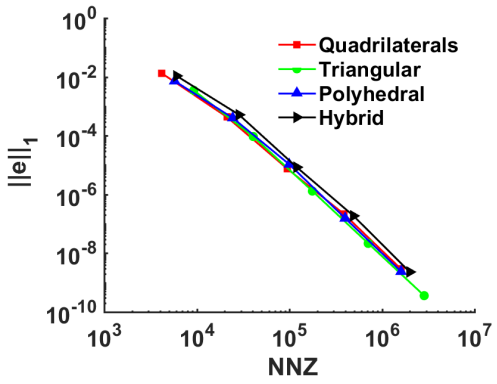


Figure 6: The $\|e\|_1$ and NNZ for sixth order scheme, on irregular grid types.

Considering the regular group, the most memory efficient grid for the second order scheme is the polyhedral one, while for fourth and sixth orders the triangular is the best option. The eighth order scheme has its best efficiency for the Cartesian

Table 4: Estimated memory in MB used for an norm-1 error of 1E-05, on regular grid types.

Grid type	FLS2	FLS4	FLS6	FLS8
Cartesian	31,36	1,11	0,66	0,69
Triangular	31,49	0,93	0,61	0,90
Polyhedral	23,65	2,45	0,66	0,72

Table 5: Estimated memory in MB used for an norm-1 error of 1E-05, on irregular grid types.

Grid type	FLS2	FLS4	FLS6	FLS8
Quadrilateral	34,16	1,90	1,07	1,02
Triangular	40,88	1,51	1,07	1,14
Polyhedral	28,63	2,99	1,22	1,11
Hybrid	52,49	2,15	1,41	1,35

grid. From those same tables it is also possible to conclude that if requiring a mean error of 1E-05 or lower a sixth order scheme is better than an eighth order one, since the memory requirements are lower. Regarding the best grid for the irregular category, for the second order scheme, the polyhedral one is the best, while for fourth order, the irregular triangular grid has the best efficiency. For the sixth order scheme the irregular quadrilaterals and triangles require a similar amount of memory, whereas for the eighth order, the quadrilaterals have a better performance. Figure 6 presents the results obtained for a sixth order scheme, using the irregular grids type.

8.3. Solver Run Time

The solver run time (SRT) is the time (in seconds) required for the solver to reach a converged solution and is considered another parameter associated with scheme efficiency. Some high-order comparison studies with the SRT have already been made by Gooch *et al.* [26] and Costa *et al.* [13], never exceeding a sixth order scheme. By applying the previous fitting procedure, it is possible to obtain the optimal coefficients that adjust the function to the data points. Once those coefficients were computed, it was possible to determine the convergence order and compare the different grids SRT's, for the same accuracy level.

In figure 7, a plot with all schemes for the regular triangular grid is shown. It is clear that the highest order is not always the most advantageous, since the eighth order scheme only surpasses all other orders for an accuracy level of approximately 5.0E-07, which is a very low value.

Performing the same SRT comparison for the irregular grids, in figure 8, the plot of the irregular quadrilaterals grid is shown. As in the regular type, it is possible to check the accuracy level for which each scheme outperforms its lower scheme.

Table 6 presents the estimated SRT's for an accuracy of 1E-05, on regular grids. Comparing the various grid types for each scheme, the triangular grid is the fastest for second and fourth order schemes, while for the sixth order scheme the best grid is the Cartesian one. For the eighth order, the polyhedral grid is the most efficient, despite in the table presenting the same SRT as the Cartesian one. Table 7, depicts the predicted SRT's for the same norm-1 error, on the irregular grids type. In this case, the irregular quadrilaterals are the best for all schemes.

