AIAA-2003-0253

Evaluation of Discontinuous Galerkin and Spectral Volume Methods for Conservation Laws on Unstructured Grids

Yuzhi Sun and Z.J. Wang
Michigan State University
East Lansing, MI

41st Aerospace Sciences Meeting and Exhibit
6-9 January, 2003
Reno, Nevada

EVALUATION OF DISCONTINUOUS GALERKIN AND SPECTRAL VOLUME METHODS FOR CONSERVATION LAWS ON UNSTRUCTURED GRIDS

Yuzhi Sun[‡] and Z.J. Wang[§]

Department of Mechanical Engineering
Michigan State University,
East Lansing, MI 48824

ABSTRACT

The discontinuous Galerkin (DG) and spectral volume (SV) methods are two classes of high-order methods for conservation laws capable of handling unstructured grids. In this paper, we evaluate their performance for scalar conservation laws in both one and two dimensions in terms of accuracy, and efficiency. We first review the basic features of the two methods, and compare their similarities and differences. Then we estimate the number of operations for each method in two dimensions. Finally, some numerical results in accuracy and CPU time are presented to support our estimates.

1. INTRODUCTION

It is well known that nature is governed by many conservation laws. The motion of fluids is no exception. The governing principles for fluids in motion are the conservation of mass, momentum and energy. Therefore, progresses made in computational methods for conservation laws can significantly impact the numerical simulation of numerous physical phenomena in nature, including fluid dynamics, combustion, acoustic waves, electromagnetics. "Real world" fluid mechanics and transport problems of interest have invariably very complex *physics and geometry*. Over the last one and half decades, unstructured grid methods [1-6] have demonstrated their potential in handling complex geometries with or without complex physics. There is a definite trend in moving from structured grids (body-fitted-coordinate grids) to unstructured grids for problems with complex geometries in computational fluid dynamics (CFD), because of the geometric flexibility offered by unstructured grids. However, most of these unstructured grid methods have only second-order accuracy. In many applications such as computational aeroacoustics (CAA), computational electromagnetics (CEM), large eddy simulation and direct numerical simulation of turbulence, higher-order accurate methods are required. There have been extensive research efforts into high-order methods for conservative laws on unstructured grids since the mid 1980s. Successful examples include the spectral element method [7] or multi-domain spectral method [8], k-exact finite volume (KEFV) method [2] and WENO method [9], the DG method [10-11], unstructured spectral method [12], fluctuation-splitting (FS) method [13], and recently the spectral volume method [14-15]. These methods have been used successfully in a variety of applications. Among these methods, the DG, KEFV, WENO and SV methods appear to be capable of handling unsteady discontinuities, while the DG and SV methods seem to be the most efficient. Therefore, our focus will be on the DG and SV methods in this paper.

The DG method is a finite element method using discontinuous solution and test spaces (usually piecewise polynomials of suitable degree), which means that the state variable usually is not continuous across element boundaries. The fluxes through the element boundaries are the computed using an approximate Riemann solver, mimicking the successful Godunov finite volume method [16]. Due to the use of Riemann fluxes across element boundaries, the DG method is fully conservative. However, high order surface and volume integrals are necessary in the DG method, which may be expensive to compute.

The SV method [14,15] is ultimately a finite volume method. Each triangular grid cell is partitioned into subcells named control volumes (CVs). The macro triangular cell is named a spectral volume (SV). Mean state-variables at the CVs inside a SV are employed to construct a high-order polynomial, which is then utilized to update the means at the CVs. The SV method is fully conservative at the control volumes.

In this paper, we perform a detailed evaluation of the DG and the SV methods for conservation laws on unstructured grids. In the next two sections, we review the major features of these two methods for scalar conservation laws in two dimensions. In Section 4, we estimate the number of operations for both methods. In Section 5, several numerical test cases in both 1D and 2D are presented. The numerical order of accuracy and CPU timing are shown to verify the estimates. Finally, several concluding remarks based on the current study are summarized in section 6.

[‡] Graduate Student (sunyuzhi@egr.msu.edu)
[§] Associate Professor (zjw@egr.msu.edu), AIAA Senior Member

2. DISCONTINUOUS GALERKIN METHOD

Consider the following two-dimensional conservation laws

$$u_t + \nabla \bullet F = 0, \quad \Omega \times (0,T) \tag{1}$$

equipped with proper initial and boundary conditions. In (1), $F = (f, g)$ is the flux vector. By multiplying by a test function v, integrating over the domain Ω, and performing an integration by parts we obtain the following weak statement of the problem

$$\int_{\Omega} v u_t dV + \oint_{\partial\Omega} v F(u) \bullet \boldsymbol{n} dS - \int_{\Omega} \nabla v \bullet F(u) dV = 0, \forall v \tag{2}$$

A discrete analogue of (2) can be obtained by subdividing Ω into N non-overlapping triangular elements $\{T_i\}$, and by considering functions u_h and v_h, defined within each element, given by the combination of n polynomial shape functions ϕ_j,

$$u_h(x,t) = \sum_{j=1}^{n} u_h^j(t)\phi_j(x), \; v_h = \sum_{j=1}^{n} v_h^j \phi_j . . \tag{3}$$

The expansion coefficients u_h^j denote the degrees of freedom (DOFs) of the numerical solution for element T_i. Note that there is no global continuity requirement for u_h, which is generally discontinuous across the element boundaries. By splitting the integral over Ω in (2) into the sum of integrals over the elements and by admitting only the functions u_h and v_h, we obtain the semi-discrete equation for T_i

$$\frac{d}{dt}\int_{T_i} v_h u_h dV + \oint_{\partial T_i} v_h F(u_h) \bullet \boldsymbol{n} dS - \int_{T_i} \nabla v_h \bullet F(u_h) dV = 0. \tag{4}$$

