OPTIMAL NUMERICAL PARAMETERIZATION OF DISCONTINUOUS GALERKIN METHOD APPLIED TO WAVE PROPAGATION PROBLEMS

Nicolas Chevaugeon [a,*], Koen Hillewaert [c], Xavier Gallez [d], Paul Ploumhans [d], Jean-François Remacle [a,b]

a Université catholique de Louvain, Department of Civil Engineering, Place du Levant 1, 1348 Louvain-la-Neuve, Belgium
b Center for Systems Engineering and Applied Mechanics (CESAME), Université catholique de Louvain, 1348 Louvain-la-Neuve, Belgium
c CENAERO CFD and Multiphysics Group, Bâtiment Mermoz 1, Av. J. Mermoz 30, b: 6041 Gosselies, Belgium
d Free Field Technologies SA, Place de l'Université, 1348 Louvain-la-Neuve, Belgium
** Corresponding author : E-mail addresses: chevaugeon@gce.ucl.ac.be (N. Chevaugeon), remacle@gce.ucl.ac.be (J.-F. Remacle).*

Abstract

This paper deals with the high-order discontinuous Galerkin (DG) method for solving wave propagation problems. First, we develop a one-dimensional DG scheme and numerically compute dissipation and dispersion errors for various polynomial orders. An optimal combination of time stepping scheme together with the high-order DG spatial scheme is presented. It is shown that using a time stepping scheme with the same formal accuracy as the DG scheme is too expensive for the range of wave numbers that is relevant for practical applications. An efficient implementation of a high-order DG method in three dimensions is presented. Using 1D convergence results, we further show how to adequately choose elementary polynomial orders in order to equi-distribute a priori the discretization error. We also show a straightforward manner to allow variable polynomial orders in a DG scheme. We finally propose some numerical examples in the field of aero-acoustics.

Keywords

Discontinuous Galerkin; Aero-acoustics; Variable p; Runge–Kutta; Dispersion analysis

1. Introduction

The discontinuous Galerkin (DG) method is a compact finite element method that provides a practical framework for the development of high-order accurate methods for unstructured grids. The method is well suited for large-scale time-dependent computations in which high accuracy is required.

An application for which the DG method is considered to be highly efficient is aero-acoustics [6,13]. It has been shown that using DGM together with an explicit time domain solver allows to solve sound propagation problems that are out of reach for classical frequency domain methods. Using high-order discontinuous finite elements allows to reduce dramatically the number of elements per wavelength while the quadrature free implementation of the method drastically diminishes both computational cost and algorithmic complexity.

The combination of an explicit Runge–Kutta time stepping scheme together with the DG method for space discretization has been widely documented in the literature [7,8]. With such a strategy, the time-step must be selected to satisfy a Courant–Friedrichs–Lewy (CFL) stability condition [15]. In a method of lines (MOL) approach, the allowable time step of an explicit time stepping scheme depends on the size of the smallest element of the mesh, on the maximum signal speed and on the polynomial order of approximation.

In the case of aero-acoustics, a given accuracy is reached if the mesh contains enough elements per wavelength. Typically, 2 or 3 elements per wavelength are sufficient for polynomial orders of 4 or 5. Unfortunately, the geometry of the problem has also to be approximated with sufficient accuracy and smaller elements are usually required near boundaries. A large disparity in element sizes may then result in an excessive number of time-steps. Some investigations [5,9,11,17,18] have sought to overcome this inefficiency by using a local refinement strategy where spatially-dependent time steps are chosen as a function of the local Courant condition on an element. In the case of high-order schemes, in both space and time, local time stepping is difficult to design and to implement. We choose here another approach: changing polynomial order locally in order to equally distribute the discretization error. As a result of this *p*-adaptation, the maximum time step imposed by the CFL condition will be more uniform.

In a first section, we discuss convergence properties of DG methods and stability of RK–DGM schemes. We show that a fourth-order accurate Runge–Kutta is usually sufficiently accurate, even for spatial DG of higher orders, e.g. for polynomial order $p = 6$.

In the next section, we propose a variable polynomial order technique that ensures that the RK–DGM scheme provides a given accuracy with a time step that is up to 10 times bigger than the usual method of lines. We show examples that demonstrate the efficiency of the method and the fact that the accuracy is also maintained.

In what follows, we call the *m - p + 1 scheme* a RK–DGM scheme with polynomials of order p in the space discretization and an explicit Runge–Kutta of order m (involving m evaluations of the residual) for the time discretization.

2. Runge–Kutta-discontinuous Galerkin discretizations in one dimension

Let us consider the following model problem [10]: find u(x,t) 2 (0,1) · (0,T) solution of the 1D wave equation

$$\partial_t u + c\partial_x u = 0, \tag{1}$$

with initial condition

$$u(x,0) = u_0(x) \tag{2}$$

and periodic boundary conditions. This is a very simple model problem for a first-order hyperbolic PDE. In (1), *c* is a positive number. Waves travel from left to right, in the positive x direction, at the speed *c*.

We consider a partition of the space (0,1) into *N* segments of size $\Delta x = 1/N$ (Fig. 1). In each element, we approximate the unknown, *u*, using Legendre polynomials of orders less or equal to *p*. For that, we consider a reference segment $\xi \in (-1,1)$ and the unknown u^k in segment k going from $x^k = k/(N+1)$ to $x^{k+1} = (k+1)/(N+1)$ is approximated as

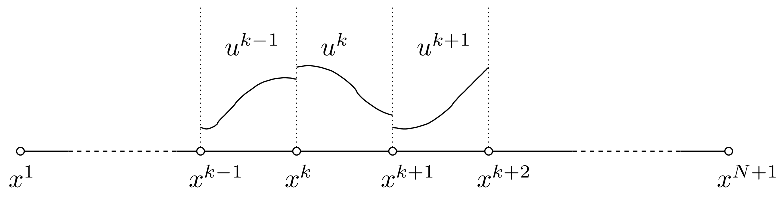

Fig. 1. One-dimensional discretization.

$$u^k(x(\xi),t) = \sum_{i=0}^{p} \mathcal{P}_i(\xi) u_i^k(t), \tag{3}$$

with

$$x(\xi) = x^k \frac{1-\xi}{2} + x^{k+1} \frac{1+\xi}{2} \tag{4}$$

A discontinuous Galerkin formulation of (1), that can be found in the early work of [14], consists in solving the following discrete variational formulation in every segment k

$$\begin{aligned} b(u^k, v) = \int_{x^k}^{x^{k+1}} \partial_t u^k(x,t) v(x)\, \mathrm{d}x \int_{x^k}^{x^{k+1}} c\, \partial_t u^k(x,t) v(x)\, \mathrm{d}x + \mu c\left[u^k(x^k,t) - u^{k-1}(x^k)\right] v(x^k) \\ +(1-\mu) c\left[u^{k+1}(x^{k+1},t) - u^k(x^{k+1},t)\right] v(x^{k+1}) = 0 \quad \forall v, \end{aligned} \tag{5}$$

where l is the upwind parameter. If $\mu = 0.5$, the scheme is centered and if $\mu = 1$, the scheme is full upwind. Taking into account the expansion (3) and choosing Legendre polynomials $\mathcal{P}_j(\xi)$ up to order $j \le p$ as test functions v, we have

$$\begin{aligned} b\left(\sum_{i=0}^{p} u_i^k \mathcal{P}_i, \mathcal{P}_j\right) = \sum_{i=0}^{p} \left(\partial_t u_i^k(t) \Delta x \int_{-1}^{1} \mathcal{P}_i \mathcal{P}_j \mathrm{d}x + c u_i^k(t) \int_{-1}^{1} \partial_\xi \mathcal{P}_i \mathcal{P}_j dx\right) \\ + \sum_{i=0}^{p} \mu c\left[\mathcal{P}_i(-1) u_i^k(t) - \mathcal{P}_i(1) u_i^{k-1}(t)\right] \mathcal{P}_j(-1) \\ + \sum_{i=0}^{p} (1-\mu) c\left[\mathcal{P}_i(-1) u_i^{k-1}(t) - \mathcal{P}_i(1) u_i^k(t)\right] \mathcal{P}_i(1) = 0, \qquad j = 0, \dots, p \end{aligned} \tag{6}$$

If $\mathbf{u}$ is a column vector of size $N \times (p+1)$ that contains all unknowns of every segment, the discontinuous Galerkin formulation can be written in a more compact form

$$\partial_t \mathbf{u} = \frac{c}{\Delta x} \mathbf{M}^{-1} \mathbf{L} \mathbf{u},$$

where **M** a diagonal matrix whose condition number grows like 2p + 1 and **L** a band matrix.

2.1. Fourier analysis

Fourier analysis considers wave-like solutions

$$u(x,t) = Re\left(Ce^{2i\pi(kx-ft+i\sigma t)}\right)$$

where *C* is a complex constant, *k* is the wave number, *f* the frequency and *σ* is the damping parameter. The term ''wave number'' refers to the number of complete wave cycles that exist *in one meter of linear space*. The wave number *k* is dimensional: it is the inverse of a distance. The frequency *f* is dimensional too: it is the inverse of a time. It is the number of complete wave cycles that are completed in one second. Substituting in (1), we have the dispersion relation

$$f = ck$$

and

$$\sigma = 0.$$

The speed of the waves is *c* and the solution is neither amplified nor damped.

If Δx is the mesh size and k is the dimensional wave number, we define a non-dimensional wave number as $k_h = k\Delta x$. The non-dimensional wave number k_h is interpreted as a wave number where the length measure is taken as the mesh size Δx. For example, a non-dimensional wave number of $k_h = 1/5$ correspond to a wave length $5\Delta x$ i.e. of 5 element sizes, i.e. $5\Delta x$.

