Int. J. Appl. Math. Comput. Sci., 2007, Vol. 17, No. 3, 375–393
DOI: 10.2478/v10006-007-0031-2

amcs

ARBITRARY HIGH-ORDER FINITE ELEMENT SCHEMES AND HIGH-ORDER MASS LUMPING

SÉBASTIEN JUND, STÉPHANIE SALMON

Institut de Recherche Mathématique Avancée, UMR 7501 de l'ULP et du CNRS
7, rue René Descartes, 67084 Strasbourg Cedex, France
e-mail: jund,salmon@math.u-strasbg.fr

Computers are becoming sufficiently powerful to permit to numerically solve problems such as the wave equation with high-order methods. In this article we will consider Lagrange finite elements of order k and show how it is possible to automatically generate the mass and stiffness matrices of any order with the help of symbolic computation software. We compare two high-order time discretizations: an explicit one using a Taylor expansion in time (a Cauchy-Kowalewski procedure) and an implicit Runge-Kutta scheme. We also construct in a systematic way a high-order quadrature which is optimal in terms of the number of points, which enables the use of mass lumping, up to $P5$ elements. We compare computational time and effort for several codes which are of high order in time and space and study their respective properties.

Keywords: wave equation, finite element method, mass lumping, Cauchy-Kowalewski procedure

1. Introduction

The second-order wave equation is a fundamental equation modeling several phenomena such as acoustic, elastic, or electromagnetic waves. In the DFG-CNRS project *Noise generation in turbulent flows* we need to solve aeroacoustic problems where sources of the wave equation come from fluid dynamics. We then need to couple a code solving fluid dynamics (HYDSOL (Dumbser, 2005)) with a code solving the propagation of the acoustic waves. As the acoustic waves propagate very far and as we work on realistic though complex geometries, we want a very precise solver on unstructured meshes, i.e., high-order schemes in both time and space.

To develop high-order schemes in space, we had to choose between different methods. First, the finite difference method, which is a popular numerical technique because of its relative simplicity of implementation, is obviously less adapted to complex geometries than those for aeroacoustic problems. Accordingly, we chose to develop high-order finite element schemes. Finite element methods are well adapted for unstructured meshes but have the main drawback that the mass matrix, which is full, has to be inverted at each time step. This problem can be circumvented by the use of lumped finite elements, which allows us to approximate the mass matrix by a diagonal one using a quadrature formula rather than exact integration (Ciarlet, 1978). The development of high-order mass lumping in 1D was discussed in (Cohen *et al.*, 1994), the case of tensor product elements such Q_k in 2D in (Cohen *et al.*, 1993), and all of the results for mass-lumping are summarized in the book (Cohen, 2002).

For obtaining high-order time schemes, we shall study two types of discretization: explicit and implicit. An explicit time scheme is based on a Taylor expansion of the unknown and its time derivatives and can be seen as a Cauchy-Kovalewski procedure (Titarev and Toro, 2002). Notice that, in this explicit case, the mass-lumping is necessary as the mass matrix has to be inverted at each time step. So we will compare the results and the efficiency of the schemes with a diagonally implicit Runge-Kutta method (Butcher, 2003), which does not need the inversion of the full mass matrix at each time step.

A comparison between high-order finite elements and finite differences was already realized (Mulder, 1996). In this paper we will consider Lagrange triangular finite elements P_k (and their lumped versions up to P_5) in order to compare high and low-order methods. The question we want to answer is as follows: Does the superior accuracy of high-order finite element methods permit a sufficient reduction in the number of degrees of freedom to balance its higher cost, especially if this cost can be limited by the use of high-order mass-lumping?

In the first part of this paper, we show how it is possible to automatically generate the mass and stiffness matrices at any order using symbolic calculation software in order to implement high-order schemes with less effort. In Section 2, we expose a method which generalizes the works (Cohen *et al.*, 2001) and (Tordjman, 1995), for determining high-order symmetric positive definite quadrature formulas. Our approach to this construction is slightly different from those of (Chin-Joe-Kong *et al.*, 1999) as it allows for a systematic minimization of the number of quadrature points. Since a high-order discretization in space has to be coupled with a high-order discretization in time in order not to lose its accuracy and convergence rate, in the third part we discuss two high-order discretizations in time: one explicit and the diagonally implicit Runge-Kutta method (DIRK). The final sections are dedicated to numerical results and to conclusions to our study.

2. Arbitrary High-Order Lagrange Finite Elements

2.1. Finite Element Method and its Implementation. To illustrate our aims, we first consider the homogeneous wave equation with a vanishing Dirichlet boundary and initial conditions. Let $\Omega \in \mathbb{R}^2$ be an open bounded and polygonal domain with boundary $\Gamma = \partial\Omega$. The goal is the following: Find u: $\Omega \times \mathbb{R}^+ \to \mathbb{R}$ such that

$$\begin{cases} \partial_t^2 u - \triangle u = f, & (x,t) \in \Omega \times \mathbb{R}^+, \\ u(x,0) = u_0(x), & x \in \Omega, \\ \partial_t u(x,0) = u_1(x), & x \in \Omega, \\ u(x,t) = 0, & (x,t) \in \Gamma \times \mathbb{R}^+. \end{cases}$$

Let T_h be a mesh (triangulation) of Ω. We denote by $H^1(\Omega)$ the first-order Sobolev space of square integrable functions whose first derivatives (in the distribution sense) are also square integrable and by $H_0^1(\Omega)$ the first-order Sobolev space with vanishing boundary conditions. We denote by $\{K\}$ the set of triangles and by $V_h = \{\psi \in C^0(\Omega) \mid \forall i, \psi_{|K_i} \in P_k\} \cap H_0^1(\Omega)$ the k-th order Lagrange finite element approximation subspace of continuous functions associated with T_h. We recall that P_k is a $\frac{1}{2}(k+1)(k+2)$ dimensional space which consists of polynomials of degrees less than or equal to k for which it is possible to form a basis $\{\psi_i\}$ such that $\psi_i(a_j) = \delta_{ij}$, where $\{a_i\}$ are the degrees of freedom on a triangle. The problem can be rewritten in a variational form (where we have neglected the initial condition): Find $u_h(t) : \mathbb{R}^+ \to V_h$ such that

$$\frac{\mathrm{d}^2}{\mathrm{d}t^2} \int_\Omega u_h(x,t) v_h(x)\,\mathrm{d}x + \int_\Omega \nabla u_h(x,t) \cdot \nabla v_h(x)\,\mathrm{d}x = \int_\Omega f(x,t) v_h(x)\,\mathrm{d}x, \quad \forall v_h \in V_h.$$

Let $\{a_i\}$ be the set of points on which we define the Lagrange degrees of freedom (i.e., over the whole triangulation) and $\{\psi_i\}$ the set of the associated basis functions. If U denotes the vector containing the coordinates of u_h in the basis $\{\psi_i\}$, and $(F(t))_i = f(a_i,t)$, the projection on the finite element space of the external force, then the problem is equivalent to the system of ordinary differential equations:

$$M\frac{\mathrm{d}^2}{\mathrm{d}t^2}U(t) + KU(t) = MF(t),$$

where the mass and stiffness matrices, respectively denoted by M and K, are defined by

$$\begin{cases} M_{ij} = \displaystyle\int_\Omega \psi_i(x)\psi_j(x)\,\mathrm{d}x, \\ K_{ij} = \displaystyle\int_\Omega \nabla\psi_i(x).\nabla\psi_j(x)\,\mathrm{d}x. \end{cases}$$

In order to implement the method, we rewrite these matrices in the form

$$M_{ij} = \sum_K \int_K \psi_i(x,y)\psi_j(x,y)\,\mathrm{d}x\,\mathrm{d}y,$$
$$K_{ij} = \sum_K \int_K \nabla\psi_i(x,y).\nabla\psi_j(x,y)\,\mathrm{d}x\,\mathrm{d}y.$$

It is now possible to compute the integrals on each triangle using the integrals on a reference triangle and an affine mapping which transforms one into the other. More precisely, consider a triangle whose vertices $S(i)$, $i = 1,\ldots,3$, have coordinates (x_i, y_i) and the reference element $\hat{K}$ whose vertices have coordinates $(0,0)$, $(1,0)$ and $(0,1)$. The affine mapping

$$F(\hat{x},\hat{y}) = A\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} + b$$

with

$$A = \begin{pmatrix} x_2 - x_1 & x_3 - x_1 \\ y_2 - y_1 & y_3 - y_1 \end{pmatrix}, \qquad b = \begin{pmatrix} x_1 \\ y_1 \end{pmatrix}$$

transforms $\hat{K}$ into K. Thus, by the change of variables $(x,y) = F(\hat{x},\hat{y})$, we get

$$\begin{aligned} M_{ij} &= \int_K \psi_i(x,y)\psi_j(x,y)\,\mathrm{d}x\,\mathrm{d}y \\ &= \int_{\hat{K}} \psi_i(F(\hat{x},\hat{y}))\psi_j(F(\hat{x},\hat{y}))(\det A)\,\mathrm{d}\hat{x}\,\mathrm{d}\hat{y} \\ &= (\det A)\int_{\hat{K}} \hat{\psi}_i(\hat{x},\hat{y})\hat{\psi}_j(\hat{x},\hat{y})\,\mathrm{d}\hat{x}\,\mathrm{d}\hat{y} \end{aligned}$$

and

$$\begin{aligned} K_{ij} &= \int_K \nabla\psi_i(x,y).\nabla\psi_j(x,y)\,\mathrm{d}x\,\mathrm{d}y \\ &= (\det A)\int_{\hat{K}} ({}^tA)^{-1}\nabla\hat{\psi}_i(\hat{x},\hat{y}) \\ &\quad \cdot({}^tA)^{-1}\nabla\hat{\psi}_j(\hat{x},\hat{y})\,\mathrm{d}\hat{x}\,\mathrm{d}\hat{y}. \end{aligned} \tag{1}$$

As

$$({}^tA)^{-1} = \frac{1}{\det A}\begin{pmatrix} y_3 - y_1 & y_1 - y_2 \\ x_1 - x_3 & x_2 - x_1 \end{pmatrix},$$

from (1) we obtain

$$\begin{aligned} K_{ij} &= \int_K \nabla\psi_i(x,y)\cdot\nabla\psi_j(x,y)\,\mathrm{d}x\,\mathrm{d}y \\ &= \frac{1}{\det A}\Big\{((y_3-y_1)^2+(x_1-x_3)^2) \\ &\quad \int_{\hat K}\partial_{\hat x}\hat\psi_i\partial_{\hat x}\hat\psi_j\,\mathrm{d}\hat x\,\mathrm{d}\hat y \\ &\quad +((y_1-y_2)^2+(x_2-x_1)^2)\int_{\hat K}\partial_{\hat y}\hat\psi_i\partial_{\hat y}\hat\psi_j\,\mathrm{d}\hat x\,\mathrm{d}\hat y \\ &\quad +((y_3-y_1)(y_1-y_2)+(x_1-x_3)(x_2-x_1)) \\ &\quad \int_{\hat K}(\partial_{\hat x}\hat\psi_i\partial_{\hat y}\hat\psi_j+\partial_{\hat y}\hat\psi_i\partial_{\hat x}\hat\psi_j)\,\mathrm{d}\hat x\,\mathrm{d}\hat y\Big\}. \end{aligned}$$

It becomes now clear that it is possible to calculate the global mass and stiffness matrices from the local mass and stiffness matrices calculated on a reference triangle, i.e., from the matrices

$$\int_{\hat K}\hat\psi_i(\hat x,\hat y)\hat\psi_j(\hat x,\hat y)\,\mathrm{d}\hat x\,\mathrm{d}\hat y,$$

$$\int_{\hat K}\partial_{\hat x}\hat\psi_i\partial_{\hat x}\hat\psi_j\,\mathrm{d}\hat x\,\mathrm{d}\hat y,$$

$$\int_{\hat K}\partial_{\hat y}\hat\psi_i\partial_{\hat y}\hat\psi_j\,\mathrm{d}\hat x\,\mathrm{d}\hat y,$$

$$\int_{\hat K}(\partial_{\hat x}\hat\psi_i\partial_{\hat y}\hat\psi_j+\partial_{\hat y}\hat\psi_i\partial_{\hat x}\hat\psi_j)\,\mathrm{d}\hat x\,\mathrm{d}\hat y.$$

2.2. Automatic Generation of the Basis Functions. Let us consider the polynomial function space P_k. We now want to automatically generate the matrices which are necessary to assemble the global matrices. It is then enough to notice that to compute these matrices, we only need to know the basis functions on the reference element. Once these functions are known, the matrices we need can be computed using any symbolic computation software (e.g., $Maple^{©}$). First we have to choose a relative classification of $\frac{1}{2}(k+1)(k+2)$ degrees of freedom per triangle.