Equ. (4) must be satisfied for any function v_h. Since ϕ_j is the basis for v_h, (4) is equivalent to the following system of n equations

$$\frac{d}{dt}\int_{T_i} \phi_j u_h dV + \oint_{\partial T_i} \phi_j F(u_h) \bullet \boldsymbol{n} dS - \int_{T_i} \nabla \phi_j \bullet F(u_h) dV = 0. \quad 1 \le j \le n \tag{5}$$

Because the approximate solution is discontinuous along the element boundaries, the interface flux is not uniquely defined. It is at this stage the Riemann fluxes used in Godunov finite volume schemes are borrowed. The interface flux function $F(u_h) \bullet \boldsymbol{n}$ is replaced by a Riemann flux $\hat{F}(u_h^-, u_h^+, \boldsymbol{n})$. In order to guarantee consistency and conservation, the Riemann flux must satisfy

$$\hat{F}(u,u,\boldsymbol{n}) = F(u) \bullet \boldsymbol{n}, \; \hat{F}(u,v,\boldsymbol{n}) = -\hat{F}(v,u,-\boldsymbol{n}) . \tag{6}$$

The surface and volume integrals are computed with Gauss quadrature formulas of suitable order of accuracies.

By assembling together all the elemental contributions, the system of ordinary differential equations which govern the evolution in time of the discrete solution can be written as

$$\frac{dU}{dt} = R(U) . \tag{7}$$

where U is the global vector of the degrees of freedom, and *R(U)* is the residual vector.

Time Integration

An explicit multi-stage third-order TVD (total variation diminishing) Runge-Kutta scheme is employed for time integration [17]. The Runge-Kutta scheme can be expressed in the following form:

$$\begin{aligned} U^{(1)} &= U^n + \Delta t R(U^n) ; \\ U^{(2)} &= \frac{3}{4}U^n + \frac{1}{4}[U^{(1)} + \Delta t R(U^{(1)})]; \\ U^{n+1} &= \frac{1}{3}U^n + \frac{2}{3}[U^{(2)} + \Delta t R(U^{(2)})]. \end{aligned} \tag{8}$$

3. SPECTRAL VOLUME METHOD

In the SV method, the element T_i is named a *spectral volume*, which is further partitioned into subcells named *control volumes* (CVs), indicated by $C_{i,j}$, as shown in Figure 1. To represent the solution as a polynomial of degree m in two dimensions (2D) we need $M = (m+1)(m+2)/2$ pieces of independent information, or degrees of freedom (DOFs). The DOFs in a SV method are the volume-averaged mean variables at the M CVs. For example, to build a quadratic reconstruction in 2D, we need at least (2+1)(3+1)/2 = 6 DOFs. There are numerous ways of partitioning a SV, and not every partition is admissible in the sense that the partition may not be capable of producing a degree m polynomial. Once M mean solutions in the CVs of an admissible SV are given, a unique polynomial reconstruction can be obtained from

$$p_i(x,y) = \sum_{j=1}^{M} L_j(x,y)\overline{u}_{i,j}\,, \tag{9}$$

where $p_i(x,y) \in P_m$ (the space of polynomials of degree m or less), $L_j(x,y) \in P_m, j = 1,\cdots,N$ are the "shape" functions satisfying

$$\int_{C_{i,j}} L_k(x,y)dV = V_{i,j}\delta_{jk}\,. \tag{10}$$

where $V_{i,j}$ is the volume of $C_{i,j}$. This high-order polynomial reconstruction facilitates a high-order update for the mean solution of each CV. Integrating (1) in each *CV*, we obtain

$$\frac{d\overline{u}_{i,j}}{dt}V_{i,j} + \sum_{r=1}^{K}\int_{A_r}(F \bullet \boldsymbol{n})dA = 0, \tag{11}$$

where K is the total number of faces in $C_{i,j}$, and $\overline{u}_{i,j}$ is the volume-averaged solution at $C_{i,j}$. The flux integral in (11) is then replaced by a Gauss-quadrature formula which is exact for polynomials of degree m

$$\int_{A_r}(F \bullet \boldsymbol{n})dA \approx \sum_{q=1}^{ne} w_{rq}F(u(x_{rq,}y_{rq})) \bullet \boldsymbol{n}_r A_r, \tag{12}$$

where *ne* is the number of quadrature points on the *r-th* face, w_{rq} are the Gauss quadrature weights, (x_{rq}, y_{rq}) are the Gauss quadrature points. Since the reconstructed polynomials are piece-wise continuous, the solution is usually discontinuous across the boundaries of a SV, although it is continuous across interior CV faces. The fluxes at the interior faces can be computed directly based on the reconstructed solutions at the quadrature points. The fluxes at the boundary faces of a SV are computed using approximate Riemann solvers given the left and right reconstructed solutions.

It has been shown [14] that order of accuracy of this SV scheme is *(m+1)th* order. In addition, the scheme is compact in the sense that a high-order polynomial is reconstructed in each SV without using any data from neighboring SVs. This property can potentially translate into significantly reduced communication cost compared to a k-exact FV scheme (for example) when implemented on parallel computers.