We consider the semi-discrete form of the model problem (1):

$$\partial_t \mathbf{u} = \mathbf{A}\mathbf{u}, \tag{7}$$

with

$$\mathbf{A} = -\frac{c}{\Delta x}\mathbf{M}^{-1}\mathbf{L}.$$

We seek how accurately the DGM scheme, i.e. **A**, is able to approach the $c\partial_x$operator.

It is well known that, for finite differences, Fourier modes are the eigenfunctions of the discrete operator. Then, Fourier analysis makes perfect sense. In the case of the DG method, Fourier modes are not the eigenfunctions of the discrete operator. This means that, if a Fourier mode is projected into the discrete DG function space, it will excite more than one eigenvectors of **A**. Conversely, eigenvectors of **A** are not Fourier modes. Hence, it can not be shown that they there exists a one-to-one correspondence between Fourier modes and the eigenvectors of **A**. However, an approximate matching is possible.

It is easy to compute the spectrum of A for different values of p, N and l using MATLAB or ARPACK [1]. We compute

$$\mathbf{AV} = \mathbf{DV},$$

where **D** is a diagonal matrix, D_{ii} being the ith complex eigenvalue of **A** and where V_i, the *i*th column of **V** is the corresponding *i*th eigenvector of **A**.

Then, we compute the fast Fourier transform of each mode V_i and identify the corresponding wave number k_i. Fig. 2 shows how modes are extracted. Operator **A** correspond to a mesh of 5 elements and a polynomial order of $p = 4$. Two numerical modes are shown at the left side of the figure. They correspond to $k\Delta x = 3/5$ and $k\Delta x = 7/5$. The power spectrum of the **FFT** applied to the numerical mode is shown on the center of the figure. Each

Published in : Journal of Computational Physics (2007), vol. 223, issue 1, pp. 188-207
doi.org/10.1016/j.jcp.2006.09.005
Status : Postprint (Author's version)

eigenmodes can clearly be correlated with one Fourier mode but we see that, for the mode with the highest wave number, higher harmonics are present. The right side of the figure shows the difference $e_i(x)$ between the exact and the numerical mode.

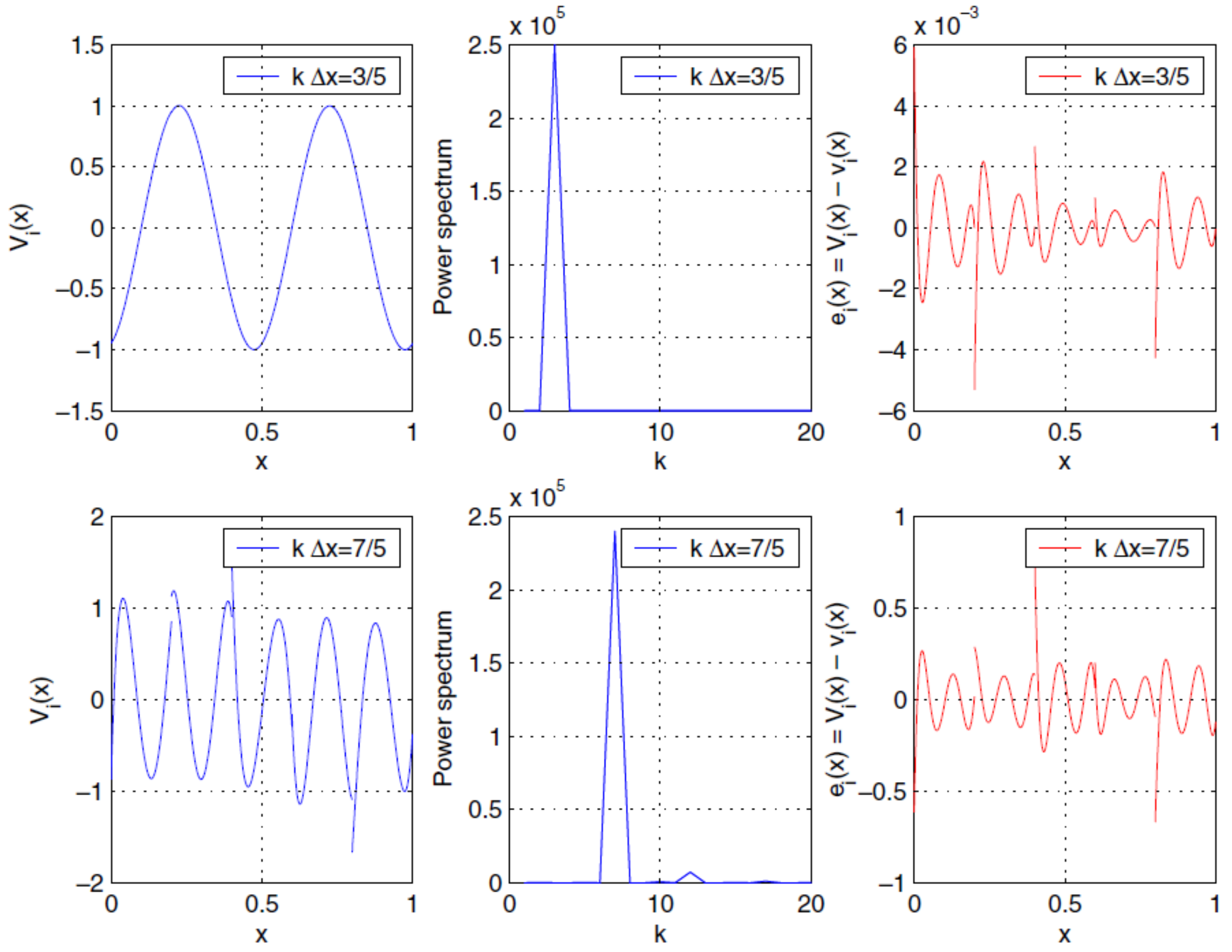

Fig. 2. Two numerical modes of the 1D DG operator with $N = 5$, $p = 4$ and $\mu = 1$. Left figures show the mode. Center figures show the result of a fast Fourier transform (power spectrum) applied to the mode. Right figures show the difference between numerical and exact modes.

Our approach for analyzing the dispersion properties of DG schemes is advantageous with respect to the one of Atkins [13] or Ainsworth [3]. The main advantage of this new method of analysis is that both spectral and spatial accuracy are analysed together. In particular, the Radau spatial structure of the error that has been demonstrated in [2] can be established in the same framework.

2.1.1. Spectral accuracy

The dispersion error usually defines how accurately the scheme is able to predict numerical wave speeds. In other words, the dispersion error of mode i is the difference between the exact and the approximate imaginary parts of wave numbers:

$$2\pi k_i - Im(D_{ii}).$$

The dissipation error usually defines how accurately the scheme is able to predict numerical damping. The dissipation error is the error in the real part of the eigenvalues, D_{ii}. In our case, the exact dissipation of the wave equation is null and all eigenvalues should all be on the imaginary axis. For spectral accuracy, our analysis reaches the same conclusions Atkins [13] or Ainsworth [3]. Convergence rates are of order $\mathcal{O}(h^{2p+2})$for the real part and of order $\mathcal{O}(h^{2p+3})$ in the case of the upwind scheme. For the centered scheme, there is no dissipation error and dispersion error is of order $\mathcal{O}(h^{2p+3})$ for even polynomial orders and of order $\mathcal{O}(h^{2p+1})$ for odd polynomial orders.

2.1.2. Spatial accuracy

The spatial error looks on how the scheme is able to provide an accurate approximation of $v_i(x)$, the ith exact eigenfunctions. Our approach of dispersion analysis is advantageous here because it allows to treat both spectral and spatial accuracy with the same method.

We define the pointwise spatial error of mode i

$$e_i(x) = V_i(x) - v_i(x).$$

The L^2 norm of the error for mode i (with wave number k) is computed as

$$E_k = \sqrt{\int_0^1 e_1^2(x)dx = \mathcal{O}(k\Delta x)^{p+1}}$$

Yet, the structure of the spatial error is well known to be of the form of Radau polynomials, (see Fig. 2). Within this framework, we are able to exhibit the Radau structure of the spatial error: in Fig. 2, the mode $k\Delta x = 3/5$ is resolved and the difference between the exact mode and the numerical mode has the form of a Radau polynomial: $R_j(\xi) = \mathcal{P}_{j+1}(\xi) - \mathcal{P}_j(\xi)$.

2.2. Time discretizations

Let us now consider our system of ordinary differential equations:

$$\partial_t \mathbf{u} = \mathbf{A}\mathbf{u}.$$

The system is autonomous, i.e. **A** does not depend on time and linear i.e. **A** does not depend on **u**. Note that the ultimate goal of aero-acoustics is to predict the noise generated by turbulent unsteady flows, which requires the use of acoustic analogies if one computes the sound-generating turbulent flow field and the resulting acoustic field in two separated steps. Then, the resulting governing equations for the acoustics are not autonomous due to the presence of a source term that depends on the turbulent unsteady flow field. At the end of this paper, we will see that the optimal parametrization of the space–time scheme can be applied to non-autonomous systems.

Taking into account time discretization, the numerical scheme in space and time can be written in a compact form as

$$\mathbf{u}(t + \Delta t) = \mathbf{Z}\mathbf{u}(t),$$

with the space–time amplification matrix **Z**. Clearly, eigenvalues Z_k of **Z** must have a norm smaller or equal to 1 for ensuring stability and should be as close as possible as

$$Z_k = \mathrm{e}^{2i\pi kc\Delta t}$$

for ensuring good accuracy.