We know that the degrees of freedom are located on a regular lattice of the triangle (see Fig. 2). As we only know the vertex coordinates, we shall express the degrees of freedom in terms of barycentric coordinates, and choose a classification as symmetric as possible (symmetric in the triangle). The choice we made is exposed in Fig. 1. This choice makes the enumeration of the degrees of freedom by three indices possible: one for the sub-triangle (m), one for the edge (i) of the m-th sub-triangle and one (j) for the j-th degree of freedom located on the i-th edge of the m-th sub-triangle. It is now possible to locate the degrees of freedom by iterating. We first define $\tilde k$ as the integer part of $k/3$. Then the algorithm is as follows:

for m from 0 to $\tilde k$,
  for j from 0 to $k-(3m+1)$,
    for i from 1 to 3,
      the ξ-th degree of freedom a_ξ is located by

$$a_\xi = \frac{mS(i-1)+(k-2m-j)S(i)+(j+m)S(i+1)}{k},$$

where $\{S(i), i=1,2,3\}$ are the three vertices to which we add $S(0)=S(3)$ and $S(4)=S(1)$, and where the number ξ of the degree of freedom, which, of course, depends on k, m, j and i, is given by

$$\xi(k,m,j,i) = \left(3k - \frac{9(m-1)}{2}\right)m + 3j + i.$$

Remark 1. If $k \equiv 0[3]$, then we cannot do exactly the same because the last sub-triangle is in fact a single degree of freedom, the center of gravity (which corresponds to the last degree of freedom in our classification), so that we only need to use m from 0 to $\tilde k - 1$ and define by hand

$$a_{\frac{(k+1)(k+2)}{3}} = \frac{S(1)+S(2)+S(3)}{3}.$$

In the same manner we define the associated basis functions. By $\{\lambda_i, i=1,2,3\}$ we denote the three barycentric functions associated with the three vertices. The basis functions are given by

$$\Psi_\xi(\lambda_1,\lambda_2,\lambda_3) = \frac{\psi_\xi(\lambda_1,\lambda_2,\lambda_3)}{\psi_\xi(\lambda_1(a_\xi),\lambda_2(a_\xi),\lambda_3(a_\xi))},$$

where

$$\psi_\xi(\lambda_1,\lambda_2,\lambda_3) = \prod_{l=0}^{m-1}\prod_{n=1}^{3}\left(\lambda_n - \frac{l}{k}\right), \prod_{l=m}^{k-2m-(j+1)}\left(\lambda_i - \frac{l}{k}\right)\prod_{l=m}^{j+m-1}\left(\lambda_{i+1} - \frac{l}{k}\right),$$

and, of course, if $k \equiv 0[3]$, we define the last basis function by

$$\Psi_{\frac{(k+1)(k+2)}{2}}(\lambda_1,\lambda_2,\lambda_3) = \frac{\prod_{l=0}^{\tilde k-1}\prod_{n=1}^{3}\left(\lambda_n - \frac{l}{k}\right)}{\left(\prod_{l=1}^{\tilde k}\left(\frac{l}{k}\right)\right)^3}.$$

We thus have explicited the $\frac{1}{2}(k+1)(k+2)$ basis functions of the polynomial space P_k and it is now possible to generate automatically the four matrices we previously introduced by using any symbolic calculation software.

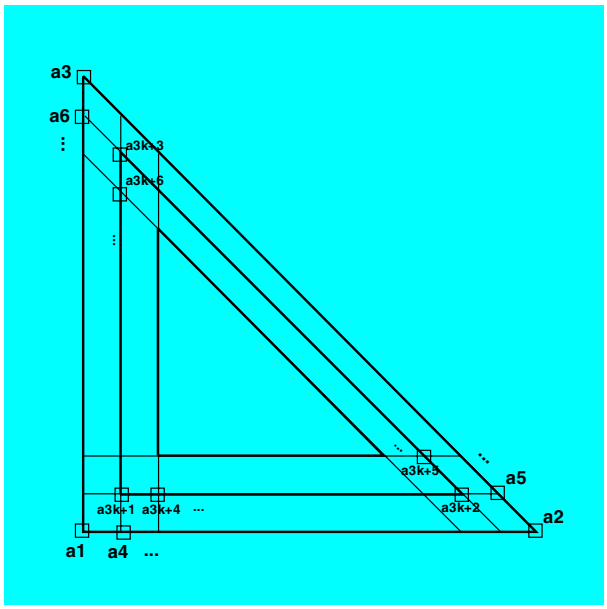

Fig. 1. Classification of degrees of freedom for P_k.

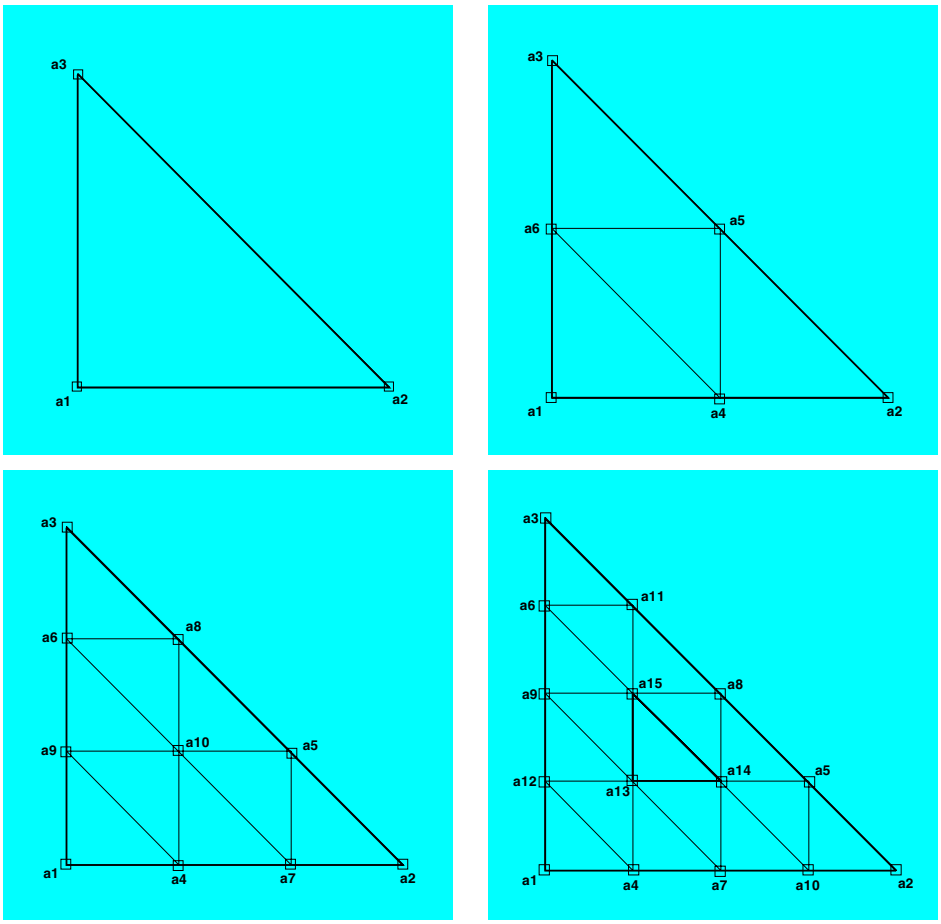

Fig. 2. Lattice of degrees of freedom for P_1 to P_4.

3. High-Order Mass-Lumping

This section recalls some important facts from the article (Cohen *et al.*, 2001). The problem when we try to use high-order methods is the increase in the computation time, especially for time-dependent problems like the wave equation. Indeed, the use of finite element methods reveals a mass matrix which has to be inverted at each time step. The idea of mass-lumping, introduced by Cohen and Joly to circumvent this problem, is to approximate this matrix by a diagonal one using a well-chosen quadrature formula. More precisely, if one can determine a quadrature formula accurate enough so as not to decrease the order of the method, then all we need to do is to put the Lagrange degrees of freedom at the quadrature points to diagonalize the mass matrix. The point is then to determine quadrature formulas in the triangle with several constraints.

3.1. Preliminary Remarks on Symmetric Quadrature Formulas in the Triangle. In this section we discuss the practical construction of quadrature formulas in the triangle. More particularly, we show how to significantly reduce the system whose unknowns determine the quadrature formula. For chosing a quadrature formula, we must choose the order of polynomials it must integrate exactly. But, generally, it is better to ask for more than this precision. For example, in one space dimension on the segment $[-1,1]$, knowing that any odd function has an integral equal to zero, one should require the numerical approximation of the integral of such functions to be zero. This is done very easily by using symmetric quadrature formulas. This means that the quadrature points are symmetrically located in the interval $[-1,1]$ and that the weights of two symmetric points are the same. Another remarquable property of symmetric quadrature formulas is that the numerical approximations of the integrals of symmetric functions (i.e., functions f and g such that $g(x) = f(-x), \quad \forall x \in [-1,1]$) are equal. It is exactly this idea of symmetry we will generalize on the triangle.

We now want to construct a quadrature formula which integrates exactly all polynomials in P_k. By now, $x = (x_1, x_2)$ denotes a variable of $\mathbb{R}^2$ and K a triangle whose vertices are S_1, S_2 and S_3. Let $\Lambda_1(x)$, $\Lambda_2(x)$ and $\Lambda_3(x)$ be three polynomials in P_1 such that $\Lambda_i(S_j) = \delta_{ij}$. The main result is that the set $B(P_k)$ defined by

$$B(P_k) = \{\Lambda_1^l(x)\Lambda_2^m(x)\Lambda_3^n(x) \mid 0 \le l,m,n \le k, \quad l+m+n = k\}$$

is a basis of P_k (it is quite easy to prove that the family forming $B(P_k)$ is free and of cardinality equal to the dimension of P_k). This implies that a given function $f \in P_k$ can be expressed by a unique linear combination of functions in $B(P_k)$, i.e., that there exists a unique function $\hat{f}$ such that

$$f(x) = \hat{f}(\Lambda_1(x), \Lambda_2(x), \Lambda_3(x)), \qquad \forall x \in \mathbb{R}^2.$$

Let $\mathbb{S}_3$ be the permutation group on $\{1,2,3\}$. Since

$$\int_K \Lambda_i(x)\,\mathrm{d}x = \int_K \Lambda_{\sigma(i)}(x)\,\mathrm{d}x, \quad \forall i = 1,2,3 \text{ and } \forall \sigma \in \mathbb{S}_3,$$

one can see that for every $\sigma \in \mathbb{S}_3$ and for every $(l,m,n) \in \mathbb{N}^3$

$$\int_K \Lambda_1^l(x)\Lambda_2^m(x)\Lambda_3^n(x)\,\mathrm{d}x = \int_K \Lambda_{\sigma(1)}^l(x)\Lambda_{\sigma(2)}^m(x)\Lambda_{\sigma(3)}^n(x)\,\mathrm{d}x,$$

and thus for any $\sigma \in \mathbb{S}_3$

$$\int_K f(x)\,\mathrm{d}x = \int_K \hat{f}(\Lambda_1(x), \Lambda_2(x), \Lambda_3(x))\,\mathrm{d}x = \int_K \hat{f}(\Lambda_{\sigma(1)}(x), \Lambda_{\sigma(2)}(x), \Lambda_{\sigma(3)}(x))\,\mathrm{d}x.$$

We shall require the quadrature formula to verify this symmetry property (2).