4. NUMBER OF OPERATIONS FOR DG AND SV

In order to provide a precise estimate of the number of operations for both methods, we need to specify the form of the flux vector. We assume scalar conservation laws, i.e., $F = (au, bu)$, where a and b are given constants. In addition, the numerical flux is computed using the Lax-Friedrich's flux, i.e.,

$$\hat{F}(u^-,u^+,\boldsymbol{n}) = \begin{cases} u^-(an_x + bn_y) & if\,(an_x + bn_y) > 0 \\ u^+(an_x + bn_y) & otherwise \end{cases} \tag{13}$$

where $\boldsymbol{n} = (n_x, n_y)$. We use the total number of multiplications and if statements as the number of operations. Therefore, once the state variable is computed, the analytical flux takes 3 operations while the Riemann flux costs 4 operations to compute (one operation is the 'if' statement).

4.1 Number of Operations of the DG Method

We consider linear, quadratic and cubic elements, which yield second, third and fourth-order spatial accuracy respectively. The DOFs for these elements are given in Figure 2. The residual vector for element T_i takes the following form

$$R_i = W^{-1}\begin{bmatrix} \int_{T_i}(a\frac{\partial\phi_1}{\partial x} + b\frac{\partial\phi_1}{\partial y})u_h dV - \oint_{\partial T_i}\hat{F}(u^-,u^+,\boldsymbol{n})\phi_1 dS \\ \vdots \\ \int_{T_i}(a\frac{\partial\phi_n}{\partial x} + b\frac{\partial\phi_n}{\partial y})u_h dV - \oint_{\partial T_i}\hat{F}(u^-,u^+,\boldsymbol{n})\phi_n dS \end{bmatrix}$$

where W is the mass matrix. Let k be the order of accuracy, n be the number of degrees of freedom on each element, nv be the number of quadrature points for the volume integral, and ns be the number of quadrature points for the surface integral on each face. The total number of operations can be divided into three main parts, corresponding to the cost for surface integral (N_1), volume integral (N_2), and mass matrix multiplication (N_3). N_1 consists the number of operations needed to compute the state variable at the quadrature points *(n*ns*3)*, the operations to compute the Riemann fluxes *[(2+2*ns)*3/2]*, and the operations to multiply the Riemann fluxes with the test functions (*3*n*ns*). Note that the Riemann fluxes are shared by two neighboring elements. Therefore, we have

$$N_1 = 6 \cdot ns \cdot n + 3 \cdot ns + 3\,.$$

N_2 is composed of the number of operations to compute the state variable at the quadrature points (*n*nv*), plus the operations to calculate the fluxes at these quadrature points (*nv*4*), and the operations to multiply the fluxes with the gradients of the test functions (*2*nv*n*). So we get

$$N_2 = nv \cdot (3 \cdot n + 4)\,.$$

N_3 is simply the cost of a square matrix multiplying a vector, which is *n*n*. Therefore the total cost to compute the residual vector for a single element is

$$N = N_1 + N_2 + N_3$$
$$= 6 \cdot ns \cdot n + 3 \cdot ns + 3 + nv \cdot (3 \cdot n + 4) + n \cdot n$$

The operations for the DG schemes of second to fourth orders are listed in Table 1.

Table 1. Number of Operations for the DG Method

k (order)	n	nv	ns	N
2	3	3	2	93
3	6	6	3	288
4	10	12	4	763

4.2 Number of Operations of the SV Method

The degrees of freedom in the SV method are the mean state variables at the control volumes. They are updated directly with a finite volume algorithm. Therefore the residual vector takes the following form

$$R_i = \begin{bmatrix} -\oint_{\partial C_{i,1}} \hat{F}(u^-, u^+, \boldsymbol{n}) dS \\ \vdots \\ -\oint_{\partial C_{i,n}} \hat{F}(u^-, u^+, \boldsymbol{n}) dS \end{bmatrix}$$

There are two kinds of faces in a SV. The faces that lie on the SV boundaries are called Riemann faces, because the state variables are discontinuous across these faces. The other faces that lie inside a SV are named continuous faces because the state variables are continuous across these faces. Denote the total number of faces in a SV with *nf*, and number of Riemann faces *nr*. Then the number of continuous faces is (*nf-nr*). Let the number of quadrature points on each face (edge) be *ne*. Then the number of operations to compute the state variables at all the quadrature points is *nf*ne*n*. The number of operations to compute the Riemann fluxes is *ne*nr*3/2*. Again the Riemann fluxes are shared by two different SVs. The number of operations to compute the analytical fluxes is simply *ne*(nf-nr)*2*. Therefore the total number of operations to compute the residuals of a spectral volume is

$$N = nf \cdot ne \cdot n + ne \cdot nr \cdot 3/2 + ne \cdot (nf - nr) \cdot 2$$

The operations for the SV schemes of second to fourth orders are listed in Table 2.

Table 2. Number of Operations for the SV Method

k (order)	n	ne	nf	nr	N
2	3	1	9	6	42
3	6	2	15	9	231
4	10	2	27	12	636

Note that the SV method is about 20-100% faster than the DG method.

5. NUMERICAL RESULTS

All of the computations were performed on a Pentium IV 2.0 GHz PC running the Redhat Linux 7.2 operating system. The code was written in C++, optimized and compiled with the default gcc compiler.

5.1 One-Dimensional Cases

We first test the DG and SV methods in one dimension for the following linear wave equation

$$u_t + u_x = 0 \qquad -1 < x < 1$$
$$u(x,0) = \sin(\pi x)$$

with periodic boundary conditions. The computation was performed until t = 1. The errors are computed based on the cell-averaged state variable on the element or the SV. Table 3 and 4 present the errors and CPU times for the DG and SV schemes respectively. Figure 4 shows that the SV method requires much less CPU time to achieve a given level of accuracy. Note that both methods have the expected numerical order accuracy. The DG method gives slightly smaller magnitude than the SV method in several cases. However, SV method is significantly faster than the DG method.