Let us define the Courant number

$$C = \frac{c\Delta t}{\Delta x}.$$

The explicit Runge–Kutta scheme of order *m* can be written formally as

$$\mathbf{u}(t + \Delta t) = \underbrace{\left(\sum_{j=0}^{m} \frac{C^j}{j!} \mathbf{A}^j\right)}_{\mathbf{Z}} \mathbf{u}(t). \tag{8}$$

If Λ = D_{ii} is the *i*th eigenvalue of **A**, then the corresponding amplification factor of the time stepping scheme is

$$Z(C\Lambda) = \sum_{j=0}^{m} \frac{(C\Lambda)^j}{j!}$$

For linear and autonomous systems, a Runge–Kutta of order *m* can always be written in with *m* stages, i.e. evaluating m times **Au**.[1] In the general case (non-linear/non-autonomous systems), more than *m* steps are necessary to obtain *m*th-order of accuracy if *m* > 4. This is the reason why fourth-order RK's are so popular. The minimum number of stages necessary for an explicit method to attain order *m* is still an open problem. For example, a ninth-order RK method requires between nine and 17 stages. In what follows, we will assume the systems to be linear and autonomous so that a *m*th-order RK can be done in m stages. In what follows, the RKmm scheme is the Runge–Kutta scheme of order *m* with *m* steps.

2.2.1. Stability

It is possible to predict the maximal Courant number available for a given combination of p and m (see e.g [8]). In Table 1, we see that the RK11 and RK22 schemes are not suitable for high-order computations because they have no stability on the imaginary axis. In some of the literature that deals with RK–DGM scheme, the choice *m* = *p* + 1 and *μ* = 1 has been studied. This choice seems quite obvious because both schemes have the formal accuracy of Δx^{p+1} and Δt^{p+1} and Dx and Dt are proportional to one another. The following condition

$$C < \frac{1}{2p+1} \tag{9}$$

ensures the stability of the scheme for any p. This rule on the time step is a good bound for low orders but becomes weaker as p increases. For example, at *p* = 8, 1/17 = 0.059 < 0.0626 and the simple stability condition predicts a time step 6% smaller than the actual stability limit.

2.2.2. Accuracy

It is possible to compute eigenvalues of **Z** and compare them to exact amplification factors. Fig. 3 shows eigenvalues of **Z** for different values of *p*.

The explicit Runge–Kutta schemes used here are accurate up to order Δt^m, so that the formal accuracy of the ''*p* + 1 - *p*'' RK–DGM scheme reduces asymptotically (i.e. for very low *k*Δ*x*'s) to order *p* + 1. This is due to the fact that the DG spatial scheme converges faster that the RK scheme.

Another possible choice is the ''2*p* + 2 - *p*'' scheme where both time and space discretizations asymptotically converge at the rate 2*p* + 2. In what follow, we demonstrate that this choice is not adequate for practical computations.

[1] In our MATLAB implementation, we have computed Z directly using (8) i.e. computing powers of A.

Published in : Journal of Computational Physics (2007), vol. 223, issue 1, pp. 188-207
doi.org/10.1016/j.jcp.2006.09.005
Status : Postprint (Author's version)

Table 1
Maximum available Courant number *C* for different combination of polynomial orders *p* and Runge–Kutta orders *m*

	$m = 1$	$m = 2$	$m = 3$	$m = 4$	$m = 5$	$m = 6$	$m = 7$	$m = 8$	$m = 9$
$p = 0$	1.0000	1.0000	1.2564	1.3926	1.6085	1.7767	1.9771	2.1568	2.3504
$p = 1$	0	0.3333	0.4096	0.4642	0.5348	0.5922	0.6590	0.7189	0.7835
$p = 2$	0	0	0.2098	0.2352	0.2716	0.3001	0.3339	0.3643	0.3970
$p = 3$	0	0	0.1301	0.1454	0.1679	0.1855	0.2064	0.2252	0.2454
$p = 4$	0	0	0.0897	0.1000	0.1155	0.1276	0.1420	0.1549	0.1688
$p = 5$	0	0	0.0661	0.0736	0.0851	0.0982	0.1045	0.1140	0.1243
$p = 6$	0	0	0.0510	0.0568	0.0656	0.0702	0.0806	0.0879	0.0958
$p = 7$	0	0	0.0407	0.0453	0.0523	0.0585	0.0643	0.0702	0.0765
$p = 8$	0	0	0.0334	0.0371	0.0428	0.0490	0.0527	0.0575	0.0626
$p = 9$	0	0	0.0279	0.0310	0.0358	0.0391	0.0440	0.0480	0.0523
$p = 10$	0	0	0.0237	0.0264	0.0304	0.0332	0.0374	0.0408	0.0445

This table refers to the full upwind case, i.e. $\mu = 1$.

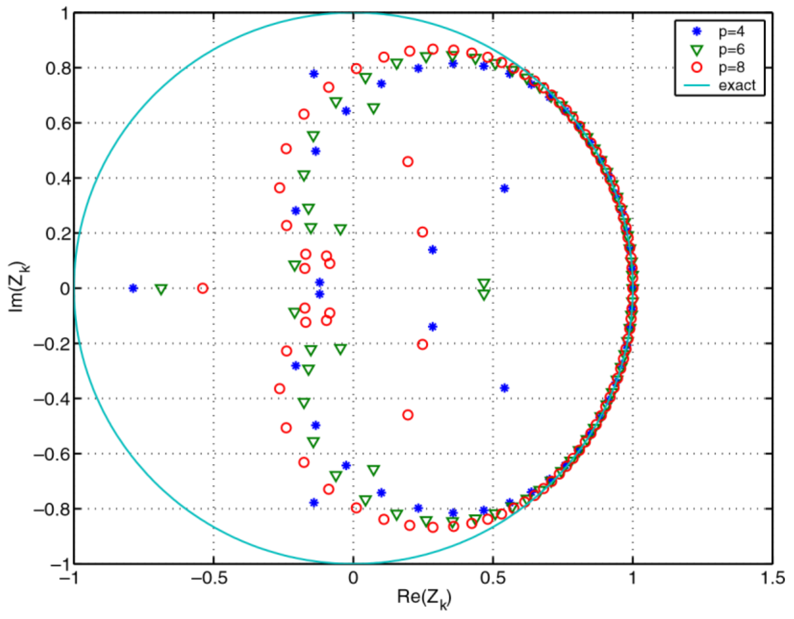

Fig. 3. Spectrum of **Z** for different values of p and using the explicit $p + 1 - p$ scheme and using upwind fluxes ($\mu = 1$).

We must highlight that time and space discretizations are very different. It is indeed possible to use 2 or 3 elements per wavelength with a polynomial order of $p = 4$. In the time domain, if a Runge–Kutta of order $m = 5$ is used, the condition that can be found in Table 1 tells us that we have to choose a Courant number of C ≃ 1/11 for ensuring the stability of the explicit scheme. This means that *we need about 11 times more time steps per period than the number of elements per wavelength*. In this sense, the mesh is much denser in time than it is in space. Asymptotically, the error of the RK time scheme certainly dominates the one of the DG space scheme but, for a certain range of numerical wave numbers, we should observe that errors in the space discretization dominate errors in the time discretization.

To demonstrate this, we have split the discretization error in two parts. First, the error in the time discretization is computed for each exact mode v_i i.e. for $z_i = 2ik_i\pi\Delta t$ with Δt such that the high-order CFL condition is fulfilled (see Table 1). Then, the RK–DGM dispersion error is evaluated by evaluating the spectrum of **Z**. Fig. 4 shows convergence plots of numerical amplification factors for RK-only (i.e. for RK with the exact modes) as well as for RK–DGM. As it was conjectured, two distinct behaviors appear on the convergence graph in the resolved range. For relatively large $k\Delta x$, the spatial DGM error dominates and convergence slope of the RK–DGM scheme is the one of the DGM, i.e. $2p + 2$.

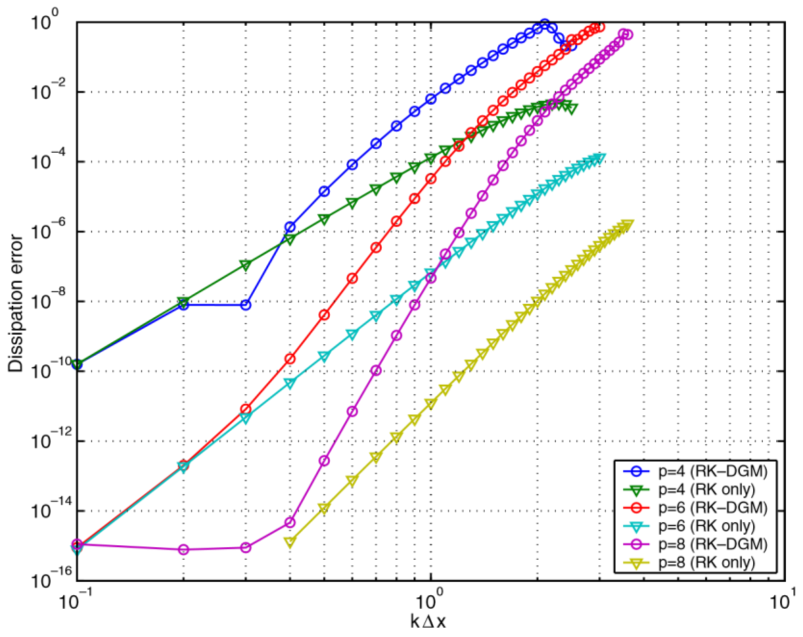

Fig. 4. Convergence plots of numerical amplification factors $|1 - |Z_k||$ as a function of numerical wave numbers. The Z_k's were calculated using Runge–Kutta only and using RK and DGM with the $p + 1 - p$ scheme.

Note that, for the range of numerical wave numbers that are computationally interesting (e.g. two elements per wavelength and $p = 4$), the spatial error dominates and is between 1 and 2 orders of magnitude larger than the temporal error. For smaller wave numbers, the RK error dominates and the RK–DGM convergence curve is superimposed with the RK curve. This range of wave numbers correspond to modes that are over-resolved in practical computations. An important conclusion of this is that the time discretization error is negligible for the $p + 1 - p$ scheme. From a practical point of view, the usefulness of such high-order time discretization is very limited. For example, the classical RK44 scheme can be used for high values of p. This scheme, the $4 - p$ scheme, enables to reduce the number of evaluations of the residual while being sufficiently accurate up to $p = 7$. Moreover, the RK44 scheme exists for non-linear/non-autonomous space operators.

