20th Australasian Fluid Mechanics Conference
Perth, Australia
5-8 December 2016

Verification of the Least-Squares Procedure within an Unstructured-Grid Flow Solver

K. A. Damm[1], **R. J. Gollan**[1] **and P. A. Jacobs**[2]

[1]School of Mechanical and Mining Engineering, Centre for Hypersonics, University of Queensland, Brisbane, Queensland 4072, Australia

[2]School of Mechanical and Mining Engineering, Queensland Geothermal Energy Centre of Excellence, University of Queensland, Brisbane, Queensland, 4072, Australia

Abstract

The least-squares procedure for gradient fitting within an unstructured flow solver requires the selection of a cloud of local points, referred to as a stencil. In this paper, we present the choice of least-squares stencil for both the flow field reconstruction for convective fluxes and viscous derivative computations in our flow solver, `Eilmer`. The importance of block boundary stencil choice is discussed with regards to retaining second-order accuracy in multi-block simulations. We use the method of manufactured solutions to verify the implementation and choice of stencils by showing that the solver achieves the expected second-order accuracy for both the Euler and Navier-Stokes equations on single-block and multi-block grids.

Introduction

One planned application of our flow solver, `Eilmer` [3], is CFD-based optimised design of hypersonic vehicles. It is envisioned that the complex geometries expected as a result of the optimisation will require an unstructured grid solver. The code, written in the D programming language [1], is a finite-volume cell-centred solver for compressible flows. For finite-volume solvers, such as `Eilmer`, gradient reconstruction is required at two stages throughout the flow update procedure. Firstly, to achieve higher then first-order accuracy, the primitive flow variables stored at the cell-centres must be reconstructed to a cell interface via some interpolation function. This reconstruction procedure requires an estimation of the gradients of the primitive flow variables at each cell centre. Secondly, for viscous simulations, the spatial gradients (for example the spatial derivatives of velocities, $\partial\vec{u}/\partial\vec{x}$) are required at the cell interface mid-points for calculating the diffusive fluxes through cell interfaces [4].

Two popular methods for approximating the gradients are Gauss' divergence theorem and the least-squares procedure. In a recent critical review of gradient approximation methods, Syrakos *et al* [8] determined that the cell-centred weighted least-squares method was the most robust gradient fitting method for unstructured grids. In the current paper, we will present the weighted least-squares procedure implemented within the unstructured flow solver. The focus of this work is to present our lessons learnt on selecting stencils which retain second-order accuracy for multi-block simulations of the Euler and Navier-Stokes equations. The Method of Manufactured Solutions [7] is used to verify the accuracy of the new stencils. In recent years, we have used manufactured solutions extensively as a verification technique during the development of the code [2, 3, 9].

Numerical Method

We begin by presenting the least-squares procedure for a generic cloud of points before delving into the selection of our stencils used in the least-squares procedure for gradient estimation of flow quantities.

Least-squares Gradient Fitting

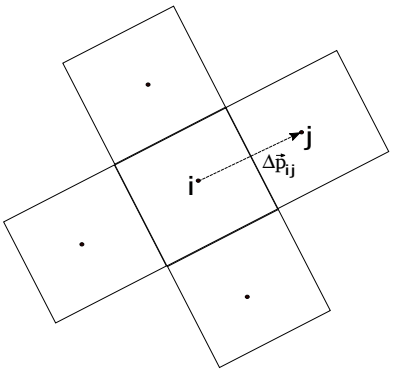

Figure 1: Least-squares point cloud stencil.

Consider the cloud of points, j, centred around the i^{th} cell in Figure 1. For a primitive flow variable q_i, the least-squares error term (S) can be defined as,

$$S = \sum_{j=1}^{N} w_j^2 (\vec{\nabla} q_i \cdot \Delta\vec{p}_{ij} - q_j)^2, \tag{1}$$

where w_j is the weight applied to the j^{th} cell. To minimise the error over the cloud, the error term is differentiated with respect to the unknown gradients and set to zero,

$$\frac{\partial S}{\partial \nabla q_i} = 0. \tag{2}$$

The linear system of equations which form (shown for two dimensions only) can then be solved to estimate the gradients,

$$\nabla q_i = [M]^{-1}\vec{b}, \tag{3}$$

where,

$$[M] = \begin{bmatrix} \sum_{j=1}^{N} w_j^2 \Delta x_{ij}\Delta x_{ij} & \sum_{j=1}^{N} w_j^2 \Delta x_{ij}\Delta y_{ij} \\ \sum_{j=1}^{N} w_j^2 \Delta x_{ij}\Delta y_{ij} & \sum_{j=1}^{N} w_j^2 \Delta y_{ij}\Delta y_{ij} \end{bmatrix}, \tag{4}$$

and,

$$\vec{b} = \begin{bmatrix} \sum_{j=1}^{N} w_j^2 \Delta q \Delta x_{ij} \\ \sum_{j=1}^{N} w_j^2 \Delta q \Delta y_{ij} \end{bmatrix}. \tag{5}$$

