A Conservative Mesh-Free Scheme and Generalized Framework for Conservation Laws

SIAM J. SCI. COMPUT.
Vol. 34, No. 6, pp. A2896–A2916

A CONSERVATIVE MESH-FREE SCHEME AND GENERALIZED FRAMEWORK FOR CONSERVATION LAWS*

EDMOND KWAN-YU CHIU†, QIQI WANG‡, RUI HU‡, AND ANTONY JAMESON†

Abstract. We present a novel mesh-free scheme for solving partial differential equations. We first derive a conservative and stable formulation of mesh-free first derivatives. We then show that this formulation is a special case of a general conservative mesh-free framework that allows flexible choices of flux schemes. Necessary conditions and algorithms for calculating the coefficients for our mesh-free schemes that satisfy these conditions are also discussed. We include numerical examples of solving the one- and two-dimensional inviscid advection equations, demonstrating the stability and convergence of our scheme and the potential of using the general mesh-free framework to extend finite volume discretization to a mesh-free context.

Key words. conservation law, advection equation, mesh-free scheme, finite difference, finite volume

AMS subject classifications. 35L65, 65M06, 65M08, 65M12, 65M50

DOI. 10.1137/110842740

1. Introduction.

Despite significant improvement in technology and software tools, efficient generation of high quality meshes has remained the frequent bottleneck in scientific computing, especially when domain boundaries are characterized by nontrivial geometry.

To circumvent mesh generation, many have developed various classes of meshless algorithms. One class of these algorithms originated from the strong form of the governing equations. Monaghan and Gingold [18] developed the smooth particle hydrodynamics method, which uses integral approximations of functions. Oñate et al. [21] proposed the finite point method (FPM), whose derivation actually somewhat parallels those of finite element methods, although FPM uses point collocation in its final discretization to avoid the computation of integrals involving test functions. Löhner et al. [15] and many others have used FPM on fluid and structural mechanics problems. Batina [2] had also previously used local least squares with polynomial basis to obtain a similar formulation. Also employing least squares techniques, Deshpande et al. [6] invented the least squares upwind kinetic method (LSKUM), which Ghosh and Despande [8], Ramesh and Despande [23], and many others later modified or used. Starting from Taylor series, Sridar and Balakrishan [25] and Katz and Jameson [14] also developed meshless methods that resemble traditional finite difference methods. Using radial basis functions, Kansa [12, 13], and later Shu et al. [24] and Tota and Wang [26], also developed various collocation meshless schemes.

Another class of meshless algorithms resulted from the discretization of the weak form of the governing equations. Nayroles, Touzot, and Villom [19] developed the diffuse element (DE) method, which Belytschko, Lu, and Gu [3] extended to obtain the

*Submitted to the journal's Methods and Algorithms for Scientific Computing section July 29, 2011; accepted for publication (in revised form) August 7, 2012; published electronically November 29, 2012.

http://www.siam.org/journals/sisc/34-6/84274.html

†Department of Aeronautics and Astronautics, Stanford University, Stanford, CA 94305 (dnomdec@gmail.com, ajameson@stanford.edu). The first author's research was supported by the PACCAR Inc. Stanford Graduate Fellowship.

‡Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA 02139 (qiqi@mit.edu, hurui@mit.edu).

element-free Galerkin (EFG) method. Because of the underlying weak form in their formulations, DE and EFG both require background grids for computing integrals. Later, Atluri and Zhu [1] introduced the meshless local Petrov–Galerkin (MLPG) method based on a local variational formulation. MLPG reduces the need for the background mesh to a local one, improving on that of DE and EFG. Duarte and Oden [7] and Melenk and Babuska [16] also developed meshless methods in the more general partition-of-unity framework.