Fig. 5 shows convergence curves for the $p - 4$ scheme.

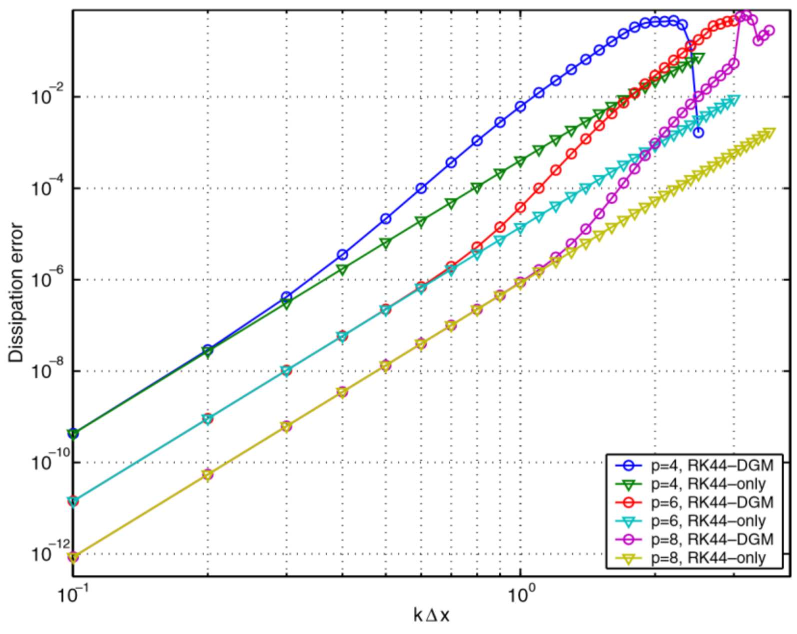

Fig. 5. Convergence plots of numerical amplification factors $|1 - |Z_k||$ as a function of numerical wave numbers. The Z_k's were calculated using Runge–Kutta only and using RK and DGM with the p - 4 scheme.

3. RK–DGM for the linearized Euler equations in two and three dimensions

Propagation of waves in arbitrary mean flows are governed by linearized Euler equations. The array of field variables

$$u = \begin{pmatrix} p' \\ v'_x \\ v'_y \\ v'_z \end{pmatrix} \tag{10}$$

defines acoustic pressure p' and velocity perturbations $v'_x;v'_y;v'_z$ around a mean flow v_{0x}, v_{0y}, v_{0z}. If the flow is homentropic, then $p' = c_0^2 p'$ and linearized Euler equation can be written in conservative form as

$$\begin{aligned} &\frac{\partial p'}{\partial t} + \nabla \cdot (p_0 \vec{v}' + \vec{v}_0 p') = 0, \\ &\frac{\partial \vec{v}'}{\partial t} + \nabla \cdot \left(\vec{v}' \otimes \vec{v}_0 + \frac{c_0^2}{p_0} |p' \right) = -\frac{1}{p_0} \left(\vec{v}_0 \cdot \nabla \vec{v}_0 + \frac{c_0^2}{p_0} \nabla p_0 \right) p' + (\nabla \cdot \vec{v}_0) \vec{v}' - \vec{v}' \cdot \nabla \vec{v}_0. \end{aligned} \tag{11}$$

Equations can be written in a more concise form as

$$\frac{\partial u}{\partial t} = -\nabla \cdot \vec{F} + s. \tag{12}$$

The arrays of flux vectors $\vec{F}$ are defined as

Published in : Journal of Computational Physics (2007), vol. 223, issue 1, pp. 188-207
doi.org/10.1016/j.jcp.2006.09.005
Status : Postprint (Author's version)

$$\vec{F} = \begin{pmatrix} \frac{1}{c_0^2} v_{0x} p' + p_0 v_x' & \frac{1}{c_0^2} v_{0y} p' + p_0 v_y' & \frac{1}{c_0^2} v_{0x} p' + p_0 v_x' \\ v_x' v_{0x} + \frac{p'}{p_0} & v_x' v_{0y} & v_x' v_{0z} \\ v_y' v_{0x} & v_y' v_{0y} + \frac{p'}{p_0} & v_y' v_{0z} \\ v_z' v_{0x} & v_z' v_{0y} & v_z' v_{0z} + \frac{p'}{p_0} \end{pmatrix} \tag{13}$$

3.1. Spatial discretization

To obtain the DG formulation, one multiplies Eq. (12) by a test function $\hat{u}$ and integrates over the domain, Ω. The divergence theorem is then applied to obtain the following variational formulation:

$$\int_{\Omega} \partial_t u \hat{u} dv + \int_{\Omega} F_x(u)\, \partial_x \hat{u} dv + \int_{\Omega} F_y(u)\, \partial_y \hat{u} dv + \int_{\Omega} F_z(u)\, \partial_z \hat{u} dv - \int_{\partial\Omega} f \hat{u} ds = \int_{\Omega} s \hat{u} dv, \qquad \forall \hat{u}, \tag{14}$$

Where $f = \vec{F} \cdot \vec{n}$ is the normal trace of the fluxes.
The physical domain, Ω, is discretized into a collection of N_e elements

$$\mathcal{T} = \bigcup_{e=1}^{\mathcal{N}e} e \tag{15}$$

called a mesh. In element e of $\mathcal{T}$, each component of u is discretized using a finite expansion, usually in a polynomial basis. It is common, in the finite element world, to distinguish reference coordinates ξ, η, ζ and real coordinates *x, y, z*. As we did it in 1D (3), we use piecewise continuous approximations on each element

$$u^e(\xi,\eta,\zeta) = \sum_{k=1}^{d} \mathcal{P}_k(\xi,\eta,\zeta) u_k^e \tag{16}$$

In (16), d is the size of the space of polynomials used for approximating u^e. For each interface in the mesh, we define a unique normal **n** and we denote by u_L and u_R the field on the left and the right side of this face, with **n** going from the ''left'' to the ''right''. The numerical flux $f(u_L,u_R)$, is computed using a Riemann solver that is constructed by computing characteristics of the left and right fields. Only the upwind quantities are kept to compute the normal flux.

3.1.1. Quadrature free implementation of the space operator

The problem (12) that we aim to solve is a linear hyperbolic PDE with non-constant coefficients if the mean flow defined by v_0, p_0 and c_0 is non-uniform. The following assumption is done for treating the mean flow. If $g(u^e)$ is a function of the unknown u^e, we apply the following rule to compute g:

$$g(u^e) = g\left(\sum_{k=1}^{d} \mathcal{P}_k u_k^e\right) \simeq \sum_{k=1}^{d} \mathcal{P}_k g(u_k^e) \tag{17}$$

This assumption allows to derive the quadrature free DGM [4,16], an approximation that permits to achieve a much better efficiency than the direct integration of each term by means of a summation over quadrature points.

The general idea of assumption (17) is that any function is approximated on the ''same grid as u''. For example, if $g = u^2$ and if $u^e = \sum_k \mathcal{P}_k u_k^e$ is in $\mathbb{P}^p$, then $(u^e)^2 = (\sum_k \mathcal{P}_k u_k^e)^2$ is in $\mathbb{P}^{2p}$. With assumption (17), $u^2 \simeq \sum_k \mathcal{P}_k (u_k^e)^2 \in \mathbb{P}^p$. Eq. (17) is, of course, only exact for linear functions *F*. Note that, even for fully non-linear problems, it is still possible to make this assumption with minor stabilizations for singular points [12].

In this work, we use Lagrange shape functions for interpolating *u*. This choice is certainly not the only one available. Orthogonal shape functions [19], for example, have specific interest. Here, we have *d* Lagrange points in each element. We note f_k^e for the value of a given function *f* at node *k* of element *e*. We have therefore[2]

$$f(u) = \sum_k f_k^e \mathcal{P}_k = f_k^e \mathcal{P}_k. \tag{18}$$

The interpolations being disconnected, it is possible to write (14) for each element e of $\mathcal{T}$ as

$$\begin{aligned}\partial_t u_k^e \int_e \mathcal{P}_k \mathcal{P}_j \mathrm{d}v = F_x(u_k^e) \int_e \mathcal{P}_k \partial_x \mathcal{P}_j \mathrm{d}v + F_y(u_k^e) \int_e \mathcal{P}_k \partial_y \mathcal{P}_j \mathrm{d}v + F_z(u_k^e) \int_e \mathcal{P}_k \partial_z \mathcal{P}_j \mathrm{d}v \\ - \sum_{l=1}^{n_e} f(u_\mathrm{L}, u_\mathrm{R})_k^{e,l} \cdot \vec{n} \int_{\partial e_l} \mathcal{P}_k \mathcal{P}_j \mathrm{d}s \\ = 0 + s(u_k^e) \int_e \mathcal{P}_k \mathcal{P}_j \mathrm{d}v, \qquad \forall i,j\end{aligned} \tag{19}$$

where we have decomposed the boundary, ∂*e*, of element *e* into n_e parts ∂e_l corresponding, in 3D, to the four triangular faces of a tetrahedron. In (19), $f(u_\mathrm{L}, u_\mathrm{R})_k^{e,l} \cdot \vec{n}$ is the numerical flux computed using the Riemann solver between the fields on the left and right of the face.
We further assume that the Jacobian matrix

$$J = \begin{bmatrix} \partial_\xi x & \partial_\eta x & \partial_\zeta x \\ \partial_\xi y & \partial_\eta y & \partial_\zeta y \\ \partial_\xi z & \partial_\eta z & \partial_\zeta z \end{bmatrix}$$

is constant for any element *e*. This is true, for example, if triangular/tetrahedral meshes are used and if all the edges of the mesh are straight sided. We note $\|e\|$ = det*J*. Some important DGM operators appear in (19). We define two mass matrix operators, one relative to element *e*,

$$M_{ij}^e = \int_e \mathcal{P}_i \mathcal{P}_j \mathrm{d}x\mathrm{d}y\mathrm{d}z = \int_e \mathcal{P}_i \mathcal{P}_j \mathrm{d}\xi \mathrm{d}\eta \mathrm{d}\zeta \, \|e\| = M_{ij} \|e\|$$

and one relative to element boundaries ∂e_l,

$$M_{ij}^{\partial e_l} = \int_{\partial e_l} \mathcal{P}_i \mathcal{P}_j \mathrm{d}x\mathrm{d}y\mathrm{d}z = \int_{\partial e_l} \mathcal{P}_i \mathcal{P}_j \mathrm{d}\xi \mathrm{d}\eta \mathrm{d}\zeta \, \|\partial e_l\| = M_{ij}^l \|\partial e_l\|$$

