Preprint accepted for publication in Journal of Computational Physics, 2014

http://dx.doi.org/10.1016/j.jcp.2014.10.033

Accuracy-Preserving Boundary Flux Quadrature for Finite-Volume Discretization on Unstructured Grids

Hiroaki Nishikawa *
National Institute of Aerospace, Hampton, VA 23666

Abstract

In this paper, we derive a general accuracy-preserving boundary flux quadrature formula for the second- and third-order node-centered edge-based finite-volume discretizations on triangular and tetrahedral grids. It is demonstrated that the general formula reduces to some well-known formulas for the second-order scheme in the case of linear fluxes. Some special boundary grids are also examined. In particular, a simple one-point quadrature formula, which is typically used for quadrilateral grids, is shown to be exact for quadratic fluxes on triangular with uniformly spaced straight boundaries. Numerical results are presented to demonstrate the accuracy of the general formula, and accuracy deterioration caused by incompatible boundary flux quadrature formulas. In general, the third-order accuracy is lost everywhere in the domain unless the third-order accuracy is maintained at boundary nodes. It is also demonstrated that the third-order scheme does not require high-order curved elements for curved boundaries but requires accurate surface normal vectors defined, which can be estimated by a quadratic interpolation from a given grid, to deliver the designed third-order accuracy.

1 Introduction

Second-order node-centered edge-based finite-volume schemes are widely used in state-of-the-art Computational Fluid Dynamics (CFD) codes, and have been well studied by many researchers in the past decades. The basic algorithm was established in the early work represented by Refs.[1, 2, 3], and further studies have been performed in recent work by Refs.[4, 5, 6, 7, 8]. One of the key elements in the node-centered formulation is the boundary flux quadrature formula. In the node-centered edge-based discretization, the residual needs to be closed at a boundary node, where a boundary condition is imposed weakly, by integrating the flux along the dual boundary faces. It is well known that the second-order accuracy degrades unless the boundary residual is exact for linear fluxes, i.e., fluxes varying linearly in space. Boundary flux quadrature formulas that ensure the exactness for linear fluxes (linearity-preserving formulas) on triangular and tetrahedral grids have been known for a long time, but a formal proof did not seem available in the literature. Recently, a simple proof was presented in Appendix B in Ref.[9], where the linearity-preserving boundary flux quadrature formulas are derived for triangles and tetrahedra, and extended to quadrilaterals, hexahedra, prisms, and pyramids under certain geometrical regularities.

A recent work by Katz and Sankaran [5, 6] revealed that the node-centered edge-based finite-volume discretization achieves third-order accuracy on triangles (tetrahedra in three dimensions) if the nodal gradients are estimated such that they are exact for quadratic functions and both the solution and the flux are linearly extrapolated to the dual-volume face. Since then, there have been efforts for constructing a third-order scheme for the Navier-Stokes equations based on the third-order edge-based scheme [10, 11]. However, our experience shows that third-order accuracy is lost unless the boundary residual is designed to be exact for quadratic fluxes. In an attempt to achieve third-order accuracy at boundary nodes, we found that a straightforward application of the linearly-preserving boundary flux quadrature formulas does not provide third-order accuracy and extensions of the proof described in Ref.[9] to third-order accuracy is not straightforward. In this paper, therefore, we propose another approach to derive boundary flux quadrature that preserves the accuracy of the edge-based

*National Institute of Aerospace, 100 Exploration Way, Hampton, VA 23666-6147, USA. Tel.:+1 757 864 7244. *E-mail address:* hiro@nianet.org

finite-volume discretization at boundary nodes. The basic idea is to generate a boundary stencil from an interior stencil by shrinking an edge. The procedure has been employed routinely in the residual-distribution method for implementing a weak boundary condition (see, e.g., Ref.[12]), but does not seem to have been utilized in the node-centered edge-based finite-volume method. This paper demonstrates that the approach can be applied to the finite-volume method on triangular and tetrahedral grids, and generates a general formula that preserves the order of accuracy of the interior scheme through the boundary nodes. It is demonstrated that the well-known formula for second-order schemes can be easily derived from the general formula. Extension to tetrahedra is also quite straightforward, and a general formula for tetrahedral grids is also presented in this paper. Numerical results demonstrate that the accuracy-preserving boundary quadrature is critical to the construction of the third-order scheme. Third-order accuracy can be lost completely through the domain by an incompatible boundary flux quadrature formula. Also, it is shown that the third-order finite-volume scheme considered in this paper does not require curved elements for problems involving curved boundaries.

The paper is organized as follows. First, we define the node-centered edge-based finite-volume discretization to be investigated. Second, the accuracy of the discretization is discussed for interior stencils, and the need for boundary fluxes is briefly discussed. Then, we derive a general formula for the boundary flux quadrature that preserves the order of accuracy of the interior discretization at boundary nodes. Some special cases are discussed subsequently. Numerical results are then presented to verify the theoretical results and to examine the impact of low-order formulas. Finally, the paper concludes with remarks.

2 Node-Centered Edge-Based Finite-Volume Discretization

2.1 Discretization

Consider a steady conservation law in two dimensions:

$$\operatorname{div} \mathbf{f} = 0, \tag{1}$$

where $\mathbf{f} = (f, g)$ is the flux vector, which is generally a nonlinear function of the solution u. All discussions that follow are based on the scalar equation, but will be valid for systems. Note that the steady conservation law represents various types of equations, including diffusion equations, source terms, time-dependent equations, as they all can be cast in the form of steady conservation laws [13, 14, 15, 16]. We consider the node-centered edge-based finite-volume scheme for Equation (1) in the form:

$$\frac{1}{V_j} \sum_{k \in \{k_j\}} \phi_{jk}(\mathbf{n}_{jk}) = 0, \tag{2}$$

where V_j is the measure of the dual control volume around node j, $\{k_j\}$ is a set of neighbors of j, $\phi_{jk}(\mathbf{n}_{jk})$ is a numerical flux evaluated at the midpoint of the edge $[j, k]$, and $\mathbf{n}_{jk}$ is the scaled directed area vector associated with the edge, i.e., $\mathbf{n}_{jk} = (n_x, n_y) = \mathbf{n}^{\ell}_{jk} + \mathbf{n}^{r}_{jk}$ (see Figure 1). For the numerical flux, we consider the upwind flux of the form:

$$\phi_{jk}(\mathbf{n}_{jk}) = \frac{1}{2}(\mathbf{f}_L + \mathbf{f}_R) \cdot \hat{\mathbf{n}}_{jk} |\mathbf{n}_{jk}| - \frac{1}{2} |a_n| (u_R - u_L) |\mathbf{n}_{jk}|, \tag{3}$$

where $\hat{\mathbf{n}}_{jk} = (\hat{n}_x, \hat{n}_y)$ denotes the unit directed area vector, and $a_n = (\partial f / \partial u)\, \hat{n}_x + (\partial g / \partial u)\, \hat{n}_y$. The left and right solutions and fluxes, $(u_L, u_R, \mathbf{f}_L, \mathbf{f}_R)$, are computed by the linear extrapolation:

$$u_L = u_j + \frac{1}{2} \overline{\nabla} u_j \cdot \Delta \mathbf{r}_{jk}, \quad u_R = u_k - \frac{1}{2} \overline{\nabla} u_k \cdot \Delta \mathbf{r}_{jk}, \tag{4}$$

$$\mathbf{f}_L = \mathbf{f}_j + \frac{1}{2} \overline{\nabla} \mathbf{f}_j \cdot \Delta \mathbf{r}_{jk}, \quad \mathbf{f}_R = \mathbf{f}_k - \frac{1}{2} \overline{\nabla} \mathbf{f}_k \cdot \Delta \mathbf{r}_{jk}, \tag{5}$$

where $\Delta \mathbf{r}_{jk} = (\Delta x_{jk}, \Delta y_{jk}) = (x_k - x_j, y_k - y_j)$, and the over-bar indicates that the gradients are approximate, typically provided by a least-squares (LSQ) fit. The flux extrapolation is required for third-order accuracy; it is not necessary for second-order accuracy [6]. In this paper, we consider two LSQ gradient constructions: one with a linear fit, and the other with a quadratic fit. Linear LSQ gradients are exact for linear functions, and quadratic LSQ gradients are exact for quadratic functions. It is known that the edge-based finite-volume discretization yields second-order accuracy with linear LSQ gradients, and third-order accuracy with quadratic LSQ gradients on triangular/tetrahedral grids [6]. The edge-based finite-volume discretization is applicable to and widely used for triangular, quadrilateral, and mixed grids, but here we focus on triangular and tetrahedral grids, only on which third-order accuracy can be achieved.

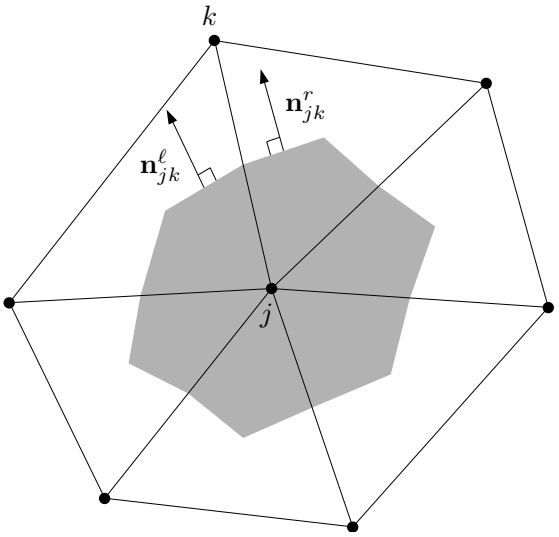

Figure 1: Stencil of the edge-based finite-volume discreetization with scaled outward normals associated with an edge, $\{j, k\}$.

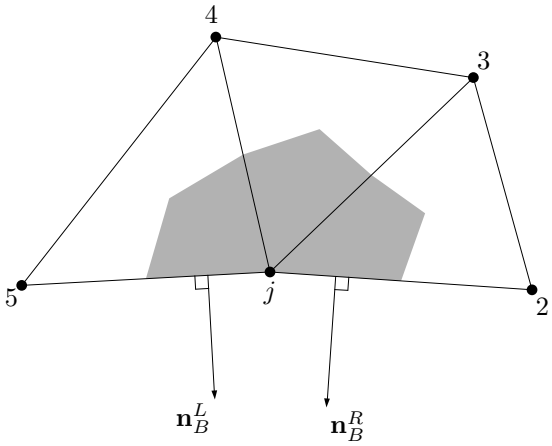

Figure 2: Stencil of the edge-based finite-volume discretization at a boundary node.

2.2 Accuracy of Edge-Based Finite-Volume Scheme

In this section, we discuss the accuracy of the edge-based finite-volume scheme on triangular grids. Accuracy-preserving boundary-flux quadrature formulas that we discuss later will be derived based on the result of this section. Accuracy analyses for the edge-based finite-volume discretization have been performed by various researchers in the past decades. But only a few analyses are available for third-order accuracy: Refs.[5, 6] based on an order analysis, and Ref.[17] based on a Taylor expansion approach. Here, we follow the latter, but the proof is outlined, for simplicity, based on the assumption that the solution is quadratic, which is sufficient for proving third-order accuracy.

For simplicity, we first consider the accuracy of the edge-based discretization for the solution gradient in the integral form:

$$(\overline{\nabla} u_j)_{EB} = \frac{1}{V_j} \sum_{k \in \{k_j\}} \frac{1}{2}(u_L + u_R)\mathbf{n}_{jk}. \tag{6}$$

Let us consider a quadratic solution defined around the node j:

$$u_j + (x - x_j)\partial_x u_j + (y - y_j)\partial_y u_j + \frac{1}{2}(x - x_j)^2 \partial_{xx} u_j + (x - x_j)(y - y_j)\partial_{xy} u_j + \frac{1}{2}(y - y_j)^2 \partial_{yy} u_j, \tag{7}$$

where all the derivatives involved in the above expression are constants defined at the node j. The solution value at the neighbor k is then exactly expressed by

$$u_k = u_j + \partial_{jk} u_j + \frac{1}{2}\partial_{jk}^2 u_j, \tag{8}$$

where we have introduced the notation:

$$\partial_{jk} \equiv \Delta\mathbf{r}_{jk} \cdot (\partial_x, \partial_y). \tag{9}$$

For the nodal gradients used in the linear extrapolation, we assume that they are computed exactly for quadratic solutions:

$$\overline{\nabla} u_j = \nabla u_j, \quad \overline{\nabla} u_k = \nabla u_j + \partial_{jk} \nabla u_j. \tag{10}$$

Consider the difference:

$$\begin{aligned} u_R - u_L &= \left(u_k - \frac{1}{2}\overline{\nabla}u_k \cdot \Delta\mathbf{r}_{jk}\right) - \left(u_j + \frac{1}{2}\overline{\nabla}u_j \cdot \Delta\mathbf{r}_{jk}\right) \\ &= u_j + \partial_{jk}u_j + \frac{1}{2}\partial_{jk}^2 u_j - \frac{1}{2}\left(\nabla u_j + \partial_{jk}\nabla u_j\right)\cdot\Delta\mathbf{r}_{jk} - \left(u_j + \frac{1}{2}\partial_{jk}u_j\right) \\ &= u_j + \partial_{jk}u_j + \frac{1}{2}\partial_{jk}^2 u_j - \frac{1}{2}\partial_{jk}u_j - \frac{1}{2}\partial_{jk}^2 u_j - u_j - \frac{1}{2}\partial_{jk}u_j \\ &= 0. \end{aligned} \tag{11}$$

Hence, the linearly extrapolated solutions, u_L and u_R, are identical to each other at the midpoint of the edge. Note that they are identical but not exact for quadratic solutions. Consequently, we have for the average:

$$\frac{1}{2}(u_R + u_L) = u_L = u_j + \frac{1}{2}\overline{\nabla}u_j \cdot \Delta\mathbf{r}_{jk} = u_j + \frac{1}{2}\nabla u_j \cdot \Delta\mathbf{r}_{jk} = u_j + \frac{1}{2}\partial_{jk}u_j. \tag{12}$$

Summing over the edges, we obtain

$$\begin{aligned} (\overline{\nabla}u_j)_{EB} &= \frac{1}{V_j}\sum_{k\in\{k_j\}}\frac{1}{2}(u_L + u_R)\mathbf{n}_{jk} \\ &= \frac{1}{V_j}\sum_{k\in\{k_j\}}\left(u_j + \frac{1}{2}\partial_{jk}u_j\right)\mathbf{n}_{jk} \\ &= \frac{u_j}{V_j}\sum_{k\in\{k_j\}}\mathbf{n}_{jk} + \frac{\partial_x u_j}{2V_j}\sum_{k\in\{k_j\}}\Delta x_{jk}\mathbf{n}_{jk} + \frac{\partial_y u_j}{2V_j}\sum_{k\in\{k_j\}}\Delta y_{jk}\mathbf{n}_{jk}. \end{aligned} \tag{13}$$

On triangular grids, the following identities hold:

$$\sum_{k\in\{k_j\}}\mathbf{n}_{jk} = \mathbf{0}, \quad \sum_{k\in\{k_j\}}\frac{1}{2}\Delta x_{jk}\, n_x = \sum_{k\in\{k_j\}}\frac{1}{2}\Delta y_{jk}\, n_y = V_j, \quad \sum_{k\in\{k_j\}}\frac{1}{2}\Delta y_{jk}\, n_x = \sum_{k\in\{k_j\}}\frac{1}{2}\Delta x_{jk}\, n_y = 0. \tag{14}$$

Hence, the edge-based formula gives the exact gradient:

$$(\overline{\nabla}u_j)_{EB} = \frac{1}{V_j}\sum_{k\in\{k_j\}}\frac{1}{2}(u_L + u_R)\mathbf{n}_{jk} = \nabla u_j, \tag{15}$$

for quadratic solutions. It implies that the gradient can be estimated to second-order accuracy. However, the formula cannot be used as an explicit formula for second-order gradients because it requires second-order gradients in the linear extrapolation step. An implicit gradient computation may be possible, but it is beyond the scope of the present paper. The above derivation also shows that the exactness of the edge-based integration relies on the non-exactness of the extrapolated solutions at the edge-midpoint. If the extrapolated solutions were exact for quadratic solutions, first-order error terms proportional to the second derivatives would be introduced in Equation (13). Note that the x-derivative and the y-derivative are independently computed to the true values. Thus, the edge-based integration of the flux divergence is also exact for quadratic fluxes on triangular grids:

$$\frac{1}{V_j}\sum_{k\in\{k_j\}}\frac{1}{2}(\mathbf{f}_L + \mathbf{f}_R)\cdot\mathbf{n}_{jk} = \operatorname{div}\mathbf{f}_j. \tag{16}$$

Note also that the dissipative term proportional to $u_R - u_L$ may be added to the average flux without affecting the result because $u_R - u_L = 0$ for quadratic solutions as shown earlier. Therefore, we can write

$$\frac{1}{V_j}\sum_{k\in\{k_j\}}\phi_{jk}(\mathbf{n}_{jk}) = \operatorname{div}\mathbf{f}_j. \tag{17}$$

In general, the flux is a nonlinear function of the solution. In such cases, even for quadratic solutions, errors are committed in the flux gradient, which then generates a second-order error term proportional to the third-derivatives of the fluxes.

$$\frac{1}{V_j}\sum_{k\in\{k_j\}}\phi_{jk}(\mathbf{n}_{jk}) = \operatorname{div}\mathbf{f}_j + Ch^2, \tag{18}$$

where C is a constant depending on the geometry and the third-derivatives of the flux. However, the second-order error vanishes on regular triangulations as shown in Refs.[18, 17, 14]. On a triangular grid composed of right isosceles triangles, we have, for general non-quadratic fluxes,

$$\frac{1}{V_j}\sum_{k\in\{k_j\}}\phi_{jk}(\mathbf{n}_{jk}) \quad = \quad \operatorname{div}\mathbf{f}_j + \frac{h^2}{12}\left[\partial_{xx}(\operatorname{div}\mathbf{f})_j + \partial_{xy}(\operatorname{div}\mathbf{f})_j + \partial_{yy}(\operatorname{div}\mathbf{f})_j\right] + O(h^3), \tag{19}$$

where h corresponds to the length of the two equal sides [14]. The second-error truncation error term contains the second derivatives of the conservation law and thus vanishes for exact solutions. As a result, the truncation error becomes third-order, and the discretization error is also third-order on regular grids, where the truncation error order matches the discretization error. For irregular grids, it is known that the truncation error is one order lower than the discretization error for a first-order system of conservation laws [4, 19]. The second-order error term may no longer vanish, but at the same time Equation (18) guarantees second-order truncation error on arbitrary triangular grids, implying that the discretization error is third order. Finally, we remark that the flux gradients may be evaluated by utilizing the chain rule:

$$\overline{\nabla}\mathbf{f}_j = \left(\frac{\partial\mathbf{f}}{\partial u}\right)_j \overline{\nabla}u_j, \tag{20}$$

where $\left(\frac{\partial\mathbf{f}}{\partial u}\right)_j$ is the flux Jacobian evaluated at node j. As long as the solution gradient is exact for quadratic solutions, the flux gradient is second-order accurate, which is sufficient to ensure the second-order truncation error.