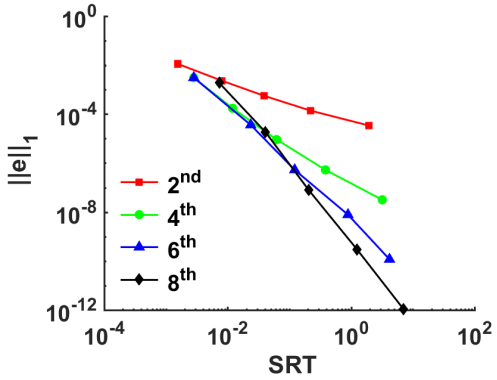


Figure 7: The $\|e\|_1$ as function of SRT for all schemes, on regular triangular grids.

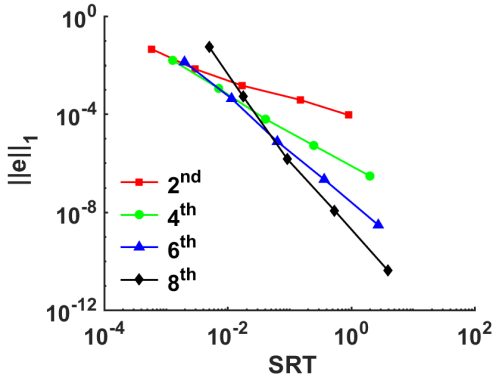


Figure 8: The $\|e\|_1$ as function of SRT for all schemes, on irregular quadrilaterals grids.

Table 6: Estimated SRT for an norm-1 error of 1E-05, on regular grid types.

Grid	FLS2	FLS4	FLS6	FLS8
Cart.	9,3E+0	7,9E-2	3,4E-2	3,4E-2
Triang.	8,3E+0	6,6E-2	3,9E-2	4,8E-2
Poly.	8,4E+0	1,8E-1	3,5E-2	3,4E-2

8.4. Face Vs Vertex Neighbors Stencils

The advantages of using face neighbors instead of vertex ones (for stencil generation) can be seen in figure 9, in relation to memory usage, and in figure

Table 7: Estimated SRT for an norm-1 error of 1E-05, on irregular grid types.

Grid	FLS2	FLS4	FLS6	FLS8
Quadri.	1,9E+1	1,7E-1	6,2E-2	5,8E-2
Triang.	2,8E+1	1,7E-1	7,2E-2	6,3E-2
Poly.	2,9E+1	3,1E-1	7,7E-2	6,9E-2
Hybrid	3,6E+1	2,7E-1	1,3E-1	8.0E-2

10, regarding the SRT. In these figures, the dashed lines represent the face neighbor stencils, while the filled lines show the vertex neighbor ones. Although different efficiency parameters and grid types are shown, in both figures the face neighbors stencils present lower NNZ and SRT values for a given accuracy level, proving that this stencil type is more efficient

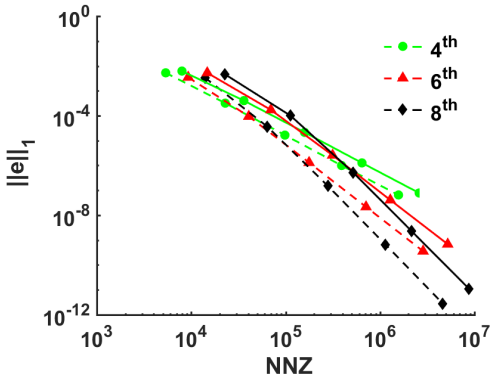


Figure 9: Solution accuracy ($\|e\|_1$) versus number of non-zeros of the global matrix, on irregular triangular grids. The dashed lines represent face neighbors stencils, while the filled lines the vertex neighbor ones.

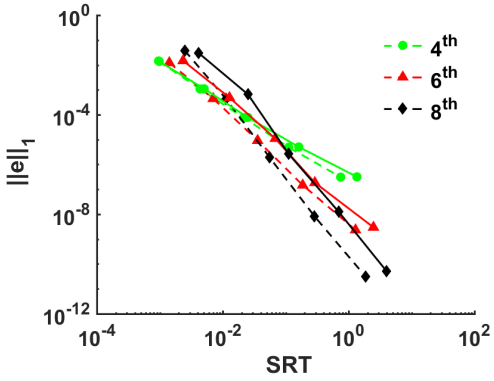


Figure 10: Solution accuracy ($\|e\|_1$) versus the SRT, on Cartesian grids. The dashed lines represent face neighbors stencils, while the filled lines the vertex neighbor ones.