M_{ij} and M_{ij}^l are constant matrices, independent of *e*, and of size *d* x *d*. We finally define three derivatives operators

[2] Starting here, we use the Einstein summation rule for repeated indices.

$$D_{ij}^{\xi} = \int_e \mathcal{P}_i \partial_\xi \mathcal{P}_j \mathrm{d}\xi \mathrm{d}\eta \mathrm{d}\zeta, \quad D_{ij}^{\eta} = \int_e \mathcal{P}_i \partial_\eta \mathcal{P}_j \mathrm{d}\xi \mathrm{d}\eta \mathrm{d}\zeta, \quad D_{ij}^{\zeta} = \int_e \mathcal{P}_i \partial_\zeta \mathcal{P}_j \mathrm{d}\xi \mathrm{d}\eta \mathrm{d}\zeta.$$

These operators are square matrices of size *d* x *d*. They are independent of *e*.
The choice of a specific family of shape functions makes the structure of some DGM operators computationally interesting. For Lagrange shape functions, boundary operators M_{ij}^{e} are sparse. In case of orthogonal polynomials, derivative operators are upper triangular matrices.

Note that

$$D_{ij}^{x} = \int_e \mathcal{P}_i \partial_x \mathcal{P}_j \mathrm{d}x\mathrm{d}y\mathrm{d}z = \|e\| \left(D_{ij}^{\xi} J_{11}^{-1} + D_{ij}^{\eta} J_{12}^{-1} + D_{ij}^{\zeta} J_{13}^{-1} \right).$$

Thanks to the previous definitions, the DGM formulation is written, for element *e*, as

$$\partial_t u_k^e M_{kj} = \left(F_\xi(u)\right)_k^e D_{kj}^{\xi} + \left(F_\eta(u)\right)_k^e D_{kj}^{\eta} + \left(F_\zeta(u)\right)_k^e D_{kj}^{\zeta} - \frac{1}{\|e\|} \sum_{l=1}^{n_e} \|\partial e_l\| f_k^{e,l} M_{kj}^{l} + s_k^e M_{kj}. \qquad (20)$$

Note that the computational cost of the source term is quite low because it only involves evaluation of the function and no matrix–matrix multiplications.
In (20), we have computed fluxes in the reference system of coordinates, i.e.

$$\left(F_\xi(u)\right)_k^e = \left(F_x(u)\right)_k^e J_{11}^{-1} + \left(F_y(u)\right)_k^e J_{21}^{-1} + \left(F_z(u)\right)_k^e J_{31}^{-1}$$

$$\left(F_\eta(u)\right)_k^e = \left(F_x(u)\right)_k^e J_{12}^{-1} + \left(F_y(u)\right)_k^e J_{22}^{-1} + \left(F_z(u)\right)_k^e J_{32}^{-1}$$

$$\left(F_\zeta(u)\right)_k^e = \left(F_x(u)\right)_k^e J_{13}^{-1} + \left(F_y(u)\right)_k^e J_{23}^{-1} + \left(F_z(u)\right)_k^e J_{33}^{-1}$$

Formulation (20) is a quadrature free version of the DGM. For sufficiently large *p*, most of the computation time is spent in the computation of the derivative operator. An efficient way to implement (20) on a computer is to use Basic Linear Algebra Subroutines (BLAS). In particular, (20) involves three large matrix–matrix multiplications. Indeed, since all elements have the same matrices D^{ξ}, D^{η}, D^{ζ}, one can first compute the fluxes in each element frame of reference for all interpolation points and all elements, using Eq. (21). The fluxes computed in this way can be stored in a matrix with *d* rows and $4\mathcal{N}_e$ columns, where 4 is the number of fields in *u* for our acoustic case. The derivative operator can then be applied through three matrix–matrix multiplications where we fully take advantage of level 3 BLAS (BLAS3) subroutines. Typically, for the DGM of order *p* = 6 on tetrahedra, we have *d = (p* + 1*)(p* + 2*)(p* + 3)/6 = 84. Lagrange shape functions for each element and the matrices of problem (20) have a size 84 x 84. The more this elementary matrix are big, the more we can get closer to the peak performance of the processor, and the more we need to rely on the quality of the BLAS implementation.

4. Variable-*p* discontinuous Galerkin method

In the kind of simulations we are interested in, we know a priori the desired mesh sizes at any point of the domain. Mesh size depends on both the wave length of the excitation and on the local Mach number of the mean flow. The polynomial order p is chosen using our knowledge about spectral accuracy of the DGM scheme: for a certain p, we need a given number of elements per wavelength.

4.1. CHOICE OF THE POLYNOMIAL ORDER

Fig. 4 can be used to determine the number of elements per wavelength that are necessary to obtain a prescribed spectral accuracy for a given polynomial order. In Table 2, we have chosen a finite number of desired accuracies ϵ and we have computed, for different values of p, the number of elements per wavelength that are necessary to obtain this desired accuracy. Therefore, we define the following non-dimensional dissipation error measure:

$$\epsilon = \frac{|\Re(A_k)|\Delta x}{c_0}.$$

We choose the dissipation error as an indicator because in DGM it dominates the dispersion error.
A signal of frequency f and wavelength k = c_0/f will decrease in amplitude by a factor

$$r = e^{-\frac{\epsilon c_0}{\Delta x}T} = e^{-\epsilon\frac{\lambda}{\Delta x}}$$

over one period $T = 1/f = \lambda/c_0$.
The numbers listed in Table 2 were computed using a linear interpolation in a log–log scale graph. In case of a mesh with variable element sizes, Table 2 allows to choose the local polynomial order that equi-distributes the dissipation error.

4.2. Filtering the solution for varying *p*

We have shown in the previous section that high-order interpolation bases allow an efficient implementation of the DG scheme and coarser meshes. On the other side, the geometry of the domain has to be discretized with sufficient accuracy for controlling the error due to a to coarse approximation of the geometry. It is indeed possible to use both high-order functional and geometrical discretization, i.e. use geometrically high-order elements. Disappointingly, this has two major disadvantages

- the generation of geometrically high-order curved meshes is not trivial at all [20];
- the quadrature free approximation does not apply as is to elements with varying Jacobians.

In our approach, we start with a mesh that is geometrically accurate by reducing element size in regions where the geometry exhibits a high curvature. Then, we apply p-coarsening in those small elements for which the highest polynomial order leads to over-resolved fields.

In principle, the DG method allows to deal with variable p meshes [19]. However, the implementation of an efficient variable p DG method is not at all straightforward because

- since the number of degrees of freedom differs from one element to another, the computation volume terms has to be split into separate BLAS-3 matrix–matrix multiplications, one for each order;

Published in : Journal of Computational Physics (2007), vol. 223, issue 1, pp. 188-207
doi.org/10.1016/j.jcp.2006.09.005
Status : Postprint (Author's version)

Table 2
Number of elements per wavelength necessary to obtain a desired accuracy (non-dimensional diffusion error $\epsilon = \frac{|\Re(A_k)|\Delta x}{c_0}$) for various polynomial orders *p*

	p = 1	*p* = 2	*p* = 3	*p* = 4	*p* = 5	*p* = 6
$\epsilon = 10^{-1}$	3.6572	1.9098	1.2594	0.9295	0.7328	0.6031
$\epsilon = 10^{-2}$	6.6670	2.9408	1.7807	1.2482	0.9487	0.7605
$\epsilon = 10^{-3}$	12.0408	4.4239	2.4513	1.6265	1.1926	0.9312
$\epsilon = 10^{-4}$	21.5166	6.5626	3.3220	2.0871	1.4765	1.1234
$\epsilon = 10^{-5}$	38.3263	9.6978	4.4638	2.6568	1.8124	1.3437
$\epsilon = 10^{-6}$	68.1900	14.2637	5.9792	3.3685	2.2144	1.5999

- since the quadrature free implementation of surface fluxes involving left and right elements of different orders is not as simple as for the constant order case.

Because only a few elements will see their order reduced, we propose here to keep all elements at their maximal order and only project both solutions and residual in a lower-order polynomial function space and then project them back in the high-order space. This back-and-forth operation will effectively filter out the highorder part of the solution.