If the weights are dependent only on the point positions, the least-squares estimates for the gradients can be reduced to a summation of the Δq values, with coefficients determined from the equations above. This is the form coded within `Eilmer`.

Stencil for Flow Field Reconstruction

To achieve second-order accuracy, cell-centred finite-volume codes need to reconstruct the primitive flow variables up to an interface mid-point before computing the convective-flux update,

$$q_{i+1/2} = q_i + \phi \cdot \nabla q_i \cdot \frac{1}{2}\Delta\vec{p}_{ij}, \tag{6}$$

where ϕ is some limiting factor which we will assume is unity.

The selection of a stencil for the least-squares gradient fitting procedure is an open debate, with various methodologies found in the literature. We have focused on retaining second-order accuracy for both single block and multi-block topologies. As a result, we have chosen a compact, nearest neighbour stencil. This stencil is in fact the same generic stencil used in the least-squares description, illustrated in Figure 1. Our reconstruction procedure computes a gradient for all internal cells only. For multi-block simulations, this gradient is communicated to the ghost-cells at block connection boundaries such that consistent inviscid fluxes are computed at shared interfaces of these connections. This increases the complexity of the code as a result of the tightly coupled communication between neighbouring blocks. However, failing to do this results in small inconsistencies in flux estimates across boundary interfaces. The compact nature of the stencil is memory efficient and requires only one ring of ghost-cells. This also allows reuse of the same stencil for all cells in the domain.

Stencil for Spatial Derivatives in Viscous Fluxes

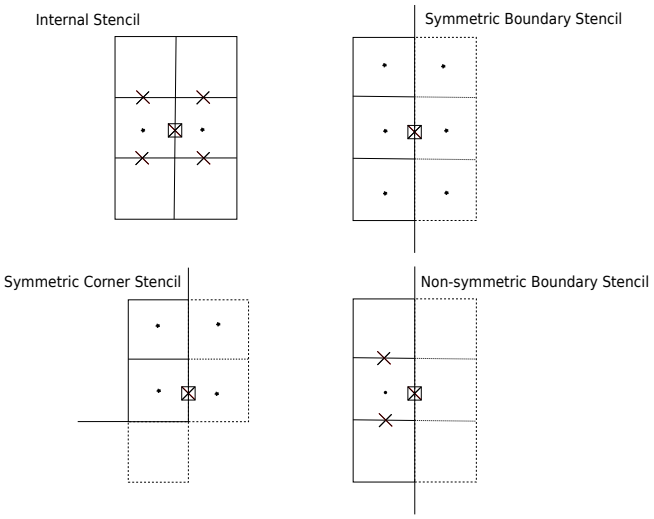

Figure 2: Stencils for point clouds used in the least-squares estimates of gradients. ☒ interface where derivatives are being estimated. × cell interface mid-point data. • cell-centre data.

The viscous flux update for a cell requires computation of the cell diffusive fluxes. The diffusive flux terms in turn require an estimation of the spatial derivatives at the cell interface mid-points. These gradients are approximated using the least-squares method outlined earlier, but with an inverse-distance weighting for the points within the cloud. The suite of stencils used in the viscous gradient calculation are presented in Figure 2. The stencil denoted as the internal stencil is applied to all interfaces which are not located on a boundary. The rationale behind this stencil is once again a compact, nearest neighbour approach. Not only does this reduce memory requirements, it retains the physically local nature of the viscous fluxes. In `Eilmer`, the internal stencil is not compatible with boundary interfaces because we do not construct nor store ghost-interfaces.

To minimise communication required between blocks, our first naive approach to handling boundary stencils was to not use any ghost cell data (stencil denoted as non-symmetric boundary stencil in Figure 2). It will be shown in a later section that this resulted in inconsistent gradients at boundary interfaces, and reduced the order of accuracy from the expected second-order to less than first-order.