To better understand the advantages, the obtained results were used in a fitting to find the optimal coefficients, as in the previous sections. Once these coefficients were obtained, it was estimated the number of non-zeros and SRT that each stencil type would require for a norm-1 error of 1E-05. The ratios shown in table 8 are computed by dividing the NNZ of face neighbors by the NNZ of the vertex neighbors. In table 9, the same procedure is applied to obtain the SRT ratios. As it can be seen, all the ratios are smaller than one, showing that the face neighbors stencils are more memory and time efficient. The Cartesian grids present the highest ratios, meaning that the gains are smaller, however in terms of memory, they range from 14% to 33%. Regarding time savings for this grid type, they range from 26% up to 53%. The best case are the irregular triangles with a memory gain of 46% for fourth order and 60% on eighth order. This grid type also presents the best results regarding the SRT improvements, with gains up to 79% on the eighth order scheme. This means that, at best, the time required for a simulation is almost five times less than what it would be needed if using a vertex neighbor stencil. These results restate the important choice of using face neighbors for high-order stencil generation with unstructured grids.

Table 8: Estimated NNZ ratios, for an norm-1 error of 1E-05. The values are obtained by dividing the NNZ of face neighbors stencils by the NNZ of vertex ones.

Grid type	FLS4	FLS6	FLS8
Cartesian	0,86	0,72	0,67
Irregular triangles	0,54	0,45	0,40
Hybrid	0,64	0,54	0,45

Table 9: Estimated SRT ratios, for an norm-1 error of 1E-05. The values are obtained by dividing the SRT of face neighbors stencils by the SRT of vertex ones.

Grid type	FLS4	FLS6	FLS8
Cartesian	0,74	0,55	0,47
Irregular triangles	0,40	0,27	0,21
Hybrid	0,58	0,51	0,30

8.5. Final Remarks

The study made on grid efficiency is solution dependent, so it should not be considered as a complete standard for any other analytical solution. Instead it should be viewed as reference or guide-line, but repeating this study for a more close to desired solution or domain, may be relevant. From what was shown, using only as comparison the norm-1 error as function of the reference length is not enough, since it is does not take into account the stencil configuration generated by each grid type. These different stencil configurations, although not directly

visible to the user, can affect the memory usage and time consumption of the solver. These small gains in memory and solver time can be perceived as having little relevance for steady simulations, but they can translate into significant gains when solving time dependent problems. Finally, it should not be concluded that the highest order is always the best one. It all depends on the accuracy level. As seen before, for accuracy levels around 1E-05 higher orders (such as eighth) can turn to be disadvantageous, since the memory and time used are greater when compared with lower orders schemes. On the other hand, when requiring high-accuracy levels, high order schemes can save a large amount of computing resources. Taking as example a comparison study made with an irregular polyhedral grid and requiring an mean error of 1E-09. The memory decreases from around 250Gb to 11Mb, when comparing a second order scheme with an eighth order one. The solver time, despite changing from computer to computer, also decreases from around 3720 hours to about one second.

9. Conclusions

A new high-order convective scheme was developed and combined with a high-order diffusive one, previously presented by Vasconcelos [25]. This allowed to test convective-diffusive equations with different coefficients and therefore, testing several Peclet numbers. The proposed change in the stencil generation algorithm, that is now based on a face neighbors expansion rather than vertex ones, has led to significantly memory and time savings. Nonetheless, the introduction of this new and more compact stencil type meant that the shape factor chosen by Vasconcelos [25] was no longer the ideal one. A new optimization of this parameter was made, resulting in a unique shape factor for all scheme orders. Finally, in the last sections, it was shown that all schemes follow their expected order for all grid types at test, proving the flexibility of the new stencil generation algorithm. It was also analyzed which grid type is better, for both the case of regular and irregular grids, obtaining results dependent on the scheme order. The Author's work, also demonstrated that using high-order schemes is more efficient in terms of memory and solver time, proving that this technique has the potential to reduced the computational cost of highly accurate steady simulations or time dependent ones.