Next the performance of both methods for the Burger's equation

$$u_t + (\frac{1}{2}u^2)_x = 0 \qquad -1 < x < 1$$
$$u(x,0) = 1 + \frac{1}{2}\sin(\pi x)$$

is investigated. The boundary condition is again periodic. The computational was carried out until t = 0.3, when the solution is still smooth. The errors for both methods are summarized in Tables 5 and 6. Again the expected order of accuracy is obtained for both methods. It is interesting to note that the 4th order SV schemes has lower errors than the 4th order DG scheme, while DG gives smaller errors in the other cases. Generally speaking, the DG and SV methods have comparable accuracy.

5.2 Two-Dimensional Cases

We first test the performance of both methods for the following linear equation

$$u_t + u_x + u_y = 0; \quad 0 < x < 2, \quad 0 < y < 2$$
$$u(x, y, 0) = \sin(\pi(x + y)) \text{ , periodic boundary condition}$$

The numerical simulation was carried out until t = 1 on two different grids, one regular and one irregular as shown in Figure 3. The errors are computed based on the cell-averaged state variable on the element or the SV. Tables 7 and 8 present the errors of both methods on the regular mesh, while Tables 9 and 10 displays the errors on the irregular mesh. Note that both methods achieved the designed numerical order of accuracy. The DG method gives smaller error magnitudes, while the SV method is faster.

Finally we test the performance of both methods for the non-linear Burger's equation

$$u_t + uu_x + uu_y = 0 \text{ , } 0 < x < 2, \quad 0 < y < 2$$
$$u(x, y, 0) = \frac{1}{4} + \frac{1}{2}\sin(\pi(x + y)) \text{ , periodic.}$$

Only the results on the irregular mesh are presented here. In the first test, the simulation was performed until t = 0.1, when the solution is still smooth. The errors are documented in Tables 11 and 12. To test the performance of the TVB limiters [15], the simulation was also performed until t = 0.45, when a shock wave appeared in the solution. The solution errors in the smooth region [-0.2, 0.4]x[-0.2,0.4] are computed and presented in Tables 13 and 14. Again, both methods achieved the designed numerical order of accuracy. The DG method gives smaller error magnitudes, while the SV method is faster.

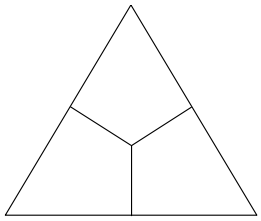
6. CONCLUDING REMARKS

The DG and SV methods are evaluated for scalar linear and nonlinear conservation laws in both one and two dimensions. Both methods can achieve the designed order of accuracy in both L_1 and L_∞ norms for the problems simulated. For 2D scalar conservation laws, the SV method is about 20% - 100% more efficient than the DG method. However, with the same number of degrees of freedom, the DG method has lower discretization errors. Taking into account of larger stability limit and increased resolution for discontinuities, the SV method compares favorably against the DG method in terms of the achievable accuracy versus CPU time.

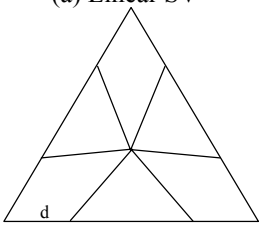
7. REFERENCES

1. D.J. Mavriplis and A. Jameson, Multigrid solution of the Navier-Stokes equations on triangular meshes, *AIAA J.* **28**, 1415-1425 (1990).
2. T.J. Barth and P.O. Frederickson, High-order solution of the Euler equations on unstructured grids using quadratic reconstruction," AIAA Paper No. 90-0013, 1990.
3. Y. Kallinderis, A. Khawaja and H. McMorris, Hybrid prismatic/tetrahedral grid generation for complex geometries, *AIAA Journal,* **34**, 291-298 (1996).
4. M. Delanaye and Y. Liu, Quadratic reconstruction finite volume schemes on 3D arbitrary unstructured polyhedral grids, AIAA Paper No. 99-3259-CP, 1999.
5. H. Luo, D. Sharov, J.D. Baum and R. Lohner, On the computation of compressible turbulent flows on unstructured grids, AIAA Paper No. 2000-0927, Jan. 2000.
6. Z.J. Wang, and R.F. Chen, "Anisotropic Solution-Adaptive Viscous Cartesian Grid Method for Turbulent Flow Simulation," *AIAA Journal*, Vol. 40, pp. 1969-1978, 2002.
7. A.T. Patera, A Spectral element method for fluid dynamics: laminar flow in a channel expansion, *J. Comput. Phys.* **54** 468-488 (1984).
8. D.A. Kopriva, Multidomain spectral solutions of the Euler gas-dynamics equations, *J. Comput. Phys.* **96**, 428 (1991).
9. C. Hu and C.-W. Shu, Weighted essentially non-oscillatory schemes on triangular meshes, *J. Comput. Phys.* **150**, 97-127 (1999).
10. B. Cockburn, S. - Hou and C. -W. Shu, TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws IV: the multidimensional case, *Mathematics of Computation 54,545-581 (1990).*
11. F. Bassi and S. Rebay, High-order accurate discontinuous finite element solution of the 2D Euler equations, *J. Comput. Phys*. **138**, 251-285 (1997).
12. J.S. Hesthaven and C.H. Teng, Stable spectral methods on tetrahedral elements, SIAM J. Sci. Comput. **21**, 2352-2380 (2000).
13. R. Abgrall and P.L. Roe, High order fluctuation splitting schemes on triangular mesh, submitted.
14. Z.J. Wang, Spectral (finite) volume method for conservation laws on unstructured grids: basic formulation, *J .Comput. Phys.178,210-251(2002)*.
15. Z.J. Wang and Yen Liu, Spectral volume method for conservation laws on unstructured grids II: extension to 2D scalar equation, to appear in *J.Compt. Phys.*