It is well known that any point $(x_1, x_2) \in K$ can be defined by its three barycentric coordinates $(\lambda_1, \lambda_2, \lambda_3) \in \mathbb{R}_+^3$ relative to the three vertices S_1, S_2 and S_3, and that this localisation is unique since $\lambda_1 + \lambda_2 + \lambda_3 = 1$. By now two points $(x_1, x_2) \in K$ and $(\hat{x}_1, \hat{x}_2) \in K$, located respectively by the barycentric coordinates $(\lambda_1, \lambda_2, \lambda_3)$ and $(\hat{\lambda}_1, \hat{\lambda}_2, \hat{\lambda}_3)$, will be said symmetric if there exists $\sigma \in \mathbb{S}_3$ such that

$$(\lambda_1, \lambda_2, \lambda_3) = (\hat{\lambda}_{\sigma(1)}, \hat{\lambda}_{\sigma(2)}, \hat{\lambda}_{\sigma(3)}).$$

Now it becomes clear that any quadrature formula I_K defined by a symmetric set of quadrature points Q, i.e., a set such that

$$(\lambda_1, \lambda_2, \lambda_3) \in Q \Rightarrow (\lambda_{\sigma(1)}, \lambda_{\sigma(2)}, \lambda_{\sigma(3)}) \in Q, \ \forall \sigma \in \mathbb{S}^3,$$

and weights satisfying

$$w(\lambda_1, \lambda_2, \lambda_3) = w(\lambda_{\sigma(1)}, \lambda_{\sigma(2)}, \lambda_{\sigma(3)}), \qquad \forall \sigma \in \mathbb{S}^3,$$

where $w(\lambda_1, \lambda_2, \lambda_3)$ corresponds to the weight associated to the point $(\lambda_1, \lambda_2, \lambda_3)$, satisfies the symmetry condition (2), i.e.,

$$I_K(\hat{f}(\Lambda_1(x), \Lambda_2(x), \Lambda_3(x))) = I_K(\hat{f}(\Lambda_{\sigma(1)}(x), \Lambda_{\sigma(2)}(x), \Lambda_{\sigma(3)}(x))), \quad \forall \sigma \in \mathbb{S}_3.$$

So, the only property we have to verify is

$$I_K(\Lambda_i(x)) = I_K(\Lambda_{\sigma(i)}(x)), \quad \forall \sigma \in \mathbb{S}_3, i = 1,2,3,$$

which is trivial, and which implies that

$$I_K(\Lambda_1^l(x)\Lambda_2^m(x)\Lambda_3^n(x)) = I_K(\Lambda_{\sigma(1)}^l(x)\Lambda_{\sigma(2)}^m(x)\Lambda_{\sigma(3)}^n(x)), \quad \forall\sigma \in \mathbb{S}_3, \forall(l,m,n) \in \mathbb{N}^3.$$

Such quadrature formulas will be said symmetric. If now we want to ensure that a symmetric quadrature formula is exact for all polynomials in P_k, all we have to do is to ensure that any basis function $\Lambda_1^l(x)\Lambda_2^m(x)\Lambda_3^n(x) \in B(P_k)$ is exactly integrated. But since

$$I_K(\Lambda_1^l(x)\Lambda_2^m(x)\Lambda_3^n(x)) = I_K(\Lambda_{\sigma(1)}^l(x)\Lambda_{\sigma(2)}^m(x)\Lambda_{\sigma(3)}^n(x)), \quad \forall\sigma \in \mathbb{S}_3, \Lambda_1^l(x)\Lambda_2^m(x)\Lambda_3^n(x) \in B(P_k),$$

it is trivial that we do not need to verify the exact integration for all $\frac{1}{2}(k+1)(k+2)$ polynomials in $B(P_k)$ but only for one representant of each equivalence class in $B(P_k)/\sim$, where, of course, the equivalence relation $\sim$ is defined by

$$\Lambda_1^l(x)\Lambda_2^m(x)\Lambda_3^n(x) \sim \Lambda_1^{\hat{l}}(x)\Lambda_2^{\hat{m}}(x)\Lambda_3^{\hat{n}}(x) \iff \exists\sigma \in \mathbb{S}_3, \Lambda_1^l(x)\Lambda_2^m(x)\Lambda_3^n(x) = \Lambda_{\sigma(1)}^{\hat{l}}(x)\Lambda_{\sigma(2)}^{\hat{m}}(x)\Lambda_{\sigma(3)}^{\hat{n}}(x).$$

3.2. Mass-Lumping Implementation. We will now discuss how the use of quadrature formulas will help us to diagonalize the mass matrix and what other restrictions they must satisfy in order not to decrease the order of the method. We have already seen that the finite element method reveals a mass matrix defined by

$$M_{ij} = \int_\Omega \psi_i(x)\psi_j(x)\,\mathrm{d}x.$$

If now, rather than exactly calculate the integrals, we evaluate them by using quadrature formulas:

$$(u,v)_h \equiv \sum_l w_l u(a_l)v(a_l),$$

where $\{a_l\}$ denotes the set of quadrature points over the whole domain and $\{w_l\}$ is the set of the associated quadrature weights, one can see that the new mass matrix will be diagonal as soon as the degrees of freedom are set as the quadrature points. Indeed, each basis function ψ_i (associated with the new degrees of freedom) will be nonzero only on one degree of freedom, so that we will have

$$\tilde{M}_{ij} := (\psi_i,\psi_j)_h = \textstyle\sum_l w_{l,h}\psi_i(a_l)\psi_j(a_l) = 0,\ \forall i \neq j.$$

This means that all of the terms not located on the diagonal of the matrix will be zero.

Accordingly, the crux of mass-lumping is in the determination of quadrature formulas. By modifying P_k (the space of standard Lagrange polynomials of order k) in $\tilde{P}_k$ in the following way:

$$P_k \subseteq \tilde{P}_k \subset P_{k'}, \quad k \leq k',$$

we get the same accuracy with $\tilde{P}_k$ as with standard P_k elements if the quadrature formula is exact in $P_{k+k'-2}$ (Fix, 1972). Thus we shall construct spaces $\tilde{P}_k$ and the associated integration formulas using the following guidelines:

- the space $\tilde{P}_k$ should be as small as possible with $P_k \subseteq \tilde{P}_k \subset P_{k'}$,
- the set of quadrature points should be $\tilde{P}_k$-unisolvent,
- the number of degrees of freedom on the edges should be sufficient to ensure the H^1-conformity,
- the quadrature weights should be strictly positive.

The first condition is for efficiency and aims at minimizing the total number of degrees of freedom. The next two conditions are purely mathematical and related to the fact that the degrees of freedom will be set as the quadrature points, and the last consideration is related to stability (Tordjman, 1995). We will now explain how we use all these considerations to construct new finite element spaces and quadrature formulas, which allows for mass-lumping up to 5-th order schemes.

3.3. Practical Construction of Lumped Finite Elements. The first things we have to discuss in this section are the different equivalence classes the symmetric condition allows us to consider and the number of parameters their use implies. The equivalence classes of points located on the edges are only of three types:

- The first type is, of course, the class made by the **three vertices** S_i. This class can be uniquely defined by a single parameter w_s which corresponds to the weight associated with the three points.
- The second type is the class made by the **three midpoints of the edges** M_i, to which we associate a weight parameter w_m.
- The last type is made by a set of **six points** M_{ij} located by a parameter α. More precisely, $M_{ij}(\alpha)$ **is the barycenter of the two vertices** S_i **and** S_j **respectively weighted by** α **and** $1-\alpha$, where $\alpha \in]0,1/2[$. This equivalence class is uniquely defined by two parameters, the weight w_α associated with the six points and the localization parameter α (see Fig. 3);

The three equivalence classes of points located in the interior of the triangle are:

- The class made by the single **barycenter**, weighted by a parameter w_g.

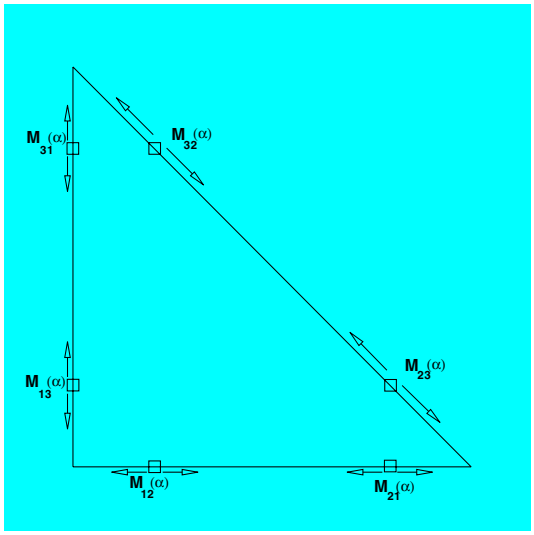

Fig. 3. Six-point equivalence class on the edge.

- The class made by **three points** G_i **located at the medians** by a parameter β. Then, $G_i(\beta)$ **is the barycenter of one of the vertex** S_i **weighted by** β **and the other two** S_j $(j \neq i)$ **weighted by** $\frac{1-\beta}{2}$, where $\beta \in]0,1[$ and $\beta \neq \frac{1}{3}$, in order not to degenerate into the barycenter. Such an equivalence class is thus defined by a weight parameter w_β and the localization parameter β (see Fig. 4).

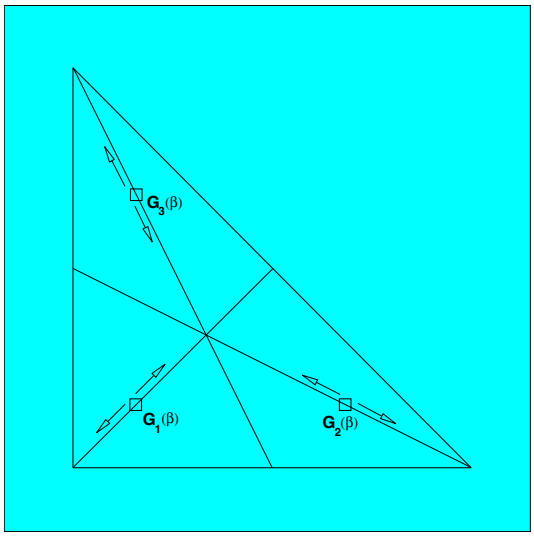

Fig. 4. Three-point equivalence class in the interior.

- The last class made by six interior points G_{ij} located this time by **two parameters** ω_1 and ω_2 in $]0,1[$ where $\omega_1 \neq \omega_2$ (in order not to degenerate into one of the two classes already mentioned). The $G_{ij}(\omega_1,\omega_2)$ **are the barycenters of the three vertices** S_i**,** S_j **and** S_k **respectively weighted by** ω_1**,** ω_2 **and** $1-\omega_1-\omega 2$. Thus this equivalence class is defined by three parameters, a weight w_ω, and two localization parameters ω_1 and ω_2 (see Fig. 5).

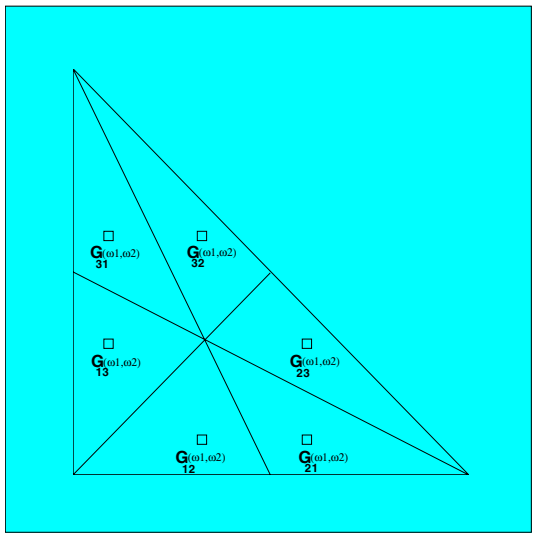

Fig. 5. Six-point equivalence class in the interior.