2.3 Boundary Flux Quadrature

At a node located on a boundary, a boundary condition needs to be imposed. It can be imposed either strongly or weakly. If the boundary condition is imposed strongly, then the edge-based discretization is replaced by an algebraic equation such as $u_j - u_B = 0$, where u_j is a solution value at the boundary node j and u_B is a value specified by the boundary condition. In this case, no boundary flux quadrature is required. On the other hand, if the boundary condition is imposed weakly through a boundary flux as described in Ref.[20], then the edge-based discretization needs to be closed by boundary flux contributions in order to define the residual at the boundary node. The solution at the boundary node is then determined by solving the residual equation. The weak boundary approach thus requires boundary flux quadrature. It is the main focus of the discussion that follows.

Consider a boundary stencil shown in Figure 2. Applying the edge-based discretization to this stencil, we obtain

$$\sum_{k=2}^{5}\phi_{jk}(\mathbf{n}_{jk}) = \phi_{j2}(\mathbf{n}^l_{j2}) + \phi_{j3}(\mathbf{n}_{j3}) + \phi_{j4}(\mathbf{n}_{j4}) + \phi_{j5}(\mathbf{n}^r_{j5}). \tag{21}$$

This does not include, yet, the contribution from the boundary faces defined by the face normal vectors, $\mathbf{n}^L_B = (y_j - y_5, x_5 - x_j)/2$ and $\mathbf{n}^R_B = (y_2 - y_j, x_j - x_2)/2$. The edge-based discretization needs to be closed by adding some approximation to the boundary fluxes:

$$\sum_{k=2}^{5}\phi_{jk}(\mathbf{n}_{jk}) + [\text{Boundary Fluxes}] = \phi_{j2}(\mathbf{n}^l_{j2}) + \phi_{j3}(\mathbf{n}_{j3}) + \phi_{j4}(\mathbf{n}_{j4}) + \phi_{j5}(\mathbf{n}^r_{j5}) + \phi_L(\mathbf{n}^L_B) + \phi_R(\mathbf{n}^R_B), \tag{22}$$

where $\phi_L(\mathbf{n}^L_B)$ and $\phi_R(\mathbf{n}^R_B)$ are the boundary flux contributions estimated over the left and right boundary faces, respectively. Note that the boundary contributions are approximations to the flux integration over the two boundary faces of the dual control volume:

$$\phi_L(\mathbf{n}^L_B) \approx \int \mathbf{f}\cdot d\mathbf{n}^L_B, \quad \phi_R(\mathbf{n}^R_B) \approx \int \mathbf{f}\cdot d\mathbf{n}^R_B. \tag{23}$$

The boundary flux integration is discretized typically by a quadrature formula. It is important to note, however, that the accuracy of the quadrature formula by itself does not guarantee the accuracy of the edge-based discretization. The accuracy is determined by the exactness of the overall discretization (22). Therefore, the boundary flux quadrature must be designed to ensure that the overall discretization (22) is exact for linear

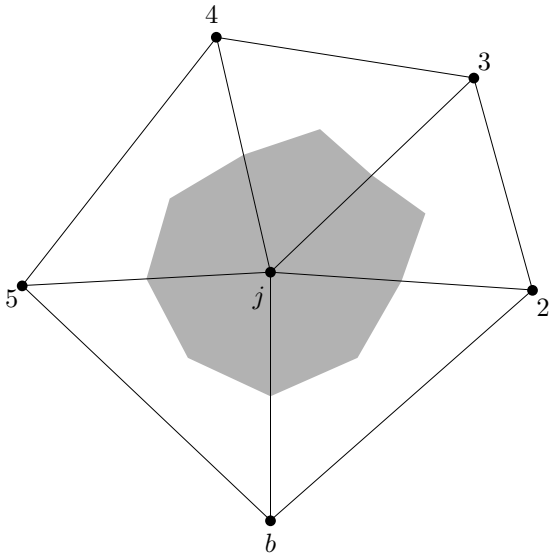

Figure 3: Interior stencil of the edge-based finite-volume discretization.

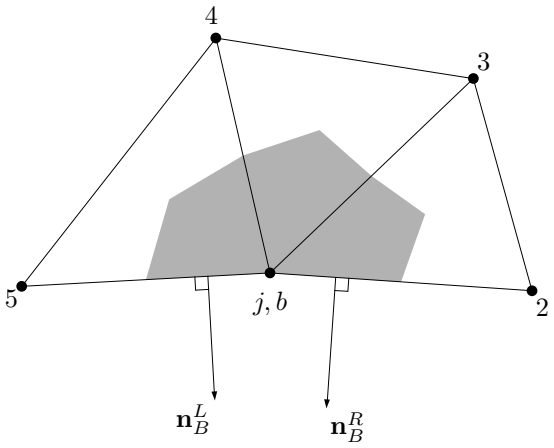

Figure 4: Interior stencil made a boundary stencil by moving the node b to the node j.

and quadratic fluxes for second- and third-order accuracy, respectively. This is the reason, incidentally, that the accuracy-preserving boundary flux quadrature formula depends on the type of elements; see Appendix B in Ref.[9] for the quadrature formulas derived for triangles, quadrilaterals, tetrahedra, hexahedra, prisms, and pyramids for second-order accuracy. For first-order accuracy, it is sufficient to evaluate them as

$$\phi_L(\mathbf{n}_B^L) = \phi_j(\mathbf{n}_B^L), \quad \phi_R(\mathbf{n}_B^R) = \phi_j(\mathbf{n}_B^R). \tag{24}$$

Note that the boundary flux is evaluated at a single node, and thus they carry a single subscript. In practice, it is a function of the current solution at the node and the solution specified by a boundary condition [20]. For second-order accuracy, it is well known (see, e.g., Refs.[2, 4, 9]) that the following quadrature formula guarantees the exactness for linear fluxes on triangular grids:

$$\phi_L(\mathbf{n}_B^L) = \frac{5}{6}\phi_j(\mathbf{n}_B^L) + \frac{1}{6}\phi_5(\mathbf{n}_B^L), \quad \phi_R(\mathbf{n}_B^R) = \frac{5}{6}\phi_j(\mathbf{n}_B^R) + \frac{1}{6}\phi_2(\mathbf{n}_B^R). \tag{25}$$

As we will show later, this formula does not guarantee the exactness for quadratic fluxes, and therefore, if used with the third-order scheme, the third-order accuracy will be lost. A quadrature formula that guarantees third-order accuracy is, to the author's knowledge, not known, and a simple derivation method presented in Appendix B in Ref.[9] does not seem easily extended to quadratic fluxes. The purpose of this paper is, therefore, to propose a new derivation method that extends to third-order accuracy, and to derive a quadrature formula that guarantees the third-order accuracy.

3 Accuracy-Preserving Boundary Flux Quadrature

In this section, we derive a general formula for the boundary flux quadrature that preserves the order of accuracy of the edge-based discretization at boundary nodes. Consider a full interior stencil shown in Figure 3. Applying the edge-based discretization to this stencil, we obtain

$$\sum_{k\in\{k_j\}} \phi_{jk}(\mathbf{n}_{jk}) = \phi_{j2}(\mathbf{n}_{j2}) + \phi_{j3}(\mathbf{n}_{j3}) + \phi_{j4}(\mathbf{n}_{j4}) + \phi_{j5}(\mathbf{n}_{j5}) + \phi_{jb}(\mathbf{n}_{jb}). \tag{26}$$

As shown in the previous section, this is second- and third-order accurate with the gradients computed exactly for linear and quadratic functions, respectively. We propose to convert the interior stencil to a boundary stencil by moving the node b to the node j while keeping the edge-connections. During the process, we observe

$$\mathbf{n}_{j2}^r \to \frac{1}{3}\mathbf{n}_B^R, \quad \mathbf{n}_{jb}^l \to \frac{2}{3}\mathbf{n}_B^R, \quad \mathbf{n}_{jb}^r \to \frac{2}{3}\mathbf{n}_B^L, \quad \mathbf{n}_{j5}^l \to \frac{1}{3}\mathbf{n}_B^L. \tag{27}$$

The stencil then approaches the one shown in Figure 4. In the limit, the edge-based discretization becomes

$$\sum_{k\in\{k_j\}} \phi_{jk}(\mathbf{n}_{jk}) = \phi_{j2}(\mathbf{n}^l_{j2}) + \phi_{j3}(\mathbf{n}_{j3}) + \phi_{j4}(\mathbf{n}_{j4}) + \phi_{j5}(\mathbf{n}^r_{j5}) + \phi_{jb}\left(\frac{2}{3}\mathbf{n}^L_B + \frac{2}{3}\mathbf{n}^R_B\right) + \phi_{j2}\left(\frac{1}{3}\mathbf{n}^R_B\right) + \phi_{j5}\left(\frac{1}{3}\mathbf{n}^L_B\right). \quad (28)$$

The stencil has now become a boundary stencil, which is similar to the one we considered earlier. Compare Figure 4 with Figure 2. Note that we have

$$\phi_{jb}\left(\frac{2}{3}\mathbf{n}^L_B + \frac{2}{3}\mathbf{n}^R_B\right) = \frac{2}{3}\phi_{jb}\left(\mathbf{n}^L_B + \mathbf{n}^R_B\right), \quad \phi_{j2}\left(\frac{1}{3}\mathbf{n}^R_B\right) = \frac{1}{3}\phi_{j2}\left(\mathbf{n}^R_B\right), \quad \phi_{j5}\left(\frac{1}{3}\mathbf{n}^L_B\right) = \frac{1}{3}\phi_{j5}\left(\mathbf{n}^L_B\right). \quad (29)$$

Recall also that the leading error term comes from the average flux term. Thus, we can write

$$\phi_{jb}(\mathbf{n}^L_B + \mathbf{n}^R_B) = \phi_{jb}(\mathbf{n}^L_B) + \phi_{jb}(\mathbf{n}^R_B), \quad (30)$$

without degrading the order of the accuracy. The edge-based discretization for the collapsed interior stencil is then written as

$$\sum_{k\in\{k_j\}} \phi_{jk}(\mathbf{n}_{jk}) = \phi_{j2}(\mathbf{n}^l_{j2}) + \phi_{j3}(\mathbf{n}_{j3}) + \phi_{j4}(\mathbf{n}_{j4}) + \phi_{j5}(\mathbf{n}^r_{j5}) + \frac{2}{3}\phi_{jb}\left(\mathbf{n}^L_B\right) + \frac{2}{3}\phi_{jb}\left(\mathbf{n}^R_B\right) + \frac{1}{3}\phi_{j2}\left(\mathbf{n}^R_B\right) + \frac{1}{3}\phi_{j5}\left(\mathbf{n}^L_B\right). \quad (31)$$

Comparing the resulting expression with Equation (22), which is repeated below,

$$\sum_{k=2}^{5} \phi_{jk}(\mathbf{n}_{jk}) + [\text{Boundary Fluxes}] = \phi_{j2}(\mathbf{n}^l_{j2}) + \phi_{j3}(\mathbf{n}_{j3}) + \phi_{j4}(\mathbf{n}_{j4}) + \phi_{j5}(\mathbf{n}^r_{j5}) + \phi_L(\mathbf{n}^L_B) + \phi_R(\mathbf{n}^R_B), \quad (32)$$

we find

$$\phi_L(\mathbf{n}^L_B) = \frac{2}{3}\phi_{jb}(\mathbf{n}^L_B) + \frac{1}{3}\phi_{j5}(\mathbf{n}^L_B), \quad \phi_R(\mathbf{n}^R_B) = \frac{2}{3}\phi_{jb}(\mathbf{n}^R_B) + \frac{1}{3}\phi_{j2}(\mathbf{n}^R_B). \quad (33)$$

This boundary discretization preserves the accuracy of the interior discretization because no modifications, which would affect the order of accuracy, have been made to the initial interior discretization. Therefore, second- and third-order accuracy is maintained at a boundary node for the second- and third-order accurate edge-based finite-volume discretizations, respectively.

Several remarks are in order. First, the idea of vanishing ghost cells is borrowed from the residual-distribution method, where the vanishing ghost cells are widely utilized for implementing weak boundary conditions (see, e.g., Ref.[12]). The use of a ghost node (or edge) is found in Ref.[21], but it was employed for creating an isotropic stencil for LSQ gradient reconstructions on boundaries, and thus it is not collapsed onto the boundary node. As is done in the residual-distribution method as well as in the finite-volume methods, the node b can serve as a ghost node where physical conditions are specified, and the boundary flux is determined through a numerical flux evaluated by the solution at node j and the ghost state at b [20]. It is emphasized that there are more than one way to define the boundary flux at a boundary node and it may influence the quality of the solution, but it has no impact on the order of accuracy as long as it is consistent with the physical flux. It is beyond the scope of the present work to compare various methods for evaluating the boundary flux. Note that the fluxes ϕ_{j2} and ϕ_{j5} must be evaluated, just like the interior scheme, by the solutions and fluxes linearly extrapolated to the midpoint of the right and left boundary face, respectively. If these fluxes are evaluated simply by the average flux, the general formula reduces to the well-known formula (25), and the scheme will reduce to a second-order accurate scheme. It should be emphasized that these fluxes are evaluated exactly as in the interior scheme and thus do not involve any boundary condition. It is the nodal flux ϕ_{jb} that takes into account boundary conditions. Furthermore, these fluxes are defined at the midpoint of the boundary face, which will not be located at a physical curved boundary, but curved elements are not necessary and should not be introduced. Third-order accuracy relies on a special property on linear triangles, i.e., Equations (14), and therefore it would be lost on curved elements. The derived formula (33) is general, and perfectly matches the order of accuracy between the interior and boundary discretizations: second/third-order accuracy for second/third-order schemes through the boundaries. For tetrahedral grids, the edge-based discretization is known to achieve second- and third-order accuracy with linear and quadratic LSQ fits, respectively. Hence, in the same way, a general formula can be derived for a tetrahedral grid, which is given in Appendix A.

4 Special Cases

4.1 Exact Quadrature for Linear Fluxes

The well-known formula (25), which is exact for linear fluxes, can be derived from the general formula. Consider the general formula for ϕ_L:

$$\phi_L(\mathbf{n}_B^L) = \frac{2}{3}\phi_{jb}(\mathbf{n}_B^L) + \frac{1}{3}\phi_{j5}(\mathbf{n}_B^L). \tag{34}$$

Without loss of generality, we assume zero dissipation and $u_b = u_j$, and take a local coordinate, denoted by s, along the edge $[j, 5]$ from the node j to the node 5, to write

$$\phi_L(\mathbf{n}_B^L) = \frac{2}{3}\mathbf{f}_j \cdot \mathbf{n}_B^L + \frac{1}{6}\left(\mathbf{f}_j + \mathbf{f}_5 - \frac{\partial_s \mathbf{f}_5 - \partial_s \mathbf{f}_j}{2\Delta s}\Delta s^2\right) \cdot \mathbf{n}_B^L, \tag{35}$$

where ∂_s denote the partial differential along the local coordinate and $\Delta s = |\mathbf{n}_B^L|$. The finite-difference term in the parenthesis corresponds to the curvature that is zero for linear fluxes. Therefore, we obtain

$$\phi_L(\mathbf{n}_B^L) = \frac{2}{3}\mathbf{f}_j \cdot \mathbf{n}_B^L + \frac{1}{6}\left(\mathbf{f}_j + \mathbf{f}_5\right) \cdot \mathbf{n}_B^L = \left(\frac{5}{6}\mathbf{f}_j + \frac{1}{6}\mathbf{f}_5\right) \cdot \mathbf{n}_B^L, \tag{36}$$

which is equivalent to the well-known formula (25). In this paper, this formula is referred to as the two-point formula. The same result can be obtained similarly for the other contribution, $\phi_R(\mathbf{n}_B^R)$. Note that the physical flux may be replaced by the numerical flux without degrading the order of accuracy. More importantly, the above derivation shows that this well-known formula cannot be exact for quadratic fluxes because it ignores the curvature term that cannot vanish for quadratic fluxes. This is true even for a simple geometry that we now consider in the following sections.