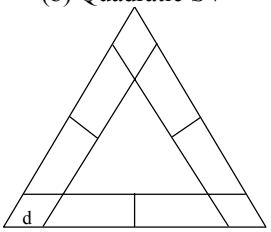
16. S.K. Godunov, A finite-difference method for the numerical computation of discontinuous solutions of the equations of fluid dynamics, *Mat. Sb*. **47**, 271 (1959).
17. C.-W. Shu, Total-Variation-Diminishing time discretizations, *SIAM Journal on Scientific and Statistical Computing* **9**, 1073-1084 (1988).



(a) Linear SV



(b) Quadratic SV



(c) Cubic SV

Figure 1. SVs of various degrees

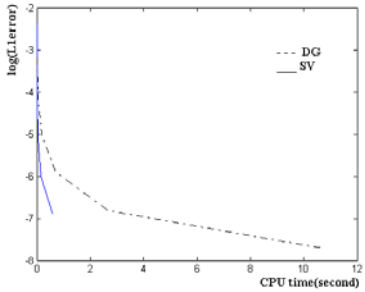


Figure 4a. Error versus CPU Time , 3rd order

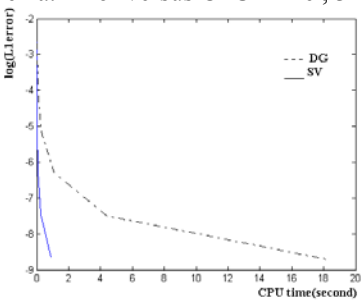
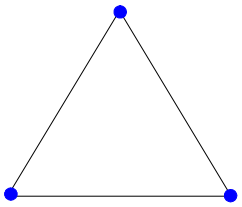
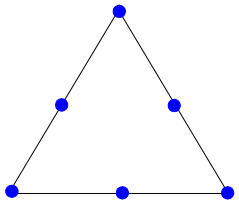


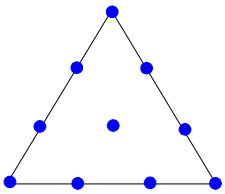
Figure 4b. Error versus CPU Time , 4th order



(a) Linear element

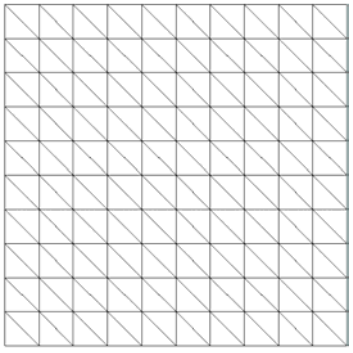


(b) Quadratic element

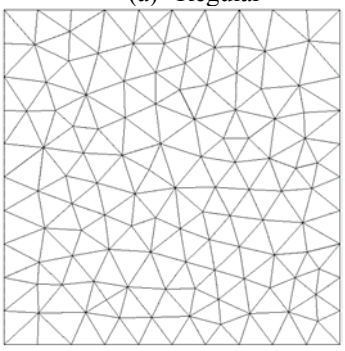


(c) Cubic element

Figure 2. The Degrees of Freedom in DG



(a) Regular



(b) Irregular

Figure 3. Regular and irregular meshes

Table 3. Errors and CPU time on the 1D linear wave equation at t = 1 using the DG method

Order of accuracy	NDOF	L_1 error	L_1 order	L_∞ error	L_∞ order	CPU (seconds)
2	20	1.45e-02	---	2.19e-02	---	2.97e-03
	40	3.21e-03	2.18	5.01e-03	2.13	1.13e-02
	80	7.39e-04	2.12	1.16e-03	2.11	4.75e-02
	160	1.76e-04	2.07	2.77e-04	2.07	1.76e-01
	320	7.96e-05	2.03	1.26e-04	2.03	6.91e-01
	640	1.07e-05	2.01	1.67e-05	2.02	2.77e+00
3	30	6.40e-04	---	9.59e-04	---	1.08e-02
	60	8.00e-05	3.00	1.23e-04	2.96	4.21e-02
	120	9.92e-06	3.01	1.55e-05	2.99	1.69e-01
	240	1.24e-06	3.00	1.94e-06	3.00	6.97e-01
	480	1.55e-07	3.00	2.43e-07	3.00	2.68e+00
	960	1.93e-08	3.01	3.04e-08	3.00	1.07e+01
4	20	2.09e-03	---	3.15e-03	---	1.82e-02
	40	1.30e-04	4.01	2.01e-04	3.97	7.16e-02
	80	8.10e-06	4.00	1.27e-05	3.98	2.76e-01
	160	5.06e-07	4.00	7.95e-07	4.00	1.09e+00
	320	3.16e-08	4.00	4.97e-08	4.00	4.38e+00
	640	1.98e-09	4.00	3.12e-09	3.99	1.82e+01

Table 4. Errors and CPU time on the 1D linear wave equation at t = 1 using the SV method

