J Eng Math (2006) 56:289–306
DOI 10.1007/s10665-006-9065-y

ORIGINAL PAPER

# Hierarchical spectral basis and Galerkin formulation using barycentric quadrature grids in triangular elements

**L. Grinberg · G. E. Karniadakis**

Received: 28 December 2005 / Accepted: 12 July 2006 /
Published online: 11 January 2007


**Abstract** Triangular spectral elements with modal shape functions are considered. The use of different types of nodal sets as quadrature grids for the construction of operators involved in the Galerkin formulation of the Navier–Stokes equations is investigated. Specifically, a Jacobi-polynomial tensor-product hierarchical basis is employed and the numerical integration, differentiation and projection on a Cartesian grid, and on two barycentric grids proposed by Blyth and Pozrikidis (2005, IMA J. Appl. Math. 71:153–169) and Taylor et al. (2005, SIAM J. Numer. Anal, Under review) are examined. A comparison of accuracy, efficiency and stability for a standard flow problem with exact solution is presented.

**Keywords** High-order · Navier–Stokes · Quadrature crimes · Spectral elements

## 1 Introduction

Spectral-element methods in two-dimensional domains consisting of triangular elements have only been developed in the last 10 years [1, 2]. They are particularly effective for solving time-dependent partial differential equations in truly complex-geometry domains. The spectral expansions employed in each triangle to represent the data, solution and geometry can be either of *modal* or *nodal* type. Hierarchical modal expansions can be cast in a tensor-product form if properly formulated, see [3]; they are typically used in conjunction with Galerkin projections. Nodal expansions are nonhierarchical and are usually employed in collocation-type methods; computing the proper set of nodes in a triangle has been the subject of several recent papers; see [4–7].

The present work focuses on modal spectral expansions and the Galerkin formulation for the divergence-free Navier–Stokes equations [2]. In the past, the discrete system has been formulated by employing Gauss-type quadratures on a Cartesian grid that is mapped to a standard square domain. Here we attempt, for

L. Grinberg (✉)
Division of Applied Mathematics, Brown University, Providence, RI 02912, USA
e-mail: lgrinb@dam.brown.edu

G. E. Karniadakis,
e-mail: gk@cfm.brown.edu

first time, to reformulate the discrete system based on barycentric quadrature grids formed by the nodal sets of points employed in the aforementioned nodal spectral expansions. Thus, the set of nodes which are used for interpolation for spectral collocation methods in the triangle will be used here for numerical integration and differentiation. We are particularly interested in examining the *quadrature crimes* [8] that may arise due to the nonlinear (advection) terms in the Navier–Stokes equations. By *quadrature crimes* we mean violations associated with quadrature rules for consistent numerical integration and differentiation of a polynomial function.

In the following, we first define the various quadrature grids and subsequently analyze the three basic operations, namely integration, differentiation and projection. We then perform a comparison of accuracy and efficiency for the three different approaches for the Kovasznay flow and analyze the numerical stability of the corresponding discrete system. We summarize the main findings in the last section.

## 2 Local coordinate systems and grids

### 2.1 Cartesian grid on a triangle

The Cartesian grid on a triangular region can be constructed by the mapping:

$$\xi_1 = \frac{(1+\eta_1)(1-\eta_2)}{2} - 1, \quad \xi_2 = \eta_2.$$

In the standard quadrilateral element the coordinates $\eta_1, \eta_2$ are bounded by constant limits, i.e.,

$$Q^2 = (\eta_1, \eta_2) | -1 \le \eta_1, \quad \eta_2 \le 1,$$

as shown in Fig. 1.

The two-dimensional Cartesian grid on $Q^2$ is constructed by superposition of two one-dimensional ones, namely $\eta_{1_i}, \eta_{2_j}, i = 1, 2, \ldots, Q_1, j = 1, 2, \ldots, Q_2$. Now consider a standard triangular element in the Cartesian coordinate system $(\xi_1, \xi_2)$

$$T^2 = (\xi_1, \xi_2) | -1 \le \xi_1, \xi_2; \quad \xi_1 + \xi_2 \le 0,$$

as shown in Fig. 1. Then, the new coordinate system on a triangular element is obtained by mapping of the standard coordinate system by the transformation

$$\eta_1 = 2\frac{1+\xi_1}{1-\xi_2} - 1, \quad \eta_2 = \xi_2. \tag{1}$$

The transformation (1) defines the *collapsed coordinate system*. It allows efficient numerical integration and differentiation. The collapsed Cartesian grid was successfully implemented in spectral/*hp* element

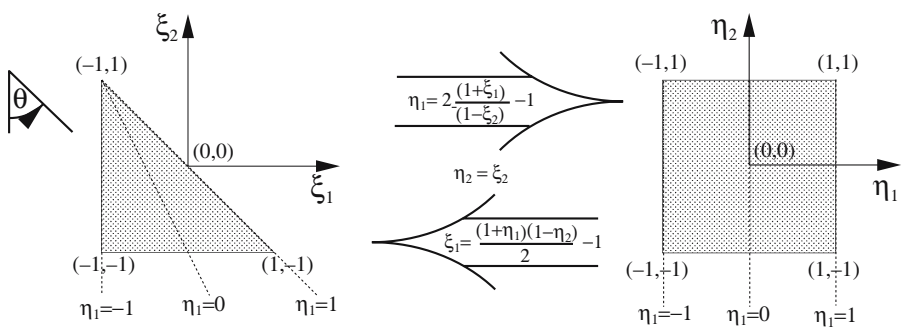


**Fig. 1** Cartesian $(\xi_1, \xi_2)$ and collapsed Cartesian $(\eta_1, \eta_2)$ coordinate systems

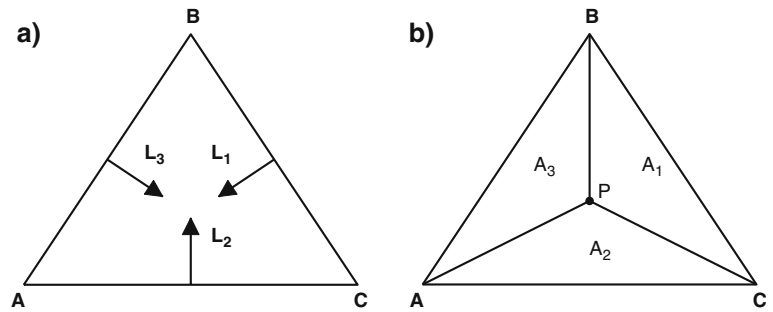


**Fig. 2** Triangular elements and barycentric coordinate system

methods and the Navier–Stokes solver *Nektar* [1]. The main drawback of such a grid is the nonsymmetrical distribution of quadrature points with respect to vertices of the triangular element. In the rest of the paper we will use the notation CCG for the collapsed Cartesian grid.

## 2.2 Barycentric grids on a triangle

The barycentric coordinate system on a triangular region is defined by three *nonindependent* coordinates $L_1, L_2, L_3$ as illustrated in Fig. 2a.

The coordinates of a grid point $P(L_{1_i}, L_{2_i}, L_{3_i})$ are computed as a ratio of areas of triangles $ABP, BCP, CAP$ (Fig. 2b) to the area of an element $ABC$, i.e.,

$$L_i = \frac{A_i}{A}, \quad i = 1, 2, 3.$$

In the present study we consider two types of barycentric grids. The first was developed by Blyth and Pozrikidis [7], and we will refer to this grid as a barycentric grid of a type $A$ (BGA); a two-dimensional grid is generated from a one-dimensional *master grid*. Here we employ the Gauss grid in a nontensorial coordinate system, whereas in [7] the Gauss–Lobatto grid was used. The total number of nodal points over a triangular element is $N = m(m+1)/2$, where $m$ is the number of grid points in one dimension. The highest order of polynomials, $P$, that can be exactly integrated and differentiated is limited by $P = m - 1$. The main advantage of the BGA is that we obtain a *symmetric* distribution of quadrature points with respect to the vertices of a triangular element.

The second type of barycentric grid was developed by Taylor et al. [6] and we will refer to this grid as a barycentric grid of a type $B$ (BGB). The number of quadrature points, $N$, is defined by the order of cardinal functions—polynomials of order $P$—as $N = (P+1)(P+2)/2$. The distribution of quadrature points over a triangular element and their associated weights are optimized in order to obtain exact integration of polynomials of order higher than $P$. The distribution of quadrature points is not always symmetrical; however, even in the case of asymmetrical distribution, there is no clustering of quadrature points as in the case of the collapsed Cartesian grid. All quadrature points computed with the cardinal function of degree up to 14 are located inside the triangle and have positive weights. Integration on the BGB grid of polynomials up to order 25 is exact.