4.2 Co-Linear Boundary Edges

Consider a straight boundary, where two adjacent boundary edges are co-linear. Assume that $\mathbf{n}_B^R = c\,\mathbf{n}_B^L$, where c is a constant, representing a general non-uniform boundary grid, which includes a boundary grid generated by geometric stretching. For quadratic fluxes, the curvature is constant, and therefore we can write the boundary contributions as

$$\phi_L(\mathbf{n}_B^L) + \phi_R(\mathbf{n}_B^R) = \frac{2}{3}\mathbf{f}_j \cdot \mathbf{n}_B^L + \frac{1}{6}\left(\mathbf{f}_j + \mathbf{f}_5 - \frac{\partial_{ss}\mathbf{f}}{2}\Delta s^2\right) \cdot \mathbf{n}_B^L + \frac{2}{3}\mathbf{f}_j \cdot \mathbf{n}_B^R + \frac{1}{6}\left(\mathbf{f}_j + \mathbf{f}_2 - \frac{\partial_{ss}\mathbf{f}}{2}c^2\Delta s^2\right) \cdot \mathbf{n}_B^R, \tag{37}$$

where $\partial_{ss}\mathbf{f}$ is the constant curvature and we have assumed again zero dissipation in the numerical flux. By a quadratic fit, we obtain the exact curvature for quadratic fluxes as

$$\partial_{ss}\mathbf{f} = \frac{2}{\Delta s^2}\left(\frac{1}{1+c}\mathbf{f}_5 - \frac{1}{c}\mathbf{f}_j + \frac{1}{c(1+c)}\mathbf{f}_2\right). \tag{38}$$

Substituting it into Equation (37), we obtain

$$\begin{aligned}\phi_L(\mathbf{n}_B^L) + \phi_R(\mathbf{n}_B^R) &= \frac{1}{6}\left(5\mathbf{f}_j + \mathbf{f}_5 - \frac{1}{1+c}\mathbf{f}_5 + \frac{1}{c}\mathbf{f}_j - \frac{1}{c(1+c)}\mathbf{f}_2\right) \cdot \mathbf{n}_B^L \\ &+ \frac{1}{6}\left(5\mathbf{f}_j + \mathbf{f}_2 - \frac{c^2}{1+c}\mathbf{f}_5 + c\mathbf{f}_j - \frac{c^2}{c(1+c)}\mathbf{f}_2\right) \cdot \mathbf{n}_B^R. \end{aligned} \tag{39}$$

For the sake of convenience in the implementation within a boundary-edge loop, we arrange the result into the following form by using $\mathbf{n}_B^R = c\,\mathbf{n}_B^L$:

$$\phi_L(\mathbf{n}_B^L) + \phi_R(\mathbf{n}_B^R) = \frac{1}{6}\left[\left(5 + \frac{1}{c}\right)\mathbf{f}_j + c(1-c)\mathbf{f}_5\right] \cdot \mathbf{n}_B^L + \frac{1}{6}\left[(5+c)\,\mathbf{f}_j + \frac{c-1}{c^2}\mathbf{f}_2\right] \cdot \mathbf{n}_B^R. \tag{40}$$

This formula is exact for quadratic fluxes on a straight boundary. The nodes can be randomly distributed. At each node, the ratio of two adjacent edge lengths, c, needs to be available. In this paper, this formula is referred to as the two-point(c) formula.

4.3 Co-Linear Boundary Edges with Uniform Spacing

It is easy to derive a formula for a straight boundary with uniform spacing. We simply set $c = 1$ in the formula obtained in the previous section:

$$\phi_L(\mathbf{n}_B^L) + \phi_R(\mathbf{n}_B^R) = \mathbf{f}_j \cdot \mathbf{n}_B^L + \mathbf{f}_j \cdot \mathbf{n}_B^R. \tag{41}$$

This is identical to a well-known boundary quadrature formula for regular quadrilateral grids, which is exact for linear fluxes for a regular quadrilateral boundary stencil. We have just shown that this formula is exact for quadratic fluxes on triangular grids with uniform boundary grids. This is not a very useful formula for practical problems involving complex geometries, but it is simple and useful for problems in rectangular geometries allowing uniformly spaced boundary grids. In this paper, this formula is referred to as the one-point formula.

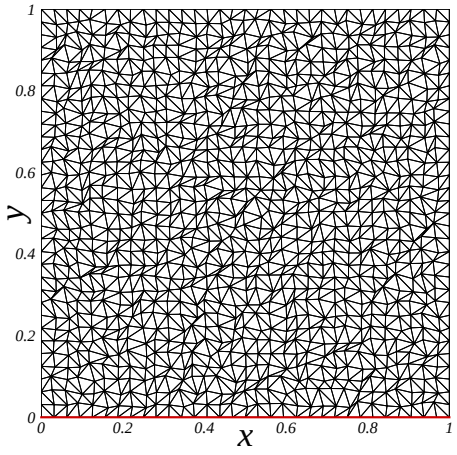

Figure 5: Coarsest grid. 1089 nodes. Red line indicates the boundary where the accuracy is measured.

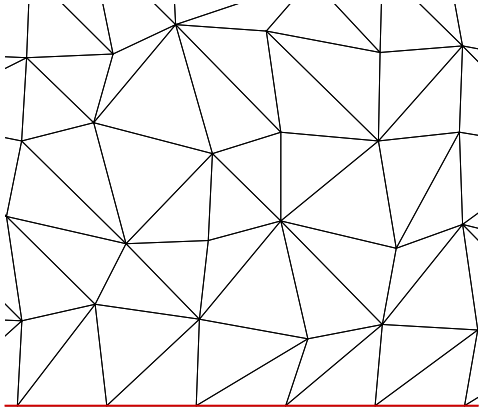

Figure 6: Uniformly spaced boundary.

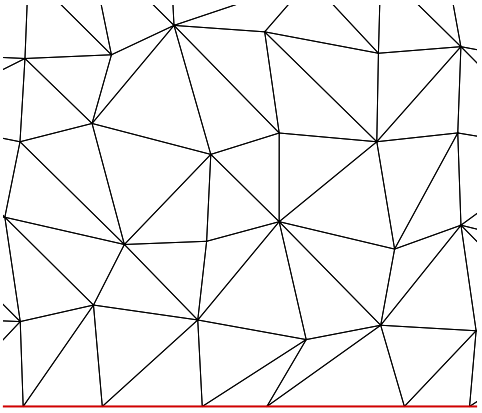

Figure 7: Non-uniformly spaced boundary.

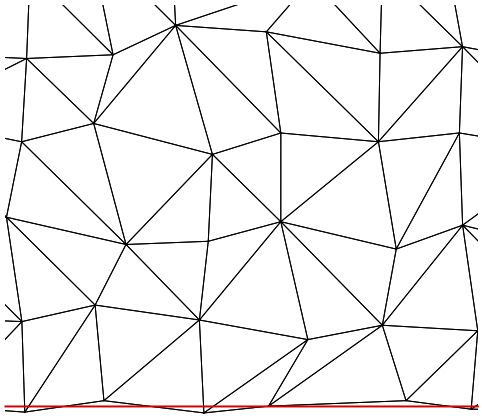

Figure 8: Fully irregular boundary.

5 Results

Numerical experiments have been performed on irregular triangular grids in a square domain and in a curved domain. We consider the linear advection equation, the advection-diffusion equation, Burgers' equation, and the Laplace equation. The advection-diffusion equation and the Laplace equation are written as a first-order hyperbolic system as in Refs.[18, 22]. A source term is added to Burgers' equation, but it is written in the divergence form as in Ref.[14]. Therefore, all equations are written in the form of a steady conservation law as in Equation (1), and discretized in the same way as described in Section 2.1. The nodal gradients are computed by unweighted linear and quadratic LSQ fits for the second- and third-order discretizations, respectively. For the quadratic fit, a two-step implementation as described in Ref.[18] is employed to avoid storing the information about the neighbors of the neighbors and to enable a full implementation in a parallel environment. Truncation errors (TE) are evaluated by substituting an exact solution into the residual. Discretization errors (DE) are computed by solving the finite-volume discretization by an implicit solver as described in [18]. For all cases, the discrete problem is fully solved such that the residuals are reduced by 10 orders of magnitude or to 10^{-15} in the L_1 norm for all equations. Both TE and DE are measured in the discrete L_1 norm, i.e., the arithmetic average of the absolute values of the nodal residuals and errors, respectively. Order of convergence is determined with a representative mesh spacing, h, which is computed as the discrete L_1 norm of the square root of the dual control volume.

5.1 Advection

We consider the linear advection equation,

$$a\,\partial_x u + b\,\partial_y u = 0, \tag{42}$$

where $(a, b) = (1.23, -0.91)$ for the exact solution $u(x, y) = \sin(\pi(bx - ay))$. The domain is taken to be a square: $(x, y) = [0, 1] \times [0, 1]$. On the boundary, the exact solution is specified (i.e., the residual equation is not solved) at $x = 0, 1$, and $y = 1$, and the solution is computed by the numerical scheme at $y = 0$ (see Figure 5). The boundary flux quadrature is required, therefore, only on the boundary $y = 0$, and the boundary flux is evaluated by the solution at the boundary node, which is upwind and suitable for the outflow boundary. The right boundary is also an outflow boundary, but we strongly impose the Dirichlet condition for simplicity; the upwind discretization recognizes the characteristic direction and will not take the solution from the downwind side anyway. The edge-based discretization is constructed as described in Section 2.1. The gradients are computed by an unweighted linear LSQ fit for the second-order scheme, and by an unweighted quadratic LSQ fit for the third-order scheme. We performed the truncation and discretization error computations over a series of ten irregular triangular grids: 1089, 4225, 16641, 66049, 103041, 148225, 198025, 263169, 332929, 409600 nodes, with three different types of boundary grids: uniformly spaced (Figure 6), non-uniformly spaced (Figure 7), and fully irregular (Figure 8). Note that some boundary nodes are located outside the domain in the fully irregular case. But the exact solution is chosen such that it is valid outside the square domain, and it will not cause any problem just like a far-field outer boundary defined quite arbitrarily in a typical external flow problem. To investigate the impact of reduced order of accuracy at boundary nodes on the interior domain, we computed discretization and truncation errors separately for the interior and boundary nodes.

Formula	Interior		Boundary	
	DE	TE	DE	TE
One-point	$O(h^2)$	$O(h)$	$O(h^2)$	$O(h)$
Two-point	$O(h^2)$	$O(h)$	$O(h^2)$	$O(h)$
Two-point(c)	$O(h^2)$	$O(h)$	$O(h^2)$	$O(h)$
General	$O(h^2)$	$O(h)$	$O(h^2)$	$O(h)$

Table 1: Summary of accuracy orders for the second-order advection scheme on triangular grids with uniformly spaced boundary grids.

Formula	Interior		Boundary	
	DE	TE	DE	TE
One-point	$O(h^3)$	$O(h^2)$	$O(h^3)$	$O(h^2)$
Two-point	$O(h^3)$	$O(h^2)$	$O(h^2)$	$O(h)$
Two-point(c)	$O(h^3)$	$O(h^2)$	$O(h^3)$	$O(h^2)$
General	$O(h^3)$	$O(h^2)$	$O(h^3)$	$O(h^2)$

Table 2: Summary of accuracy orders for the third-order advection scheme on triangular grids with uniformly spaced boundary grids.

Results are summarized in Tables 1-6; actual error convergence plots are given in Appendix B (Figures B.1-B.12). It is clearly seen from the tables that the truncation error is one order lower than the discretization error

Formula	Interior		Boundary	
	DE	TE	DE	TE
One-point	$O(h^2)$	$O(h)$	$O(h)$	$O(1)$
Two-point	$O(h^2)$	$O(h)$	$O(h^2)$	$O(h)$
Two-point(c)	$O(h^2)$	$O(h)$	$O(h^2)$	$O(h)$
General	$O(h^2)$	$O(h)$	$O(h^2)$	$O(h)$

Table 3: Summary of accuracy orders for the second-order advection scheme on triangular grids with non-uniformly spaced boundary grids.

Formula	Interior		Boundary	
	DE	TE	DE	TE
One-point	$O(h^3)$	$O(h^2)$	$O(h)$	$O(1)$
Two-point	$O(h^3)$	$O(h^2)$	$O(h^2)$	$O(h)$
Two-point(c)	$O(h^3)$	$O(h^2)$	$O(h^3)$	$O(h^2)$
General	$O(h^3)$	$O(h^2)$	$O(h^3)$	$O(h^2)$

Table 4: Summary of accuracy orders for the third-order advection scheme on triangular grids with non-uniformly spaced boundary grids.

Formula	Interior		Boundary	
	DE	TE	DE	TE
One-point	$O(h^2)$	$O(h)$	$O(h)$	$O(1)$
Two-point	$O(h^2)$	$O(h)$	$O(h^2)$	$O(h)$
General	$O(h^2)$	$O(h)$	$O(h^2)$	$O(h)$

Table 5: Summary of accuracy orders for the second-order advection scheme on triangular grids with fully irregular boundary grids.

Formula	Interior		Boundary	
	DE	TE	DE	TE
One-point	$O(h^3)$	$O(h^2)$	$O(h)$	$O(1)$
Two-point	$O(h^3)$	$O(h^2)$	$O(h^2)$	$O(h)$
General	$O(h^3)$	$O(h^2)$	$O(h^3)$	$O(h^2)$

Table 6: Summary of accuracy orders for the third-order advection scheme on triangular grids with fully irregular boundary grids.

for all cases as expected. We then focus on the discretization error. Tables 1 and 2 summarize the results for uniformly spaced boundary grids. As expected, the one-point formula yields second- and third-order accuracy on uniformly spaced boundary grids. Also shown is that the two-point formula gives only second-order accuracy on the boundary for the third-order scheme. The two-point(c) formula, on the other hand, yields third-order accuracy through the boundary. Similarly, the general formula also achieves the design accuracy. Tables 3 and 4 show the results for non-uniformly spaced boundary grids. It can be seen that the one-point formula leads to the accuracy deterioration on the boundary to first-order as expected. The two-point formula gives the same results as those for uniformly spaced boundary grids. It does not yield third-order accuracy on the boundary. The two-point(c) formula and the general formula produce the same consistent results as before: second/third-order accuracy for the second/third-order scheme everywhere. The same results are obtained, as expected, for fully irregular boundary grids as shown in Tables 5 and 6. The results for the two-point(c) formula are not shown because it is not applicable to these grids; it is a special formula for flat boundaries. We remark that the perturbation in the y-direction is small compared with that in the x-direction as shown in Figure 8, but it is so significant that all schemes become inconsistent at boundary nodes with the two-point(c) formula. It is noteworthy that the order of accuracy in the interior is not affected by any accuracy deterioration on the boundary for all cases. Seemingly, this result is expected for the advection equation because errors generated at the outflow boundary are hard to propagate back into the domain. This is not the case for other equations having isotropic nature, which we will consider in the next section.

5.2 Advection-Diffusion

We consider the advection-diffusion equation,

$$a\,\partial_x u + b\,\partial_y u = \nu(\partial_{xx} u + \partial_{yy} u). \tag{43}$$

It is known that the advection-diffusion equation is equivalent to the following system [18]:

$$\partial_x \mathbf{F} + \partial_y \mathbf{G} = \mathbf{0}, \tag{44}$$

where

$$\mathbf{F} = \begin{bmatrix} au - \nu p \\ -u/T_r - (y - y_j)\, q/T_r \\ (x - x_j)\, q/T_r \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} bu - \nu q \\ (y - y_j)\, p/T_r \\ -u/T_r - (x - x_j)\, p/T_r \end{bmatrix}, \tag{45}$$

where $Tr = (2\sqrt{\nu}\pi)^{-2}$ as described in Ref.[18], and (x_j, y_j) denotes the location of a node at which the system is discretized. The third-order edge-based discretization is constructed based on this system as described in Section 2.1. For the second-order scheme, the terms involving $(x - x_j)$ and $(y - y_j)$, which represent source terms, are not used, and a point-integration has been used for the source terms as described in details in Ref.[18]. Note that u satisfies the advection-diffusion equation (43), and also the gradients are obtained as $(p, q) = (\partial_x u, \partial_y u)$ to the same order of accuracy as demonstrated in Ref.[18]. The nodal gradients, as required in the edge-based discretization, are computed in the same way as in the previous case except that we take $\overline{\nabla} u_j = (p_j, q_j)$ and avoid the gradient computation for u [18]. The resulting second- and third-order edge-based finite-volume schemes are referred to as SchemeII(2nd) and SchemeII(3rd), respectively, as in Ref.[18]. The domain is a square as in the previous case. For this problem, we set $(a, b) = (1.23, 0.12)$, and $\nu = \frac{\sqrt{a^2+b^2}}{Re}$ with $Re = 10$. The following exact solution [23] is specified everywhere on the boundary:

$$u(x, y) = \cos(2\pi\eta) \exp\left(\frac{-2\pi^2\nu}{1 + \sqrt{1 + 4\pi^2\nu^2}}\, \xi\right), \tag{46}$$

where $\xi = ax + by$, $\eta = bx - ay$. For the variables, (p, q), we specify the exact gradients on the boundaries except on the bottom ($y = 0$), where the variable q, corresponding to the normal derivative, is computed by the numerical scheme solving the third equation for q; the boundary flux quadrature is then required. Truncation errors are, therefore, computed for the third equation. In this problem, we focus on the accuracy of q at boundary nodes for isotropic and anisotropic grids.