In order to lump P_k finite elements, we choose to consider a polynomial space $\tilde{P}_k$ such that $P_k \subseteq \tilde{P}_k \subset P_{k'}$. Once $\tilde{P}_k$ is fixed, the number of quadrature points to use for the associated quadrature formula is determined, as it corresponds to the dimension of the polynomial space. Thus we can divide the quadrature points into a certain number of equivalence classes and exhibit the associated formal quadrature formula as well as its number of parameters. Up to now, we still do not know suitable $\tilde{P}_k$. Actually, we have to verify the conforming condition. It leads us to consider a set of symmetric points with exactly as many points on the edges as standard P_k has (i.e., $3k$), in order to maintain the P_k continuity. This leads us to a more precise polynomial space determination: if we want to ensure the uniqueness, we have to enrich P_k by polynomials of orders greater than k which are identically zero on the edges. This can be written by

$$\tilde{P}_k = P_k + bP_{k'},$$

where b is the so-called bubble function $\lambda_1\lambda_2\lambda_3$ which vanishes on the edges of the triangle. We underline that this last expression is well defined for $k \geq 1$ and $k' \geq k-2$, and that

$$P_k \subset \tilde{P}_k \subset P_{k'+3}.$$

Another way to enrich the polynomial space is, e.g., to consider

$$\tilde{P}_k = P_k + b^2P_{k'},$$

which is well defined for $k \geq 4$ and $k' \geq k-5$, and such that

$$P_k \subset \tilde{P}_k \subset P_{k'+6}.$$

It is quite easy to generalize this construction when k is large enough. We can also combine these two constructions by paying attention to the fact that the spaces are well defined. For example,

$$\tilde{P}_k = P_k + b(P_{k'} + bP_{k''}),$$

or

$$\tilde{P}_k = P_k + b(P_{k'} + b^2P_{k''}),$$

or

$$\tilde{P}_k = P_k + b^2(P_{k'} + bP_{k''}).$$

Now we can give our practical approach to lump the P_k elements, which is the following:

- we first choose a polynomial space $\tilde{P}_k$ between P_k and $P_{k'}$,
- we determine the number of parameters for the quadrature formula (weights and localization parameters) in order to be accurate enough,
- we formally try to construct a symmetric quadrature formula using a symmetric set of quadrature points which involves this number of parameters,
- we determine the parameters of the quadrature formula by solving a polynomial system.

The problems that can appear are only of two types:

- the polynomial space we consider does not permit us to get enough parameters for the construction of the quadrature formula, i.e., the set of equivalence classes Q made by the points associated with the polynomial space does not permit us to get enough weights and localization parameters,
- the polynomial system does not have any suitable solution.

If we encounter one of these problems, we must consider a larger polynomial space and go over the whole procedure.

The initial choice of the polynomial space is, of course, $\tilde{P}_k = P_k$, and for this choice we have to determine a quadrature formula exact for polynomials up to the order $2k-2$. If this is not possible, we have to enrich $\tilde{P}_k$. To do this, we exhibit all of the spaces $\tilde{P}_k$ previously described such that

$$P_k \subset \tilde{P}_k \subset P_{k+1},$$

and try for each one to determine a quadrature formula which is exact for polynomials up to order $k+(k+1)-2$. If this is not possible, we continue this procedure by considering greater k' and all of the spaces $\tilde{P}_k$ such that

$$P_k \subset \tilde{P}_k \subset P_{k'}.$$

We point out that the number of quadrature points we have to use corresponds to the dimension of the space considered.

The number of degrees of freedom which must appear in the quadrature formula is determined by calculating the number $\tilde{M}$ of equivalence classes in $B(P)$, where P denotes the symmetric polynomials of order $k+k'-2$.

Once we know the number of parameters we need, we can see if it is possible to construct a quadrature formula which makes this number of parameters appear, with the given number of quadrature points. If it is, the formal shape of the quadrature formula is known, and all we have to do is determine the parameters by solving the polynomial system composed by the equalities between the evaluation of the formal quadrature formula and the exact integration of one representant of each equivalence class $\tilde{C}_\beta$ of the polynomials the quadrature formula has to integrate exactly.

3.4. Examples. The use of our approach to construct lumped spaces has given the same results for the lumped P1 to lumped P4 as the spaces already known (see (Cohen *et al.*, 2001; Chin-Joe-Kong *et al.*, 1999) for details of these spaces). We give directly the lumped P5 construction and our progression towards the lumped P6 space.

3.4.1. Lumped P5 Calculation. We first try, *a priori*, $k'=5$ and thus P_5 itself:

$$\tilde{P}_5 = P_5.$$

We need to construct a twenty-one-point quadrature formula ($\frac{6\times 7}{2}$), exact up to order eight ($2\times 5-2$), for which we have ten parameters (corresponding to the ten equivalence classes):

$$\begin{aligned}
&order = 8,\\
C_1 &= \{\lambda_1^8, \lambda_2^8, \lambda_3^8\},\\
C_2 &= \{\lambda_1^7\lambda_2, \lambda_1^7\lambda_3, \lambda_2^7\lambda_1, \lambda_2^7\lambda_3, \lambda_3^7\lambda_1, \lambda_3^7\lambda_2\},\\
C_3 &= \{\lambda_1^6\lambda_2^2, \lambda_1^6\lambda_3^2, \lambda_2^6\lambda_1^2, \lambda_2^6\lambda_3^2, \lambda_3^6\lambda_1^2, \lambda_3^6\lambda_2^2\},\\
C_4 &= \{\lambda_1^6\lambda_2\lambda_3, \lambda_2^6\lambda_1\lambda_3, \lambda_3^6\lambda_1\lambda_2\},\\
C_5 &= \{\lambda_1^5\lambda_2^3, \lambda_1^5\lambda_3^3, \lambda_2^5\lambda_1^3, \lambda_2^5\lambda_3^3, \lambda_3^5\lambda_1^3, \lambda_3^5\lambda_2^3\},\\
C_6 &= \{\lambda_1^5\lambda_2^2\lambda_3, \lambda_1^5\lambda_3^2\lambda_2, \lambda_2^5\lambda_1^2\lambda_3,\\
&\quad \lambda_2^5\lambda_3^2\lambda_1, \lambda_3^5\lambda_1^2\lambda_2, \lambda_3^5\lambda_2^2\lambda_1\},\\
C_7 &= \{\lambda_1^4\lambda_2^4, \lambda_1^4\lambda_3^4, \lambda_2^4\lambda_3^4\},\\
C_8 &= \{\lambda_1^4\lambda_2^3\lambda_3, \lambda_1^4\lambda_3^3\lambda_2, \lambda_2^4\lambda_1^3\lambda_3, \lambda_2^4\lambda_3^3\lambda_1,\\
&\quad \lambda_3^4\lambda_1^3\lambda_2, \lambda_3^4\lambda_2^3\lambda_1\},\\
C_9 &= \{\lambda_1^4\lambda_2^2\lambda_3^2, \lambda_2^4\lambda_1^2\lambda_3^2, \lambda_3^4\lambda_1^2\lambda_2^2\},\\
C_{10} &= \{\lambda_1^3\lambda_2^3\lambda_3^2, \lambda_1^3\lambda_3^3\lambda_2^2, \lambda_2^3\lambda_3^3\lambda_1^2\}.
\end{aligned}$$

We notice that we cannot construct a ten-degrees-of-freedom quadrature formula: we will, at best, only have nine parameters:

- five weights w_s, w_{α_1}, w_{α_2}, w_{β_1}, w_{β_2} associated respectively with the three vertices, six points on the edges (twice) and three interior points (twice),
- four localization parameters α_1, α_2, β_1 and β_2 associated with the six points on the edges (twice) and the three interior points (twice).

Then, we consider $k' = 6$, and all of the spaces $\tilde{P}_5$ such that

$$P_5 \subset \tilde{P}_5 \subset P_6.$$

The only solutions are

$$\tilde{P}_5 = P_5 + bP_3 \quad \text{and} \quad \tilde{P}_5 = P_5 + b^2P_0.$$

We now have to determine a ninth-order quadrature formula $(5 + 6 - 2)$ with twelve degrees of freedom (corresponding to twelve equivalence classes):

$$\begin{aligned}
&order = 9,\\
C_1 &= \{\lambda_1^9, \lambda_2^9, \lambda_3^9\},\\
C_2 &= \{\lambda_1^8\lambda_2, \lambda_1^8\lambda_3, \lambda_2^8\lambda_1, \lambda_2^8\lambda_3, \lambda_3^8\lambda_1, \lambda_3^8\lambda_2\},\\
C_3 &= \{\lambda_1^7\lambda_2^2, \lambda_1^7\lambda_3^2, \lambda_2^7\lambda_1^2, \lambda_2^7\lambda_3^2, \lambda_3^7\lambda_1^2, \lambda_3^7\lambda_2^2, \},\\
C_4 &= \{\lambda_1^7\lambda_2\lambda_3, \lambda_2^7\lambda_1\lambda_3, \lambda_3^7\lambda_1\lambda_2\},\\
C_5 &= \{\lambda_1^6\lambda_2^3, \lambda_1^6\lambda_3^3, \lambda_2^6\lambda_1^3, \lambda_2^6\lambda_3^3, \lambda_3^6\lambda_1^3, \lambda_3^6\lambda_2^3\},\\
C_6 &= \{\lambda_1^6\lambda_2^2\lambda_3, \lambda_1^6\lambda_3^2\lambda_2, \lambda_2^6\lambda_1^2\lambda_3, \lambda_2^6\lambda_3^2\lambda_1,\\
&\quad \lambda_3^6\lambda_1^2\lambda_2, \lambda_3^6\lambda_2^2\lambda_1\},\\
C_7 &= \{\lambda_1^5\lambda_2^4, \lambda_1^5\lambda_3^4, \lambda_2^5\lambda_1^4, \lambda_2^5\lambda_3^4, \lambda_3^5\lambda_1^4, \lambda_3^5\lambda_2^4\},\\
C_8 &= \{\lambda_1^5\lambda_2^3\lambda_3, \lambda_1^5\lambda_3^3\lambda_2, \lambda_2^5\lambda_1^3\lambda_3, \lambda_2^5\lambda_3^3\lambda_1,\\
&\quad \lambda_3^5\lambda_1^3\lambda_2, \lambda_3^5\lambda_2^3\lambda_1\},\\
C_9 &= \{\lambda_1^5\lambda_2^2\lambda_3^2, \lambda_2^5\lambda_1^2\lambda_3^2, \lambda_3^5\lambda_1^2\lambda_2^2\},\\
C_{10} &= \{\lambda_1^4\lambda_2^4\lambda_3, \lambda_1^4\lambda_3^4\lambda_2, \lambda_2^4\lambda_3^4\lambda_1\},\\
C_{11} &= \{\lambda_1^4\lambda_2^3\lambda_3^2, \lambda_1^4\lambda_3^3\lambda_2^2, \lambda_2^4\lambda_1^3\lambda_3^2, \lambda_2^4\lambda_3^3\lambda_1^2,\\
&\quad \lambda_3^4\lambda_1^3\lambda_2^2, \lambda_3^4\lambda_2^3\lambda_1^2\},\\
C_{12} &= \{\lambda_1^3\lambda_2^3\lambda_3^3\}.
\end{aligned}$$

The dimension of the space $\tilde{P}_5 = P_5 + bP_3$ is twenty five. Thus we have to construct a quadrature formula using twenty five points knowing that fifteen of them are on the edges. According to our approach, the most natural way to consider the set of points is the following and it only gives us eleven degrees of freedom:

- six weights w_s, w_{α_1}, w_{α_2}, w_{β_1}, w_ω and w_g respectively associated with the three vertices, six points on the edges (twice), three interior points on the medians, six other interior points, and the barycenter,
- five localization parameters α_1, α_2, β_1, ω_1 and ω_2 respectively associated with the six points on the edges (twice), three interior points on the medians and six other interior points.

Once again we do not have enough parameters to construct the requested ninth-order quadrature formula. The dimension of the space $\tilde{P}_5 = P_5 + b^2P_0$ is twenty two. One can see that the twenty two resulting points will give us only ten parameters for the quadrature points, which is less than the twelve needed (six weights and four localisation parameters).