We emphasize that the BGA grid was constructed for interpolation purposes, while the BGB grid was optimized for integration.

Another set of symmetrical nodes, optimized for integration on a triangular element was suggested by Wandzura and Xiao [9]. The highest order of polynomial function that can be exactly integrated using this set is limited by $P = 30$. It was argued in [6] that integration on BGB is generally less expensive than integration using the Wandzura and Xiao grid. The Vandermonde matrix constructed from modal bases defined on a triangular element and Wandzura and Xiao points is not a square matrix, which provides

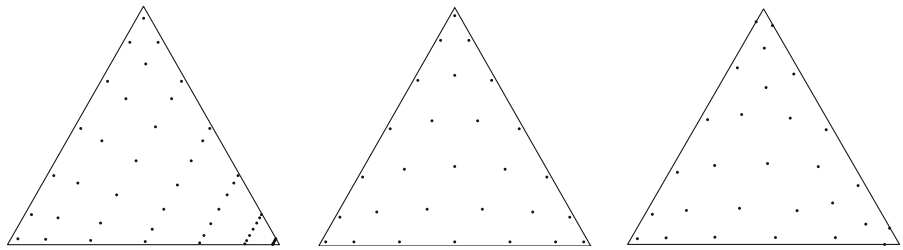

**Fig. 3** Quadrature grids: Left—CCG; Middle—BGA; Right—BGB. The total number of quadrature points, $N$, is defined as: $N = (P+1)^2$ for CCG, $N = (P+1)(P+2)/2$ for BGA and BGB; $P = 6$

additional complexity in numerical differentiation. In the current work we have not studied the Wandzura and Xiao points.

In Fig. 3 we show typical distributions of Cartesian and barycentric grid points over an equilateral triangle.

## 3 Integration and differentiation

In this section we review the methods for numerical integration and differentiation using the collapsed Cartesian and barycentric grids. We compare the computational cost of these basic numerical operations with respect to different computational grids.

### 3.1 Numerical integration

The exact integration of a function $u$ on triangular region $T^2$

$$I = \int_{T^2} u \, \mathrm{d}A$$

can be approximated by numerical integration, i.e.,

$$I = \sum_{i=1}^{N} \omega_i u_i + R,$$

where $\omega_i$ is the $i$th quadrature weight, $u_i$ is the value of a function $u$ computed at a point $i$ and $R$ is the quadrature error. Different types of grids defined on a triangular region lead to different integration techniques.

#### *3.1.1 Integration on the collapsed Cartesian grid*

The exact integration of a function $u$ on the standard triangular region using the collapsed coordinate system $(\eta_1, \eta_2)$ is expressed as

$$I = \int_{T^2} u(\xi_1, \xi_2) \mathrm{d}\xi_1 \, \mathrm{d}\xi_2 = \int_{-1}^{1} \int_{-1}^{-\xi_2} u(\xi_1, \xi_2) \mathrm{d}\xi_1 \mathrm{d}\xi_2 = \int_{-1}^{1} \int_{-1}^{1} u(\eta_1, \eta_2) J \, \mathrm{d}\eta_1 \, \mathrm{d}\eta_2, \tag{2}$$

where $J$ is the Jacobian of the transformation (1)

$$J = \left| \frac{\partial(\xi_1, \xi_2)}{\partial(\eta_1, \eta_2)} \right|. \tag{3}$$

The numerical counterpart of the last term in (2) is

$$I = \sum_{i=1}^{i=Q_1} \omega_{1_i} \left[ \sum_{j=1}^{j=Q_2} \omega_{2_j} u(\eta_{1_i}, \eta_{2_j}) J(\eta_{1_i}, \eta_{2_j}) \right] + R. \tag{4}$$

It is efficient to use Gauss quadrature points and weights for numerical integration. The Jacobian, $J$, is usually included into the weights $\omega_1, \omega_2$. In the case where $u$ is a polynomial of degree $P_1(P_2)$ in $\xi_1(\xi_2)$, we need only $Q_1 = (P_1 + 1)/2$, $Q_2 = (P_2 + 1)/2$ Gauss quadrature points to perform the exact numerical integration. Thus, the precise cost of numerical integration of a polynomial of order $P$ (here and thereafter we assume that $P_1 = P_2 = P$ and $Q_1 = Q_2 = Q$) on the collapsed-coordinate system is $(3Q^2 + Q)$ multiplications and summations, with function evaluations at $Q^2$ quadrature points.

*3.1.2 Integration on the barycentric grids*

The numerical integration on a barycentric grid over a triangular region with area $A$ is given by

$$I = 2A \sum_{i=1}^{N} \omega_i u(L_{1_i}, L_{2_i}) + R'$$

with the weights, $\omega_i$, for BGA computed from

$$\mathbf{V}\omega = \int \Phi \, \mathrm{d}A,$$

where $\Phi$ is a vector formed from the polynomial basis $\phi(m, n) \in \mathcal{P}$

$$\mathcal{P} = \mathrm{span}\{\xi_1^n \xi_2^m, m + n \leq P\},$$

and $\mathbf{V}$ is the corresponding Vandermonde matrix constructed from $\phi(m, n)$.

For BGB the procedure of computing the location of quadrature points and weights is more complicated; a detailed explanation can be found in [6].

The computational cost of the exact integration of a polynomial of degree $P$ on BGA is $(P + 1)(P + 2)$ summations and multiplications, while the cost of function evaluations scales as $(P + 1)(P + 2)/2$. On the other hand, for the exactly same computational cost we can integrate polynomials of degree higher than $P$ using BGB. We summarize this section by comparing the number of quadrature points and the number of numerical operations for integration of polynomial of degree $P$. In Fig. 4a, b we plot the results for the three types of quadrature. We assume that the computational cost for function evaluation at each quadrature point is one unit. Clearly, the quadrature grid BGB is more efficient than BGA.

## 3.2 Numerical differentiation

Differentiation on Cartesian and barycentric coordinate systems in a triangular element is based on chain rule. In the case of CCG we apply the chain rule twice: first due to transformation from a global Cartesian coordinate system $(x, y)$ to a local $(\xi_1, \xi_2)$, and then due to transformation from a local Cartesian system to a collapsed one $(\eta_1, \eta_2)$. In the case of barycentric grid we have only one transformation, that is, from a global coordinate system to a local one.

*3.2.1 Differentiation on collapsed Cartesian grid*

The mapping from a local coordinate to a global coordinate system is defined by

$$x_i = x_i^A \frac{-\xi_1 - \xi_2}{2} + x_i^B \frac{1 + \xi_1}{2} + x_i^C \frac{1 + \xi_2}{2},$$

here $x_1 = x, x_2 = y$ and $x_i^A, x_i^B, x_i^C$ are coordinates of element vertices. Thus, differentiation on the CCG is defined by

$$\begin{bmatrix} \dfrac{\partial}{\partial x} \\ \dfrac{\partial}{\partial y} \end{bmatrix} = \frac{2}{A} \begin{bmatrix} \dfrac{\partial y}{\partial \xi_2} & -\dfrac{\partial y}{\partial \xi_1} \\ -\dfrac{\partial x}{\partial \xi_2} & \dfrac{\partial x}{\partial \xi_1} \end{bmatrix} \begin{bmatrix} \dfrac{\partial}{\partial \xi_1} \\ \frac{\partial}{\partial \xi_2} \end{bmatrix}, \tag{5}$$

where the partial derivatives in $\xi_i$ are computed from:

$$\begin{bmatrix} \dfrac{\partial}{\partial \xi_1} \\ \dfrac{\partial}{\partial \xi_2} \end{bmatrix} = \begin{bmatrix} \dfrac{2}{1-\eta_2} \dfrac{\partial}{\partial \eta_1} \\ \dfrac{1+\eta_1}{1-\eta_2} \dfrac{\partial}{\partial \eta_1} + \dfrac{\partial}{\partial \eta_2} \end{bmatrix}. \tag{6}$$

The derivatives of $u(\eta_1, \eta_2)$ may also be computed using the collocation differentiation matrix $\mathbf{D}_{\eta_i}$ as $\mathbf{u}(\eta_1, \eta_2)_{\eta_i} = \mathbf{D}_{\eta_i}\mathbf{u}(\eta_1, \eta_2)$.

In order to differentiate a function from $\mathcal{V}^P$ exactly (here and thereafter $\mathcal{V}^P$ denotes a polynomial space of order $P$), $Q = P + 1$ quadrature points are required. Thus, the computational cost of numerical differentiation on CCG is proportional to the cost of matrix-matrix multiplication and scales as $(P+1)^3$.