5.2.1 Isotropic Grids

As in the advection case, numerical results have been obtained for a series of ten irregular triangular grids with three different types of boundary grids: uniformly spaced (Figure 6), non-uniformly spaces (Figure 7), and fully irregular (Figure 8). Results are summarized in Tables 7-12. Again, the actual error convergence plots are given in Appendix B (Figures B.13-B.24). Tables 7 and 8 show the results for uniformly-spaced boundary grids. As expected, for the second-order scheme, all boundary quadrature formulas give second-order discretization errors and first-order truncation errors. But for the third-order scheme, the two-point formula, which is not exact for quadratic fluxes, results in second-order discretization error everywhere. It shows that in contrast to the previous test case, the discretization error in the interior is now affected by degraded accuracy on the boundary. The diffusion part of Equation (44) describes an isotropic wave, modeling the isotropic diffusion, and therefore allows the error committed on the boundary to propagate back into the domain. Note that the system may also be considered as a model for the acoustic wave in the Euler equations, and thus a similar accuracy deterioration is expected for subsonic inviscid flow calculations. Tables 9 and 10 show the results for non-uniformly-spaced boundary grids. As expected, the one-point formula now gives first-order error on the boundary. The discretization error, although deteriorated, stays somewhere between first-order and second-order for the second-order scheme, which is indicated by $O(h^{1.5})$ in the table (see Figures B.15 and B.16 for details). It is interesting that the first-order accuracy on the boundary does not greatly affect the accuracy of the second-order scheme in the interior. For the third-order scheme, however, the discretization error deteriorates clearly to first-order with the one-point formula (see Figures B.21 and B.22). Although not shown, similar accuracy deterioration has been observed for other variables in the interior domain. The accuracy preservation at boundary nodes, therefore, has a serious impact on the third-order scheme. For the two-point formula, the discretization error deteriorates to second-order everywhere for the third-order scheme. On the other hand, the two-point(c) and the general formulas yield second- and third-order accuracy everywhere as expected. The same results have been obtained, as expected, for fully irregular boundary grids; except that the two-point(c) formula is not applicable. See Tables 11 and 12. The results indicate that only the general formula can achieve the third-order accuracy for arbitrary boundary grids.

5.2.2 Anisotropic Grids

To further demonstrate the impact of incompatible boundary flux quadrature, we performed computations on skewed triangular grids. The test problem is taken from Ref.[18]. The exact solution is still given by Equation

Formula	Interior		Boundary	
	DE	TE	DE	TE
One-point	$O(h^2)$	$O(h)$	$O(h^2)$	$O(h)$
Two-point	$O(h^2)$	$O(h)$	$O(h^2)$	$O(h)$
Two-point(c)	$O(h^2)$	$O(h)$	$O(h^2)$	$O(h)$
General	$O(h^2)$	$O(h)$	$O(h^2)$	$O(h)$

Table 7: Summary of accuracy orders in $q(=\partial_y u)$ for the second-order advection-diffusion scheme on triangular grids with uniformly spaced boundary grids.

Formula	Interior		Boundary	
	DE	TE	DE	TE
One-point	$O(h^3)$	$O(h^2)$	$O(h^3)$	$O(h^2)$
Two-point	$O(h^2)$	$O(h^2)$	$O(h^2)$	$O(h)$
Two-point(c)	$O(h^3)$	$O(h^2)$	$O(h^3)$	$O(h^2)$
General	$O(h^3)$	$O(h^2)$	$O(h^3)$	$O(h^2)$

Table 8: Summary of accuracy orders in $q(=\partial_y u)$ for the third-order advection-diffusion scheme on triangular grids with uniformly spaced boundary grids.

Formula	Interior		Boundary	
	DE	TE	DE	TE
One-point	$O(h^{1.5})$	$O(h)$	$O(h)$	$O(1)$
Two-point	$O(h^2)$	$O(h)$	$O(h^2)$	$O(h)$
Two-point(c)	$O(h^2)$	$O(h)$	$O(h^2)$	$O(h)$
General	$O(h^2)$	$O(h)$	$O(h^2)$	$O(h)$

Table 9: Summary of accuracy orders in $q(=\partial_y u)$ for the second-order advection-diffusion scheme on triangular grids with non-uniformly spaced boundary grids.

Formula	Interior		Boundary	
	DE	TE	DE	TE
One-point	$O(h)$	$O(h^2)$	$O(h)$	$O(1)$
Two-point	$O(h^2)$	$O(h^2)$	$O(h^2)$	$O(h)$
Two-point(c)	$O(h^3)$	$O(h^2)$	$O(h^3)$	$O(h^2)$
General	$O(h^3)$	$O(h^2)$	$O(h^3)$	$O(h^2)$

Table 10: Summary of accuracy orders in $q(=\partial_y u)$ for the third-order advection-diffusion scheme on triangular grids with non-uniformly spaced boundary grids.

Formula	Interior		Boundary	
	DE	TE	DE	TE
One-point	$O(h^{1.5})$	$O(h)$	$O(h)$	$O(1)$
Two-point	$O(h^2)$	$O(h)$	$O(h^2)$	$O(h)$
General	$O(h^2)$	$O(h)$	$O(h^2)$	$O(h)$

Table 11: Summary of accuracy orders in $q(=\partial_y u)$ for the second-order advection-diffusion scheme on triangular grids with irregular boundary grids.

Formula	Interior		Boundary	
	DE	TE	DE	TE
One-point	$O(h)$	$O(h^2)$	$O(h)$	$O(1)$
Two-point	$O(h^2)$	$O(h^2)$	$O(h^2)$	$O(h)$
General	$O(h^3)$	$O(h^2)$	$O(h^3)$	$O(h^2)$

Table 12: Summary of accuracy orders in $q(=\partial_y u)$ for the third-order advection-diffusion scheme on triangular grids with irregular boundary grids.

(46) with $Re = 10$, but the domain is taken as $(x, y) = [0, 1] \times [0, 0.01]$. Highly skewed grids have been generated by mapping the grids generated in a unit square domain into the new domain. Five levels of grids have been used: 1089, 4225, 16641, 66049, 103041 nodes. As in the previous case, all variables are specified by the exact solution on all boundaries except that q, corresponding the y-derivative of u, is computed on the bottom boundary by the numerical scheme. Note that the y-derivative on the bottom boundary is the normal derivative, modeling the viscous stress in viscous boundary-layer computations. The residual at a boundary node, therefore, requires the boundary flux contribution. For this problem, the boundary grid is uniformly spaced; we thus consider the one-point and two-point formulas (exact for quadratic and linear fluxes, respectively) and focus on the impact of the accuracy of the boundary flux quadrature formula.

Figure 9 shows the distribution of q on the bottom boundary (for the coarsest grid of 1089 nodes) for second-order schemes: SchemeII(2nd) and Galekrin. Galerkin is included to illustrate the severe skewness present in the grid; Galerkin is known to produce oscillatory solutions on such grids [9, 24]. Note that Galerkin still uses the second-order edge-based finite-volume scheme for the advective term, and only the diffusive term is discretized by the Galerkin method. For Galerkin, the y-derivative is computed from the solution by a linear least-squares method. SchemeII(2nd) is applied with two different boundary flux quadrature formulas. Both yield very

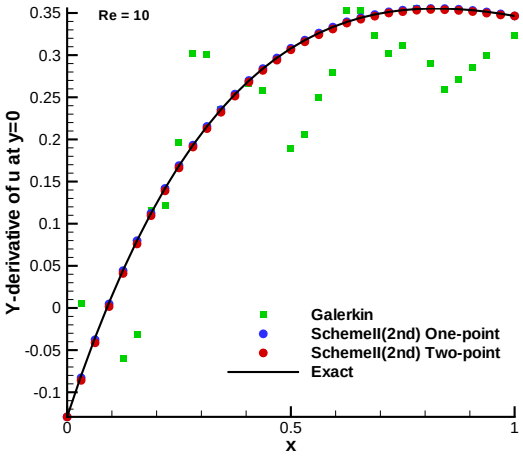

Figure 9: Second-order results. $q(= u_y)$, on the bottom boundary.

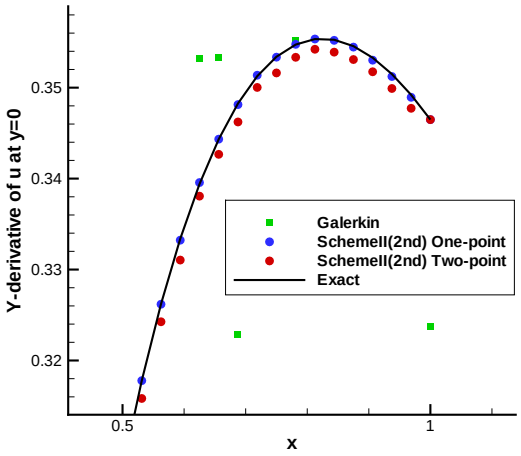

Figure 10: Blow up of Figure 9.

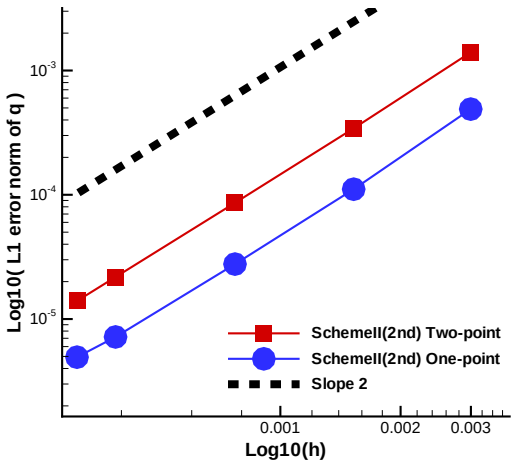

Figure 11: Second-order results. DE for $q(= u_y)$, in the interior.

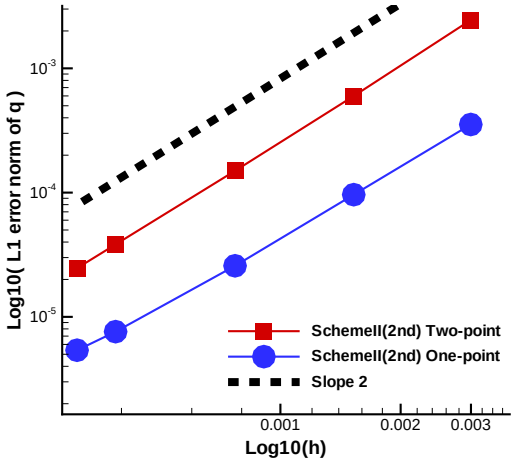

Figure 12: Second-order results. DE for $q(= u_y)$, on the bottom boundary.

accurate derivatives on the boundary, but the one-point formula gives more accurate results as shown in Figure 10. This result was not expected because the errors obtained with these two formulas are almost at the same level in the case of isotropic grids. It implies that the impact of the accuracy of the boundary flux quadrature formula can be large for highly-skewed grids. In terms of the order of accuracy, both quadrature formulas give second-order accuracy for the second-order scheme as shown in Figures 11 and 12. Accuracy deterioration is much more serious for the third-order scheme. Third-order results are shown in Figures 13 and 14. Here, Galerkin(3rd) is a third-order accurate version of Galerkin as described in Ref.[18]; it still exhibits, although small, oscillations in q, which is computed by a quadratic least-squares method. Note that Galerkin(3rd) uses the third-order edge-based finite-volume scheme for the advective part, and only the diffusive term is discretized by the third-order Galerkin method [18]. On the other hand, SchemeII(3rd) gives very accurate results with the one-point formula, but significant accuracy deterioration can be observed for the two-point formula; see Figures 13 and 14. In fact, as expected, the scheme reduces to second-order accurate with the two-point formula, which is confirmed by the error convergence plots in Figures 15 and 16. Note that the deterioration is so severe that it leads to the solution less accurate than those obtained by the second-order scheme.

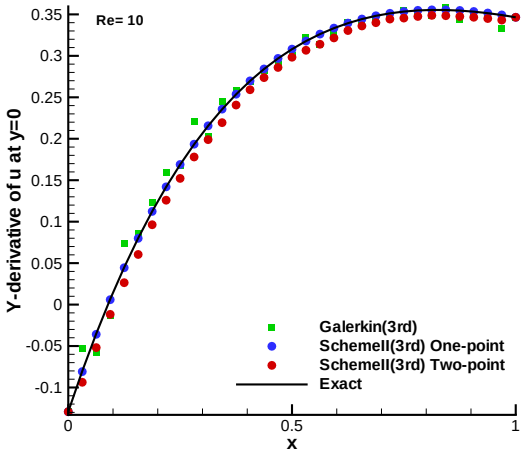

Figure 13: Third-order results. $q(= u_y)$, on the bottom boundary.

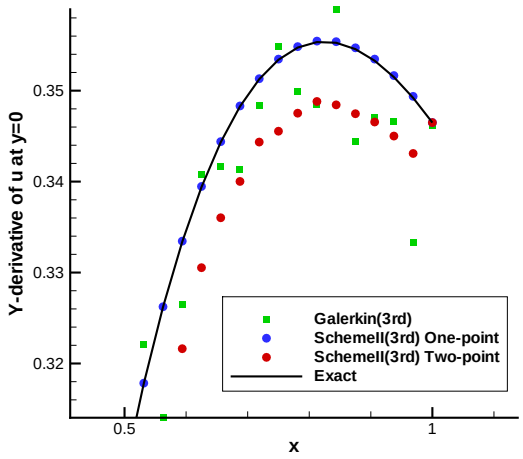

Figure 14: Blow up of Figure 13.

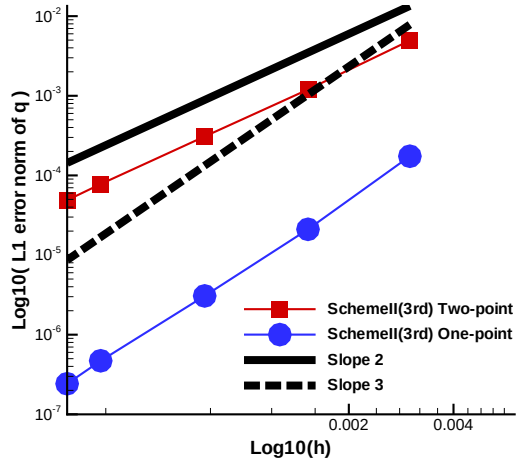

Figure 15: Third-order results. DE for $q(= u_y)$, in the interior.

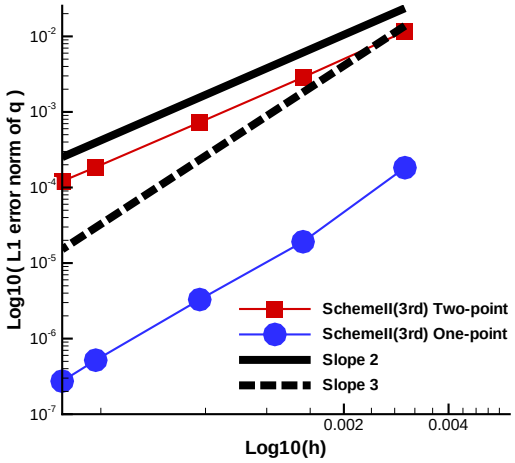

Figure 16: Third-order results. DE for $q(= u_y)$ on the bottom boundary.

5.3 Burgers Equation

As an example of nonlinear equations, we consider Burgers' equation:

$$\partial_x f + \partial_y g = s(x, y), \tag{47}$$

where $(f, g) = (u^2/2, u)$, and $s(x, y) = (1 + \sin(x - y)) \cos(x - y)$. The source term has been introduced, so that the exact solution is given by $u(x, y) = \sin(x - y) + 2$. The equation can be written as a steady conservation law without a source term:

$$\partial_x (f - f^s) + \partial_y (g - g^s) = 0, \tag{48}$$

where (f^s, g^s) is the source flux vector defined by

$$f^s = (x - x_j)s - \frac{1}{2}(x - x_j)^2 \partial_x s + \frac{1}{6}(x - x_j)^3 \partial_{xx} s, \quad g^s = 0, \tag{49}$$

and x_j denotes the x-coordinate of the node at which the residual is defined. Equation (48) is a third-order accurate approximation to the original equation, and therefore the third-order finite-volume scheme for the original equation can be constructed based on Equation (48) without losing third-order accuracy. See Ref.[14] for more details. The edge-based discretization is then constructed as in Section 2.1 by the upwind flux; the source divergence part, $\partial_x f^s + \partial_y g^s = 0$, is discretized by the central flux. Note that the source divergence part is evaluated just once and stored as a forcing term. For the second-order scheme, we discretize the original equation by the upwind flux with a point-integration of the source term. Again, we consider a square domain, and specify the exact solution at the left, right, and bottom boundaries, and compute the solution on the top boundary. The choice of the top boundary is due to the characteristic running upward in the positive y-direction (the characteristic speed is always positive in the y-direction: $\partial g/\partial u = 1$). We use the same grids as in the previous cases, but those with the fully irregular boundary are different in that the perturbation in the y-direction are applied at the top boundary instead of the bottom. The flux gradients are computed from the solution gradients as in Equation (20).

Formula	Interior		Boundary	
	DE	TE	DE	TE
One-point	$O(h^2)$	$O(h^2)$	$O(h)$	$O(1)$
Two-point	$O(h^3)$	$O(h^2)$	$O(h^2)$	$O(h)$
Two-point(c)	$O(h^3)$	$O(h^2)$	$O(h^3)$	$O(h^2)$
General	$O(h^3)$	$O(h^2)$	$O(h^3)$	$O(h^2)$

Table 13: Summary of accuracy orders for the third-order scheme for Burgers' equation on triangular grids with non-uniformly spaced boundary grids.

Formula	Interior		Boundary	
	DE	TE	DE	TE
One-point	$O(h^2)$	$O(h^2)$	$O(h)$	$O(1)$
Two-point	$O(h^3)$	$O(h^2)$	$O(h^2)$	$O(h)$
General	$O(h^3)$	$O(h^2)$	$O(h^3)$	$O(h^2)$

Table 14: Summary of accuracy orders for the third-order scheme for Burgers' equation on triangular grids with fully irregular boundary grids.