The main goal of CFD, particularly on aerospace, is to study the air flow around an object. So, incorporating these high-order schemes into Euler and Navier-Stokes solvers, would be the next step in order to understand the potential of the proposed method in fluid simulations.

References

[1] Andrew Mosedale and Dimitris Drikakis. Assessment of Very High Order of Accuracy in Implicit LES models. *Journal of Fluids Engineering*, 129(12):1497–1503, 2007.

[2] D. W. Zingg, S. De Rango, M. Nemec, and T. H. Pulliam. Comparison of Several Spatial Discretizations for the Navier-Stokes Equations. *Journal of Computational Physics*, 160(2):683 – 704, 2000.

[3] Miguel R Visbal and Datta V Gaitonde. On the Use of Higher-Order Finite-Difference Schemes on Curvilinear and Deforming Meshes. *Journal of Computational Physics*, 181(1):155 – 185, 2002.

[4] Timothy Barth and Paul Frederickson. Higher order solution of the Euler equations on unstructured grids using quadratic reconstruction. In *28th Aerospace Sciences Meeting*, Aerospace Sciences Meetings. American Institute of Aeronautics and Astronautics, 1990.

[5] Carl Ollivier-Gooch and Michael Van Altena. A High-Order-Accurate Unstructured Mesh Finite-Volume Scheme for the Advection-Diffusion Equation. *Journal of Computational Physics*, 181:729 – 752, 2002.

[6] Chris Michalak and Carl Ollivier-Gooch. Unstructured High-Order Accurate Finite-Volume Solutions of the Navier-Stokes Equations. In *47th AIAA Aerospace Sciences Meeting including The New Horizons Forum and Aerospace Exposition*, Aerospace Sciences Meetings. American Institute of Aeronautics and Astronautics, Jan 2009.

[7] Carl Ollivier-Gooch and Alireza Jalali. Higher-Order Finite Volume Solution Reconstruction on Highly Anisotropic Meshes. In *21st AIAA Computational Fluid Dynamics Conference*, Fluid Dynamics and Co-located Conferences. American Institute of Aeronautics and Astronautics, Jun 2013.

[8] Jean-Camille Chassaing, Sofiane Khelladi, and Xesús Nogueira. Accuracy assessment of a high-order moving least squares finite volume method for compressible flows. *Computers & Fluids*, 71:41 – 53, 2013.

[9] Luis Ramírez, Xesús Nogueira, Sofiane Khelladi, Jean-Camille Chassaing, and Ignasi Colominas. A new high-order finite volume method based on Moving Least Squares for the resolution of the incompressible Navier-Stokes equations on unstructured grids. *Computer*

Methods in Applied Mechanics and Engineering, 278:883 – 901, 2014.

[10] Stéphane Louis Clain, Gaspar J. Machado, and Rui M. S. Pereira. A new very high-order finite volume method for the 2D convection diffusion problem on unstructured meshes. In *IV Conferência Nacional em Mecânica dos Fluidos, Termodinâmica e Energia*. Laboratório Nacional de Engenharia Civil, May 2012.

[11] Ricardo Costa, Stéphane Clain, and Gaspar José Machado. Finite Volume Scheme Based on Cell-Vertex Reconstructions for Anisotropic Diffusion Problems with Discontinuous Coefficients. In *Computational Science and Its Applications – ICCSA 2014: 14th International Conference, Guimarães, Portugal, June 30 – July 3, 2014, Proceedings, Part I*, pages 87 – 102. 2014.