We then consider $k' = 7$ and the following space between P_5 and P_7:

$$\begin{aligned}
\tilde{P}_5 &= P_5 + b^2P_1,\\
\tilde{P}_5 &= P_5 + b(P_3 + bP_1),\\
\tilde{P}_5 &= P_5 + bP_4.
\end{aligned}$$

We must now construct a quadrature formula with fourteen degrees of freedom: we have to integrate exactly all symmetric polynomials up to the tenth degree (5+7-2):

$$\begin{aligned}
&order = 10,\\
C_1 &= \{\lambda_1^{10}, \lambda_2^{10}, \lambda_3^{10}\},\\
C_2 &= \{\lambda_1^9\lambda_2, \lambda_1^9\lambda_3, \lambda_2^9\lambda_1, \lambda_2^9\lambda_3, \lambda_3^9\lambda_1, \lambda_3^9\lambda_2\},\\
C_3 &= \{\lambda_1^8\lambda_2^2, \lambda_1^8\lambda_3^2, \lambda_2^8\lambda_1^2, \lambda_2^8\lambda_3^2, \lambda_3^8\lambda_1^2, \lambda_3^8\lambda_2^2, \},\\
C_4 &= \{\lambda_1^8\lambda_2\lambda_3, \lambda_2^8\lambda_1\lambda_3, \lambda_3^8\lambda_1\lambda_2\},\\
C_5 &= \{\lambda_1^7\lambda_2^3, \lambda_1^7\lambda_3^3, \lambda_2^7\lambda_1^3, \lambda_2^7\lambda_3^3, \lambda_3^7\lambda_1^3, \lambda_3^7\lambda_2^3\},\\
C_6 &= \{\lambda_1^7\lambda_2^2\lambda_3, \lambda_1^7\lambda_3^2\lambda_2, \lambda_2^7\lambda_1^2\lambda_3, \lambda_2^7\lambda_3^2\lambda_1,\\
&\quad \lambda_3^7\lambda_1^2\lambda_2, \lambda_3^7\lambda_2^2\lambda_1\},\\
C_7 &= \{\lambda_1^6\lambda_2^4, \lambda_1^6\lambda_3^4, \lambda_2^6\lambda_1^4, \lambda_2^6\lambda_3^4, \lambda_3^6\lambda_1^4, \lambda_3^6\lambda_2^4\},\\
C_8 &= \{\lambda_1^6\lambda_2^3\lambda_3, \lambda_1^6\lambda_3^3\lambda_2, \lambda_2^6\lambda_1^3\lambda_3, \lambda_2^6\lambda_3^3\lambda_1,\\
&\quad \lambda_3^6\lambda_1^3\lambda_2, \lambda_3^6\lambda_2^3\lambda_1\},\\
C_9 &= \{\lambda_1^6\lambda_2^2\lambda_3^2, \lambda_2^6\lambda_1^2\lambda_3^2, \lambda_3^6\lambda_1^2\lambda_2^2\},\\
C_{10} &= \{\lambda_1^5\lambda_2^5, \lambda_1^5\lambda_3^5, \lambda_2^5\lambda_3^5\},\\
C_{11} &= \{\lambda_1^5\lambda_2^4\lambda_3, \lambda_1^5\lambda_3^4\lambda_2, \lambda_2^5\lambda_1^4\lambda_3, \lambda_2^5\lambda_3^4\lambda_1,\\
&\quad \lambda_3^5\lambda_1^4\lambda_2, \lambda_3^5\lambda_2^4\lambda_1\},\\
C_{12} &= \{\lambda_1^5\lambda_2^3\lambda_3^2, \lambda_1^5\lambda_3^3\lambda_2^2, \lambda_2^5\lambda_1^3\lambda_3^2, \lambda_2^5\lambda_3^3\lambda_1^2,\\
&\quad \lambda_3^5\lambda_1^3\lambda_2^2, \lambda_3^5\lambda_2^3\lambda_1^2\},\\
C_{13} &= \{\lambda_1^4\lambda_2^4\lambda_3^2, \lambda_1^4\lambda_3^4\lambda_2^2, \lambda_2^4\lambda_3^4\lambda_1^2\},\\
C_{14} &= \{\lambda_1^4\lambda_2^3\lambda_3^3, \lambda_2^4\lambda_3^3\lambda_1^3, \lambda_3^4\lambda_1^3\lambda_2^3\}.
\end{aligned}$$

By considering $\tilde{P}_5 = P_5 + b^2P_1$ we can only have eleven parameters (six weights and five localization parameters), and with $\tilde{P}_5 = P_5 + b(P_3 + bP_1)$ we have 12 of them (six weights and six localisation parameters).

We then consider the last manner to remain between P_5 and P_7:

$$\tilde{P}_5 = P_5 + bP_4.$$

Now, we have to use thirty points for the quadrature formula. The most natural way to consider the set of points associated with the polynomial space is the following and it gives us the required fourteen degrees of freedom (see Fig. 6):

- seven weights w_s, w_{α_1}, w_{α_2}, w_{β_1}, w_{β_2}, w_{β_3} and w_ω respectively associated with the three vertices, six points on the edges (twice), three interior points on the medians (thrice) and six other interior points,

- seven localization parameters α_1, α_2, β_1, β_2, β_3, ω_1 and ω_2 respectively associated with the six points on the edges (twice), three interior points on the medians (thrice) and six other interior points.

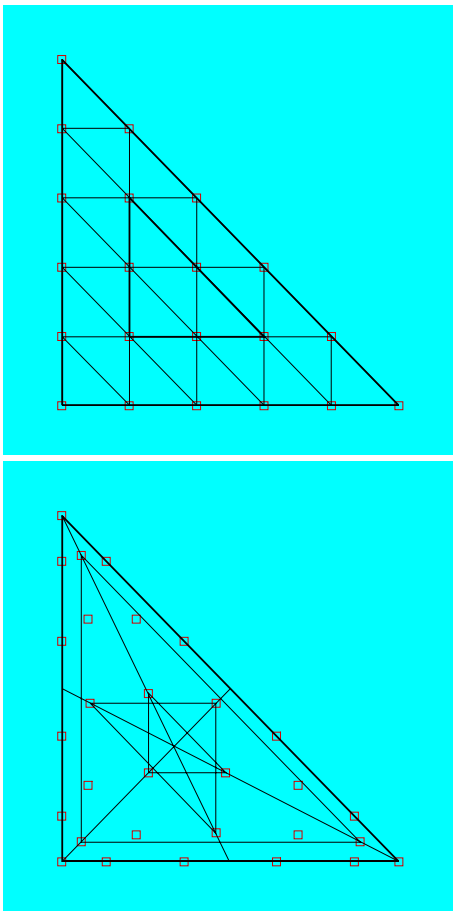

Fig. 6. From P_5 to lumped P_5 elements.

The associated quadrature formula can be given by

$$
\begin{aligned}
I_K^{app}(f) = \mathrm{mes}(K)\Big\{ & w_s \sum_{j=1}^{3} f(S_j) \\
& + w_{\alpha_1} \sum_{\substack{i,j=1 \\ i \neq j}}^{3} f(M_{ij}(\alpha_1)) \\
& + w_{\alpha_2} \sum_{\substack{i,j=1 \\ i \neq j}}^{3} f(M_{ij}(\alpha_2)) \\
& + w_{\beta_1} \sum_{j=1}^{3} f(G_j(\beta_1)) + w_{\beta_2} \sum_{j=1}^{3} f(G_j(\beta_2)) \\
& + w_{\beta_3} \sum_{j=1}^{3} f(G_j(\beta_3)) \\
& + w_{\omega} \sum_{j=1}^{6} f(G_j(\omega_1, \omega_2)) \Big\}.
\end{aligned}
$$

We solve numerically the corresponding system using $Maple^{©}$, which returns an acceptable solution:

$$
\begin{aligned}
\alpha_1 &\simeq 0.13226458163271398535388822004364 73, \\
\alpha_2 &\simeq 0.36329807415368604570550633618418 10, \\
\beta_1 &\simeq 0.88494463117717978867836496446913619, \\
\beta_2 &\simeq 0.08432632384167779612299356515186339, \\
\beta_3 &\simeq 0.48628178547608184787221833703559663, \\
\omega_1 &\simeq 0.22100121875989000797812820146484 19, \\
\omega_2 &\simeq .07819258362551702199888597846982582,
\end{aligned}
$$

and strictly positive weights:

$$
\begin{aligned}
w_s &\simeq 0.00141884794135849195859201314216756, \\
w_{\alpha_1} &\simeq 0.00696115728097842131688535505330660, \\
w_{\alpha_2} &\simeq 0.01238113000735325822823625385145974, \\
w_{\beta_1} &\simeq 0.02325227091923514227899684743112030, \\
w_{\beta_2} &\simeq 0.06906086075456558705677702806082523, \\
w_{\beta_3} &\simeq 0.09180247526152571475403829582713371, \\
w_{\omega} &\simeq 0.05455715193999251909734296553127691.
\end{aligned}
$$

We found out that this space and quadrature formula are the same as those proposed independently by in (Chin-Joe-Kong *et al.*, 1999).

3.4.2. Lumped P6 Calculation. Up to now, our attempts to obtain a lumped P6 space and the adequate quadrature formula have been unfruitful. It seems that, following our procedure, the first polynomial space which gives exactly the number of parameters needed is between P_6 and P_{11}:

$$\tilde{P}_6 = P_6 + b(P_6 + b(P_4 + bP_2)).$$

This gives us 27 parameters needed to integrate all symmetric polynomials up to the fifteenth order. The problem is that $Maple^{©}$ does not seem to be powerful enough to solve the resulting polynomial system. We also attempted unsuccessfully to solve this system by using numerical methods like conjugated gradients (we are aware, however, that such methods are very sensitive to the initialization).

4. High-Order Discretization in Time

Once we have an arbitrary high-order discretization in space, we need a suitable discretization in time in order not to decrease the overall order of the scheme. For

this purpose, we propose two solutions: one based on a Taylor expansion in time of the unknown, which can be seen as a Lax-Wendroff procedure, also known as the Cauchy-Kowalewski one (Lax and Wendroff, 1960)); and the Diagonally Implicit Runge-Kutta (DIRK) method (Butcher, 2003).

4.1. Explicit High-Order Discretization in Time. The idea of our arbitrary high discretization in time is based on a Taylor expansion in time. First, we rewrite the semi-discrete scheme in time as

$$\left\{ \begin{array}{rl} M\dot{V} + KU &= MF, \\ \dot{U} - V &= 0, \end{array} \right.$$

where U and V denote respectively the vectors of components u_i and v_i, and $\dot{U}$ and $\dot{V}$ the corresponding time derivatives. Writing $U^n = U(n\Delta t)$ and $V^n = V(n\Delta t)$, we apply a Taylor expansion in time on U and V:

$$U^{n+1} = U^n + \Delta t \dot{U}^n + \frac{\Delta t^2}{2!}\ddot{U}^n + \ldots,$$

$$V^{n+1} = V^n + \Delta t \dot{V}^n + \frac{\Delta t^2}{2!}\ddot{V}^n + \ldots,$$

and finally replace the time derivatives of U by the time derivatives of V using the first equation of the system, themselves evaluated using the second equation of the system:

$$U^{n+1} = U^n + \Delta t V^n + \frac{\Delta t^2}{2!}\dot{V}^n + \ldots,$$

$$V^{n+1} = V^n + \Delta t \underbrace{\dot{V}^n}_{F - M^{-1}KU^n} + \frac{\Delta t^2}{2!} \underbrace{\ddot{V}^n}_{\dot{F} - M^{-1}KV^n} + \ldots$$

One can remark that this discretization in time allows us to solve more general problems than the one considered until now, like problems with absorbing boundary conditions:

$$\partial_t u(x,t) + \partial_{\vec{n}} u(x,t) = 0, \quad (x,t) \in \Gamma = \partial\Omega \times \mathbb{R}^+ \ ,$$

for which we will have to solve

$$\begin{aligned} U^{n+1} &= U^n + \Delta t V^n + \frac{\Delta t^2}{2!}\dot{V}^n + \ldots \\ V^{n+1} &= V^n + \Delta t(F - M^{-1}(M_\Gamma V^n + KU^n)) \\ &\quad + \frac{\Delta t^2}{2!}(\dot{F} - M^{-1}(M_\Gamma(F - M^{-1}(M_\Gamma V^n \\ &\quad + KU^n)) + KV^n)) + \ldots, \end{aligned}$$

where

$$(M_\Gamma)_{ij} = \int_\Gamma \psi_i(x,y)\psi_j(x,y)\,\mathrm{d}x\,\mathrm{d}y.$$

At each time step, V^{n+1} and its time derivatives are determined using U^n and V^n, and are substituted in the expression for U^{n+1}. One can see that this method, which is quite easy to generalize to higher orders, has two main drawbacks. First, it is an explicit method, which implies a CFL number (the ratio between time and space steps) restriction, and second, it involves the time derivatives of the external force, which is, in general, not known analytically. For the CFL number restriction, one can ask if the use of higher-order methods balances the lower CFL number their use implies. We refer to Section 5 to see how faster high-order schemes reach a given error, even with a stronger CFL number restriction. For the external force, we computed test cases for which we know analytically the external force and its times derivatives, and compared the results when we used high-order finite differences to approximate the derivatives of the external force (high enough so as not to decrease the global order of the scheme). It seemed that the error between the two solutions was much lower than the errors between these two solutions and the exact solution.