Order of accuracy	NDOF	L_1 error	L_1 order	L_∞ error	L_∞ order	CPU
2	20	1.03e-01	---	1.306e-01	---	6.53e-04
	40	2.74e-02	1.91	3.69e-02	1.82	1.66e-03
	80	7.02e-03	1.96	9.72e-03	1.92	4.71e-03
	160	1.77e-03	1.99	2.49e-03	1.96	1.48e-02
	320	4.46e-04	1.99	6.28e-04	1.99	5.48e-02
	640	1.12e-04	1.99	1.58e-04	1.99	1.95e-01
3	30	3.96e-03	---	5.36e-03	---	1.53e-03
	60	5.05e-04	2.97	6.49e-04	3.05	4.22e-03
	120	6.34e-05	2.99	7.86e-05	3.05	1.25e-02
	240	7.95e-06	3.00	9.61e-06	3.03	4.29e-02
	480	9.95e-07	3.00	1.19e-06	3.01	1.59e-01
	960	1.24e-07	3.00	1.48e-07	3.01	5.99e-01
4	20	2.13e-03	---	2.95e-03	---	2.78e-03
	40	1.37e-04	3.96	1.83e-04	4.01	7.06e-03
	80	8.85e-06	3.95	1.11e-05	4.04	2.01e-02
	160	5.58e-07	3.99	6.78e-07	4.03	7.11e-02
	320	3.51e-08	3.99	4.18e-08	4.02	2.37e-01
	640	2.20e-09	4.00	2.59e-09	4.01	8.77e-01

Table 5. Errors and CPU time on the 1D Burger's equation at t = 0.3 using the DG method

Order of accuracy	NDOF	L_1 error	L_1 order	L_∞ error	L_∞ order	CPU
2	20	1.08e-02	---	2.10e-02	---	3.23e-03
	40	3.05e-03	1.82	8.36e-03	1.33	1.20e-02
	80	8.08e-04	1.92	2.34e-03	1.84	4.59e-02
	160	2.07e-04	1.96	6.12e-04	1.93	1.83e-01
	320	5.22e-05	1.99	1.57e-04	1.96	7.20e-01
	640	1.31e-05	1.99	3.96e-05	1.99	2.92e+00
3	30	1.75e-03	---	7.99e-03	---	1.17e-02
	60	2.01e-04	3.12	9.72e-04	3.04	4.53e-02
	120	2.44e-05	3.04	1.77e-04	2.46	1.82e-01
	240	3.09e-06	2.98	2.43e-05	2.86	7.08e-01
	480	3.90e-07	2.99	3.12e-06	2.96	2.85e+00
	960	4.86e-08	3.00	3.91e-07	3.00	1.14e+01
4	20	3.17e-03	---	7.22e-03	---	1.00e-02
	40	9.79e-04	1.70	3.38e-03	1.09	3.80e-02
	80	5.90e-05	4.05	5.15e-04	2.71	1.49e-01
	160	4.74e-06	3.64	4.76e-05	3.44	5.89e-01
	320	3.36e-07	3.82	3.39e-06	3.81	2.40e+00
	640	2.17e-08	3.95	2.24e-07	3.92	9.44e+00

Table 6. Errors and CPU time on the 1D Burger's equation at t = 0.3 using the SV method

Order of accuracy	NDOF	L_1 error	L_1 order	L_∞ error	L_∞ order	CPU
2	20	1.16e-02	---	2.74e-02	---	7.98e-04
	40	3.07e-03	1.92	9.97e-03	1.46	2.07e-03
	80	7.82e-04	1.97	3.09e-03	1.69	5.84e-03
	160	1.95e-04	2.00	8.11e-04	1.93	1.86e-02
	320	4.89e-05	2.00	2.08e-04	1.96	6.73e-02
	640	1.22e-05	2.00	5.24e-05	1.99	2.50e-01
3	30	1.44e-03	---	1.17e-02	---	1.78e-03
	60	1.92e-04	2.91	2.12e-03	2.46	5.18e-03
	120	2.71e-05	2.82	3.90e-04	2.44	1.60e-02
	240	3.67e-06	2.88	5.76e-05	2.76	5.70e-02
	480	4.81e-07	2.93	7.72e-06	2.90	4.10e-01
	960	6.17e-08	2.96	9.89e-07	2.96	1.55e+00
4	20	2.90e-03	---	1.64e-02	---	1.67e-03
	40	1.09e-04	4.73	7.80e-04	4.39	4.28e-03
	80	1.25e-05	3.12	2.59e-04	1.59	2.52e-02
	160	7.17e-07	4.12	1.67e-05	3.96	8.35e-02
	320	4.36e-08	4.04	1.10e-06	3.92	3.06e-01
	640	2.71e-09	4.01	6.95e-08	3.98	2.27e+00

Table 7. Errors and CPU time on the 2D linear equation at t = 1 using the DG method (regular mesh)

Order of accuracy	Grid	L_1 error	L_1 order	L_∞ error	L_∞ order	CPU
2	10x10x2	1.14e-02	---	2.43e-02	---	3.33e-01
	20x20x2	2.31e-03	2.30	5.83e-03	2.06	2.76e+00
	40x40x2	5.09e-04	2.19	1.42e-03	2.04	2.23e+01
	80x80x2	1.18e-04	2.11	3.49e-04	2.02	1.81e+02
	160x160x2	2.84e-05	2.06	8.65e-05	2.01	1.45e+03
3	10x10x2	3.45e-04	---	7.65e-04	---	7.590e-01
	20x20x2	4.27e-05	3.02	9.66e-05	2.99	6.37e+00
	40x40x2	5.32e-06	3.00	1.21e-05	3.00	5.09e+01
	80x80x2	6.65e-07	3.00	1.51e-06	3.00	4.27e+02
	160x160x2	8.31e-08	3.00	1.89e-07	3.00	3.37e+03
4	10x10x2	1.39e-05	---	2.43e-05	---	1.66e+00
	20x20x2	8.59e-07	4.02	1.52e-06	4.00	1.33e+01
	40x40x2	5.34e-08	4.01	9.54e-08	4.00	1.08e+02
	80x80x2	3.33e-09	4.00	5.97e-09	4.00	8.47e+02
	160x160x2	2.08e-10	4.00	3.73e-10	4.00	7.28e+03