### *3.2.2 Differentiation on the barycentric grids*

The numerical differentiation on BGA and BGB is similar. We can represent any function as

$$u(\xi_1, \xi_2) = \sum_{k=1}^{K} \hat{u}_k \phi_k(\xi_1, \xi_2), \tag{7}$$

where $K = (P+1)(P+2)/2$, $\hat{u}_k$ is the amplitude of polynomial basis $\phi_k$, $k = k(m,n)$ is an index of polynomial basis and $m$, $n$ are polynomial order of the basis with respect to two coordinates. For more details see [10]. From (7) we express the derivatives of $u$ with respect to each orthogonal direction $\xi_1, \xi_2$ as

$$\frac{\partial}{\partial \xi_i} u(\xi_1, \xi_2) = \sum_{k=1}^{K} \hat{u}_k \frac{\partial}{\partial \xi_i} \phi_k(m, n, \xi_1, \xi_2), \quad i = 1, 2. \tag{8}$$

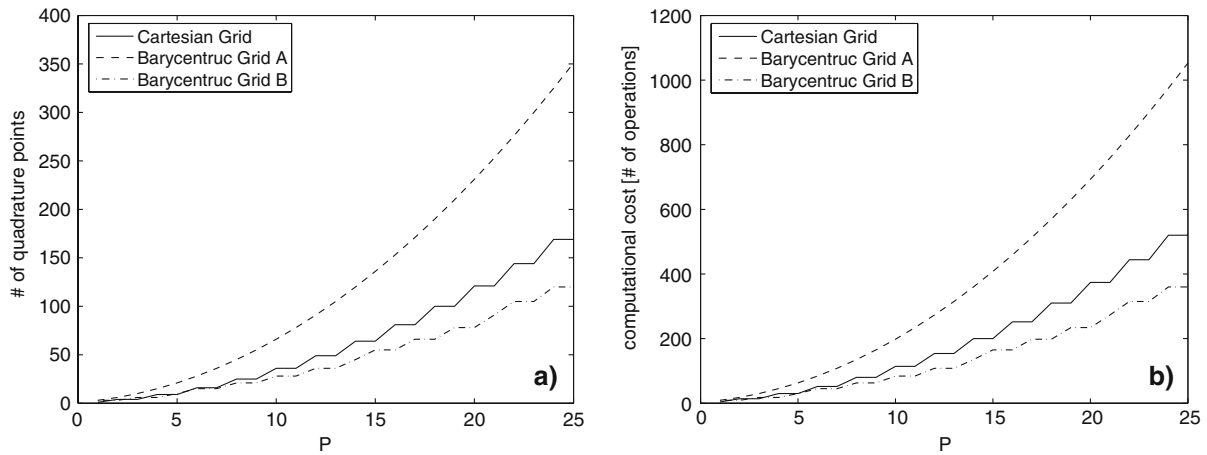


**Fig. 4** (**a**) Number of quadrature points for exact numerical integration of a polynomial of order $P$. (**b**) Corresponding computational cost

We may rewrite expansions (7) and (8) in compact form as:

$$\mathbf{u} = \mathbf{V}\hat{\mathbf{u}}, \qquad \mathbf{u}_{\xi_i} = \mathbf{V}_{\xi_i}\hat{\mathbf{u}}, \quad i = 1, 2.$$

Thus, the derivatives of a function $u$ at quadrature points $\xi_{1_i}, \xi_{2_i}$ are computed from:

$$\mathbf{u}_{\xi_i} = [\mathbf{V}_{\xi_i}\mathbf{V}^{-1}]\mathbf{u}, \quad i = 1, 2. \tag{9}$$

The Vandermonde matrix $\mathbf{V}$ is constructed on $N$ quadrature points. To perform exact differentiation of a polynomial function of order up to $P$ we need $N = (P+1)(P+2)/2$ quadrature points, i.e., the same as the number of polynomials in the basis. The transformation from barycentric to local Cartesian coordinates $-1 \leq \xi_1, \xi_2; \xi_1 + \xi_2 \leq 0$ is given by

$$\xi_1 = -L_1 + L_2 - L_3 = 2L_2 - 1, \quad \xi_2 = -L_1 - L_2 + L_3 = -1 + 2L_3. \tag{10}$$

The derivatives of a function $u(x, y)$ are expressed as linear combination of derivatives of the function in the local coordinate system $(\xi_1, \xi_2)$ using chain rule (5).

The computational cost of numerical differentiation on the barycentric grid scales as $N^2 = (P+1)^2 (P+2)^2/4$.

*3.2.3 Projection*

The projection operator from *modal to physical space* is defined by

$$u(\xi_{1_i}, \xi_{2_j}) = \sum_{k=1}^{K} \hat{u}_k \phi_k(\xi_{1_i}, \xi_{2_j}). \tag{11}$$

Thus, the computational cost of computing $u(\xi_{1_i}, \xi_{2_j})$ at a single quadrature point does not depend on the type of grid, but the overall cost depends on the number of quadrature points. This implies that using the barycentric grid with $N = (P+1)(P+2)/2$ quadrature points is almost half the cost of using CCG with $N = (P+1)^2$ quadrature points.

The projection from *physical to modal space* is performed in three steps. First, we compute the mass matrix $\mathbf{M}$; second, we integrate the function with respect to the test functions; third, we solve the algebraic system $\mathbf{M}\hat{\mathbf{u}} = \mathbf{f}$, where the unknowns $\hat{\mathbf{u}}$ are amplitudes of the expansion basis. The efficiency of the first and the second steps depends on the effectiveness of the numerical integration. However, we note that CCG is based on a tensor-product construction. For tensor-product bases we can employ the *sum-factorization* to effectively reduce the computational cost. On barycentric grids the sum-factorization is not possible [1]. We also note that the second step is repeated $(P+1)(P+2)/2$ times equal to the number of modes in a polynomial expansion. In Fig. 4b we demonstrate that the choice of BGB leads to the most efficient integration and thus the most efficient projection.

## 4 Application to a Navier–Stokes solver

This section consists of three parts. First, we provide an overview of the model problem and numerical scheme. In the second part we analyze the numerical efficiency of a Navier–Stokes solver where different quadrature grids are employed. In the third part we analyze the effect of quadrature grids on the temporal stability of the Navier–Stokes solver.

### 4.1 Kovasznay flow and numerical scheme

The aforementioned three types of quadrature grids were employed in all operations in a 2D spectral/*hp*-element code for incompressible Navier–Stokes equations. As a prototype problem we consider

the Kovasznay flow, a steady laminar flow behind a two-dimensional grid. The exact solution is given by

$$u = 1 - \mathrm{e}^{\lambda x}\cos(2\pi y), \quad v = \frac{\lambda}{2\pi}\mathrm{e}^{\lambda x}\sin(2\pi y), \quad p = p_0 - \frac{1}{2}\mathrm{e}^{2\lambda x}, \tag{12}$$

where $\lambda = \frac{\mathrm{Re}}{2} - \sqrt{\frac{\mathrm{Re}^2}{4} + 4\pi^2}$, Re being the Reynolds number. The numerical solution is obtained by solving the incompressible Navier–Stokes (NS) equations in terms of the primitive variables, i.e.,

$$\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v}\cdot\nabla\mathbf{v} = -\nabla p + \frac{1}{\mathrm{Re}}\nabla^2\mathbf{v}, \quad \nabla\cdot\mathbf{v} = 0 \tag{13}$$

with Dirichlet boundary condition for $\mathbf{v} = [u\ v]^{\mathrm{T}}$, specified by the exact solution (12). Following the standard spectral/*hp* element approach, the unknown functions, $\mathbf{f} = [u, v, p]$ (including the pressure; see [1, Chapter 8]) are approximated by $P$-order polynomial expansions:

$$\mathbf{f}_P = \sum_{k=1}^{K} \hat{\mathbf{f}}_k \phi_k(\xi_1, \xi_2), \tag{14}$$

where $\phi \in \mathcal{V}^P$. On triangular elements $K = (P+1)(P+2)/2$.