It turned out that the results are identical to those for the linear advection equation as shown in Tables 1-6, except that the third-order accuracy deteriorates in the interior with the one-point formula for non-uniform and irregular boundary grids. Tables are shown, therefore, for Burgers' equation only for the problematic cases (see Tables 13 and 14). Error convergence plots are, however, presented for all cases in Appendix B (Figures B.25-B.36). Note again that the order of accuracy in the interior is not affected by any accuracy degradation at boundary nodes in many cases. As in the linear advection case, the characteristic runs through the top boundary to outside the domain, and therefore errors generated at boundary nodes do not seem to propagate back into the domain. However, as noted above, the accuracy in the interior nodes deteriorates when first-order errors are introduced at boundary nodes as shown in Tables 13 and 14 and in Figures B.33 and B.35. The deterioration may look slight in the error convergence plots, but the asymptotic convergence rate is in fact 2.0. Although not shown, the same results have been obtained with a quadratic LSQ fit directly applied to the nonlinear flux instead of Equation (20). These results indicate that the third-order scheme tolerates one order lower accuracy at boundary nodes, but loses the design accuracy with two orders lower accuracy in the nonlinear case. Nevertheless, with the general formula, second- and third-order accuracy is maintained everywhere for arbitrary grids.

5.4 Potential Flow over Curved Boundary

To investigate effects of curved boundaries, we consider a potential flow over a cylinder, which is governed by the Laplace equation for the stream function, ψ:

$$\partial_{xx}\psi + \partial_{yy}\psi = 0. \tag{50}$$

The discretization is performed in the same way as in Section 5.2. We thus write the Laplace equation as a steady conservation law:

$$\partial_x \mathbf{F} + \partial_y \mathbf{G} = \mathbf{0}, \tag{51}$$

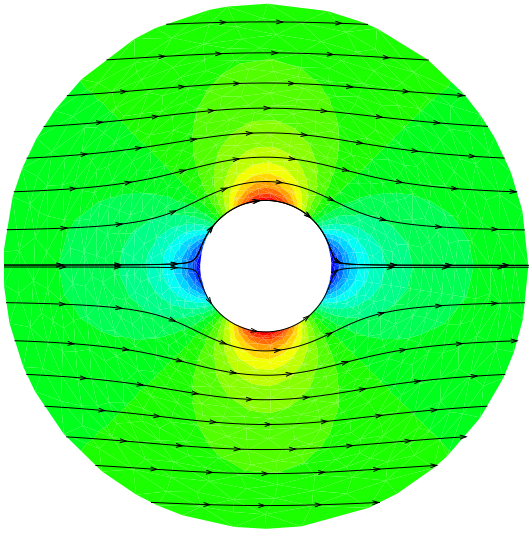

Figure 17: Exact solution contours for u and the streamlines for the cylinder problem.

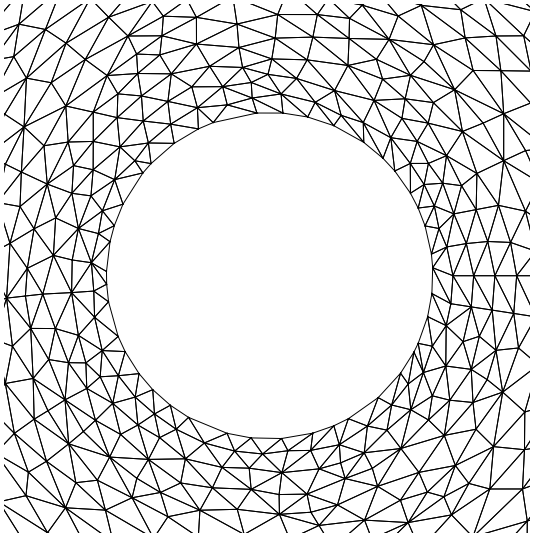

Figure 18: Coarsest irregular triangular grid for the cylinder problem (820 nodes). Boundary nodes are irregularly spaced.

where

$$\mathbf{F} = \begin{bmatrix} -p \\ -\psi/T_r - (y - y_j)\, q/T_r \\ (x - x_j)\, q/T_r \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} -q \\ (y - y_j)\, p/T_r \\ -\psi/T_r - (x - x_j)\, p/T_r \end{bmatrix}, \tag{52}$$

where $T_r = (2\pi)^{-2}$, and then apply the third-order edge-based discretization, i.e. SchemeII(3rd). Again, this scheme is capable of delivering the gradients, $(p, q) = (\partial_x\psi, \partial_y\psi)$ to the same order of accuracy as that of the main variable ψ. In this case, the gradients are equivalent to the velocity components: $u = \partial_y\psi = q$ and $v = -\partial_x\psi = -p$, where u and v are the velocity components in the x- and y-directions, respectively. The domain is taken around a cylinder of unit diameter located at the origin, and the outer boundary is a circle located at the distance of two cylinder-diameters from the origin. The exact solution is known for this problem (see, e.g., [25]). The contours of u and the streamlines are shown in Figure 17. Our focus here is on the accuracy at boundary nodes on the curved boundary of the cylinder, and therefore at the outer boundary, the exact solution is strongly imposed. At each boundary node, j, on the cylinder, the stream function, which is constant, is specified by solving

$$\psi_j = 0. \tag{53}$$

Another condition specified at the boundary node is a slip boundary condition. It is implemented by solving the following equation:

$$(q_j, -p_j) \cdot \hat{\mathbf{n}}_j = 0, \tag{54}$$

where $\hat{\mathbf{n}}_j$ denotes the unit vector normal to the boundary at the node j, whose accuracy has a significant impact on the order of accuracy as we shall see below. The tangential component of the velocity is determined by solving the residual equation projected along the tangential direction:

$$(Res_j(3), -Res_j(2)) \cdot \hat{\mathbf{t}}_j = 0, \tag{55}$$

where $Res_j(2)$ and $Res_j(3)$ are the residuals for the second and third equations, respectively, and $\hat{\mathbf{t}}_j$ is a unit vector satisfying $\hat{\mathbf{t}}_j \cdot \hat{\mathbf{n}}_j = 0$. Note that the projected residual approximates the equation for the tangential velocity, $(u, v) \cdot \hat{\mathbf{t}}_j$, which is equivalent to $(q, -p) \cdot \hat{\mathbf{t}}_j$:

$$\frac{1}{T_r}\left[(\partial_y\psi, -\partial_x\psi) \cdot \hat{\mathbf{t}}_j - (q, -p) \cdot \hat{\mathbf{t}}_j\right] = 0. \tag{56}$$

Note that $Res_j(2)$ and $Res_j(3)$ require a boundary quadrature formula, and that the boundary fluxes need to be computed, for consistency and conservation, based on the edge-normal vectors, $\mathbf{n}_B^L$ and $\mathbf{n}_B^R$ in Figure 4. The normal vector $\hat{\mathbf{n}}_j$ is used only in Equation (54) and for the associated tangent vector $\hat{\mathbf{t}}_j$ in Equation (55). Note also that the above procedure is, in effect, very similar to the weak boundary procedure described in Ref.[26]: the boundary flux is evaluated by the specified boundary condition, leading to a penalty-term formulation. Finally, in the actual implementation, the left hand sides of Equations (53), (54), and (55) form a vector of residual at the boundary node, and they are solved by the implicit solver. We point out that the above boundary condition implementation is only an example, and other implementations are possible. But it is sufficient to illustrate the importance of the boundary flux quadrature formulas, which is the main subject of the paper. Furthermore, it is general enough to examine the effect of the accuracy of the surface normal vector $\hat{\mathbf{n}}_j$, which is often required for practical boundary condition implementations.

Numerical experiments are performed for a series of nine irregular triangular grids: 820, 1830, 3240, 5050, 7260, 9870, 12880, 16290, 20100 nodes. The coarsest grid is shown in Figure 18. Note that the physical boundary is curved, but the grids are composed of linear triangles and therefore the boundary edges are straight. We do not consider high-order (curved) elements, e.g., curved P_2 elements, because the third-order scheme considered here relies on a special property of the edge-based scheme that it achieves third-order accuracy on linear triangles, which would probably be lost on curved elements. Compared with other high-order methods, which fail to achieve the design order of accuracy without curved elements such as the discontinuous Galerkin method [27, 28] and the continuous finite-element-type methods [29, 30, 31], the third-order scheme considered here does not require boundary conditions imposed over the boundary edge (they are imposed at nodes), nor extra degrees of freedom placed between two boundary nodes, which should be located at the physical boundary and thus makes a curved boundary edge inevitable on a curved boundary. We point out that similar third-order schemes exist that preserve third-order accuracy without curved elements [32, 33]. For all the results that we discuss below, the discretization errors are measured for the variable q, corresponding to the x-velocity, u. The truncation error was computed for the third component of the residual vector, i.e., Equation (55).

Formula	Interior		Boundary	
	DE	TE	DE	TE
One-point	$O(h^2)$	$O(h^2)$	$O(h)$	$O(h)$
Two-point	$O(h^2)$	$O(h^2)$	$O(h)$	$O(h)$
General	$O(h^2)$	$O(h^2)$	$O(h)$	$O(h^2)$

Table 15: Summary of accuracy orders for the third-order scheme for the curved boundary problem: Linear approximation to the surface normal vector.

Formula	Interior		Boundary	
	DE	TE	DE	TE
One-point	$O(h^2)$	$O(h^2)$	$O(h^2)$	$O(h)$
Two-point	$O(h^2)$	$O(h^2)$	$O(h^2)$	$O(h)$
General	$O(h^3)$	$O(h^2)$	$O(h^3)$	$O(h^2)$

Table 16: Summary of accuracy orders for the third-order scheme for the curved boundary problem: Exact surface normal vector.

Formula	Interior		Boundary	
	DE	TE	DE	TE
One-point	$O(h^2)$	$O(h^2)$	$O(h^2)$	$O(h)$
Two-point	$O(h^2)$	$O(h^2)$	$O(h^2)$	$O(h)$
General	$O(h^3)$	$O(h^2)$	$O(h^3)$	$O(h^2)$

Table 17: Summary of accuracy orders for the third-order scheme for the curved boundary problem: Quadratic approximation to the surface normal vector.

Formula	Interior		Boundary	
	DE	TE	DE	TE
One-point	$O(h^2)$	$O(h^2)$	$O(h^2)$	$O(h)$
Two-point	$O(h^2)$	$O(h^2)$	$O(h^2)$	$O(h)$
General	$O(h^3)$	$O(h^2)$	$O(h^3)$	$O(h^2)$

Table 18: Summary of accuracy orders for the third-order scheme for the curved boundary problem: Cubic approximation to the surface normal vector.

We first performed the computation with the surface normal vector defined by a typical linear approximation:

$$\hat{\mathbf{n}}_j = \frac{(y_{j+1} - y_{j-1}, x_{j-1} - x_{j+1})}{\sqrt{(x_{j+1} - x_{j-1})^2 + (y_{j+1} - y_{j-1})^2}}, \tag{57}$$

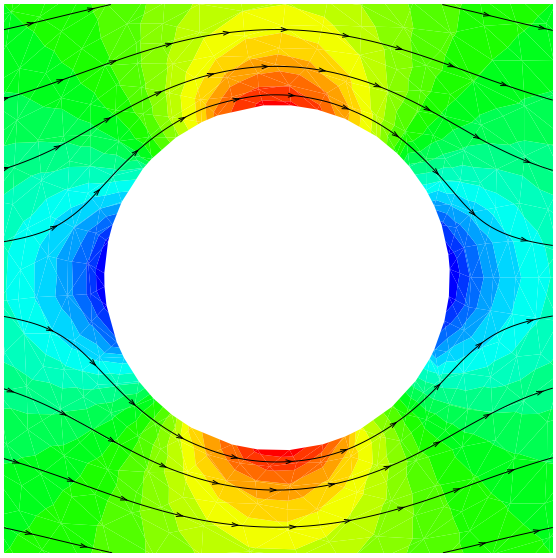

Figure 19: Contours of the x-velocity and the streamlines for the cylinder problem: the general formula and linear approximation to the surface normal vector.

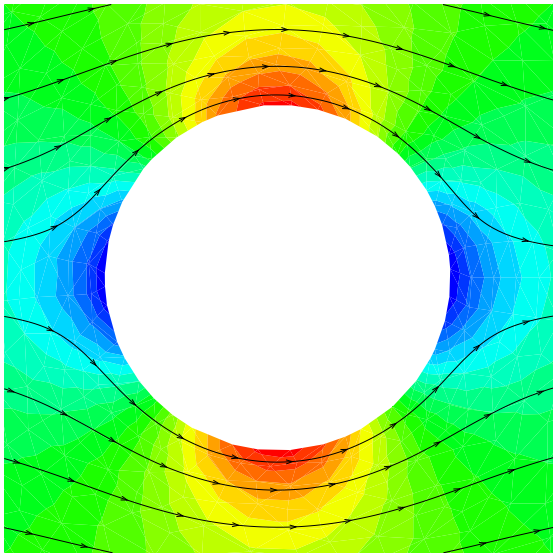

Figure 20: Contours of the x-velocity and the streamlines for the cylinder problem: the general formula and quadratic approximation to the surface normal vector.

where the boundary nodes are assumed to be ordered clockwise. The results are shown in Table 15 (and Figures B.37 and B.38). It is observed that the discretization error deteriorates to somewhere between first-order and second-order on the boundary and to second-order in the interior. It should be noted here that the results for the one-point and two-point formulas are nearly identical to each other. This is because the relevant flux components, i.e., the second and third components of the flux vectors (52), are essentially zero: $\psi = 0$ on the boundary and other components are also zero up to the accuracy of the surface normal vector. It is a very special case but it arises in a practical situation when a no-slip condition is strongly imposed at a wall in a viscous flow. The first-order truncation error on the boundary is, therefore, expected for the one-point and two-point formulas on irregularly-spaced boundary grids, but the accuracy deterioration with the general formula is somewhat unexpected. In fact, the truncation error is second-order everywhere as expected with the general formula, and thus we would expect third-order accuracy in the discretization error. The results imply that the error in the surface normal vector dominates. To confirm the effect of the approximate surface normal vector, we performed the same computation with the exact surface normal vector. Results are shown in Table 16 (and Figures B.39 and B.40). The results are now as expected and consistent with those in the previous sections: inaccurate quadrature formulas fail to achieve the third-order accuracy on the boundary as well as in the interior; only the general formula is capable of achieving the designed third-order accuracy. To recover the design accuracy without the exact normal vector, which may not be available in a general case, we employed a quadratic interpolation to compute a more accurate surface normal vector. A quadratic interpolation is constructed over three consecutive boundary nodes for x and y in the parameter space of the edge length, s, and the surface normal vector is obtained as

$$\hat{\mathbf{n}}_j = \frac{(dy/ds, -dx/ds)_j}{\sqrt{(dx/ds)_j^2 + (dy/ds)_j^2}}. \tag{58}$$

Note that the above does not give the exact normal vector to the cylinder since the exact parametrization of x and y is not quadratic in s. As shown in Table 17 (and Figures B.41 and B.42), the results are essentially the same as those obtained with the exact surface normal vector, thus demonstrating that a quadratically reconstructed normal vector is sufficiently accurate to ensure third-order accuracy. It is observed from Figures 19 and 20, which compare the solutions obtained with the linear and quadratic approximations on the coarsest grid as in Figure 18, that the solution is indeed improved near the boundary, especially near the lower right part of the cylinder, and that no anomalous behaviors [27, 28] are seen in the third-order solution on linear triangles. Even with the quadratic approximation to the surface normal vector, the use of lower-order boundary flux quadrature formulas leads to significant accuracy deterioration as can be seen in Table 17 as well as in Figures 21 and 22. We also tried a cubic interpolation by adding one more node for the interpolation, but

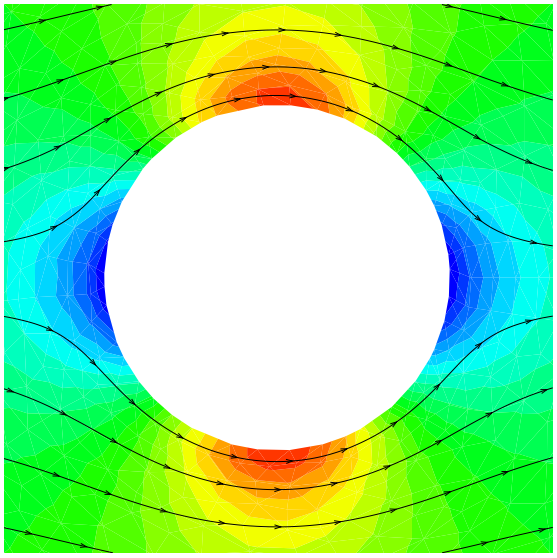

Figure 21: Contours of the x-velocity and the streamlines for the cylinder problem: the one-point formula and quadratic approximation to the surface normal vector.

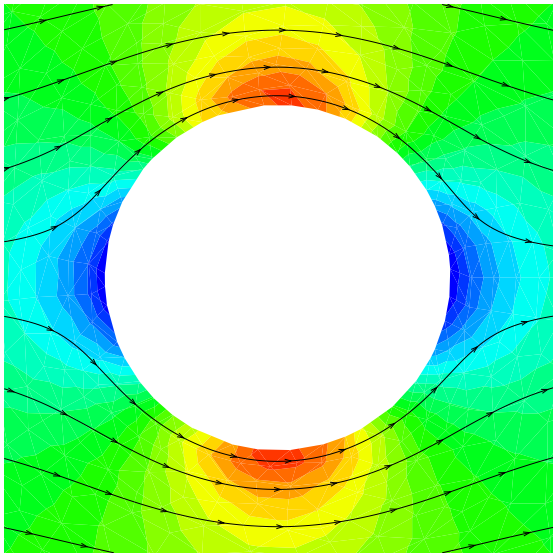

Figure 22: Contours of the x-velocity and the streamlines for the cylinder problem: the two-point formula and quadratic approximation to the surface normal vector.

the results are, again, essentially the same as those obtained with the exact surface normal vector as shown in Table 18 (and Figures B.43 and B.44). These results indicate that accurate surface normal vectors, computed at least by a quadratic interpolation, and the general flux quadrature formula are both critical to the successful implementation of the third-order edge-based scheme.