Through the excellent work, including and beyond those mentioned above, by many researchers, mesh-free methods have shown great potential and demonstrated ample practical success in scientific computing. However, they still have not become enormously popular among scientists and engineers. Challenges such as point generation and costs of adapting meshless algorithms partly explain the situation. However, one fundamental property of mesh-free methods has led to serious doubts from the scientific community: the lack of formal conservation. To the best of our knowledge, because of their local nature, existing mesh-free schemes do not preserve conservation at the discrete level, except in very limited situations (such as with uniform point distributions, with which one could obtain meshes trivially). Two important disadvantages result. First, compared to some mesh-based approaches, the lack of conservation hinders computational efficiency by precluding the computation of reciprocal fluxes, e.g., as in edge-based approaches. Then, more importantly, nonconservation leads to unpredictable errors when sharp discontinuities exist in the solution. The difficulty in formally quantifying the effects of nonconservation on the accuracy and stability of algorithms deters scientists and engineers from using mesh-free algorithms on a regular basis.

In this paper, we aim at addressing this fundamental issue by presenting a novel mesh-free scheme that possesses various formal conservation and mimetic properties at the discrete level. Designed for numerical solution of conservation laws, the new scheme also allows for a generalization to a framework that accommodates existing schemes for computing numerical fluxes. To present the scheme in detail, we organize the rest of this paper as follows: Section 2 contains the definition of the discrete derivative operator for our meshless scheme, along with the reciprocity and consistency conditions the operator satisfies. Using those conditions, we prove the scheme's global and local conservation properties in sections 3 and 4. These properties lead to the important generalized framework in section 5 that enables one to incorporate many existing flux schemes into meshless discretizations. In section 6, we outline the scheme's extra discrete geometric properties, which drive the design of the procedures in section 7 for generating the necessary meshless coefficients. Section 8 contains the numerical results that demonstrate the success and flexibility of the current meshless framework. There, one can see generated coefficients on sample domains, numerical solutions to the advection equation computed using these coefficients in both the original scheme and the generalized framework with an upwinding flux scheme, and evidence of success of the current framework in handling nonlinear conservation laws. Finally, we briefly conclude our work in section 9.

2. Differentiation operator. We discretize a complex domain Ω, with boundary $\partial\Omega$, using a cloud of N points, each with vector coordinates x_i, $i = 1, \ldots, N$. The point cloud contains points both on the boundary ($s_B = \{i \mid x_i \in \partial\Omega\}$) and in the interior of the domain ($i \notin s_B$). At each boundary point ($i \in s_B$), we define an outward-facing vector normal n_i with magnitude of the portion of area on $\partial\Omega$ associated with the point i. Each point i has a set of neighboring points s_i, which does not include point i itself.

The discrete first derivative is defined by

$$m_i \partial^k \phi_i \approx m_i \delta^k \phi_i = a_{ii}^k \phi_i + \sum_{j \in s_i} a_{ij}^k \phi_j, \tag{2.1}$$

where $k = I, II, III$ is the spatial dimension, and ∂^k and δ^k are the analytic and discrete first derivative operators in the kth spatial coordinate. Here, m_i can represent some volume associated with each point, while the coefficients a_{ij}^k for the point pairs (i, j) then have corresponding dimensions of area (we shall justify this characterization in section 4). To preserve the most generality, we perform our analysis in three dimensions. However, the results identically apply to lower dimensions.

We enforce the following two conditions on a_{ij} and m_i:

C-1. *Reciprocity of coefficients*:

$$\begin{aligned} a_{ij}^k &= -a_{ji}^k, \quad i \neq j \quad (j \in s_i \Leftrightarrow i \in s_j), \\ a_{ii}^k &= 0, \qquad i \notin s_B, \\ a_{ii}^k &= \frac{1}{2} n_i^k, \quad i \in s_B, \end{aligned}$$

where n_i^k is the kth component of the outward-facing, area-weighted boundary normal n_i.

C-2. *Consistency of order L*:

$$a_{ii}^k p(x_i) + \sum_{j \in s_i} a_{ij}^k p(x_j) = m_i \partial^k p(x_i)$$

for all multivariate polynomials p of total order L, where x_i is the coordinate of the ith point.

We shall now investigate the properties of discretizations satisfying conditions C-1 and C-2.