The NS equations were solved using a splitting scheme as follows:

$$\frac{\mathbf{v}^\star - \mathbf{v}^n}{\Delta t} = -[\mathbf{NL}]^{n+1/2}, \tag{15}$$

$$\frac{\mathbf{v}^{n+1} - \mathbf{v}^\star}{\Delta t} = \frac{1}{\mathrm{Re}}\nabla^2\mathbf{v}^{n+1} - \nabla p^{n+1}, \tag{16}$$

where $p^{n+1}$ is computed from

$$\nabla^2 p^{n+1} = (\Delta t)^{-1}\nabla\cdot\mathbf{v}^\star, \tag{17}$$

and **NL** in (15) represents the nonlinear terms. The nonlinear term can be formulated in conservative, rotational, skew-symmetric and convective forms. We performed numerical experiments using all formulations. In the paper we concentrate primarily on the conservative (18) and convective (19) forms.

$$NL_x = \frac{\partial (u)^2}{\partial x} + \frac{\partial uv}{\partial y}, \qquad NL_y = \frac{\partial uv}{\partial x} + \frac{\partial (v)^2}{\partial y}. \tag{18}$$

$$NL_x = u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y}, \qquad NL_y = u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y}. \tag{19}$$

The weak form of NS equations is obtained by multiplication of (13) by a test function $\psi_k$ and integration over domain $\Omega$; according to the Bubnov–Galerkin method $\psi_k = \phi_k$. The weak form of linear terms requires integration of polynomials of from $\mathcal{V}^{2P}$, while for the nonlinear terms the integrated polynomials are in $\mathcal{V}^{3P-1}$. The calculation of **NL** involves several steps: On the first step, values of **u** are computed on $N$ quadrature points by projection from modal to physical space. On the second step, the derivatives operator is applied to obtain $NL_x$, $NL_y$. Finally, $NL_x$ and $NL_y$ are integrated with respect to the test functions. We note that values of $NL_x$, $NL_y$ in physical space are also used for the evaluation of the right-hand side for the pressure equation, (17). The divergence of the provisional field $\mathbf{v}^\star$ is, first, computed by applying the derivative operator on $\mathbf{v}^\star$ and, second, by integration of $\nabla\cdot\mathbf{v}^\star$ with respect to the test functions.

A *consistent* implementation of the weak form of the nonlinear terms implies that the number of quadrature points should satisfy the rules for *exact* numerical integration and *exact* differentiation of polynomials from $\mathcal{V}^{3P-1}$ and $\mathcal{V}^{2P}$, respectively. Our numerical experiments were divided into three categories:

(a) Number of quadrature points is sufficient for exact integration of polynomials from $\mathcal{V}^{2P}$.
(b) Number of quadrature points is sufficient for exact integration of polynomials from $\mathcal{V}^{3P-1}$.
(c) Number of quadrature points is sufficient for exact integration of polynomials from $\mathcal{V}^{3P-1}$ and exact differentiation of polynomials from $\mathcal{V}^{2P}$.

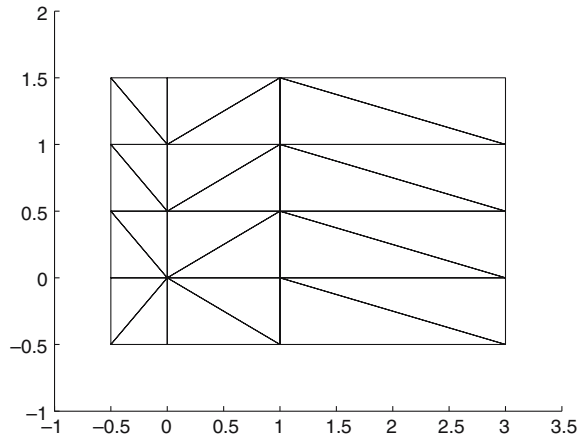


**Fig. 5** Triangular elements and domain for the solution of the Kovasznay problem

We will refer to the method in case (a) as inconsistent numerical integration; in case (b) as consistent numerical integration; and case (c) as fully consistent numerical integration and differentiation.

The computational domain was decomposed into 24 triangular elements, as shown in Fig. 5. For each simulation we started from the same initial field, used the same solver for the Helmholtz and Poisson equations (involved in the second and third substeps of the splitting scheme) and the same time stepping scheme. The size of the timestep was fixed to $\Delta t = 10^{-4}$. For the nonlinear terms we used the second-order Adams–Bashforth (AB2) time stepping formula; for the viscous terms we used an Euler Backwards scheme. To estimate the numerical efficiency, we monitored the CPU time required to obtain a solution at $T = 6$ and the $L_2$-error of a solution at $T = 1, 2, 3, 4, 5, 6$ (steady state is achieved after $T = 4$) Also, $P = 6$ and $P = 8$ were used for spectral polynomial order.

## 4.2 Numerical efficiency

Let $P_I$ denote the highest order of polynomial function that can be exactly integrated on a given set of quadrature points and $P_d$ the highest order of polynomial function that can be exactly differentiated. We also denote by $N$ the total number of quadrature points. For each type of grid the number of quadrature points is chosen according to the following principles.

*Collapsed Cartesian grid (CCG)* The number of quadrature points is set according to rules for one-dimensional integration. Using $Q$ Gauss quadrature points per direction, we can integrate exactly polynomials of order $P \leq 2Q - 1 = P_I$ and differentiate exactly polynomials of order $P \leq Q - 1 = P_d$. The total number of grid points on a triangular element is $N = Q^2$.

*Barycentric grid of type A (BGA)* The highest order of polynomial that can be exactly integrated and differentiated is defined by the number of grid points of the one-dimensional master grid from which the two-dimensional one is constructed. In order to have exact numerical differentiation and integration we need $m = P_d + 1 = P_I + 1$ grid points in the one-dimensional grid. Thus, the total number of grid points in the triangular region is defined by $N = (P_d + 1)(P_d + 2)/2$.

*Barycentric grid of type B (BGB)* The number of quadrature points for consistent differentiation is defined similarly as for BGA, i.e., $N = (P_d + 1)(P_d + 2)/2$. However, the highest order of polynomial that can be exactly integrated using the same number of grid points is greater than $P_d$. Appropriate values of $N$ for exact integration of polynomial function in $\mathcal{V}^{P_I}$ are provided in Table 7.1 of [6].

We recall that BGB was optimized for numerical integration, while BGA was not; moreover some of the weights, computed on BGA, for integration of polynomials with $P > 6$, are negative.

In handling the nonlinear terms it is convenient to perform the basic numerical operations on the same grid, that is we use the same points for integration and differentiation. In nonlinear problems we have that $P_I > P_d$, which implies that it is not always possible to have $N$ grid points and satisfy *exactly* the rules for accurate integration and differentiation without paying an extra cost by performing over-integration or differentiation (in a super-collocation fashion); see [11]. Also, in the discretization of the nonlinear terms with $\mathbf{v}_P \in \mathcal{V}^P$ we may specify the number of quadrature points for each type of grid in order to have (a) exact integration of the weak linear advection operator; (b) exact integration of the weak nonlinear advection operator; and (c) exact integration and differentiation of the weak nonlinear advection operator.

In Table 1 we summarize the values of $N$, $P_I$ and $P_d$ given the above considerations for the nonlinear problem with $\mathbf{v}_P \in \mathcal{V}^6$. The number of quadrature points on each grid was chosen such that we have: (a) inconsistent numerical integration, (b) consistent numerical integration, and (c) consistent numerical integration and differentiation. In our problem, the order of polynomial function that should be integrated is higher than the order for differentiation, thus consistent integration on BGA leads to consistent differentiation as well. For this reason the second and the third rows that correspond to BGA in Table 1 (also in Table 2) are the same. We note that, for this particular problem, the highest order of a polynomial function to be integrated is 17 and the highest order of polynomial function to be differentiated is 12, since we use the conservative formulation for the nonlinear terms.

The Kovasznay flow problem was solved on the three different grids, with the number of grid points specified according to Table 1. In Fig. 6 we present the $L_2$-error as a function of time. The left plot illustrates the behavior of the error of the solution when the number of quadrature points is sufficient to integrate a weak linear advection operator only. We note that for the collapsed Cartesian grid as well as for barycentric grid of type B this number of quadrature points is insufficient for exact numerical differentiation. On the other hand, for BGA the numerical differentiation is exact. The dashed line depicts the fully consistent case, i.e., when we have consistency in both integration and differentiation. The right plot shows the $L_2$-error for tests where the number of quadrature points is sufficient for consistent numerical integration. We observe that in terms of accuracy the choice of barycentric grid of type A is advantageous for the case of inconsistent numerical integration. Another observation is that insufficient resolution for numerical differentiation alone does not affect the overall accuracy, as shown in the right plot.

We performed the same test with rotational and skew-symmetric forms of the nonlinear term. The results were very similar to the results obtained with conservative formulation of the **NL**. The drop of accuracy was observed when *both* numerical integration and differentiation suffered from under-resolution.