[12] A. Boularas, Stéphane Louis Clain, and F. Baudoin. A sixth-order finite volume method for diffusion problem with curved boundaries. *Applied Mathematical Modelling*, 42:401 – 422, 2017.

[13] Ricardo Costa, Stphane Clain, Gaspar Machado, and Raphal Loubre. A very high-order accurate staggered finite volume scheme for the stationary incompressible Navier-Stokes and Euler equations on unstructured meshes. *Journal of Scientific Computing*, 71(3):13751411, 2017.

[14] Luis Cueto-Felgueroso, Ignasi Colominas, Xesús Nogueira, Fermín Navarrina, and Manuel Casteleiro. Finite volume solvers and Moving Least-Squares approximations for the compressible Navier-Stokes equations on unstructured grids. *Computer Methods in Applied Mechanics and Engineering*, 196:4712 – 4736, 2007.

[15] Luis Cueto-Felgueroso and Ignasi Colominas. High-order Finite Volume Methods and Multiresolution Reproducing Kernels. *Archives of Computational Methods in Engineering*, 15:185 – 228, 2008.

[16] Xesús Nogueira, Luis Cueto-Felgueroso, Ignasi Colominas, and Gómez H. Implicit Large Eddy Simulation of non-wall-bounded turbulent flows based on the multiscale properties of a high-order finite volume method. *Computer Methods in Applied Mechanics and Engineering*, 199:615 – 624, 2010.

[17] Xesús Nogueira, Luis Cueto-Felgueroso, Ignasi Colominas, Fermín Navarrina, and Manuel Casteleiro. A new shock-capturing technique based on Moving Least Squares for higher-order numerical schemes on unstructured grids. *Computer Methods in Applied Mechanics and Engineering*, 199:2544 – 2558, 2010.

[18] X. Nogueira, L. Cueto-Felgueroso, I. Colominas, H. Gómez, F. Navarrina, and M. Casteleiro. On the accuracy of finite volume and discontinuous Galerkin discretizations for compressible flow on unstructured grids. *International Journal for Numerical Methods in Engineering*, 78(13):1553 – 1584, 2009.

[19] Artur G. R. Vasconcelos, Duarte M. S. Albuquerque, and José C. F. Pereira. A Very High-Order Finite Volume Method Based on Weighted Least Squares for Elliptic Operators on Polyhedral Unstructured Grids. *Computers & Fluids*, 181:383–402, 2019.

[20] Luiz Tobaldini Neto and Clinton P. Groth. A high-order finite-volume scheme for large-eddy simulation of turbulent premixed flames. In *52nd AIAA Aerospace Sciences Meeting - AIAA Science and Technology Forum and Exposition, SciTech*. American Institute of Aeronautics and Astronautics, Jan 2014.

[21] B.I. Soemarwoto J.C. Kok and H. van der Ven. Extra-Large Eddy Simulations Using a High-Order Finite-Volume Scheme. Technical report, Nationaal Lucht- en Ruimtevaartlaboratorium, National Aerospace Laboratory NLR, 2007.

[22] H. K. Versteeg and W. Malalasekera. *An Introduction to Computational Fluid Dynamics-The finite volume method*. Pearson Education, 2nd edition, 2007.

[23] J. E. Akin. *Finite Element Analysis with Error Estimators*. Elsevier, 1st edition, 2005.

[24] Shaozhong Deng. Quadrature formulas in two dimensions, 2010. Dept. of Mathematics and Statistics, UNC at Charlotte.

[25] Artur G. R. Vasconcelos. A Very High-Order Finite Volume Method Based on Weighted Least Squares for the Solution of Poisson Equation on Unstructured Grids. Master's thesis, Instituto Superior Técnico, Universidade de Lisboa, 2017.

[26] Amir Nejat and Carl Ollivier-Gooch. A high-order accurate unstructured finite volume Newton-Krylov algorithm for inviscid compressible flows. *Journal of Computational Physics*, 227(4):2582 – 2609, 2008.