4.2. Implicit Runge-Kutta Method. In the light of the two drawbacks of the explicit time discretization, we choose to implement another method, the so-called DIRK method (Diagonally-Implicit Runge-Kutta method). It is a particular case of the implicit Runge-Kutta method for which the coefficient matrix A has a lower triangular structure. The advantage of this method is that the intermediate stages can be evaluated sequentially rather than as one large implicit system solution. We apply this method to the ordinary differential system we have already written:

$$\left\{ \begin{array}{rl} M\dot{V} + KU &= MF, \\ \dot{U} - V &= 0. \end{array} \right.$$

This means that, given $B = (b_1, \ldots, b_s)$, $T = (t_1, \ldots, t_s)$ and the square matrix $A = (a_{ij})_{1 \le i,j \le s}$ (which is triangular), we define

$$\begin{pmatrix} V^{n+1} \\ U^{n+1} \end{pmatrix} = \begin{pmatrix} V^n \\ U^n \end{pmatrix} + \Delta t \sum_{i=1}^{s} b_i \begin{pmatrix} V_i \\ U_i \end{pmatrix},$$

where U_i and V_i are the solutions to the following problem:

$$\begin{pmatrix} \begin{pmatrix} V_1 \\ U_1 \end{pmatrix} \\ \vdots \\ \begin{pmatrix} V_s \\ U_s \end{pmatrix} \end{pmatrix} = \begin{pmatrix} N \begin{pmatrix} V^n \\ U^n \end{pmatrix} \\ \vdots \\ N \begin{pmatrix} V^n \\ U^n \end{pmatrix} \end{pmatrix}$$

$$+\Delta t\begin{pmatrix} a_{11}N & & \\ \vdots & \ddots & \\ a_{s1}N & \cdots & a_{ss}N \end{pmatrix}\begin{pmatrix} \begin{pmatrix} V_1 \\ U_1 \end{pmatrix} \\ \vdots \\ \begin{pmatrix} V_s \\ U_s \end{pmatrix} \end{pmatrix}$$

$$+\begin{pmatrix} \begin{pmatrix} F(t_1) \\ 0 \end{pmatrix} \\ \vdots \\ \begin{pmatrix} F(t_s) \\ 0 \end{pmatrix} \end{pmatrix},$$

and where

$$N=\begin{pmatrix} 0 & -M^{-1}K \\ Id & 0 \end{pmatrix}.$$

This problem can be settled in s steps by solving at each step the generic problem of the form

$$\begin{cases} (M+(a_{ii}\Delta t)^2K)U_i = M(V^n+\Delta t\sum_{j=1}^{i-1}a_{ij}V_j) \\ \qquad\qquad +a_{ii}\Delta t(MF(t_i) \\ \qquad\qquad -KU^n-\Delta t\sum_{j=1}^{i-1}a_{ij}U_j)), \\ V_i=\dfrac{1}{a_{ii}}\Delta t(U_i-(V^n+\sum_{j=1}^{i-1}a_{ij}\Delta tV_j)). \end{cases}$$

Since this time the discretization is implicit, there will not be any CFL number restriction and one can see that it does not involve any time derivatives of the external force. The main drawbacks of this method are, first, that it is not the mass matrix which has to be inverted, but a linear combination of the mass and stiffness matrices, which implies that there is no interest any more in the use of mass-lumping, and second, that the external force has to be known at the intermediate time steps t_i, which could imply the use of interpolation. We refer the reader to the next section to see that, if there is theoretically no CFL restriction, using larger CFL numbers makes the schemes very dissipative and how much this dissipation costs for long-lasting computations.

5. Numerical Results

Before we test the efficiency of the schemes, we determine, for the explicit time discretization schemes, the numerical CFL condition which ensures the convergence. For this we simply consider the following test case:

$$\begin{cases} \partial_t^2u-\triangle u=0, & (x,y,t)\in\Omega\times\mathbb{R}^+, \\ u(x,y,0)=\sin(\pi x)\sin(\pi y), & (x,y)\in\Omega, \\ \partial_tu(x,y,0)=0, & (x,y)\in\Omega, \\ u(x,y,t)=0, & (x,y,t)\in\partial\Omega\times\mathbb{R}^+, \end{cases}$$

where $\Omega=[0,1]^2$. In Fig. 7 we give two examples of the meshes we used. Note that all of the test cases, which are computed on structured meshes, were also computed on unstructured meshes and produced virtually the same results, provided that the unstructured meshes are regular enough.

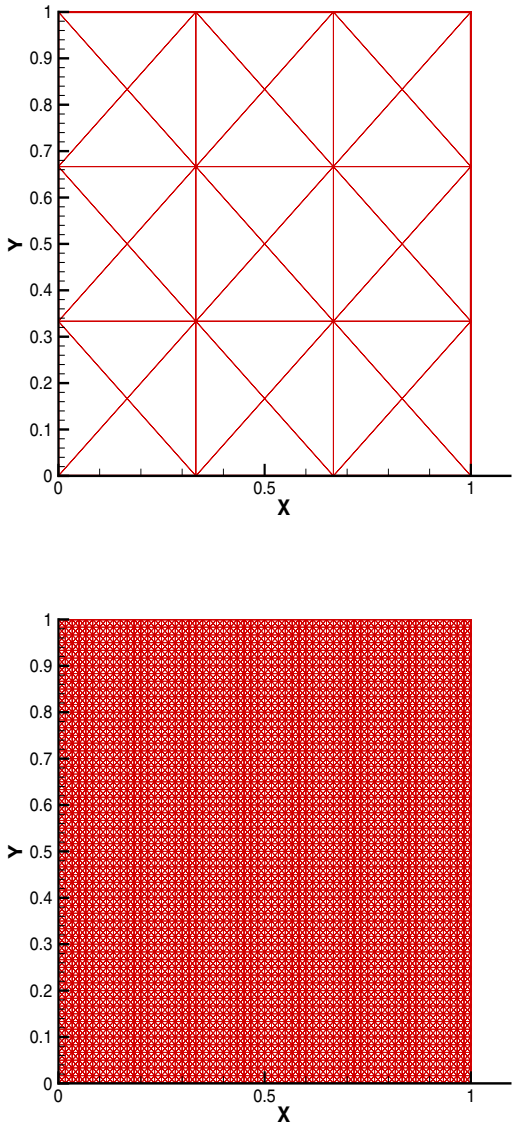

Fig. 7. Coarsest and finest meshes.

Since we know the exact solution which is given by

$$u(x,y,t)=\sin(\pi x)\sin(\pi y)\cos(\sqrt{2}\pi t),$$

we calculate the error between the numerical and exact solutions after ten periods, i.e., at $t=10\sqrt{2}$, and determine the optimal CFL number (the ratio between Δ_t and Δ_x) by increasing it until we lose convergence. These

CFL numbers are given in Table 1. Notice that increasing the space discretization order implies decreasing the CFL number (from 0.51 for P_1 elements to 0.094 for P_5 elements) and that the use of mass-lumping allows for the use of a slightly larger CFL number except for lumped P_5, which asks for much more degrees of freedom as the standard P^5 and though it yields the CFL decrease. In Figs. 8 and 9 we give the regression curves. Using these plots, we can then determine the convergence orders for the optimal CFL numbers, which are given in Table 2 and are as expected: $k+1$ for polynomials of degree k.

Since the implicit time discretization is known not to demand CFL restriction, in Fig. 10 we give plots of the regression curves (computed in the same conditions) for half, one and twice the optimal CFL numbers associated with the explicit time discretization. We calculate the convergence orders for the same CFL number as for the explicit time discretizations, and give them in Table 3. Here again, orders are $k+1$ for polynomials of degree k.

Table 1. Optimal CFL number for the explicit time discretization schemes.

Space discretization type	P_1	P_2	P_3	P_4	P_5
CFL number	0.51	0.24	0.18	0.12	0.094
Space discretization type	$\tilde{P}_1$	$\tilde{P}_2$	$\tilde{P}_3$	$\tilde{P}_4$	$\tilde{P}_5$
CFL number	1.00	0.33	0.23	0.14	0.056

Table 2. Convergence order for the explicit time discretization schemes.

Space discretization type	P_1	P_2	P_3	P_4	P_5
Order	1.96	2.99	3.97	4.95	6.03
Space discretization type	$\tilde{P}_1$	$\tilde{P}_2$	$\tilde{P}_3$	$\tilde{P}_4$	$\tilde{P}_5$
Order	2.10	3.02	4.09	4.94	5.95

Table 3. Convergence order for the implicit time discretization schemes.

Space discretization type	P_1	P_2	P_3	P_4	P_5
Order	1.98	3.00	3.98	4.94	5.98

Taking too large a CFL number makes the explicit time discretization schemes unstable. This can be verified by propagating a plane wave over Ω using periodic boundary conditions. In Fig. 11 we give an example of what happens when we take even slightly larger CFL than the optimal CFL we determined, and this for any order scheme using the explicit time discretization.

We then investigate the effects of increasing the CFL number on the implicit time discretization schemes. For a given mesh and a given scheme, we plot several pick-points of the numerical solution of the plane wave test case over two hundred periods for several CFL numbers (half, one and twice the corresponding optimal CFL calculated for the explicit time discretization). In Table 4 we

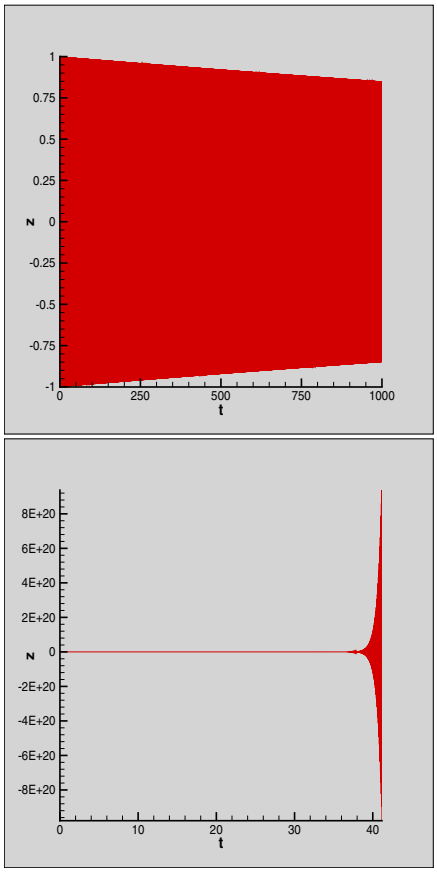

Fig. 11. Evolution of an optimal and an excessive CFL number.

give the details of the meshes we used for each space discretization and the plots are displayed in Fig. 12. There is not much influence of the CFL number on the convergence orders but the levels of errors are quite different. It seems that using an implicit scheme for getting rid of the CFL restriction is not that interesting.