To retain consistent gradients at shared interfaces on block connection boundaries and thus achieve second-order accuracy even for multi-block simulations, we developed the symmetric boundary and corner stencils presented in Figure 2. The important feature of these stencils are the symmetry about the boundary interface. By selecting such a stencil, the gradient along the boundary interfaces can be evaluated consistently between the two neighbouring blocks. It is important to note that the corner cloud must be handled differently in our implementation since diagonal block neighbours cannot access data from one another. This allows for the interface to have consistent spatial derivatives for multi-block simulations at corners.

In the next section, it is shown that the stencils presented retain second-order accuracy for both single- and multi-block simulations.

Verification using the Method of Manufactured Solutions

The Method of Manufactured Solutions is a code verification technique that can be used to assess the order of accuracy of a computational fluid dynamics code. First proposed by Roache and Steinberg [6], the method allows one to choose a purely manufactured analytical solution which is fed through the governing partial differential equations to obtain the source terms which would in turn generate the manufactured solution. The solver is then employed to simulate the derived source terms with exact Dirichlet boundary conditions applied to all boundaries by filling the ghost cells with values from the analytical solution. As grid refinement increases, the solution generated by the flow solver should approach the analytical solution in all cells. The manufactured solutions method is a more rigorous approach than simply using exact solutions because the manufactured solution can be constructed to exercise all terms in the governing equations.

The chosen analytical solution for this paper was first presented by Roy *et al* [7]. It is presented below, and the constants used are presented in Tables 1 and 2. In this paper, we test our implementation of the Euler equations and Navier-Stokes equations. For the Euler test, an inviscid supersonic flow is simulated on a uniform grid. For the Navier-Stokes test, a subsonic flow of a viscous gas is simulated on the same uniform grid. The gas is modelled as a calorically perfect gas with $\gamma = 1.4$, $R = 287.0\ J/(kg.K)$, $\mu = 10.0\ Pa.s$, and $Pr = 1.0$.

$$\rho(x,y) = \rho_0 + \rho_x \sin\left(\frac{a_{\rho x}\pi x}{L}\right) + \rho_y \cos\left(\frac{a_{\rho y}\pi y}{L}\right) + \rho_{xy}\cos\left(\frac{a_{\rho xy}\pi xy}{L^2}\right) \tag{7}$$

$$u(x,y) = u_0 + u_x \sin\left(\frac{a_{ux}\pi x}{L}\right) + u_y \cos\left(\frac{a_{uy}\pi y}{L}\right) + u_{xy}\cos\left(\frac{a_{uxy}\pi xy}{L^2}\right) \tag{8}$$

$$v(x,y) = v_0 + v_x \cos\left(\frac{a_{vx}\pi x}{L}\right) + v_y \sin\left(\frac{a_{vy}\pi y}{L}\right) + v_{xy}\cos\left(\frac{a_{vxy}\pi xy}{L^2}\right) \tag{9}$$

$$p(x,y) = p_0 + p_x \cos\left(\frac{a_{px}\pi x}{L}\right) + p_y \sin\left(\frac{a_{py}\pi y}{L}\right) + p_{xy}\sin\left(\frac{a_{pxy}\pi xy}{L^2}\right) \tag{10}$$

In theory, on an infinitely refined mesh the flow solver solution should approach the analytical solution. In practice, one calculates an observed order of convergence of the discretisa-

Equation, ϕ	ϕ_0	ϕ_x	ϕ_y	ϕ_{xy}	$a_{\phi x}$	$a_{\phi y}$	$a_{\phi xy}$
ρ (kg/m^3)	1	0.15	-0.1	0	1	0.5	0
u (m/s)	800	50	-30	0	1.5	0.6	0
v (m/s)	800	-75	40	0	0.5	$2/3$	0
p (N/m^2)	1×10^5	0.2×10^5	0.5×10^5	0	2	1	0

Table 1: Constants for supersonic Euler manufactured solution.

Equation, ϕ	ϕ_0	ϕ_x	ϕ_y	ϕ_{xy}	$a_{\phi x}$	$a_{\phi y}$	$a_{\phi xy}$
ρ (kg/m^3)	1	0.1	0.15	0.08	0.75	1.0	1.25
u (m/s)	70	4	-12	7	$5/3$	1.5	0.6
v (m/s)	90	-20	4	-11	1.5	1.0	0.9
p (N/m^2)	1×10^5	-0.3×10^5	0.2×10^5	-0.25×10^5	1.0	1.25	0.75

Table 2: Constants for subsonic Navier-Stokes manufactured solution.