Table 8. SV Errors and CPU time on the 2D linear equation at t = 1 using the SV method (regular mesh)

Order of accuracy	Grid	L_1 error	L_1 order	L_∞ error	L_∞ order	CPU
2	10x10x2	4.02e-02	---	5.86e-02	---	1.21e-01
	20x20x2	1.06e-02	1.92	1.59e-02	1.88	9.47e-01
	40x40x2	2.71e-03	1.97	4.09e-03	1.96	8.81e+00
	80x80x2	6.83e-04	1.99	1.03e-03	1.99	8.39e+01
	160x160x2	1.71e-04	2.00	2.59e-04	1.99	6.05e+02
3	10x10x2	3.73e-03	---	5.21e-03	---	4.68e-01
	20x20x2	4.77e-04	2.97	7.12e-04	2.87	4.19e+00
	40x40x2	6.04e-05	2.98	9.05e-05	2.98	3.67e+01
	80x80x2	7.59e-06	2.99	1.14e-05	2.98	2.91e+02
	160x160x2	9.51e-07	3.00	1.43e-06	2.99	2.21e+03
4	10x10x2	9.04e-05	---	1.25e-04	---	1.33e+00
	20x20x2	5.66e-06	4.00	8.00e-06	3.97	1.25e+01
	40x40x2	3.56e-07	3.99	5.02e-07	4.00	8.46e+01
	80x80x2	2.23e-08	4.00	3.14e-08	4.00	6.77e+02
	160x160x2	1.39e-09	4.00	1.98e-09	3.99	6.36e+03

Table 9. Errors and CPU time on the 2D linear equation at t = 1 using the DG method (irregular mesh)

Order	Grid	L_1 error	L_1 order	L_∞ error	L_∞ order	CPU
2	10x10	2.17e-02	---	6.05e-02	---	3.77e-01
	20x20	4.67e-03	2.22	1.64e-02	1.88	3.26e+00
	40x40	1.07e-03	2.12	4.17e-03	1.97	2.65e+01
	80x80	2.56e-04	2.06	1.05e-03	1.99	2.13e+02
	160x160	6.26e-05	2.03	2.62e-04	2.00	1.75e+03
3	10x10	7.34e-04	---	2.56e-03	---	9.12e-01
	20x20	8.72e-05	3.07	4.42e-04	2.53	7.50e+00
	40x40	1.07e-05	3.03	6.34e-05	2.80	6.00e+01
	80x80	1.33e-06	3.01	8.19e-06	2.95	4.80e+02
	160x160	1.66e-07	3.00	1.03e-06	2.99	4.29e+03
4	10x10	4.10e-05	---	1.80e-04	---	3.87e+00
	20x20	2.41e-06	4.09	1.30e-05	3.79	3.12e+01
	40x40	1.47e-07	4.04	8.54e-07	3.92	2.48e+02
	80x80	9.08e-09	4.01	5.72e-08	3.90	2.10e+03
	160x160	5.65e-10	4.01	3.72e-09	3.94	1.70e+04

Table 10. Errors and CPU time on the 2D linear equation at t = 1 using the SV method (irregular mesh)

Order	Grid	L_1 error	L_1 order	L_∞ error	L_∞ order	CPU
2	10x10	6.71e-02	---	1.18e-01	---	1.38e-01
	20x20	1.83e-02	1.87	3.40e-02	1.80	1.30e+00
	40x40	4.71e-03	1.96	9.25e-03	1.88	1.17e+01
	80x80	1.19e-03	1.98	2.42e-03	1.94	8.61e+01
	160x160	3.00e-04	1.99	6.20e-04	1.96	8.38e+02
3	10x10	8.36e-03	---	1.68e-02	---	5.59e-01
	20x20	1.15e-03	2.87	2.95e-03	2.51	4.79e+00
	40x40	1.52e-04	2.92	5.28e-04	2.48	3.88e+01
	80x80	2.01e-05	2.91	1.31e-04	2.01	3.27e+02
	160x160	2.64e-06	2.93	2.85e-05	2.20	2.71e+03
4	10x10	2.06e-04	---	4.46e-04	---	3.08e+00
	20x20	1.33e-05	3.95	3.63e-05	3.62	2.44e+01
	40x40	8.58e-07	3.96	2.85e-06	3.67	1.96e+02
	80x80	5.40e-08	3.99	1.78e-07	4.00	1.65e+03
	160x160	3.42e-09	3.98	2.66e-08	2.74	1.30e+04

Table 11. Errors and CPU time on the 2D nonlinear equation at t = 0.1 using DG (irregular mesh)