We can now compare the computational effort for reaching a given error. We consider a sinusoidal signal (in fact, the fourth power of a sinusoide) propagated through periodic boundary conditions. We use regression curves to fix the logarithm of the error we want to reach and we determine the refinement of the mesh, the number of degrees of freedom and the computational time for several final times (half, two and ten periods) as well as several space and time discretizations (standard and lumped finite elements, explicit and implicit time discretizations) and CFL numbers. The results are given in Tables 5 and 6. One can see that, although the use of mass-lumping increases the number of degrees of freedom and the use of high-order methods decreases the CFL number, there is a substantial reduction in the computational time when we use high-order schemes compared with lower order schemes and when we use lumped elements compared with the standard ones.

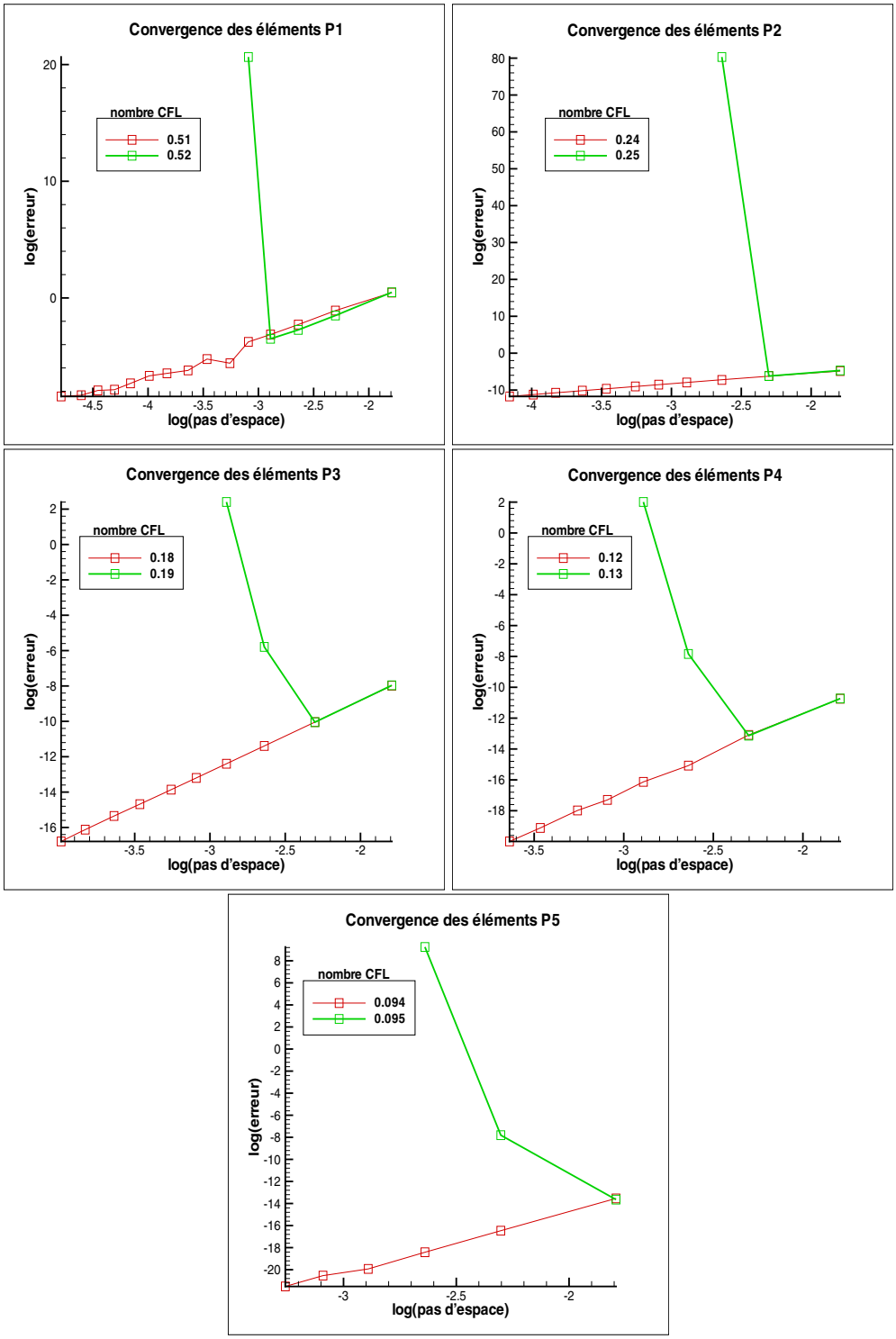

Fig. 8. Convergence rate for the explicit time discretization (standard P_k elements).

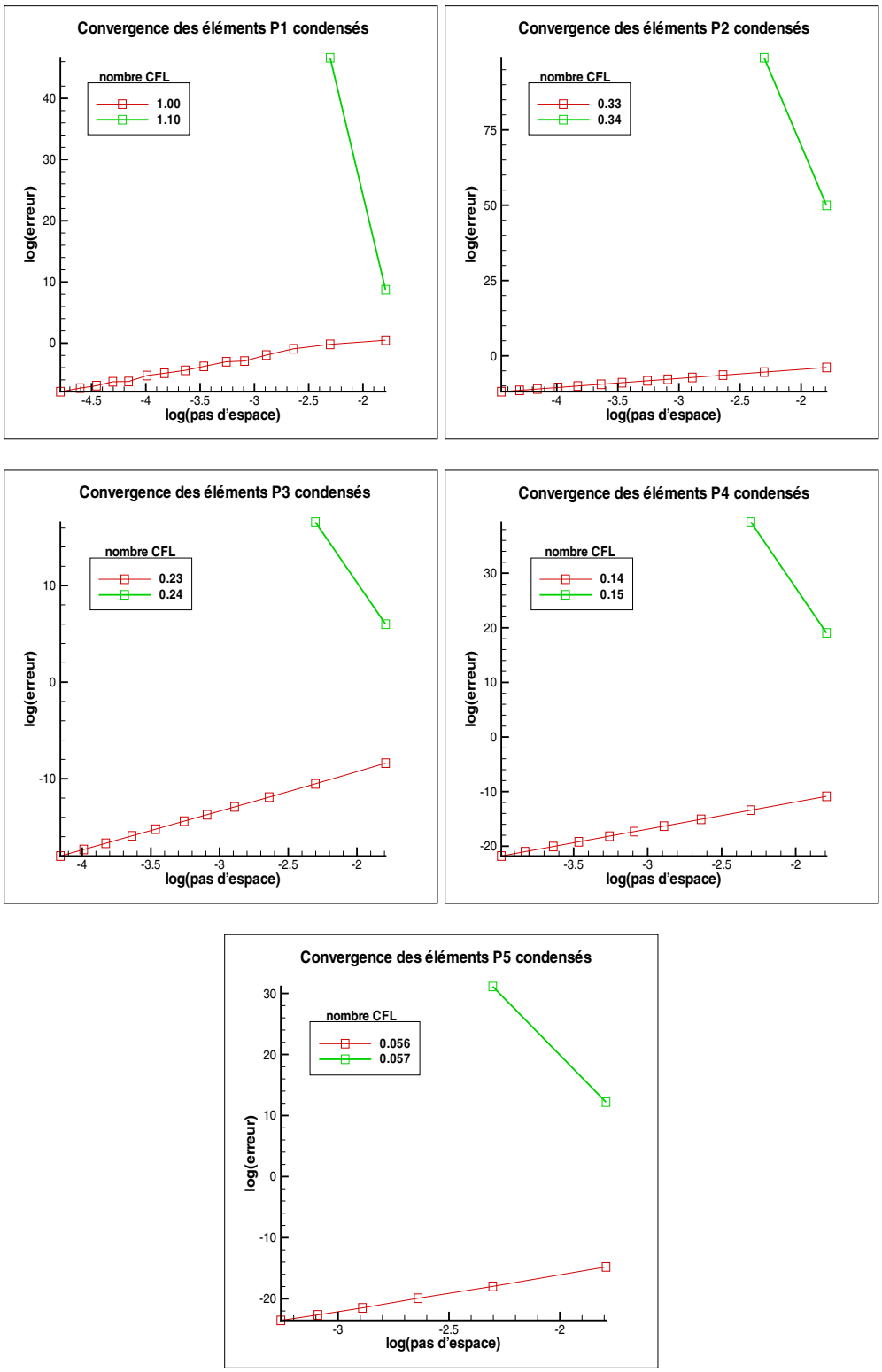

Fig. 9. Convergence rate for the explicit time discretization (lumped P_k elements).

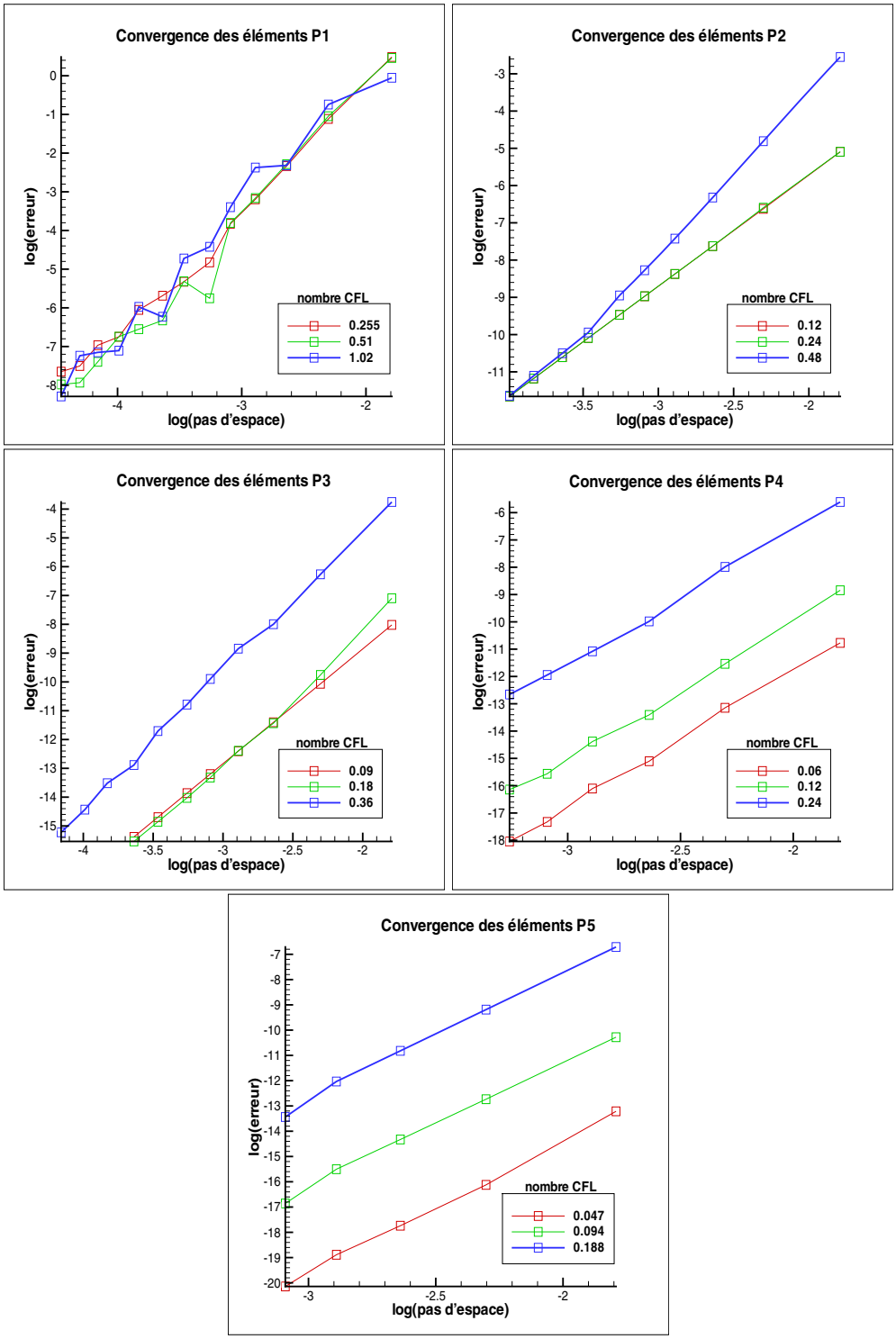

Fig. 10. Convergence rate for the implicit time discretization.