6 Concluding Remarks

In this paper, a general accuracy-preserving boundary flux quadrature formula has been derived for the node-centered edge-based finite-volume discretization on arbitrary triangular and tetrahedral grids. The general formula has been derived by collapsing an interior stencil onto a boundary stencil. The general formula is exact for linear and quadratic fluxes provided the nodal gradients are computed by linear and quadratic fits, respectively. In the case of linear fluxes, the general formula has been shown to reduce to the well-known formula for the second-order scheme. A special formula, which is exact for quadratic fluxes, has been derived from the general formula at a boundary node whose adjacent boundary edges are co-linear. It is also shown that the special formula reduces to a simple one-point quadrature formula, as typically used for quadrilateral grids, if the adjacent edges have the same length. Thereby, it has been proved that the one-point quadrature formula is exact for quadratic fluxes at a boundary node shared by two co-linear boundary edges of the same length.

The theoretical results have been verified by extensive numerical experiments. Numerical results show that the third-order accuracy cannot be achieved in the interior of the domain in general unless third-order accuracy is guaranteed at boundary nodes. Results for highly-skewed irregular triangular grids show that the accuracy in the normal derivative on a boundary, which models the viscous stress, is greatly affected by the accuracy of the boundary flux quadrature. Numerical results for a curved boundary problem have shown that the third-order node-centered edge-based finite-volume discretization does not require high-order curved elements for curved boundaries but requires accurate surface normal vectors defined at boundary nodes to deliver the designed third-order accuracy. It is also shown that sufficiently accurate surface normal vectors can be obtained conveniently by a quadratic interpolation from a given grid.

A challenge is the extension to other types of elements. The edge-based finite-volume discretization is known to be second-order accurate at best for grids other than pure triangular/tetrahedral grids. Moreover, the formal second-order accuracy can be achieved only for elements with certain geometric regularities (e.g., regular hexahedra or rectangular prisms). In that case, the idea of vanishing ghost cell may be applied to derive a general formula for the second-order scheme; but it will merely reproduce the formulas already derived in Ref.[9]. For irregular quadrilateral/prismatic/pyramidal/hexahedral grids, the second-order accuracy needs to

be established first in the interior stencil, based on which the boundary flux quadrature can be derived by the method proposed in this paper.

Acknowledgments

This work has been funded by the U.S. Army Research Office under the contract/grant number W911NF-12-1-0154. Support by Software CRADLE is also greatly acknowledged.

References

[1] T. J. Barth. Numerical aspects of computing viscous high reynolds number flows on unstructured meshes. AIAA Paper 91-0721, 1991.

[2] H. Luo, J. D. Baum, and R. Löhner. An improved finite volume scheme for compressible flows on unstructured grids. In *Proc. of 33rd Aerospace Sciences Meeting and Exhibit*, AIAA Paper 95-0348, 1995.

[3] A. Haselbacher, J. J. McGuirk, and G. J. Page. Finite volume discretization aspects for viscous flows on mixed unstructured grids. *AIAA J.*, 37(2):177–184, 1999.

[4] B. Diskin and J. L. Thomas. Accuracy analysis for mixed-element finite-volume discretization schemes. *NIA Report No. 2007-08*, 2007.

[5] A. Katz and V. Sankaran. Mesh quality effects on the accuracy of CFD solutions on unstructured meshes. *J. Comput. Phys.*, 230:7670–7686, 2011.

[6] A. Katz and V. Sankaran. An efficient correction method to obtain a formally third-order accurate flow solver for node-centered unstructured grids. *J. Sci.Comput.*, 51:375–393, 2012.

[7] B. Diskin, J. L. Thomas, E. J. Nielsen, H. Nishikawa, and J. A. White. Comparison of node-centered and cell-centered unstructured finite-volume discretizations: Viscous fluxes. *AIAA J.*, 48(7):1326–1338, July 2010.

[8] B. Diskin and J. L. Thomas. Comparison of node-centered and cell-centered unstructured finite-volume discretizations: Inviscid fluxes. *AIAA J.*, 49(4):836–854, 2011.

[9] H. Nishikawa. Beyond interface gradient: A general principle for constructing diffusion schemes. In *Proc. of 40th AIAA Fluid Dynamics Conference and Exhibit*, AIAA Paper 2010-5093, Chicago, 2010.

[10] B. B. Pincock and A. Katz. High-order flux correction for viscous flows on arbitrary unstructured grids. In *Proc. of 21st AIAA Computational Fluid Dynamics Conference*, AIAA Paper 2011-2566, San Diego, California, June 2013.

[11] H. Nishikawa. First, second, and third order finite-volume schemes for Navier-Stokes equations. In *Proc. of 7th AIAA Theoretical Fluid Mechanics Conference, AIAA Aviation and Aeronautics Forum and Exposition 2014*, AIAA Paper 2014-2091, Atlanta, GA, 2014.

[12] J. Dobeš. *Numerical Algorithms for the Computation of Steady and Unsteady Compressible Flow over Moving Geometries Application to Fluid-Structure Interaction.* PhD thesis, Von Karman Institute for Fluid Dynamics, Sint-Genesius-Rode, Belgium, 2007.

[13] H. Nishikawa. A first-order system approach for diffusion equation. I: Second order residual distribution schemes. *J. Comput. Phys.*, 227:315–352, 2007.

[14] H. Nishikawa. Divergence formulation of source term. *J. Comput. Phys.*, 231:6393–6400, 2012.

[15] A. Mazaheri and H. Nishikawa. First-order hyperbolic system method for time-dependent advection-diffusion problems. *NASA-TM–2014–218175*, March 2014.

[16] A. Mazaheri and H. Nishikawa. Very efficient high-order hyperbolic schemes for time-dependent advection-diffusion problems: Third-, Fourth, and Sixth-Order. *Comput. Fluids*, 102:131–147, October 2014.

[17] B. Diskin. Notes on a compact third order edge-based node-centered scheme. Unpublished, March 2011.

[18] H. Nishikawa. First, second, and third order finite-volume schemes for advection-diffusion. *J. Comput. Phys.*, 273:287–309, 2014.

[19] B. Diskin and J. L. Thomas. Notes on accuracy of finite-volume discretization schemes on irregular grids. *Appl. Numer. Math.*, 60:224–226, 2010.

[20] J.-R. Carlson. Inflow/outflow boundary conditions with application to FUN3D. *NASA-TM–2011-217181*, October 2011.

[21] A. Haselbacher and J. Blazek. Accurate and efficient discretization of Navier-Stokes equations on mixed grids. *AIAA J.*, 38(11):2094–2102, 2000.

[22] H. Nishikawa. First-, second-, and third-order finite-volume schemes for diffusion. *J. Comput. Phys.*, 256:791–805, 2014.

[23] H. Nishikawa and P. L. Roe. On high-order fluctuation-splitting schemes for Navier-Stokes equations. In C. Groth and D. W. Zingg, editors, *Computational Fluid Dynamics 2004*, pages 799–804. Springer-Verlag, 2004.

[24] H. Nishikawa. Robust and accurate viscous discretization via upwind scheme - I: Basic principle. *Comput. Fluids*, 49(1):62–86, October 2011.

[25] J. Katz and A. Plotkin. *Low-Speed Aerodynamics*. Cambridge University Press, second edition, 2001.

[26] J. Nordström, Karl Forsberg, Carl Adamsson, and Peter Eliasson. Finite volume methods, unstructured meshes and strict stability for hyperbolic problems. *Appl. Numer. Math.*, 45:453–473, 2003.

[27] F. Bassi and S. Rebay. High-order accurate discontinuous finite element solution of the 2D Euler equations. *J. Comput. Phys.*, 138:251–285, 1997.

[28] L. Krivodonova and M. Berger. High-order accurate implementation of solid wall boundary conditions in curved geometries. *J. Comput. Phys.*, 211:492–512, 2006.

[29] T. Erwin, W. Kyle Anderson, S. Kapadia, and L. Wang. Three dimensional stabilized finite elements for compressible navier-stokes. In *Proc. of 20th AIAA Computational Fluid Dynamics Conference*, AIAA Paper 2011-3411, Honolulu, Hawaii, 2011.

[30] R. Abgrall and D. De Santis. High order residual distribution scheme for Navier-Stokes equations. In *Proc. of 20th AIAA Computational Fluid Dynamics Conference*, AIAA Paper 2011-3231, Hawaii, 2011.

[31] Martin Vymazal, Tiago Quintino, Nadge Villedieu, and Herman Deconinck. High-order upwind residual distribution schemes on isoparametric curved elements. *J. Comput. Phys.*, 230:890–906, 2011.

[32] H. Nishikawa, M. Rad, and P. Roe. A third-order fluctuation-splitting scheme that preserves potential flow. In *Proc. of 15th AIAA Computational Fluid Dynamics Conference*, AIAA Paper 01-2595, Anaheim, 2001.

[33] H. Nishikawa. Multigrid third-order least-squares solution of Cauchy-Riemann equations on unstructured triangular grids. *Int. J. Numer. Meth. Fluids*, 53:443–454, 2007.

Appendix A: Accuracy-Preserving Boundary Flux Quadrature for Tetrahedral Grids

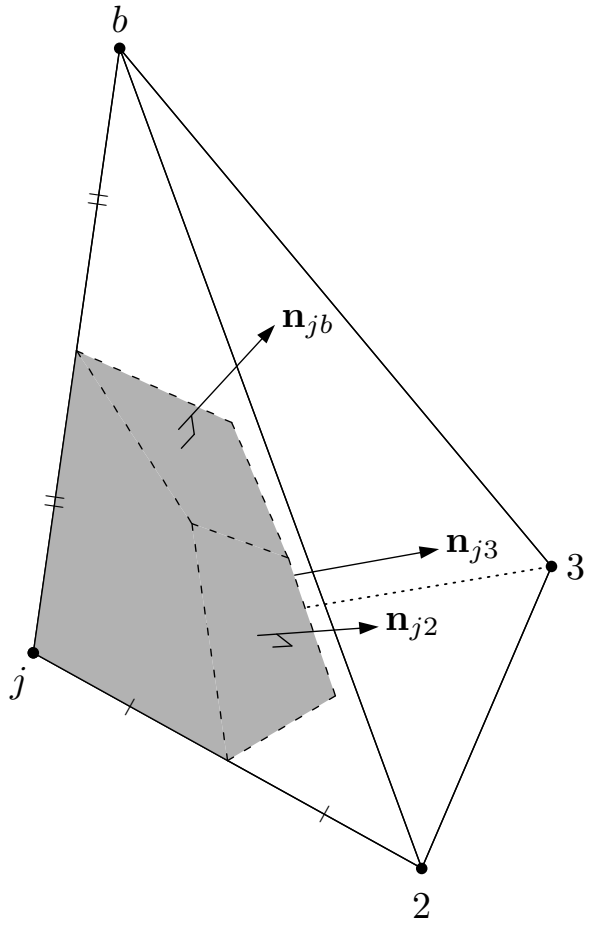

Figure A.1: A dual volume of a tetrahedron around the node j. This is a part of a full interior stencil around the node j; it is then considered as a ghost cell.

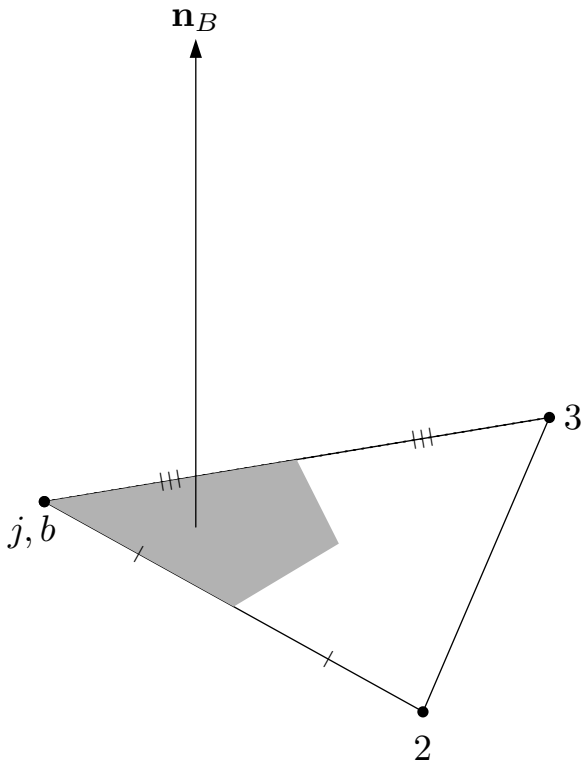

Figure A.2: A boundary element created from the cell in Figure A.1 by moving the node b to the node j. The boundary face normal vector $\mathbf{n}_B$ is pointing outward from the element.

A general boundary flux quadrature formula can be derived in the same way for tetrahedral grids as in the two-dimensional case. In an interior stencil on a tetrahedral grid, the edge-based finite-volume discretization is known to yield third-order accuracy with nodal gradients that are exact for quadratic fluxes. To derive a general boundary formula, it suffices to consider a single cell in a full tetrahedral stencil. Consider a tetrahedral cell in Figure A.1, which should be considered as a part of a full tetrahedral stencil around the node j. The flux contribution from this cell to the edge-based finite-volume discretization at j is given by

$$\phi_{j2}(\mathbf{n}_{j2}) + \phi_{j3}(\mathbf{n}_{j3}) + \phi_{jb}(\mathbf{n}_{jb}). \tag{59}$$

It is straightforward to show

$$\mathbf{n}_{j2} \to \frac{1}{4}\mathbf{n}_B, \quad \mathbf{n}_{jb} \to \frac{1}{2}\mathbf{n}_B, \quad \mathbf{n}_{j5} \to \frac{1}{4}\mathbf{n}_B, \tag{60}$$

as the node b approaches the node j (see Figure A.2). Note that we take the node b as located outside the computational domain. Therefore, by shrinking the edge $[b, j]$ from b to j, we find that the above finite-volume discretization reduces to

$$\frac{1}{4}\phi_{j2}(\mathbf{n}_B) + \frac{1}{4}\phi_{j3}(\mathbf{n}_B) + \frac{1}{2}\phi_{jb}(\mathbf{n}_B). \tag{61}$$

This is a general boundary flux quadrature formula on the dual boundary face on the triangular boundary element $[j, 2, 3]$: second/third-order accuracy is preserved in the boundary discretization. Note that the fluxes $\phi_{j2}(\mathbf{n}_B)$ and $\phi_{j3}(\mathbf{n}_B)$ are computed at the midpoint of the edge $[j, 2]$ and $[j, 3]$, respectively, just like done in the interior scheme. On the other hand, the flux ϕ_{jb} is computed based on the solution at the node j and a solution, u_b, specified by a boundary condition as described in Ref.[20]. A well-known formula that is exact for linear fluxes (valid for second-order schemes) can be derived from the general formula. For linear fluxes, the arithmetic average gives the exact flux at the edge midpoint, and therefore, we can write

$$\phi_{jb}(\mathbf{n}_B) = \mathbf{f}_j \cdot \mathbf{n}_B, \quad \phi_{j2}(\mathbf{n}_B) = \frac{1}{2}(\mathbf{f}_j + \mathbf{f}_2) \cdot \mathbf{n}_B, \quad \phi_{j3}(\mathbf{n}_B) = \frac{1}{2}(\mathbf{f}_j + \mathbf{f}_3) \cdot \mathbf{n}_B, \tag{62}$$

where we have assumed $\mathbf{f}_b = \mathbf{f}_j$ without loss of generality. Substituting these expressions into the general formula, we find

$$\left(\frac{6}{8}\mathbf{f}_j + \frac{1}{8}\mathbf{f}_2 + \frac{1}{8}\mathbf{f}_3\right) \cdot \mathbf{n}_B, \tag{63}$$

which is the well-known formula for the second-order edge-based finite-volume discretization [2].

Appendix B: Error Convergence Plots

Actual discretization error (DE) and truncation error (TE) convergence results are listed. In each figure, DE convergence is shown on the left, and TE convergence is shown on the right. The accuracy orders observed in these plots have been summarized in Tables 1-18. In the plots, DE-I refers to the discretization error in the interior, and DE-B refers to the discretization error on the boundary, and similarly for the truncation errors. Also, UB, NUB, and IRRG refer to uniformly-spaced, non-uniformly-spaced, and irregular boundary grids, respectively.

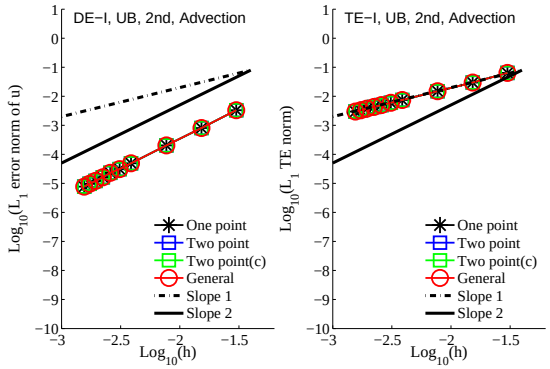

Figure B.1: DE(left) and TE(right) at **interior** nodes for the second-order advection scheme, uniformly spaced boundary grid (UB).