tion error. This is then compared to the actual order of convergence of the numerical model. To determine the observed order of accuracy, error terms from successive grid refinements are compared. Obkerkampf and Roy's textbook [5] presents the following equation for estimating the order of accuracy,

$$p = \frac{ln\left(\frac{\varepsilon_{k+1}}{\varepsilon_k}\right)}{ln(r)}, \tag{11}$$

where ε_{k+1} is the error at the coarse level and ε_k is the error at the fine level, and,

$$r = \left(\frac{N_1}{N_2}\right)^{1/d}. \tag{12}$$

Here the refinement factor, r, is defined as the ratio of the number of cells in the fine mesh (N_1) and the coarse mesh (N_2) raised to the power of $1/d$, where d is the dimension (i.e. 2 for two dimensions). Any of the error norms may be used to determine the order of accuracy. Here we have chosen to use the L_∞ norm, which is defined as the maximum absolute error over the entire domain for a single cell, and the L_2 norm, defined as,

$$\varepsilon = \| u - u_{ref} \|_2 = \left(\frac{1}{N}\sum_{n=1}^{N} | u_n - u_{ref,n} |^2\right)^{1/2}. \tag{13}$$

The levels of grid refinement used in this study are presented in Table 3. We have used the following blocking arrangements: single block, 4×2 blocks and 4×4 blocks. Due to computational resource limitations on our workstations, only the 8-block and 16-block cases were simulated on the 5th refinement level.

Grid	Dimensions	Cell width, Δx (m)
1	8×8	0.125
2	16×16	0.0625
3	32×32	0.03125
4	64×64	0.015625
5	128×128	7.8125×10^{-3}

Table 3: Levels of grid refinement used for verification.

Method of Manufactured Solutions Results

The results from the Euler test are presented in Figures 3 and 4. As expected, the L_∞ norm is larger than the L_2 norm for a given refinement. Furthermore, both norms achieve the expected second-order of accuracy for both single block and multi-block grids. It is also noted that both the L_∞ and the L_2 norm are identical for the single-block and multi-block grids. This result shows that the stencil we have chosen achieves the desirable trait of "transparency" at block interfaces.

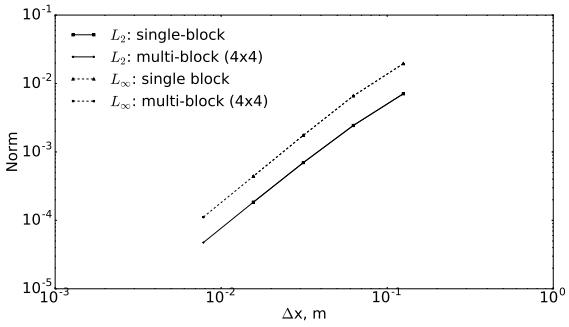

Figure 3: Norms for the inviscid flow solution.

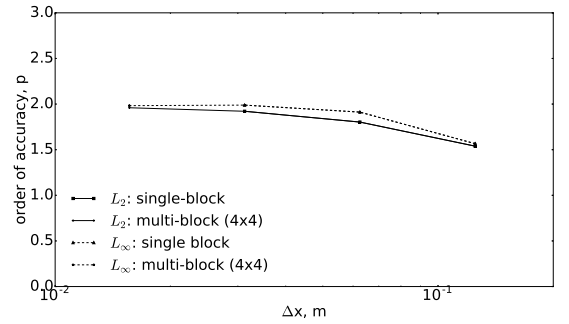

Figure 4: Order of accuracy for the inviscid flow solution

The results from the Navier-Stokes test using the symmetric stencil are presented in Figures 5 and 6. Once again as expected, the L_∞ norm is of a larger magnitude than the L_2 norm for a given refinement, however, both norms achieve the expected second-order of accuracy for both single block and multi-block grids. It should be noted that, although it appears as if the symmetric stencil achieves block transparency, agreement is only to the third decimal place for both norms. This result is to be expected since for a given refinement level, the internal stencil and symmetric boundary stencil have differing truncation errors. However, despite the difference in truncation error, both stencils should achieve the same converged order of accuracy. The results reflect this.