A different situation was observed when the convective form of **NL** was used; using the convective form always leads to exact numerical differentiation. In Fig. 7 we present the numerical error of the solution obtained using the convective form and inconsistent numerical integration. We see no drop of accuracy due to numerical under-integration.

The Kovasznay-flow problem was also solved with higher accuracy, corresponding to $\mathbf{v}_P \in \mathcal{V}^8$. In Table 2 we provide a summary of the number of quadrature points and $P_I$, $P_d$ for each simulation. We note that, in case of $\mathbf{v}_P \in \mathcal{V}^8$, the highest order of polynomial function to be integrated is 23, while the highest order of polynomial function to be differentiated is 16. For BGB the maximum number of quadrature points is 120 which corresponds to a cardinal function of a degree 14; for this reason $\max(P_d) = 14$. The results of

**Table 1** Number of quadrature points for (a) inconsistent numerical integration, (b) consistent numerical integration, and (c) consistent numerical integration and differentiation of a weak nonlinear operator; $\mathbf{v}_P \in \mathcal{V}^6$

| | Collapsed Cartesian | | | Barycentric grid A | | | Barycentric grid B | | |
|---|---|---|---|---|---|---|---|---|---|
| | $N$ | $P_I$ | $P_d$ | $N$ | $P_I$ | $P_d$ | $N$ | $P_I$ | $P_d$ |
| (a) | 49 | 13 | 6 | 91 | 12 | 12 | 36 | 13 | 7 |
| (b) | 81 | 17 | 8 | 171 | 17 | 17 | 66 | 18 | 10 |
| (c) | 169 | 25 | 12 | 171 | 17 | 17 | 120 | 21 | 12 |

convergence of the error in the $L_2$ norm are presented in Fig. 8; we observe similar behavior as in the case with $\mathbf{v}_P \in \mathcal{V}^6$.

In Figs. 9 and 10 we show the dependence of the $L_2$-error for the different methods with respect to CPU time. Here the CPU time is measured in seconds and reflects the elapsed time from the beginning of simulation (first time step, $t = 0$) to the intermediate time ($t = 1, 2, 3, 4, 5$) and final time $T = 6$. The CPU

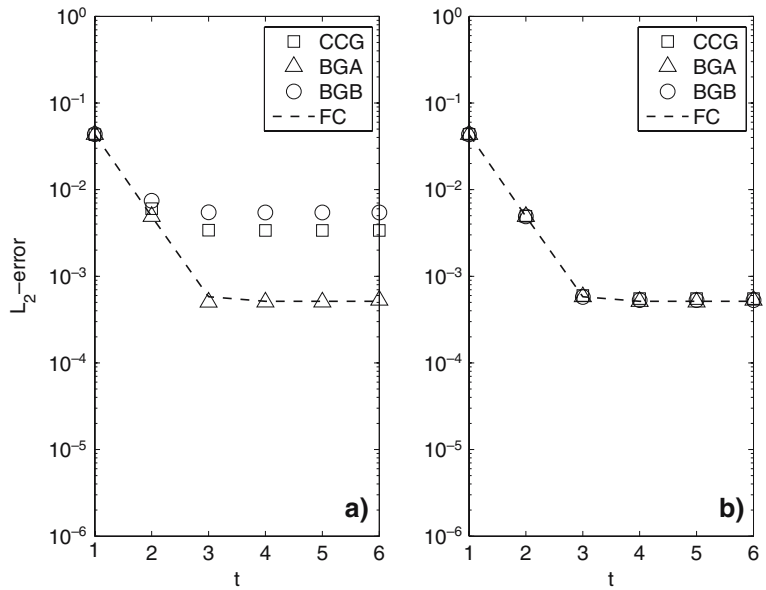


**Fig. 6** $L_2$-error versus time for the collapsed Cartesian grid (CCG), barycentric grid of type A (BGA) and B (BGB) with (**a**) inconsistent numerical integration, and (**b**) exact numerical integration. The dashed line (FC) depicts the fully consistent case; $\mathbf{v}_P \in \mathcal{V}^6$; conservative formulation of the nonlinaer term

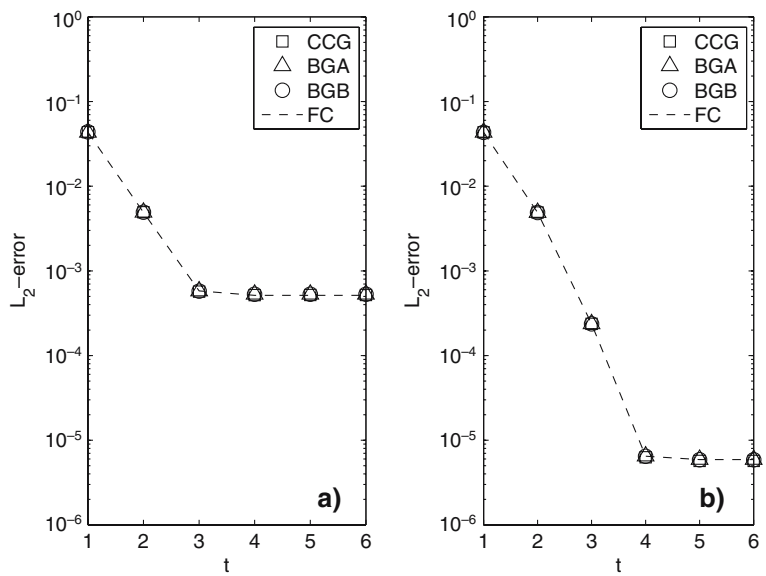


**Fig. 7** $L_2$-error versus time for the collapsed Cartesian grid (CCG), barycentric grid of type A (BGA) and B (BGB) with inconsistent numerical integration and convective formulation of the nonlinear term; (**a**) $\mathbf{v}_P \in \mathcal{V}^6$; (**b**) $\mathbf{v}_P \in \mathcal{V}^8$ The dashed line (FC) depicts the fully consistent case

**Table 2** Number of quadrature points for (a) inconsistent numerical integration, (b) consistent numerical integration, and (c) consistent numerical integration and differentiation of a weak nonlinear operator; $\mathbf{v}_P \in \mathcal{V}^8$

| | Collapsed Cartesian | | | Barycentric grid A | | | Barycentric grid B | | |
|---|---|---|---|---|---|---|---|---|---|
| | $N$ | $P_I$ | $P_d$ | $N$ | $P_I$ | $P_d$ | $N$ | $P_I$ | $P_d$ |
| (a) | 81 | 17 | 8 | 153 | 16 | 16 | 55 | 16 | 9 |
| (b) | 144 | 23 | 11 | 300 | 23 | 23 | 105 | 23 | 13 |
| (c) | 289 | 33 | 16 | 300 | 23 | 23 | 120 | 120 | 14 |

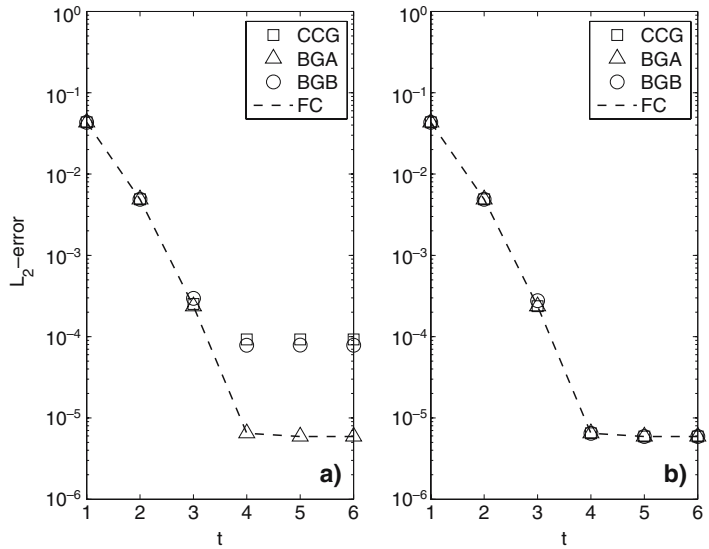


**Fig. 8** $L_2$-error versus time for the collapsed Cartesian grid (CCG), barycentric grid of type A (BGA) and B (BGB) with (**a**) inconsistent numerical integration, and (**b**) exact numerical integration. The dashed line (FC) depicts the fully consistent case; $\mathbf{v}_P \in \mathcal{V}^8$