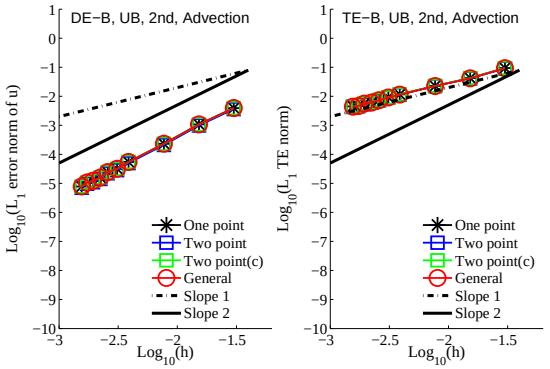

Figure B.2: DE(left) and TE(right) at **boundary** nodes for the second-order advection scheme, uniformly spaced boundary grid (UB).

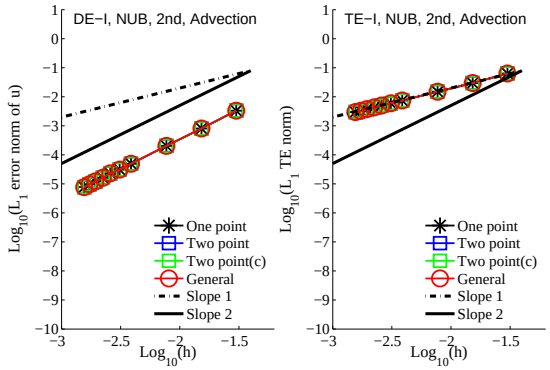

Figure B.3: DE(left) and TE(right) at **interior** nodes for the second-order advection scheme, non-uniformly spaced boundary grid (NUB).

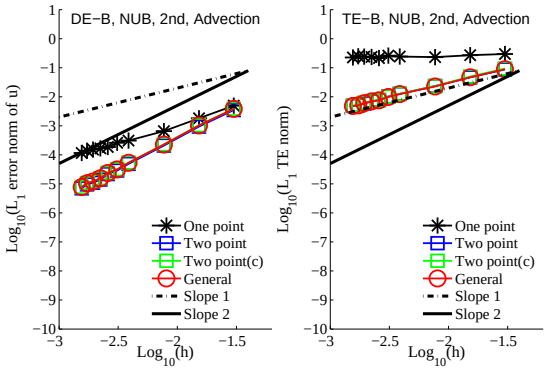

Figure B.4: DE(left) and TE(right) at **boundary** nodes for the second-order advection scheme, non-uniformly spaced boundary grid (NUB).

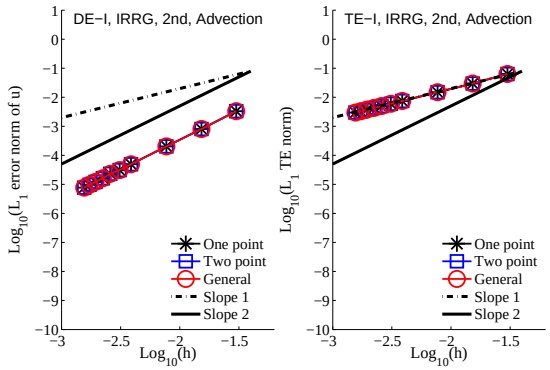

Figure B.5: DE(left) and TE(right) at **interior** nodes for the second-order advection scheme, irregular boundary grid (IRRG).

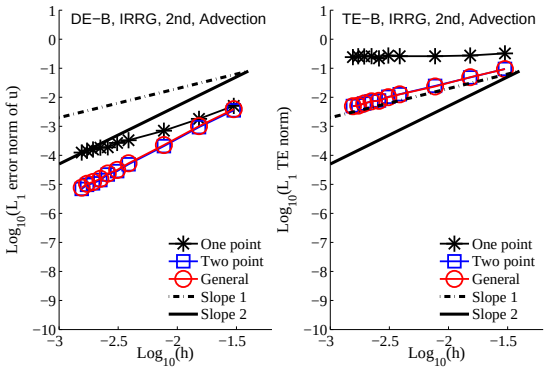

Figure B.6: DE(left) and TE(right) at **boundary** nodes for the second-order advection scheme, irregular boundary grid (IRRG).

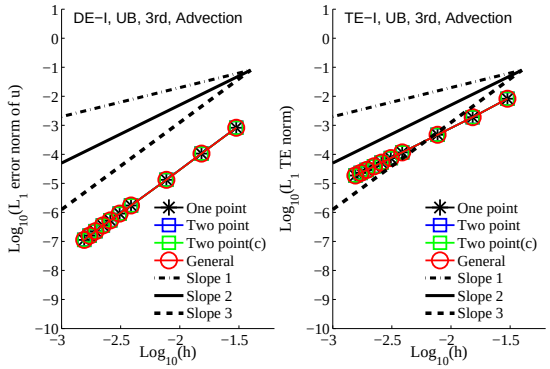

Figure B.7: DE(left) and TE(right) at **interior** nodes for the third-order advection scheme, uniformly spaced boundary grid (UB).

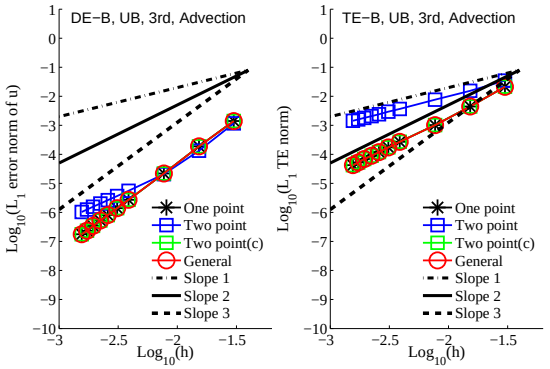

Figure B.8: DE(left) and TE(right) at **boundary** nodes for the third-order advection scheme, uniformly spaced boundary grid (UB).

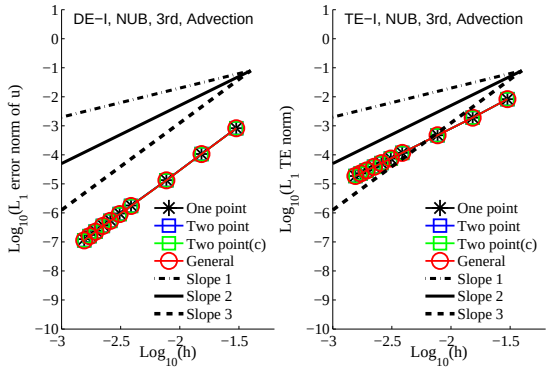

Figure B.9: DE(left) and TE(right) at **interior** nodes for the third-order advection scheme, non-uniformly spaced boundary grid (NUB).

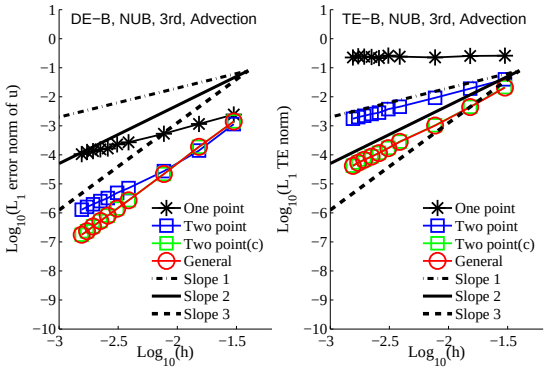

Figure B.10: DE(left) and TE(right) at **boundary** nodes for the third-order advection scheme, non-uniformly spaced boundary grid (NUB).

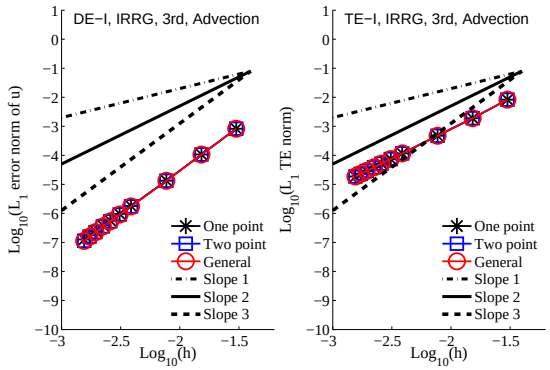

Figure B.11: DE(left) and TE(right) at **interior** nodes for the third-order advection scheme, irregular boundary grid (IRRG).

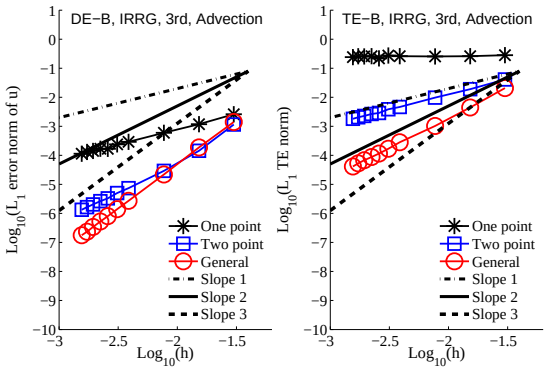

Figure B.12: DE(left) and TE(right) at **boundary** nodes for the third-order advection scheme, irregular boundary grid (IRRG).

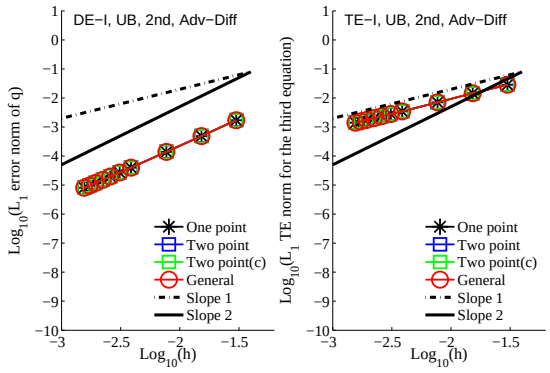

Figure B.13: DE(left) and TE(right) at **interior** nodes for the second-order advection-diffusion scheme, uniformly spaced boundary grid (UB).

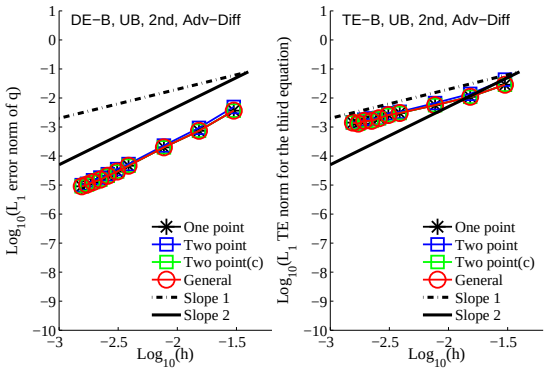

Figure B.14: DE(left) and TE(right) at **boundary** nodes for the second-order advection-diffusion scheme, uniformly spaced boundary grid (UB).

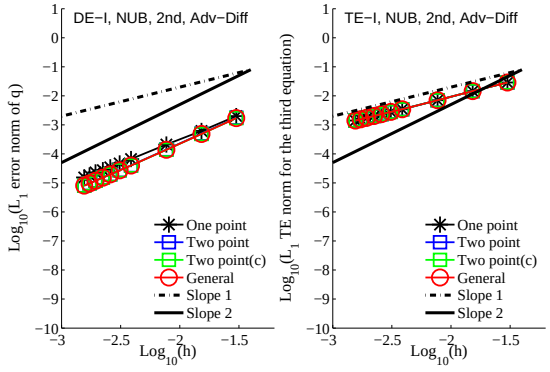

Figure B.15: DE(left) and TE(right) at **interior** nodes for the second-order advection-diffusion scheme, non-uniformly spaced boundary grid (NUB).

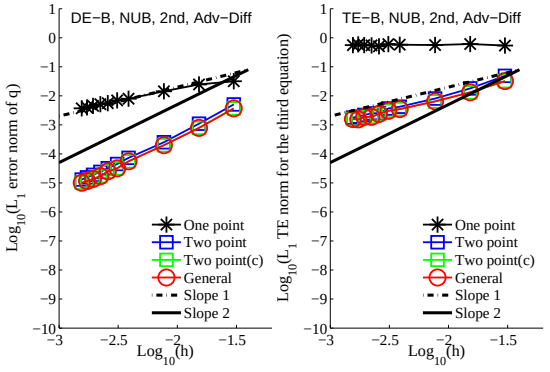

Figure B.16: DE(left) and TE(right) at **boundary** nodes for the second-order advection-diffusion scheme, non-uniformly spaced boundary grid (NUB).

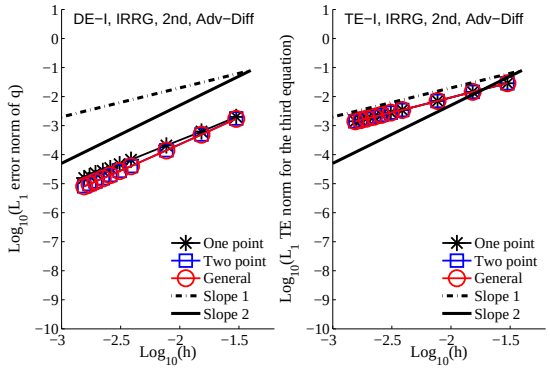

Figure B.17: DE(left) and TE(right) at **interior** nodes for the second-order advection-diffusion scheme, irregular boundary grid (IRRG).

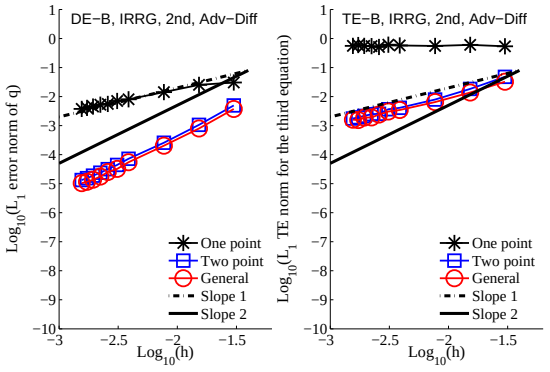

Figure B.18: DE(left) and TE(right) at **boundary** nodes for the second-order advection-diffusion scheme, irregular boundary grid (IRRG).

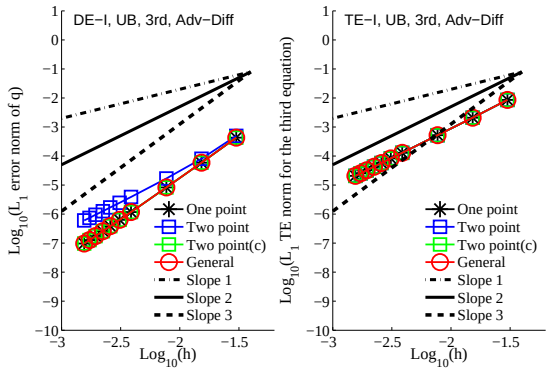

Figure B.19: DE(left) and TE(right) at **interior** nodes for the third-order advection-diffusion scheme, uniformly spaced boundary grid (UB).

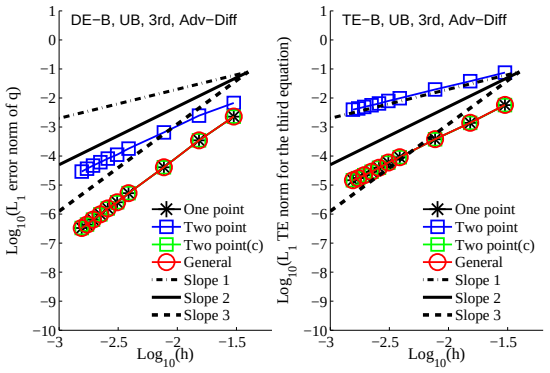

Figure B.20: DE(left) and TE(right) at **boundary** nodes for the third-order advection-diffusion scheme, uniformly spaced boundary grid (UB).

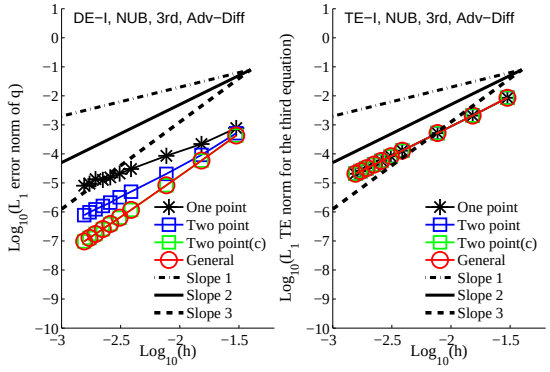

Figure B.21: DE(left) and TE(right) at **interior** nodes for the third-order advection-diffusion scheme, non-uniformly spaced boundary grid (NUB).

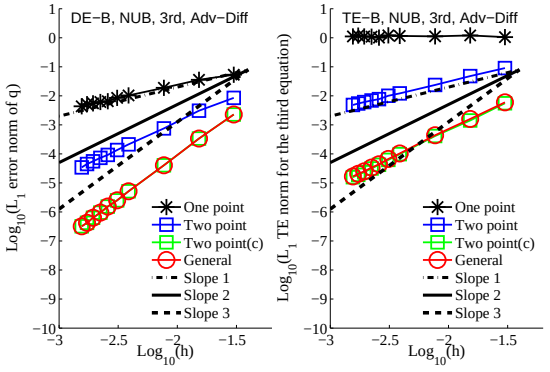

Figure B.22: DE(left) and TE(right) at **boundary** nodes for the third-order advection-diffusion scheme, non-uniformly spaced boundary grid (NUB).

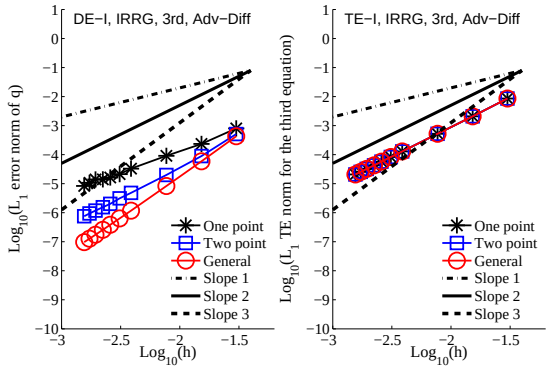

Figure B.23: DE(left) and TE(right) at **interior** nodes for the third-order advection-diffusion scheme, irregular boundary grid (IRRG).