Table 5. Efficiency of the schemes.

for an error of $2. \times 10^{-9}$ in L^2 norm after 2 periods						
space disc.	time disc.	CFL	no. elem.	no. DOF	error	CPU time
P_4	explicit	0.12	1444	11705	2.14×10^{-9}	148.s
$\tilde{P}_4$	explicit	0.14	1444	16037	1.90×10^{-9}	21.s
P_5	explicit	0.094	324	4141	1.87×10^{-9}	24.s
$\tilde{P}_5$	explicit	0.056	196	4285	2.39×10^{-9}	6.s
P_4	implicit	0.12	1600	12961	2.27×10^{-9}	155.s
P_4	implicit	0.06	1444	11705	2.21×10^{-9}	258.s
P_5	implicit	0.047	324	4141	2.19×10^{-9}	46.s
P_5	implicit	0.094	900	11401	2.23×10^{-9}	148.s

Table 6. Efficiency of the schemes.

for an error of $4. \times 10^{-5}$ in L^2 norm after 10 periods						
space disc.	time disc.	CFL	no. elem.	no. DOF	error	CPU time
P_2	explicit	0.24	1444	2965	3.86×10^{-5}	30.s
$\tilde{P}_2$	explicit	0.33	2116	6441	4.45×10^{-5}	9.s
P_3	explicit	0.18	100	481	4.36×10^{-5}	<1.s
$\tilde{P}_3$	explicit	0.23	100	681	2.67×10^{-5}	<1.s
P_2	implicit	0.24	1024	2113	4.15×10^{-5}	19.s
P_2	implicit	0.48	1024	2113	4.82×10^{-5}	9.s
P_3	implicit	0.18	100	481	5.77×10^{-5}	1.s
P_3	implicit	0.36	484	2245	5.02×10^{-5}	11.s
for an error of $3. \times 10^{-7}$ in L^2 norm after 10 periods						
space disc.	time disc.	CFL	no. elem.	no. DOF	error	CPU time
P_3	implicit	0.18	1156	5305	2.77×10^{-7}	117.s
P_3	implicit	0.36	3844	17485	2.38×10^{-7}	521.s
P_4	implicit	0.06	196	1625	2.77×10^{-7}	29.s
P_4	implicit	0.12	400	3281	3.77×10^{-7}	61.s
P_5	implicit	0.047	64	841	4.17×10^{-7}	11.s
P_5	implicit	0.094	256	3281	3.22×10^{-7}	72.s
P_3	explicit	0.18	1156	5305	3.35×10^{-7}	129.s
$\tilde{P}_3$	explicit	0.23	900	5941	3.18×10^{-7}	14.s
P_4	explicit	0.12	196	1625	2.82×10^{-7}	3.s
$\tilde{P}_4$	explicit	0.14	196	2213	2.82×10^{-7}	2.s
P_5	explicit	0.094	64	841	2.53×10^{-7}	3.s
$\tilde{P}_5$	explicit	0.056	36	805	3.77×10^{-7}	1.s
for an error of 1.5×10^{-8} in L^2 norm after 10 periods						
space disc.	time disc.	CFL	no. elem.	no. DOF	error	CPU time
$\tilde{P}_3$	explicit	0.23	4096	26817	1.52×10^{-8}	135.s
P_4	explicit	0.12	676	5513	1.55×10^{-8}	172.s
$\tilde{P}_4$	explicit	0.14	676	7541	1.28×10^{-8}	31.s
P_5	explicit	0.094	196	2521	1.00×10^{-8}	44.s
$\tilde{P}_5$	explicit	0.056	100	2201	1.55×10^{-8}	12.s
P_4	implicit	0.06	676	5513	1.46×10^{-8}	320.s
P_5	implicit	0.047	324	4141	0.63×10^{-8}	222.s

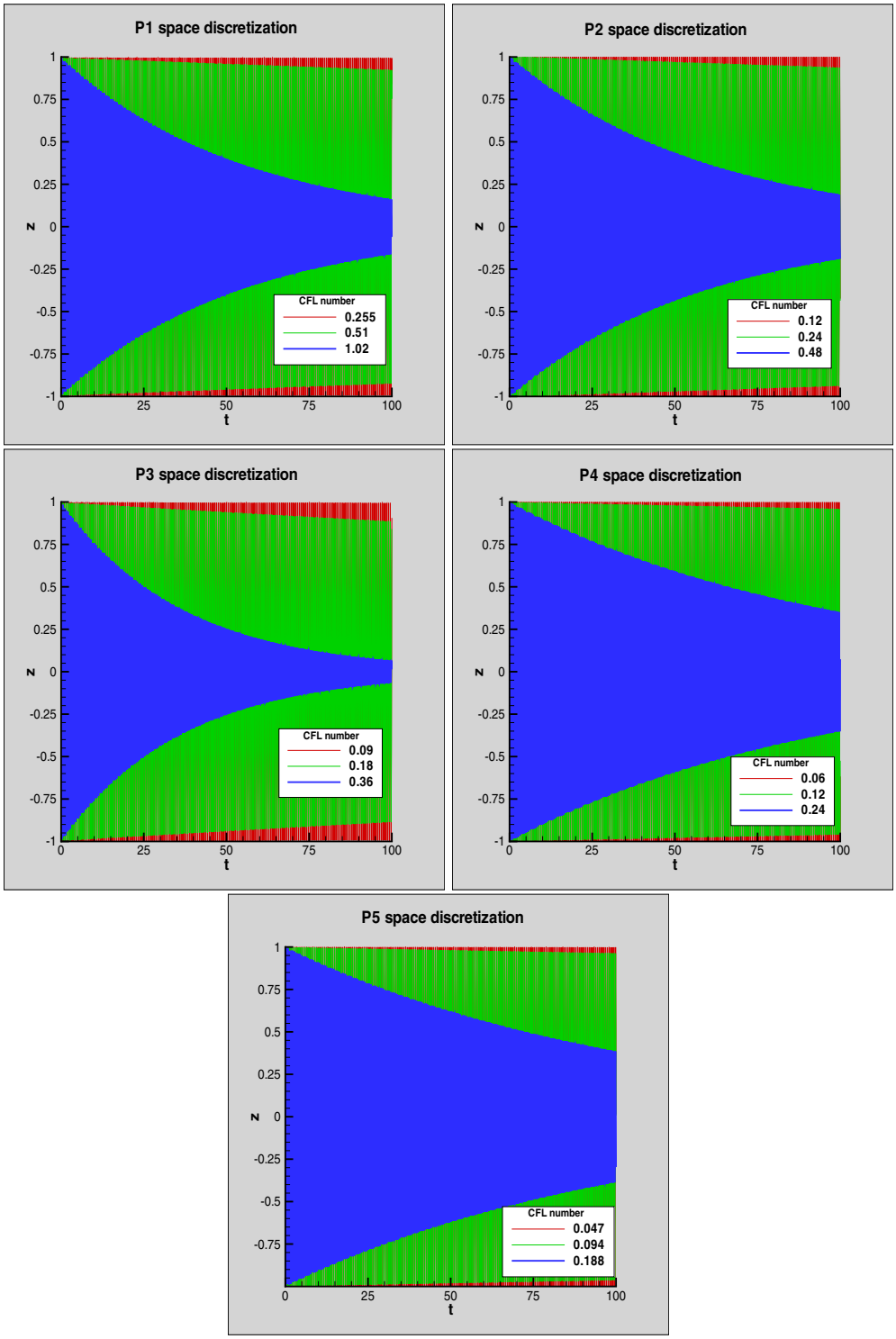

Fig. 12. CFL growing effects on the implicit time discretization.

Table 4. Details of the meshes.

space discretization	no. of triangles	no. of degrees of freedom
P1	1444	761
P2	324	685
P3	144	685
P4	100	841
P5	64	841

6. Conclusion

In this paper we have presented a numerical scheme of a high order in space and time to solve the two-dimensional wave equation. We proposed an algorithm to lump any Lagrange finite element of order k and determined lumped P_k finite elements up to $k = 5$. As was seen for the lumped P_6 calculation, the point is that we are unable, in general, to determine if the polynomial system, which gives the parameters of the quadrature formula, has an acceptable solution. In Section 5, we highlighted first the increasing CFL number restriction for increasing order explicit time discretization schemes (from 0.51 for P_1 space discretization, to 0.056 for $\tilde{P}_5$ space discretization), and then the fact that, even with a lower CFL number, high-order schemes are more efficient in terms of computational time and storage. One can see the gain in time by using lumped finite elements even if their use implies an increasing number of degrees of freedom by element. This drawback is balanced by the fact that we can use coarser meshes and that even if the order of the schemes is the same, i.e., the coefficients of the regression curves are the same, the constant defining this curve is lower for the lumped finite elements (due to a better repartition of the degrees of freedom).

One can ask if this result remains true by the use of mass-lumping in the three-dimensional space: to lump the second-order tetrahedral element, in (Cohen, 2002) the use of 23 degrees of freedom was proposed instead of 10 degrees of freedom for the standard second-order element, which means an over-cost for all matrix products that could balance the under-cost of the inversion of the mass matrix. For the implicit time discretization, the fact that theoretically there are fewer CFL restrictions is balanced by the fact that there is no more interest in the use of lumped finite elements in space, and that the schemes become very dissipative by increasing the CFL number, which is a real drawback for long-time computation.

Acknowledgements

This work is part of a DFG-CNRS project *Noise generation in turbulent flows*.

References

Butcher J.C. (2003): *Numerical Methods for Ordinary Differential Equations*. Chichester: John Wiley and Sons.

Chin-Joe-Kong M.J.S., Mulder W.A. and Van Veeldhuizen M. (1999): *Higher-order triangular and tetrahedral finite elements with mass lumping for solving the wave equation*. Journal of Engineering Mathematics Vol. 35, No. 4, pp. 405–426.

Ciarlet P.G. (1978): *The Finite Element Method for Elliptic Problems*. North-Holland, Amsterdam.

Cohen G. (2002): *Higher Order Numerical Method for Transient Wave Equations*. Berlin: Springer-Verlag.

Cohen G., Joly P., Roberts J.E. and Tordjman N. (2001): *Higher order triangular finite element with mass lumping for the wave equation*. SIAM Journal on Numerical Analysis, Vol. 38, No. 6, pp. 2047–2078.

Cohen G., Joly P. and Tordjman N.(1994): *Higher-order finite elements with mass lumping for the 1D wave equation*. Finite Elements in Analysis and Design, Vol. 16 , No. 3–4, pp. 329–336.

Cohen G., Joly P. and Tordjman N. (1993): *Construction and analysis of higher order finite elements with mass lumping for the wave equation*, In: Proceedings of the Second International Conference on Mathematical and Numerical Aspects of Wave Propagation Phenomena, SIAM, Philadelphia, pp. 152–160.

Dumbser M. (2005): *Arbitrary High Order Schemes for the Solution of Hyperbolic Conservation Laws in Complex Domains*. Ph.D. thesis, Stuttgart University.

Fix G.J. (1972): *Effect of quadrature errors in the finite element approximation of steady state, eigenvalue and parabolic problems*, In: The Mathematical Foundations of the Finite Element Method with Applications to the Partial Differential Equations, (A.K. Aziz, Ed.), New York: Academic Press, pp. 525–556.

Lax P.D. and Wendroff B. (1960): *Systems of conservation laws*. Communications on Pure Applied Mathematics, Vol. 13, pp. 217–237.

Mulder W.A. (1996): *A comparison between higher-order finite elements and finite differences for solving the wave equation*, In: Proceedings of the Second ECCOMAS Conference Numerical Methods in Engineering, (J.-A. Désidéri, P. Le Tallec, E. Oñate, J. Périaux and E. Stein, Eds.), Chichester: John Wiley and Sons, pp. 344–350.

Tordjman N. (1995): *Eléments finis d'ordre élevés avec condensation de masse pour l'équation des onde*. Ph.D. Thesis, Université Paris IX Dauphine, Paris.

Titarev V.A. and Toro E.F. (2002): *ADER: Arbitrary high order Godunov approach*. Journal of Scientific Computing, Vol. 17, No. 1-4, pp. 609–618.