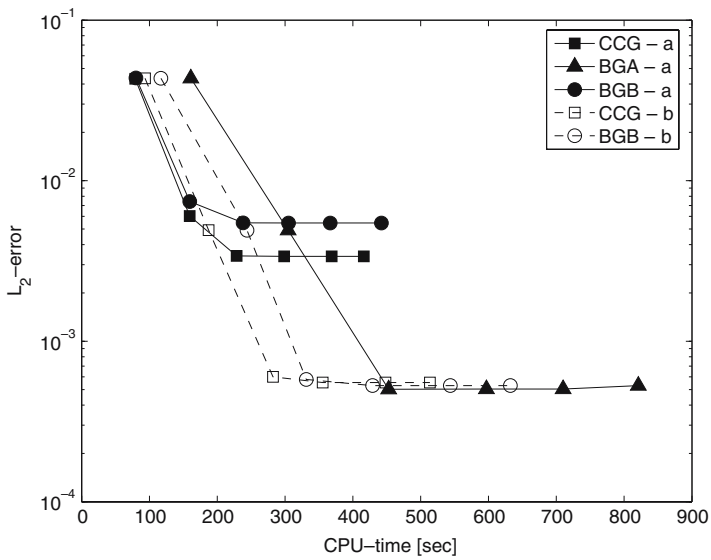


**Fig. 9** $L_2$-error versus CPU time: numerical solution of Kovasznay problem on collapsed Cartesian grid (CCG), barycentric grid of a type A (BGA) and B (BGB); inconsistent numerical integration of nonlinear term—solid line (**a**) and consistent numerical integration—dashed line (**b**); $\mathbf{v}_P \in \mathcal{V}^6$. The computations were performed on an Intel(R) Xeon(TM) CPU 3·06 GHz and 2GB of memory

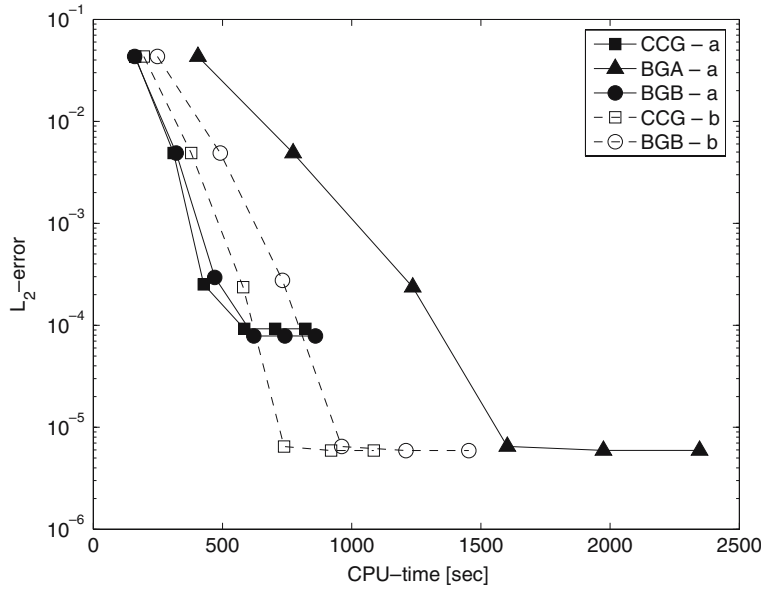


**Fig. 10** $L_2$ − error versus CPU time: numerical solution of Kovasznay problem on collapsed Cartesian grid (CCG), barycentric grid of a type A (BGA) and B (BGB); inconsistent numerical integration of nonlinear term—solid line (**a**) and consistent numerical integration—dashed line (**b**); $\mathbf{v}_P \in \mathcal{V}^8$. The computations were performed on an Intel(R) Xeon(TM) CPU 3·06 GHz and 2GB of memory

time does not reflect the time consumed for construction of the linear operators, which is done only once at the beginning of the simulation. We recall that for all tests we use the same numerical scheme, the same initial condition, and the same size of a time step. However, the nonlinear term is computed using different grids, which results in distinct performance of the solver. In this test, the sum-factorization technique for numerical integration on CCG (4) was not implemented. For solutions obtained on the Cartesian grid and on the barycentric grid of type B we present the CPU time for (a) inconsistent numerical integration and (b) consistent numerical integration. For solution on barycentric grid of a type A we show only one case, with the number of quadrature points consistent for differentiation but not sufficient for consistent integration of the nonlinear terms. The results in both figures show that, at least for the Kovasznay flow, we consider here the collapsed Cartesian grid is the most efficient, even without implementing the sum-factorization technique.

### 4.3 Stability

In this section we show that the stability criteria are practically independent of the grid type where the basic numerical operations are performed. However, when the size of timestep approaches its critical value, the BGA grid exhibits some advantages.

The Kovasznay problem was solved again using different spatial accuracy and variable timestep. The conservative formulation of the nonlinear term was employed. The critical timestep for stability ($\Delta t_{\text{stable}}$) was numerically evaluated; in Table 3 we summarize the results in terms of the $L_2$-error in the numerical solution at $T = 6$ (steady state), obtained using different grids. In the second column we show the numerical error obtained using $\Delta t_{\text{stable}}$; one can see that the numerical error is not affected by the choice of grid and is solely determined by the order of polynomial expansion, $P$, as expected. In the third column we present results obtained using marginally stable $\Delta t$; here the choice of a grid does affect the results. Columns four and five present results of simulation with slightly large size of $\Delta t$.

The relation between $P$ and the critical timestep, $\Delta t_{\text{stable}}$, was obtained heuristically in the following way: for fixed order of polynomial expansion, $P$, the timestep was continuously increased until an instability in the numerical solution appeared; the instability was characterized by a *sudden* loss of accuracy. Then, $\Delta t_{\text{stable}}$ was defined as the maximum timestep for which the stable solution was obtained. The test was performed for $P = 5, 6, 7, 8, 9$ and was repeated for Adams–Bashforth time-integration of the nonlinear terms with first-, second- and third-order ($AB1$, $AB2$ and $AB3$, respectively). The dependence of $\Delta t_{\text{stable}}$ on $P$ for the three Adams–Bashforth methods is presented in Fig. 11. It is clear that AB1 (Euler forward) method with $\Delta t_{\text{stable}} \propto P^{-1.7}$ is the most stable (this will become clear from analyzing the eigenspectra; see below). Second, we observe that the stability region for barycentric grids and collapsed Cartesian grid is not exactly the same. This is due to the different *quadrature crimes* we commit using inconsistent integration or differentiation. In Table 4 we compare the $L_2$-error of a solution at $T = 6$ with consistent integration and unresolved/resolved differentiation (CCG) to the error of a solution with inconsistent integration but consistent differentiation (BGA); the improvement in $L_2$-error is clear.

The estimate of the numerical error in computing the nonlinear advection operator on a sparse grid can be obtained from the following. First, we project the exact solution of Kovasznay problem onto the space spanned by the basis functions. Second, we use *consistent* numerical differentiation to compute $NL_x, NL_y$ (18). Third, we compute *exactly* the inner product of polynomial $NL_x, NL_y$ terms with a test function, $\phi_k$: $I_{\text{exact}} = (NL, \phi_k)$.

These three steps lead to an accurate projection of the nonlinear terms onto a modal space. Next, we compute the approximate inner product $I_\epsilon$. The nonlinear terms are evaluated and projected using the CCG, BGA and BGB grids with consistent integration on the first two and consistent differentiation on

**Table 3** $L_2$-error for different timesteps

| | $L_2$-error | $L_2$-error | $L_2$-error | $L_2$-error |
|---|---|---|---|---|
| $P = 5$ | | | | |
| | $\Delta t = 2{\cdot}500$ E−3 | $\Delta t = 2{\cdot}550$ E−3 | $\Delta t = 2{\cdot}600$ E−3 | $\Delta t = 2{\cdot}800$ E−3 |
| CCG | 5E−3 | **4E−2** | 1E+0 | unstable |
| BGA | 5E−3 | **5E−3** | unstable | unstable |
| BGB | 5E−3 | **1E−2** | 1E+0 | unstable |
| $P = 6$ | | | | |
| | $\Delta t = 1{\cdot}730$ E−3 | $\Delta t = 1{\cdot}740$ E−3 | $\Delta t = 1{\cdot}750$ E−3 | $\Delta t = 2{\cdot}000$ E−3 |
| CCG | 4E−4 | **2E−1** | 5E−1 | 2E+0 |
| BGA | 3E−4 | **3E−4** | unstable | unstable |
| BGB | 4E−4 | **4E−4** | 6E−3 | unstable |
| $P = 7$ | | | | |
| | $\Delta t = 1{\cdot}270$ E−3 | $\Delta t = 1{\cdot}275$ E−3 | $\Delta t = 1{\cdot}280$ E−3 | $\Delta t = 1{\cdot}290$ E−3 |
| CCG | 4E−5 | **3E−1** | 5E−1 | unstable |
| BGA | 4E−5 | **4E−2** | 5E−1 | unstable |
| BGB | 4E−5 | **4E−2** | 3E−1 | unstable |
| $P = 8$ | | | | |
| | $\Delta t = 9{\cdot}735$ E−4 | $\Delta t = 9{\cdot}749$ E−4 | $\Delta t = 10{\cdot}00$ E−4 | $\Delta t = 12{\cdot}00$ E−4 |
| CCG | 5E−6 | **2E−3** | 1E+0 | unstable |
| BGA | 4E−6 | **5E−6** | 1E+0 | unstable |
| BGB | 4E−6 | **6E−5** | 1E+0 | 5E+1 |
| $P = 9$ | | | | |
| | $\Delta t = 7{\cdot}600$ E−4 | $\Delta t = 7{\cdot}790$ E−4 | $\Delta t = 8{\cdot}000$ E−4 | $\Delta t = 8{\cdot}200$ E−3 |
| CCG | 3E−7 | **3E−4** | 1E+0 | unstable |
| BGA | 3E−7 | **4E−7** | 1E+0 | unstable |
| BGB | 3E−7 | **9E−5** | 2E+0 | 1E+1 |