Let us define expansions at order *p* and *q (q* < *p*), i.e.

$$u^p = \sum_{k=1}^{d_p} \mathcal{P}_k u_k^e \in \mathbb{P}^p \quad and \quad u^q = \sum_{k=1}^{d_q} \mathcal{Q}_k u_k^e \in \mathbb{P}^q$$

as well as the following matrices:

$$M_{ij}^p = \int_e \mathcal{P}_i \mathcal{P}_j dv, \quad M_{ij}^q = \int_e \mathcal{Q}_i \mathcal{P} \mathcal{Q}_j dv \quad and \quad M_{ij}^{pq} = \int_e \mathcal{P}_i \mathcal{Q}_j dv = M_{ji}^{qp} \tag{21}$$

Eq. (20) can be written in a more concise form as

$$M^p \partial_t u^p = R^p(u^p), \tag{22}$$

where R^p is the residual vector at order p involving the volume term, the surface term and the source term. The building block of the Runge–Kutta time stepping is the forward Euler step. It can be written in two steps

- Compute the residual vector: $R^p(u^p)$.
- Update the solution: $u^{p*} = u^p + \Delta t (M^p)^{-1} R^p$.

It is possible to reduce the order of the scheme locally using filters in the same way as it is done for limiters. The forward Euler step is modified as follows:

- Compute the residual vector: $R^p(u^p)$.
- Filter the residual vector: $R^p \longleftarrow F(R^p)$.
- Update the solution: $u^{p*} = u^p + \Delta t (M^p)^{-1} R^p$.
- Filter the solution: $u^{p*} \longleftarrow F(u^{p*})$.

The L^2 projection operators Π^{pq} and Π^{qp} that L^2-project, respectively, $u^p \longrightarrow u^q$ and $u^q \longrightarrow u^p$ are constructed as follows:

$$\Pi^{qp} = (M^q)^{-1} M^{qp} \quad and \quad \Pi^{pq} = (M^p)^{-1} M^{pq}.$$

To filter higher-order modes (in the sense of L^2) out of the solution, we simply apply Π^{pq} followed by Π^{qp}. This filtering operator can be written as

$$f(u^p) = \Pi^{pq}\Pi^{qp}u^p = [(M^p)^{-1}M^{pq}(M^q)^{-1}M^{qp}]u^p.$$

Filtering the residual is slightly different. The L^2 projection operators $\overline{\Pi}^{pq}$ and $\overline{\Pi}^{qp}$ that L^2-project, respectively, $R^p \longrightarrow R^q$ and $R^q \longrightarrow R^p$ are constructed as follows:

$$\overline{\Pi}^{qp} = M^{qp}(M^p)^{-1} \quad and \quad \overline{\Pi}^{pq} = M^{pq}(M^q)^{-1}.$$

The residual filter can be written as

$$F(R^p) = \overline{\Pi}^{pq}\overline{\Pi}^{qp}u^p = [M^{pq}(M^q)^{-1}M^{qp}(M^p)^{-1}]R^p.$$

Developments are given in the Appendix.
Filters *F* and *f* are independent of the metric and therefore are stored only once for every $q < p$.

5. Example

To validate our methodology, we consider the propagation of an acoustic mode in a 2D planar nacelle-like geometry, as depicted in Fig. 6(a). The geometry is actually a slice of a 3D nacelle. The inner and outer radius of the fan are 0.14 m and 0.68 m, respectively. Since only characteristic non-reflecting boundary conditions were implemented at the time of this writing, a single incident plane wave is imposed at the fan face, at a frequency f = 2.5 kHz.

The mean flow is incompressible and such that M = 0.2 at the boundary while M = 0.3 at the fan face. The highest value of the Mach number, M = 0.4, is attained near the lip.

We consider the speed of sound at rest to be c_0 = 337 m/s. The wavelength of our signal is, in the no flow case λ = 13.48 cm. We have used a mesh of variable element sizes, varying from 1 cm to 10 cm. Most of the elements are of mesh sizes of Δx = 10 cm. Small elements are concentrated at the vicinity of the nacelle lip where the geometry exhibits large curvatures. The number of elements per wavelength for the elements of size 10 cm is about 1.35.

The Helmholtz number, k_r, is usually defined as

$$k_r = 2\pi\frac{H}{\lambda}$$

where H = 1.38 m is the diameter of the nacelle. In our simulation, k_r = 64. This value is considered as a high Helmholtz number in the aero-acoustics community.

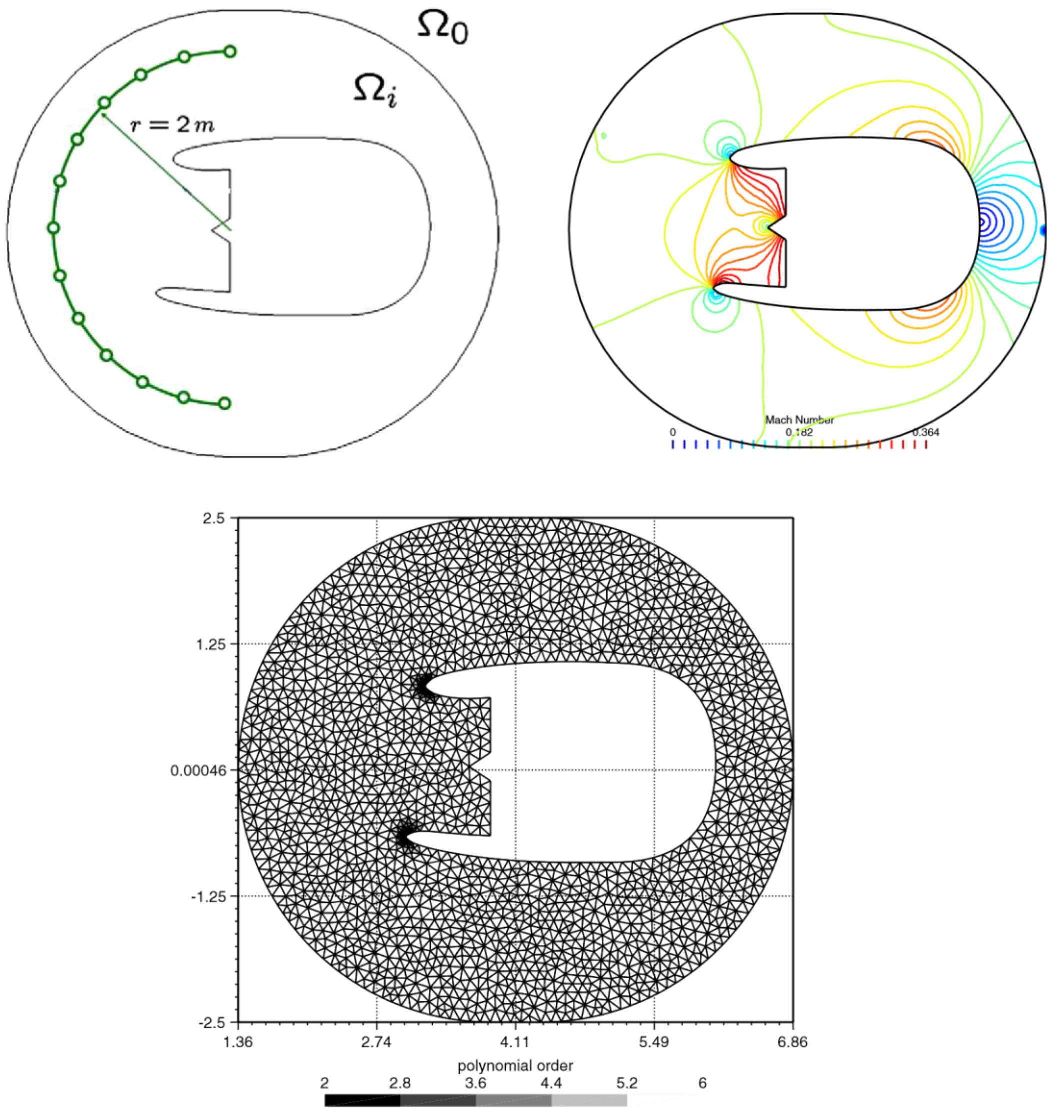

Fig. 6. Test case for the 2D plane wave propagation (top,left). Geometry and series of sampling points. The series consists of sampling points located on a quarter of a circle centered about the middle of the fan face and whose radius is equal to 2 m (top,right). Mach number of the mean flow (bottom). Mesh of the domain (3921 triangles). Element sizes ranges from 1 cm to 10 cm. Element are colored using levels of grey that are representing polynomial orders used in the optimized run.

The acoustic mode that is initiated at the fan has to propagate for a distance of about 3 m. At a frequency of 2.5 kHz, about 20 wave cycles will be necessary to travel that distance. Using the arguments developed in Section 4, it is possible to predict how the DG scheme is going to perform for different polynomial orders. The proportion of the wave amplitude that will remain after 20 cycles will be $R = r(\epsilon)^{20}$. For p = 6 and 1.35 elements per wavelength, Table 2 indicates that $\epsilon \simeq 1 \cdot e - 5$ so that $r = e^{-1 \cdot e - 5 \times 1.35} = 0.9999865$. We have therefore that $R = (0.9999865)^{20} = 0.99973$. It is possible to compute the same ratio for other orders:

- R = 1.0000 for p = 7,
- R = 0.9997 for p = 6,
- R = 0.8711 for p = 5,
- R = 0.4369 for p = 4,
- $R \simeq$ 0.0000 for p < 4.

The right polynomial choice for this specific problem is *p* = 6 if we can afford an error of a tenth of a percent. Increasing the polynomial order should not, in principle, be beneficial in terms of accuracy while degrading the performance. Also, accuracy should decrease with decreasing interpolation order.

We have solved the problem using polynomial orders ranging from *p* = 2 to *p* = 7 and using the *p* + 1 - *p* scheme. We have then computed the same problem using *p*-adaptivity. The element orders have been chosen in order to evenly distribute the dissipation error through the mesh. The resulting distribution of orders is presented in Fig. 6. Among the 3941 triangles of the mesh, 3687 are at order *p* = 6,29 at order *p* = 5,24 at order *p* = 4,38 at order p = 3, and the rest is at order p = 2. The resulting time step is 6.8 times bigger than the one for a uniform p = 6 computation. Fig. 7 depicts the Sound Pressure Level (in dB) at the sampling points for the different cases. Clearly, the results for the variable p scheme compare well with the ones of the *p* = 7 scheme, which is over-resolved. A difference of less than half of a dB is seen between the optimal p results and the *p* = 7 results. As predicted, using lower orders of approximation leads to unresolved results.