3. Global conservation and mimetic properties. This section proves that if the discrete operator satisfies the conditions in section 2, it has discrete properties corresponding to classical properties of the analytic first derivative operator ∂^k that constantly appear in conservation laws and their manipulations, namely the following:

- Conservation:

$$\int_\Omega \partial^k \phi dx = \int_{\partial\Omega} \phi n^k ds.$$

- Integration by parts:

$$\int_\Omega \psi \partial^k \phi dx = \int_{\partial\Omega} \psi \phi n^k ds - \int_\Omega \phi \partial^k \psi dx.$$

- Enegy conservation:

$$\int_\Omega \phi \partial^k \phi dx = \int_{\partial\Omega} \frac{1}{2} \phi^2 n^k ds.$$

The statements in the following three theorems each contain a discrete representation of the corresponding property above. For example, in Theorem 3.1, the summations on both sides of the equations are discrete approximations of the integral

in the continuous conservation condition. Keeping in mind that the coefficient m_i represents a volume associated with point i, one can view the quantity $m_i\delta^k\phi_i$ as the discrete approximation of the volume integral $\int_{\omega_i}\partial^k\phi dx$.[1] Similarly, $n_i^k\phi_i$ is the counterpart of the kth component of the vector integral $\int n\phi ds$ over $\partial\omega_i$, the area of the boundary with normal n associated with point i. As a result, (3.1) is the discrete version of the mathematical statement for conservation in which the continuous integrals become the corresponding discrete sums of approximate local integrals over relevant parts of the domain. The same analogy applies to the quantities in the following two theorems.

THEOREM 3.1 (discrete conservation). *If m_i and a_{ij}^k satisfy* C-1 *and* C-2*, then*

$$\sum_{i=1}^{N} m_i\delta^k\phi_i = \sum_{i\in s_B}\phi_i n_i^k, \tag{3.1}$$

where s_B is the collection of all boundary points.

Proof. We can write the discrete first derivative as

$$m_i\partial^k\phi_i \approx m_i\delta^k\phi_i = \sum_{j=1}^{N}\tilde{a}_{ij}^k\phi_j, \tag{3.2}$$

where $\tilde{a}_{ii}^k = a_{ii}^k$, $\tilde{a}_{ij}^k = a_{ij}^k$ if $j\in s_i$, and $\tilde{a}_{ij}^k = 0$ otherwise. Let $p\equiv 1$ in C-2. Using $a_{ij}^k + a_{ji}^k = 0$ from C-1, we get

$$\tilde{a}_{ij}^k = -\tilde{a}_{ji}^k \qquad \textit{only if } i\neq j, \tag{3.3}$$

$$\sum_{\substack{j=1\\ j\neq i}}^{N}\tilde{a}_{ji}^k = a_{ii}^k, \tag{3.4}$$

and also

$$\sum_{j=1}^{N}\tilde{a}_{ji}^k = 2a_{ii}^k. \tag{3.5}$$

Incorporating (3.2) into the left-hand side of (3.1) and using (3.3), we have

$$\sum_{i=1}^{N} m_i\delta^k\phi_i = \sum_{i=1}^{N}\sum_{j=1}^{N}\tilde{a}_{ij}^k\phi_j = \sum_{j=1}^{N}\left(\sum_{i=1}^{N}\tilde{a}_{ij}^k\right)\phi_j = \sum_{j=1}^{N}2a_{jj}\phi_j = \sum_{i\in s_B} n_i^k\phi_i, \tag{3.6}$$

where we changed the dummy index from j to i in the last step. ∎

THEOREM 3.2 (summation by parts). *If m_i and a_{ij}^k satisfy* C-1 *and* C-2*, then*

$$\sum_{i=1}^{N} m_i\psi_i\delta^k\phi_i + \sum_{i=1}^{N} m_i\phi_i\delta^k\psi_i = \sum_{i\in s_B}\psi_i\phi_i n_i^k. \tag{3.7}$$

[1] Note that though $\delta^k\phi_i$ approximates $\partial^k\phi$ to Lth order, $m_i\delta^k\phi_i$ may not approximate $\int_{\omega_i}\partial^k\phi dx$ to the same order of accuracy.