The number of grid points for CCG and BGB satisfies consistent numerical integration only but for BGA it satisfies consistent numerical differentiation only. $v_P \in \mathcal{V}^i$, $i = 5, 6, 7, 8$. For $v_P \in \mathcal{V}^9$ the number of quadrature points for BGB is sufficient for exact numerical integration of polynomials of order up to 25

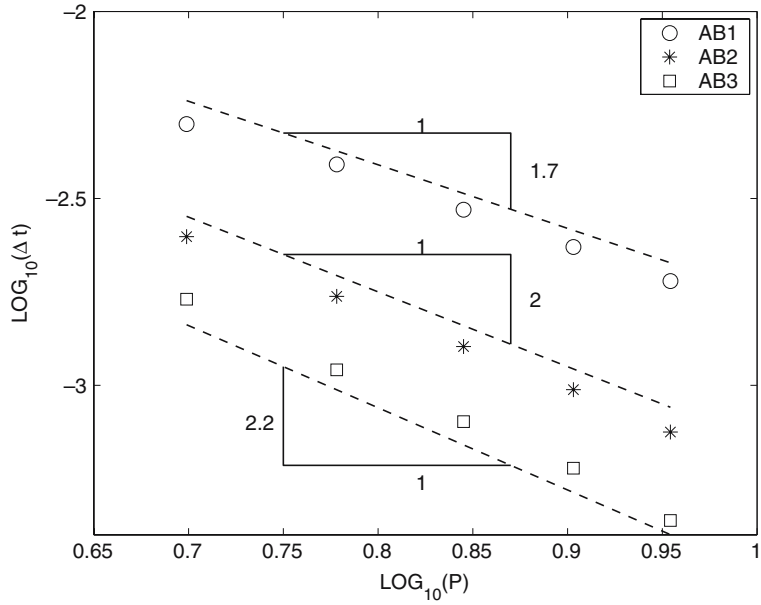


**Fig. 11** Stable $\Delta t_{\text{stable}}$ versus $P$ for solution of the Kovasznay problem. The nonlinear terms, fomulated in conservative form, were computed with Adams–Bashforth method of order 1,2 and 3

**Table 4** $L_2$-error in Kovasznay solution computed on CCG and BGA

| | a) $P_d(CCG) < P_d(BGA)$ | b) $P_d(CCG) = P_d(BGA)$ |
|---|---|---|
| $P = 6$, $\Delta t = 1{\cdot}740\ E{-}3$ | | |
| CCG | 2E−1 | 3E−4 |
| BGA | 3E−4 | 3E−4 |
| $P = 8$, $\Delta t = 9{\cdot}749\ E{-}4$ | | |
| CCG | 2E−3 | 6E−6 |
| BGA | 5E−6 | 5E−6 |
| $P = 9$, $\Delta t = 7{\cdot}790\ E{-}4$ | | |
| CCG | 3E−4 | 4E−7 |
| BGA | 4E−7 | 4E−7 |

The number of quadrature points for BGA satisfies consistent numerical differentiation only. The number of quadrature points for CCG satisfies consistent numerical integration and (a) unresolved numerical differentiation ($P_d(CCG) < P_d(BGA)$), (b) resolved numerical differentiation ($P_d(CCG) = P_d(BGA)$). $\mathrm{v}_P \in \mathcal{V}^i$, $i = 6, 8, 9$

the last one. We define the numerical error in evaluating the nonlinear advection operator as

$$\alpha_{NL} = \mathrm{MAX}_{j,\,k}(I_{\text{exact}} - I_\epsilon), \quad j = 1, .., \mathrm{Nel}, \;\; k = 1, \ldots, K$$

where Nel is a number of elements in the computational domain and $K = (P+1)(P+2)/2$ is the number of test functions. In Table 5 we present the values of $\alpha_{NL}$; note that unresolved differentiation on CCG grid leads to the highest error.

In the semi-implicit time-stepping we have employed, the stability of the three schemes is dictated by the eigenspectra of the advection operator and, of course, the type of the time-stepping method. Let us denote the advection operator by $A$ and its eigenspectra by $\Lambda(A)$. Then, the perturbed advection operator, $A + \Delta A$, has $\Lambda(A + \Delta A) = \Lambda_\epsilon(A)$ eigenspectra, which is also known as *epsilon-pseudospectra* of $A$ [12]. The definition of pseudospectra of a matrix $A$ is given by

$$\Lambda_\epsilon(A) = z \in \mathcal{C} : ||(A - zI)v|| \leq \epsilon$$

**Table 5** Error in the nonlinear advection operator; $\mathbf{v}_P \in \mathcal{V}^8$

| | N | $\alpha_{NL}$ |
|---|---|---|
| CCG | 144 | 8E−8 |
| BGA | 300 | 8E−10 |
| BGB | 105 | 5E−9 |

for some $v \in \mathcal{C}^n$ with $||v|| = 1$. If some of the eigenvalues of $A$ are located in the vicinity of the region of instability, then the *epsilon-pseudo-eigenvalues* might be located inside the unstable region. The distance between the eigenvalues and the *epsilon-pseudo-eigenvalues* depends on $||\Delta A||$. In our case, $||\Delta A||$ is a result of inconsistent numerical evaluation of the advection operator.

In order to analyze the eigenspectrum of the advection operator we simplify the problem by solving the Navier–Stokes equations on a computational domain with two elements as shown in Fig. 12. We use AB1 method for the nonlinear terms. By gradual increase of $\Delta t$ we experimentally obtain the critical timestep, $\Delta t_{\text{stable}}$, for which a stable solution is obtained. Then, we assemble the condensed global linear advection operator $A_G = M^{-1}L$, where

$$L = \left( U\frac{\partial u}{\partial x} + V\frac{\partial u}{\partial y}, \phi \right),$$

and the scalar coefficients $U$ and $V$ denote the maximum values of the corresponding velocity field, $\mathbf{v}_P$; $M$ is a mass matrix. The local, elemental, advection operator is extracted from the global one and its eigenspectra is analyzed. We choose the element where $U$ and $V$ have their largest values. In Fig. 13 we present the scaled (by $\Delta t_{\text{stable}}$) eigenspectra of the *linearized* local advection operator. The dashed line depicts the edge of a stability region for the AB1 time-stepping scheme. We note that the values of $\Delta t_{\text{stable}}$ were obtained from the solution of the *nonlinear* problem, with the advection terms computed with $O(\Delta t)$. For all numerical experiments the scaled eigenvalues fit exactly the region of stability, which suggests that we can analyze stability of the *nonlinear* problem using properly scaled eigensectra which correspond to the *linear* problem.