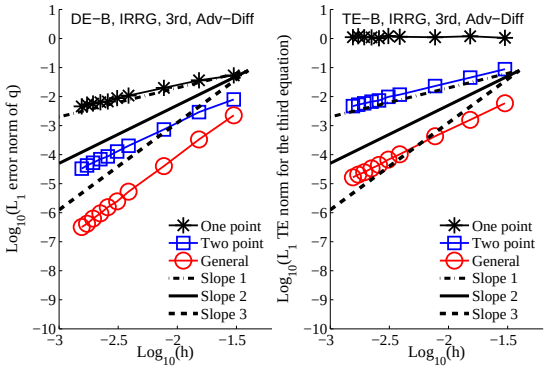

Figure B.24: DE(left) and TE(right) at **boundary** nodes for the third-order advection-diffusion scheme, irregular boundary grid (IRRG).

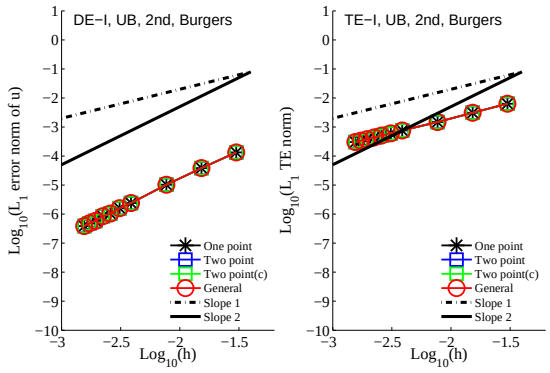

Figure B.25: DE(left) and TE(right) at **interior** nodes for the second-order scheme for Burgers' equation, uniformly spaced boundary grid (UB).

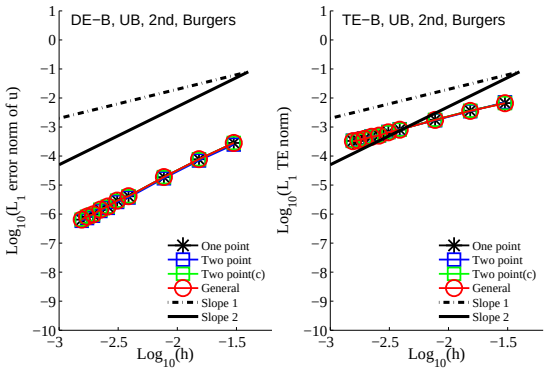

Figure B.26: DE(left) and TE(right) at **boundary** nodes for the second-order scheme for Burgers' equation, uniformly spaced boundary grid (UB).

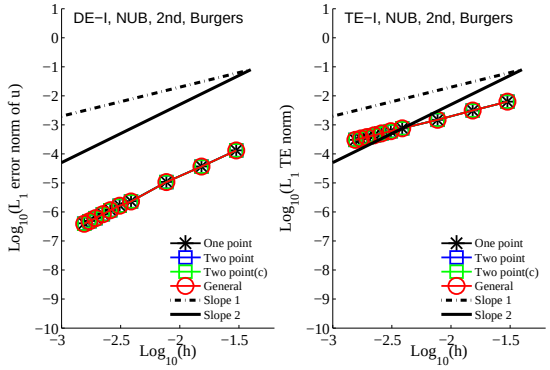

Figure B.27: DE(left) and TE(right) at **interior** nodes for the second-order scheme for Burgers' equation, non-uniformly spaced boundary grid (NUB).

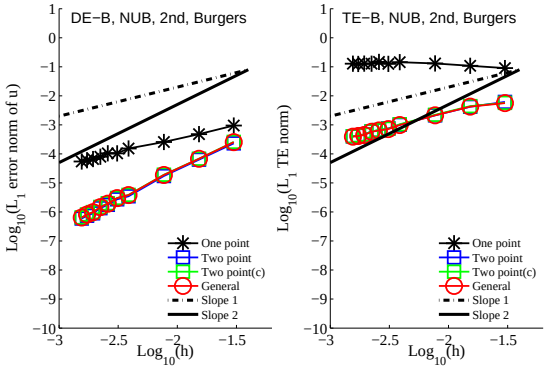

Figure B.28: DE(left) and TE(right) at **boundary** nodes for the second-order scheme for Burgers' equation, non-uniformly spaced boundary grid (NUB).

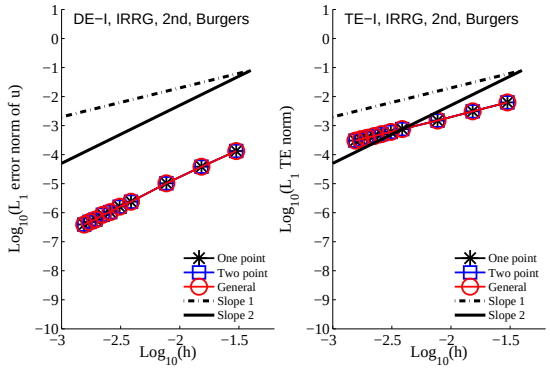

Figure B.29: DE(left) and TE(right) at **interior** nodes for the second-order scheme for Burgers' equation, irregular boundary grid (IRRG).

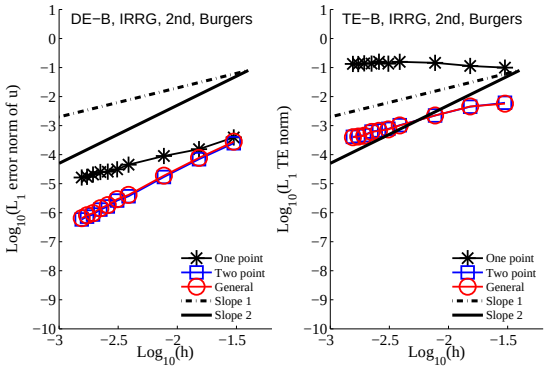

Figure B.30: DE(left) and TE(right) at **boundary** nodes for the second-order scheme for Burgers' equation, irregular boundary grid (IRRG).

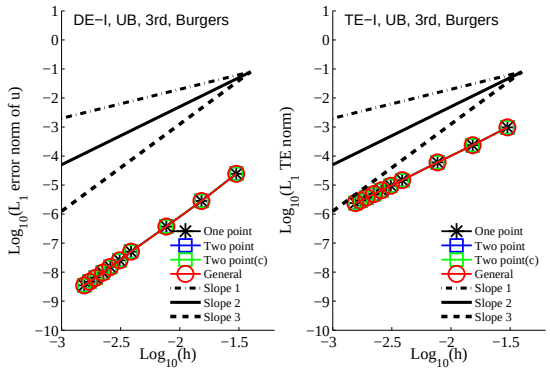

Figure B.31: DE(left) and TE(right) at **interior** nodes for the third-order scheme for Burgers' equation, uniformly spaced boundary grid (UB).

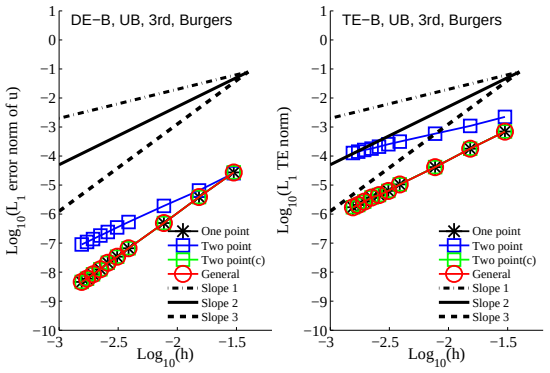

Figure B.32: DE(left) and TE(right) at **boundary** nodes for the third-order scheme for Burgers' equation, uniformly spaced boundary grid (UB).

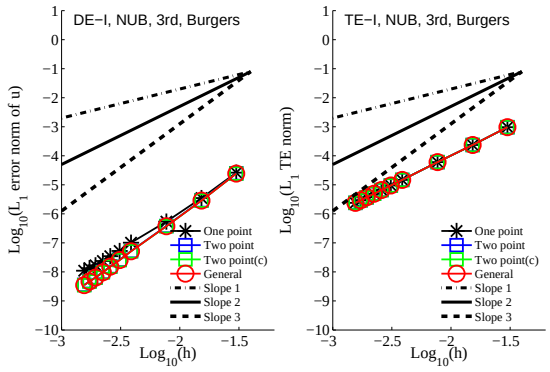

Figure B.33: DE(left) and TE(right) at **interior** nodes for the third-order scheme for Burgers' equation, non-uniformly spaced boundary grid (NUB).

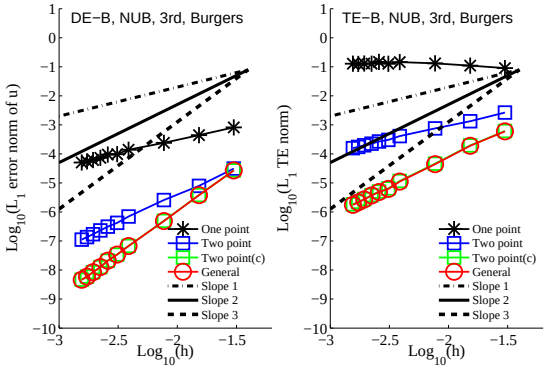

Figure B.34: DE(left) and TE(right) at **boundary** nodes for the third-order scheme for Burgers' equation, non-uniformly spaced boundary grid (NUB).

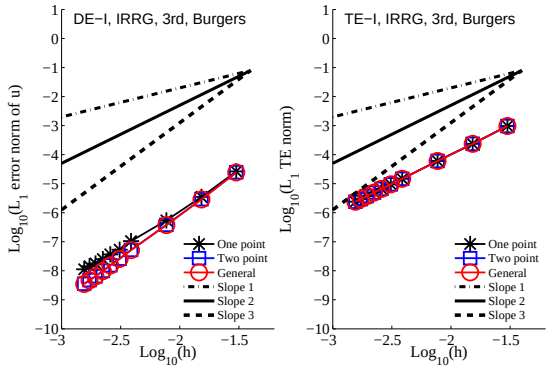

Figure B.35: DE(left) and TE(right) at **interior** nodes for the third-order scheme for Burgers' equation, irregular boundary grid (IRRG).

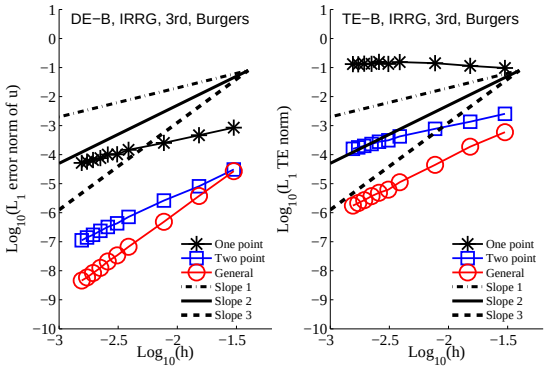

Figure B.36: DE(left) and TE(right) at **boundary** nodes for the third-order scheme for Burgers' equation, irregular boundary grid (IRRG).

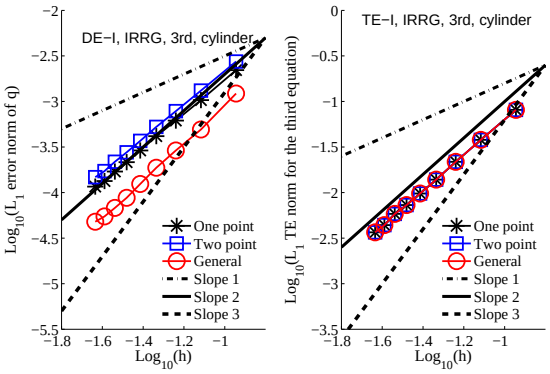

Figure B.37: DE(left) and TE(right) at **interior** nodes for the third-order scheme for the curved boundary problem: Linear approximation to the surface normal vector.

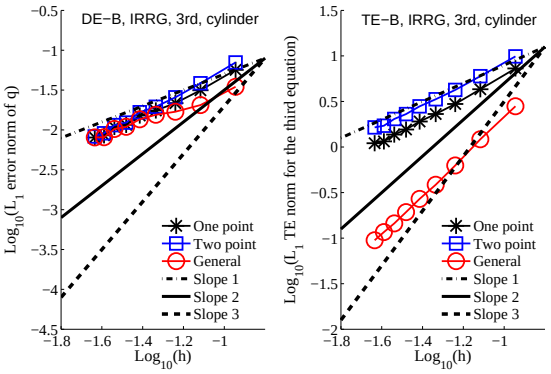

Figure B.38: DE(left) and TE(right) at **boundary** nodes for the third-order scheme for the curved boundary problem: Linear approximation to the surface normal vector.

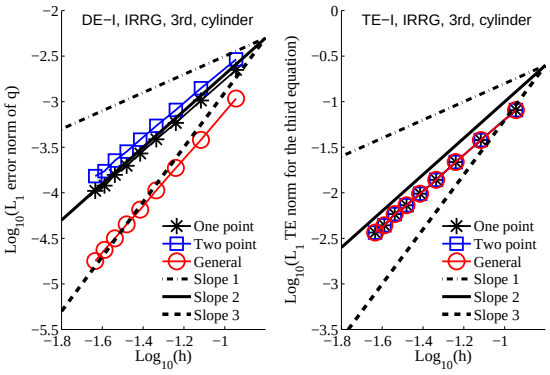

Figure B.39: DE(left) and TE(right) at **interior** nodes for the third-order scheme for the curved boundary problem: Exact surface normal vector.

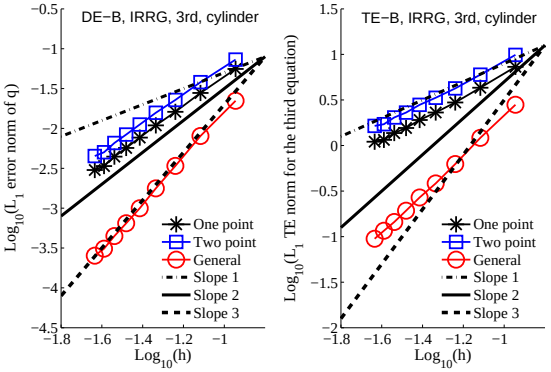

Figure B.40: DE(left) and TE(right) at **boundary** nodes for the third-order scheme for the curved boundary problem: Exact surface normal vector.

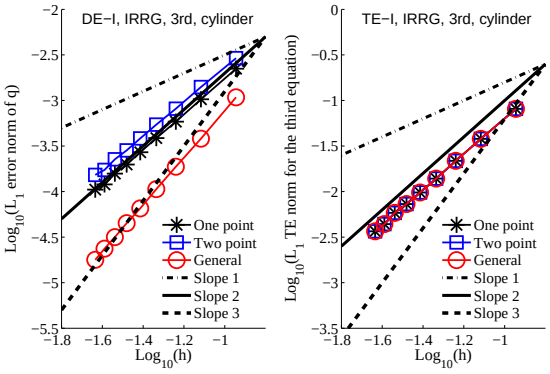

Figure B.41: DE(left) and TE(right) at **interior** nodes for the third-order scheme for the curved boundary problem: Quadratic approximation to the surface normal vector.

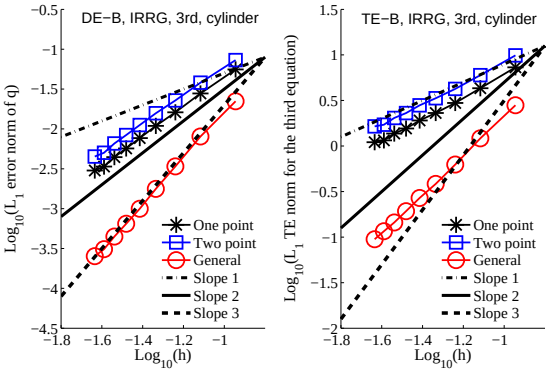

Figure B.42: DE(left) and TE(right) at **boundary** nodes for the third-order scheme for the curved boundary problem: Quadratic approximation to the surface normal vector.

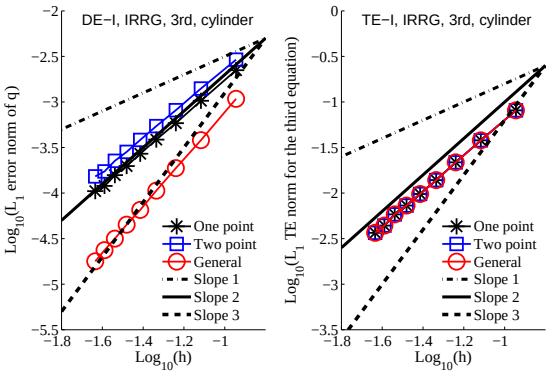

Figure B.43: DE(left) and TE(right) at **interior** nodes for the third-order scheme for the curved boundary problem: Cubic approximation to the surface normal vector.

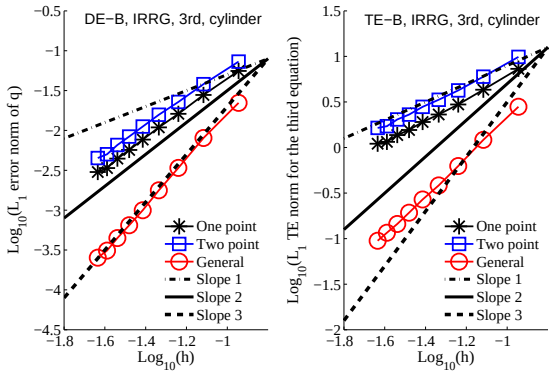

Figure B.44: DE(left) and TE(right) at **boundary** nodes for the third-order scheme for the curved boundary problem: Cubic approximation to the surface normal vector.