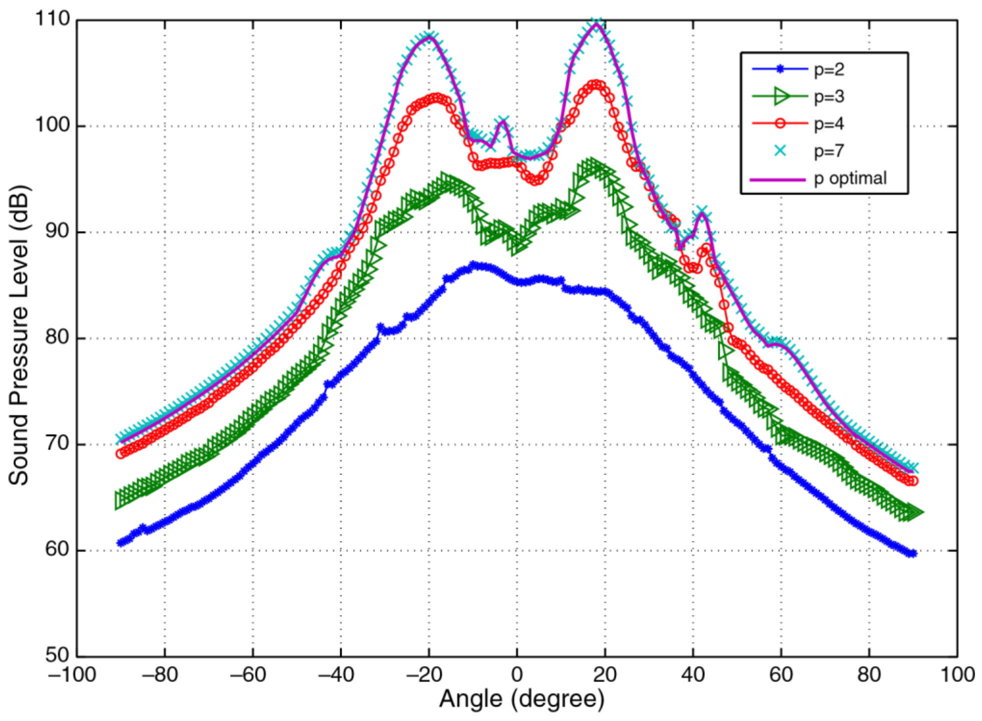

Fig. 7. Sound pressure level (in dB) at sampling points for the 2D plane wave propagation.

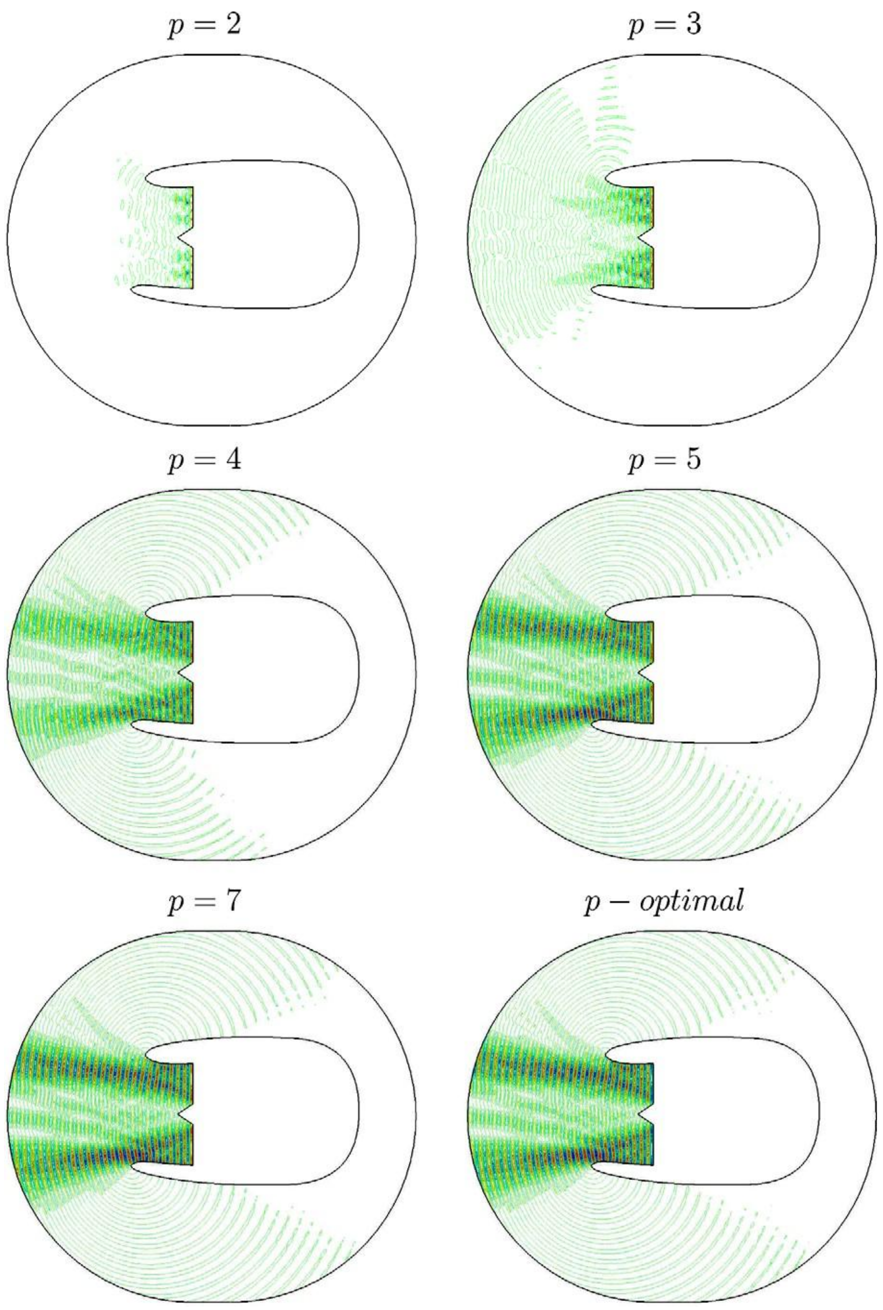

Fig. 8. Test case for the 2D plane wave propagation.

Fig. 8 shows iso-contours of the acoustic pressure field *p'* for the different cases. Details of resolved computations are compared in Table 3.

We have finally computed the same problem with a uniform mesh of triangles of size equal to 10 cm. The detail of the mesh at the vicinity of the lip of the nacelle is presented at Fig. 9. We have computed the sound propagation using both meshes and sound pressure levels at sampling points are presented at Fig. 10. Clearly, the geometrical error dominates in the case of the uniform mesh. We conjecture here that every geometrical feature of size equivalent to the wavelength should be geometrically captured by the mesh in order to have converged solutions.

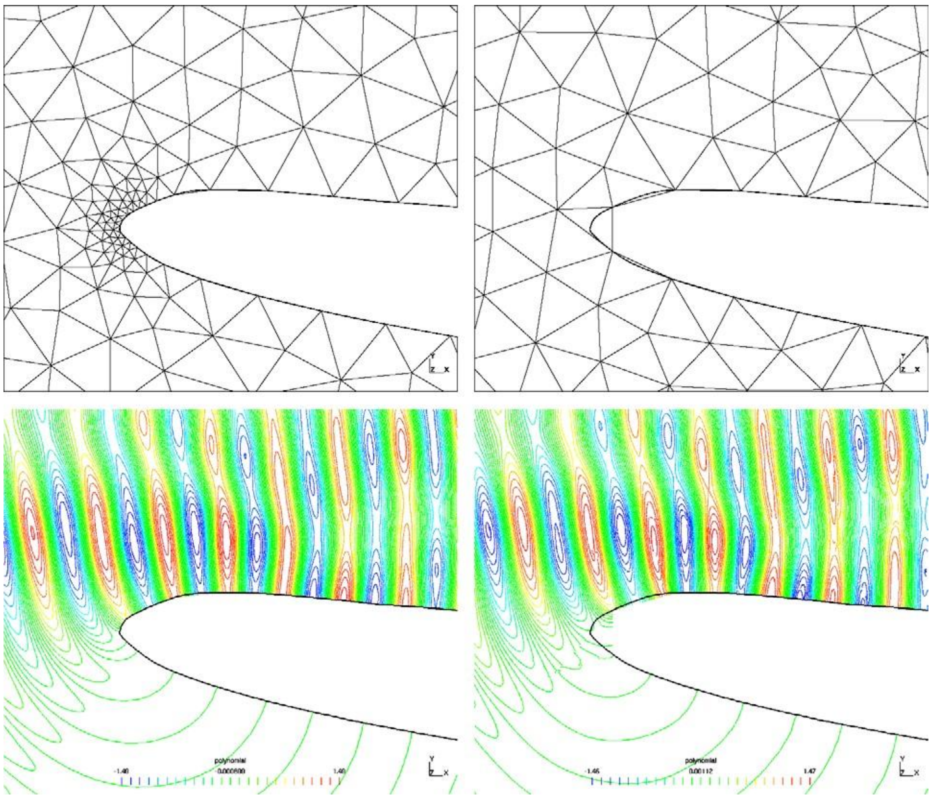

Fig. 9. Uniform (top/left) and geometrically adapted (top/right) meshes. Corresponding pressure levels are represented at bottom figures. All graphics are showing the mesh at the vicinity of the nacelle lip.