In Fig. 14 we present the pseudospectra of the linearized local advection operator $A$. Note that one of the eigenvalues is located in the neighborhood of the unstable region. The eigenspectra of the *linear* advection operator do not depend on the type of grid, since the number of grid points suffices to compute the operator exactly. However, the "effective" eigenspectra of the *nonlinear* operator is grid dependent

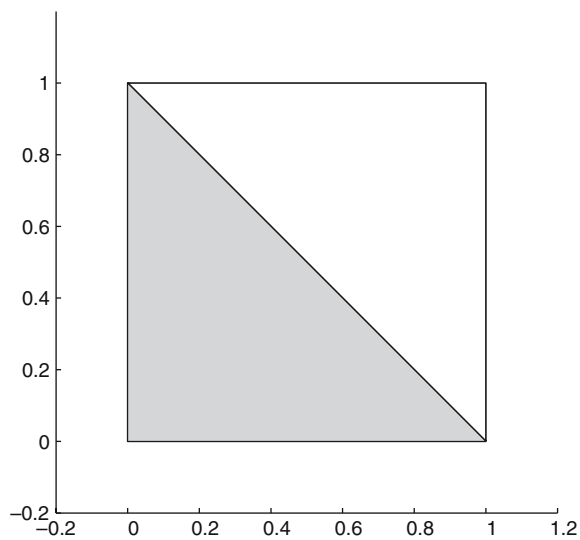


**Fig. 12** Two-element computational domain for solution of Kovasznay problem

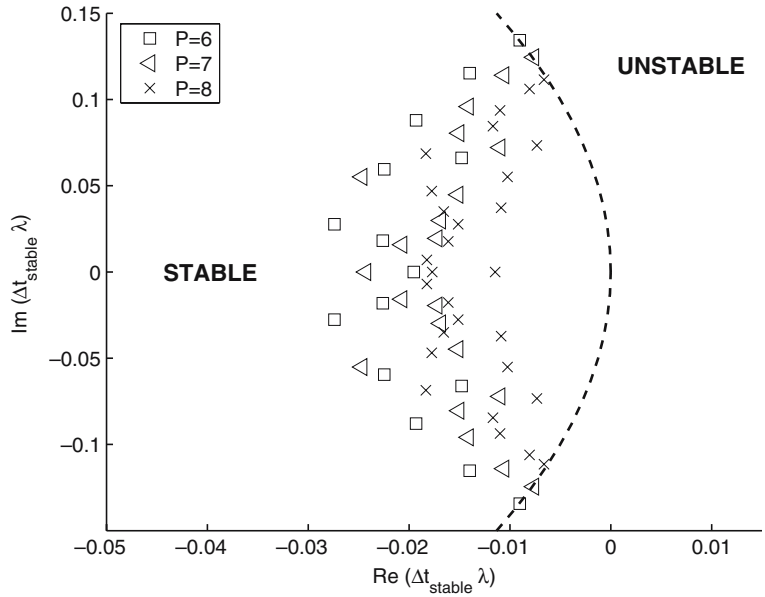


**Fig. 13** Eigenspectrum of linearized local advection operator. $\mathbf{v}_P \in \mathcal{V}^i,\ i = 6, 7, 8$

due to different *quadrature crimes* we commit. The larger the numerical error associated with incorrect integration and differentiation, the larger the distance between eigenvalues of the perturbed operator and those of the exact operator. As seen in Table 5, the error in computing the nonlinear terms on the BGA grid is smaller than the error of the nonlinear terms computed on the CCG grid. The nonlinear advection operator is computed every timestep, which introduces a random shift to the eigenvalues of the perturbed operator from those of the exact one. It appears that the BGA grid minimizes these shifts and provides

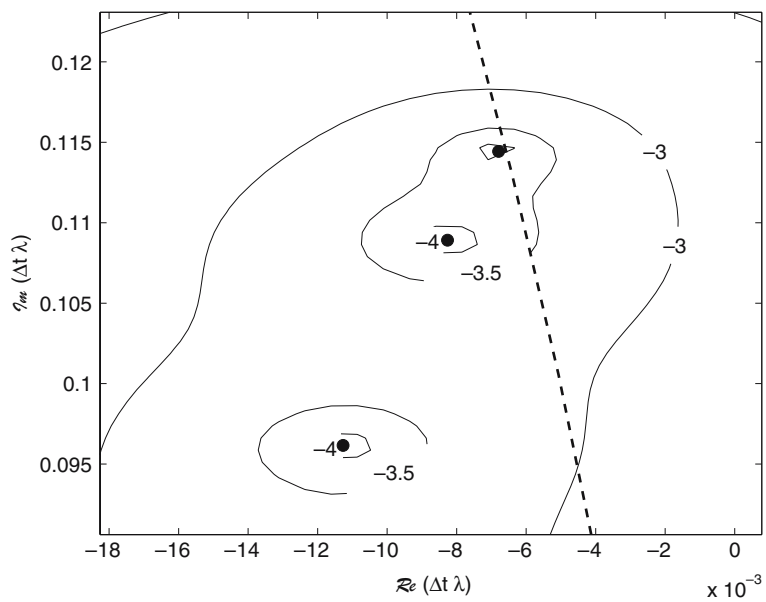


**Fig. 14** Pseudospectra of a linear local advection operator, $A$, computed on a triangular element: dots—represent the eigenvalues of $A$, solid lines—contours of $\mathrm{Log}_{10}$ of *epsilon-pseudo-spectra* for $\epsilon = 10^{-4}, 10^{-3.5}, 10^{-3}$ , dash line—edge of stability region; $\mathbf{v}_P \in \mathcal{V}^8$

a more accurate bound on $\Delta t \Lambda(A + \Delta A)$. Thus, the transition from a stable region to an unstable one depends on $\Delta t$ and less on $\Delta A$ as in the case of CCG.

## 5 Conclusions

We have employed in this paper barycentric grids, typically used in collocation spectral methods on a triangle, to perform numerical integration, differentiation and projection in the Galerkin spectral/*hp* element method. We have examined the individual operations separately as well as in the context of solving the Navier–Stokes equations for incompressible flows. The main findings of this work are the following:

- All three quadrature grids investigated can be used in constructing the discrete operators.
- The choice of the grid, (e.g., symmetric versus nonsymmetric) does not affect the accuracy of the numerical solution as long as a sufficient number of quadrature points is used.
- BGB leads to high efficiency, but the limited number of grid points defined for multivariate quadrature restricts the highest order of polynomial expansion.
- In the solution of nonlinear problems with Galerkin projection the loss of accuracy can be a result of underresolution in *both* numerical integration and differentiation. However, in the solution of the model equation chosen in this paper, if at least one of these numerical operations is consistent, the overall accuracy is maintained.
- The stability properties are nearly independent of the grid types. The performance of barycentric grids is slightly better when the size of a timestep approaches its critical value, while the performance of the collapsed Cartesian grid is the worst due to the *quadrature crimes* we commit due to unresolved differentiation of the nonlinear terms when a conservative form is used.
- Using different grids, deviation of several orders of accuracy in the error of a stable numerical solution may occur. This happens when the size of the time step approaches its critical value in terms of stability. Quadrature crimes we commit in differentiation and integration practically shrink the stability region.
- The collapsed Cartesian grid is overall the most efficient, particularly for high order of polynomial expansions.

**Acknowledgements** This work was partially supported by NSF, ONR and AFOSR

## References

1. Karniadakis GE, Sherwin SJ (2005) Spectral/hp element methods for CFD. 2nd edn. Oxford University Press Oxford
2. Karniadakis GE, Sherwin SJ (1995) A new triangular and tetrahedral basis for high-order finite element methods. Int J Num Methods Engng 38:3775–3802
3. Karniadakis GE, Sherwin SJ (1995) A triangular spectral element method; applications to the incompressible Navier–Stokes equations. Comput Methods Appl Mech Engng 123:189–229
4. Chen Q, Babuška I (1996) The optimal symmetrical points for polynomial interpolation of real functions in the tetrahedron. Comput Methods Appl Mech Engng 137(1):89–94
5. Hesthaven JS (1998) From electrostatics to almost optimal nodal sets for polynomial interpolation in a simplex. SIAM J Numer Anal 35:655–676
6. Taylor MA, Wingate BA, Bos LP (2005) Cardinal function algorithm for computing multivariate quadrature points. SIAM J Numer Anal under review
7. Blyth MG, Pozrikidis C (2005) A Lobatto interpolation grid over the triangle. IMA J Appl Math 71:153–169
8. Strang G, Fix GJ (1973) An analysis of the finite element method. Englewood cliffs Prentice-Hall
9. Wandzura S, Xiao H (2003) Symmetric quadrature rules on triangle. Comput Math Applics 45:1829–1840
10. Warburton T, Pavarino LF, Hesthaven JS (2000) A pseudo-spectral scheme for the incompressible navier-stokes equations using unstructured nodal elements. J Comp Phys 164:1–21
11. Kirby RM, Karniadakis GE (2004) Spectral element and hp methods. In: Stein E, de Borst R, Hughes T (eds) Encyclopedia of computational mechanics. John Wiley & Sons
12. Trefethen LN, Embree M (2005) Spectra and pseudospectra: The behavior of Nonnormal Matrices and operators. Princeton University Press Princeton, N.J