Order	Grid	L_1 error	L_1 order	L_∞ error	L_∞ order	CPU
2	10x10	1.06e-02	---	3.48e-02	---	3.74e-02
	20x20	2.75e-03	1.95	1.14e-02	1.61	3.22e-01
	40x40	6.82e-04	2.01	3.21e-03	1.83	2.65e+00
	80x80	1.70e-04	2.00	8.24e-04	1.96	2.18e+01
	160x160	4.24e-05	2.00	2.08e-04	1.98	1.74e+02
3	10x10	6.80e-04	---	3.17e-03	---	1.80e-01
	20x20	1.14e-04	2.57	8.32e-04	1.93	1.54e+00
	40x40	1.79e-05	2.68	1.62e-04	2.36	1.29e+01
	80x80	2.73e-06	2.71	3.45e-05	2.23	9.76e+01
	160x160	4.08e-07	2.74	5.89e-06	2.55	7.76e+02
4	10x10	6.01e-05	---	4.58e-04	---	2.91e-01
	20x20	3.68e-06	4.03	3.76e-05	3.61	2.34e+00
	40x40	2.34e-07	3.98	2.47e-06	3.93	1.93e+01
	80x80	1.61e-08	3.86	2.07e-07	3.58	1.53e+02
	160x160	1.20e-09	3.75	1.95e-08	3.41	1.21e+03

Table 12. Errors and CPU time on the 2D nonlinear equation at t = 0.1 using SV (irregular mesh)

Order	Grid	L_1 error	L_1 order	L_∞ error	L_∞ order	CPU
2	10x10	5.79e-03	---	1.76e-02	---	1.34e-02
	20x20	1.46e-03	1.99	4.91e-03	1.84	1.29e-01
	40x40	3.67e-04	1.99	1.41e-03	1.80	1.17e+00
	80x80	9.41e-05	1.96	5.60e-04	1.34	8.62e+00
	160x160	2.39e-05	1.97	2.68e-04	1.06	6.92e+01
3	10x10	6.28e-04	---	2.15e-03	---	1.08e-01
	20x20	1.18e-04	2.42	6.20e-04	1.80	9.80e-01
	40x40	1.91e-05	2.62	1.38e-04	2.17	7.99e+00
	80x80	3.02e-06	2.66	3.28e-05	2.07	6.83e+01
	160x160	4.64e-07	2.70	6.09e-06	2.43	5.09e+02
4	10x10	6.73e-05	---	4.43e-04	---	2.28e-01
	20x20	5.19e-06	3.70	5.61e-05	2.98	1.89e+00
	40x40	3.93e-07	3.72	4.46e-06	3.65	1.50e+01
	80x80	2.95e-08	3.74	3.10e-07	3.85	1.21e+02
	160x160	2.38e-09	3.63	3.13e-08	3.31	1.13e+03

Table 13. Errors and CPU time on the 2D nonlinear equation at t = 0.45 using DG (irregular mesh)

Order	Grid	L_1 error	L_1 order	L_∞ error	L_∞ order	CPU
2 (TVBM=800)	10x10	1.78e-03	---	6.11e-03	---	4.22e-01
	20x20	1.75e-04	3.35	5.90e-04	3.37	3.37e+00
	40x40	3.29e-05	2.41	1.97e-04	1.58	2.73e+01
	80x80	6.84e-06	2.27	5.06e-05	1.96	2.26e+02
	160x160	1.53e-06	2.16	9.94e-06	2.35	1.82e+03
3 (TVBM=800)	10x10	9.49e-04	---	4.85e-03	---	9.20e-01
	20x20	7.68e-05	3.63	4.62e-04	3.39	7.41e+00
	40x40	7.73e-06	3.31	5.11e-05	3.18	6.04e+01
	80x80	8.26e-07	3.23	9.77e-06	2.39	5.06e+02
	160x160	9.58e-08	3.11	2.12e-06	2.20	4.11e+03
4 (TVBM=1200)	10x10	7.18e-04	---	4.87e-03	---	1.91e+00
	20x20	8.80e-06	6.35	1.14e-04	5.42	1.55e+01
	40x40	2.63e-07	5.06	2.22e-05	2.36	1.26e+02
	80x80	1.50e-08	4.13	2.60e-08	9.73	1.20e+03
	160x160	3.63e-09	2.05	5.93e-09	2.14	9.88e+03

Table 14. Errors and CPU time on the 2D nonlinear equation at t = 0.45 using SV (irregular mesh)

Order	Grid	L_1 error	L_1 order	L_∞ error	L_∞ order	CPU
2 (TVBM=800)	10x10	1.99e-03	---	3.80e-03	---	2.05e-01
	20x20	4.84e-04	2.04	1.53e-03	1.31	1.67e+00
	40x40	1.14e-04	2.09	3.92e-04	1.97	1.81e+01
	80x80	2.87e-05	1.99	1.29e-04	1.60	1.57e+02
	160x160	7.11e-06	2.01	3.31e-05	1.97	1.21e+03
3 (TVBM=800)	10x10	1.05e-03	---	6.00e-03	---	6.73e-01
	20x20	8.03e-05	3.71	6.20e-04	3.27	5.85e+00
	40x40	7.30e-06	3.46	3.89e-05	3.99	5.82e+01
	80x80	8.68e-07	3.07	7.30e-06	2.41	4.53e+02
	160x160	1.11e-07	2.97	1.46e-06	2.32	3.61e+03
4 (TVBM=1200)	10x10	4.39e-04	---	3.25e-03	---	1.61e+00
	20x20	4.63e-06	6.57	7.14e-05	5.51	1.43e+01
	40x40	9.87e-08	5.55	1.39e-06	5.69	1.23e+02
	80x80	1.55e-08	2.67	3.56e-08	5.28	1.02e+03
	160x160	3.65e-09	2.09	7.14e-09	2.32	8.70e+03