To contrast the behaviour of the new stencils with the old, the results from the Navier-Stokes test using the initial boundary stencil is presented in Figure 7. It is clear that the inconsistent flux at the shared interfaces along block connections has

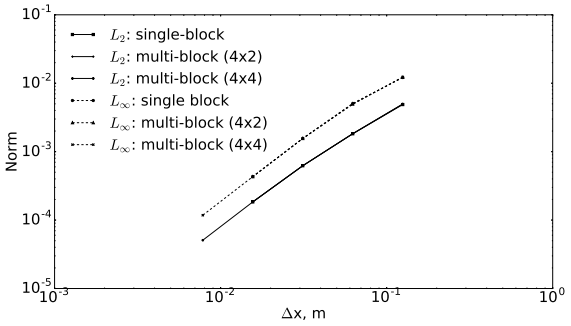

Figure 5: Norms for the viscous flow solution with symmetric boundary stencil.

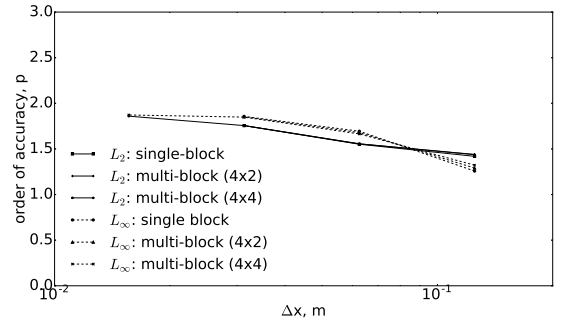

Figure 6: Order of accuracy for the viscous solution with symmetric boundary stencil.

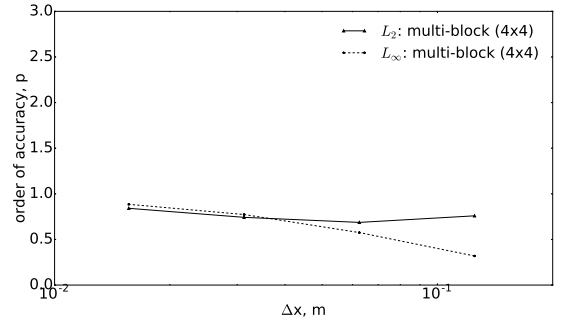

Figure 7: Order of accuracy for the viscous flow solution with non-symmetric boundary stencil.

reduced the order of accuracy from the desired value of 2. Indeed this result motivated further investigation and code development so that our stencil selection would solve this issue of inconsistency.

Conclusions

In this paper, we presented the weighted least-squares procedure used for gradient fitting in both the inviscid and viscous flux updates for our flow solver, `Eilmer`. Our choice of point cloud stencil was presented along with our rationale for enforcing consistency at block connection interfaces. We presented our results for observed order of accuracy based on the method of manufactured solutions. This verified that with our stencil selection, `Eilmer` achieves second-order accuracy even for multi-block simulations of the Euler and Navier-Stokes equations. The importance of consistent block connection interface flux handling was shown by providing evidence that using non-symmetric stencils reduced the order of accuracy to below 1st order.

Acknowledgements

The authors would like to acknowledge the support of the Cooperative Research Centre for Space Environment Management (SERC Limited) through the Australian Governments Cooperative Research Centre Programme.

References

[1] Bright, W., The D programming language, http://dlang.org.

[2] Gollan, R. and Jacobs, P., Verification of a compressible flow solver, in *18th Australasian Fluid Mechanics Conference (AFMC)*, Australasian Fluid Mechanics Society, 2012.

[3] Gollan, R. and Jacobs, P., About the formulation, verification and validation of the hypersonic flow solver Eilmer, *International Journal for Numerical Methods in Fluids*, **73**, 2013, 19–57.

[4] Mavriplis, D. J., Revisiting the least-squares procedure for gradient reconstruction on unstructured meshes, *AIAA Paper 2003-3986*.

[5] Oberkampf, W. L. and Roy, C. J., *Verification and validation in scientific computing*, Cambridge University Press, 2010.

[6] Roache, P. J. and Steinberg, S., Symbolic manipulation and computational fluid dynamics, *AIAA Journal*, **22**, 1984, 1390–1394.

[7] Roy, C. J., Nelson, C., Smith, T. and Ober, C., Verification of Euler/Navier–Stokes codes using the method of manufactured solutions, *International Journal for Numerical Methods in Fluids*, **44**, 2004, 599–620.

[8] Syrakos, A., Dimakopoulos, Y., Goulas, A. and Tsamopoulos, J., A critical analysis of some popular methods for the calculation of the gradient in finite volume methods, with suggestions for improvements, *arXiv preprint arXiv:1606.05556*.

[9] Veeraragavan, A., Beri, J. and Gollan, R., Use of the method of manufactured solutions for the verification of conjugate heat transfer solvers, *Journal of Computational Physics*, **307**, 2016, 308–320.