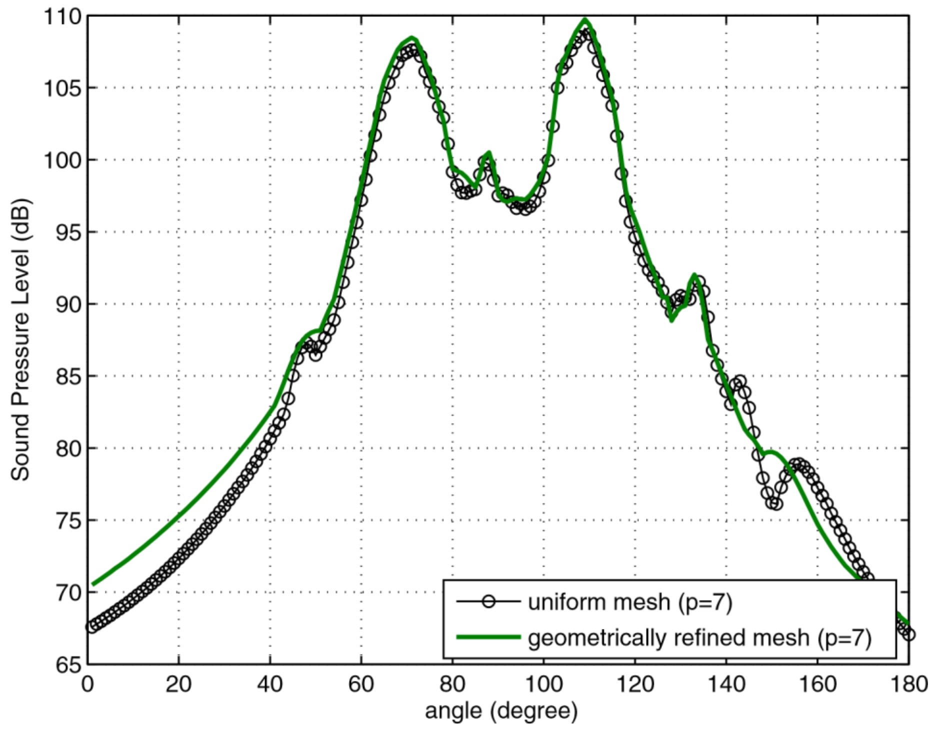

Fig. 10. Sound pressure level (in dB) at sampling points for the 2D plane wave propagation.

Table 3
CPU times for the different resolved cases (the cases that provide the right solution)

DG order	RK order	Δt	CPU/time step	Total CPU (h)
p = 6 (constant)	m = 7	1.0×10^{-6}	1.40	6.9
p = 6 (variable)	m = 7	6.7×10^{-6}	1.42	1.04
p = 6 (variable)	m = 4	4.9×10^{-6}	0.77	0.77
p = 7 (constant)	m = 8	8.6×10^{-7}	1.94	11.15

The 4 - p (variable p) scheme is the most efficient.

6. Conclusions

In this paper, we have proposed a number of rules that enable to do efficient discontinuous Galerkin computations of wave propagation problems. Some important issues were addressed like

- A methodology to choose an optimal polynomial order with respect to signal wave lengths.
- A methodology to choose an optimal time-stepping scheme that is sufficiently accurate not to degrade space accuracy for the relevant range of wavelengths of the problem.
- A way to take into account variable polynomial order in an efficient way.
- A way to distribute polynomial orders in order to equi-distribute dissipation error.

At the end, we were able to perform one real aero-acoustics application. The results confirm that a priori selected parameters values are optimal.

The computation of noise generated by turbulent flows in real 3D geometries is the final goal of our research. Some important issues still have to be addressed such as the account of acoustic liners (time domain impedance boundary conditions) as well as the stability of the LEE in the presence of shear flows.

Appendix. Projection operators

It is indeed easy to show how to project u^p from $\mathbb{P}^p$ to $\mathbb{P}^q$. We have

$$u^p = \sum_{k=1}^{d_p} \mathcal{P}_k u_k^p$$

The L^2 projection of u^p into $\mathbb{P}^q$ is written as find $u^q \in \mathbb{P}^q$ solution of

$$\int_e u^p Q_j \mathrm{d}v = \int_e u^q Q_j \mathrm{d}v \quad \forall Q_j \in \mathbb{P}^q.$$

Expanding u^q in $\mathcal{Q}_i$'s, i.e. writing $u^q = \sum_{k=1}^{d_q} \mathcal{Q}_k u_k^q$ gives

$$\sum_{k=1}^{d_p} u_k^p \int_e \mathcal{P}_k Q_j \mathrm{d}v = \sum_{k=1}^{d_q} u_k^q \int_e \mathcal{Q}_k Q_j \mathrm{d}v \quad \forall Q_j \in \mathbb{P}^q.$$

Taking into account the definition of mass matrices (21), we write the L^2 projection in matrix form

$$u^q = (M^q)^{-1} M^{pq} u^p = \Pi^{qp} u^q.$$

Now, if we aim to project the weighted residual R^p, i.e. the right hand side of (19) into a different polynomial space, the situation is different. The right hand side of (12) has the same nature as u: it is a vector r^p of $\mathbb{P}^p$. In its discrete form (that we do not know), it can be expanded as

$$r^p = \sum_{k=1}^{d_p} r_k^p \mathcal{P}_k.$$

We have the following result:

$$r^q = (M^q)^{-1} M^{pq} r^p$$

The weighted residuals are written as

$$R_j^p = \int_e r^q \mathcal{P}_j \, \mathrm{d}v = \sum_{k=1}^{d_p} r_k^p \int_e \mathcal{P}_k \mathcal{P}_j$$

which gives, in matrix form

$$R^p = M^p r^p$$

Consequently, we find

$$R^q = M^q r^q = M^q (M^q)^{-1} M^{pq} r^p = M^{pq} (M^p)^{-1} R^q = \overline{\Pi}^{qp} R^p.$$

References

[1] ARPACK (the ARnoldi PACKage), 2006.

[2] S. Adjerid, K.D. Devine, J.E. Flaherty, L. Krivodonova, A posteriori error estimation for discontinuous Galerkin solutions of hyperbolic problems, Computer Methods in Applied Mechanics and Engineering 191 (2002) 1097–1112.

[3] M. Ainsworth, Dispersive and dissipative behavior of high order discontinuous Galerkin finite element methods, Journal of Computational Physics 198 (1) (2004) 106–130.

[4] H. Atkins, C.-W. Shu, Quadrature-free implementation of the discontinuous Galerkin method for hyperbolic equations, AIAA Journal 36 (1998) 775–782.

[5] M.J. Berger, J. Oliger, Adaptive mesh refinement for hyperbolic partial differential equations, Journal of Computational Physics 53 (1984) 484–512.

[6] N. Chevaugeon, J.-F. Remacle, X. Gallez, P. Ploumhans, S. Caro, Efficient discontinuous Galerkin methods for solving acoustic problems, in: 11th AIAA/CEAS Aeroacoustics Conference (26th AIAA Aeroacoustics Conference), Monterey, 2005.

[7] B. Cockburn, C.-W. Shu, Tvd Runge–Kutta local projection discontinuous Galerkin finite element method for conservation laws, Mathematics of Computations 52 (186) (1989) 411–435.

[8] B. Cockburn, C.-W. Shu, Runge–Kutta discontinuous Galerkin methods for convection-dominated problems, Journal of Scientific Computing 16 (2001) 173–261.

[9] C. Dawson, R. Kirby, High resolution schemes for conservation laws with locally varying time steps, SIAM Journal on Scientific Computing 22 (6) (2001) 2256–2281.

[10] R.S. Falk, G.R. Richter, Explicit finite element methods for symmetric hyperbolic equations, SIAM Journal on Numerical Analysis 36 (1999) 935–952.

[11] J.E. Flaherty, R.M. Loy, M.S. Shephard, B.K. Szymanski, J. Teresco, L. Ziantz, Adaptive local refinement with octree loadbalancing for the parallel solution of three-dimensional conservation laws, Journal of Parallel and Distributed Computing 47 (1997) 139–152.

[12] K. Hillewaert, N. Chevaugeon, P. Geuzaine, J.-F. Remacle, Hierarchic multigrid iteration strategy for the discontinuous Galerkin solution of the steady euler equations, International Journal for Numerical Methods in Fluids 51 (9–10) (2006) 1157–1176.

[13] F. Hu, H. Atkins, Eigensolution analysis of the discontinuous Galerkin method with nonuniform grids *i* one space dimension, Journal of Computational Physics 182 (2) (2002) 516–545.

[14] P. Lesaint, Sur la résolution des systèmes hyperboliques du premier ordre par la méthode des éléments finis. Ph.D. Thesis, Université Pierre et Marie Curie, 1975.

[15] R. LeVeque, Numerical Methods for Conservation Laws, Birkhäuser-Verlag, 1992.

[16] D.P. Lockard, H. Atkins, Efficient implementations of the quadrature-free discontinuous Galerkin method, in: Proceeding of 14th AIAA CFD Conference, AIAA, 1999, pp. 526–536.

[17] P.K. Moore, J.E. Flaherty, A local refinement finite element method for one-dimensional parabolic systems, SIAM Journal on Numerical Analysis 27 (1990) 1422–1444.

[18] P.K. Moore, J.E. Flaherty, Adaptive local overlapping grid methods for parabolic systems in two space dimensions, Journal of Computational Physics 98 (1992) 54–63.

[19] J.-F. Remacle, J.J.E. Flaherty, M. Shephard, An adaptive discontinuous Galerkin technique with an orthogonal basis applied to compressible flow problems, SIAM Review 45 (2003) 53–72.

[20] M.S. Shephard, J.E. Flaherty, K.E. Jansen, X. Li, X. Luo, N. Chevaugeon, J.-F. Remacle, M.W. Beall, R.M. O'Bara, Adaptive mesh generation for curved domains, Applied Numerical Mathematics 52 (2005) 251–271.