Proof. Substituting (3.2) into the left-hand side of (3.7), we have

$$\begin{aligned}
&\sum_{i=1}^{N} m_i\psi_i\delta^k\phi_i + \sum_{i=1}^{N} m_i\phi_i\delta^k\psi_i \\
&\qquad = \sum_{i=1}^{N}\psi_i\sum_{j=1}^{N}\tilde{a}_{ij}^k\phi_j + \sum_{i=1}^{N}\phi_i\sum_{j=1}^{N}\tilde{a}_{ij}^k\psi_j \\
&\qquad = \sum_{i=1}^{N}\psi_i\sum_{j=1}^{N}\tilde{a}_{ij}^k\phi_j + \sum_{i=1}^{N}\phi_i\Big(\sum_{\substack{j=1\\ j\neq i}}^{N} -\tilde{a}_{ji}^k - a_{ii}^k + 2a_{ii}^k\Big)\psi_j \\
&\qquad = \sum_{i=1}^{N}\sum_{j=1}^{N}\tilde{a}_{ij}^k\psi_i\phi_j - \sum_{i=1}^{N}\sum_{j=1}^{N}\tilde{a}_{ij}^k\psi_i\phi_j + 2\sum_{i=1}^{N} a_{ii}\psi_i\phi_i \\
&\qquad = \sum_{i\in s_B}\psi_i\phi_i n_i^k, \qquad (3.8)
\end{aligned}$$

where we exchanged the dummy indices i and j in the second term on the second-to-last line. ∎

COROLLARY 3.3 (discrete energy conservation). *If m_i and a_{ij}^k satisfy* C-1 *and* C-2*, then*

$$\sum_{i=1}^{N} m_i\phi_i\delta^k\phi_i = \sum_{i\in s_B}\frac{\phi_i^2}{2}n_i^k. \qquad (3.9)$$

Proof. Equation (3.9) is obtained by setting $\psi = \phi$ in (3.7). ∎

4. Local conservation. In addition to global conservation, it is also desirable for a scheme to, at the discrete level, preserve local conservation, i.e.,

$$\int_{\omega_i}\partial^k\phi dx = \int_{\partial\omega_i}\phi n^k ds. \qquad (4.1)$$

In traditional meshes, one achieves discrete local conservation (with the use of a conservative numerical scheme) by requiring each cell C_i of volume V_i to be completely enclosed by its faces, which should not overlap. Mathematically, this means that all cells should satisfy

$$\begin{aligned}
\sum_{f\in C_i} n_f^k x_f^k &= V_i, \\
\sum_{f\in C_i} n_f^k &= 0, \qquad (4.2)
\end{aligned}$$

where $k = I, II, III$ again denotes each spatial dimension.

We shall prove that the current scheme satisfies a discrete version of (4.1). As a result, one will see that the meshless coefficients naturally lead to a version of the cell-closure condition in the mesh-free setting.

THEOREM 4.1 (local discrete conservation). *If m_i and a_{ij}^k satisfy* C-1 *and* C-2, then, defining f_{ij} to be a virtual face associated with the point pair ij, with face area $n_{f_{ij}}^k = 2a_{ij}^k$, the following conditions hold:

1. At each point i,

$$m_i\delta^k\phi_i = \sum_{j\in s_i}\phi_{f_{ij}}n^k_{f_{ij}} + \phi_i n^k_i, \qquad i \in s_B,$$
$$m_i\delta^k\phi_i = \sum_{j\in s_i}\phi_{f_{ij}}n^k_{f_{ij}}, \qquad i \notin s_B, \tag{4.3}$$

   where $\phi_{f_{ij}}$ is the function value associated with f_{ij}.
2. In addition, m_i represents a virtual volume at point i that is fully enclosed by the virtual faces f_{ij} between point i and point j (plus boundary faces if $i \in s_B$), which has vector areas $\vec{n}_{f_{ij}}$.

We use the word "virtual" to describe the meshless analogues of faces and volumes to highlight their fundamental differences from their physical counterparts in meshes. Specifically, we can treat any computed coefficients satisfying C-1 and C-2 as mesh-free equivalents of faces and volumes. They neither have nor need physical shapes, as mesh faces and volumes computed from cell vertex locations do.

To facilitate the proof, we introduce the following corollary.

COROLLARY 4.2 (local discrete geometric conservation law). *If m_i and a^k_{ij} satisfy* C-1 *and* C-2*, and the vector valued multivariate function $\vec{p}$ satisfies the divergence-free condition*

$$\nabla\cdot\vec{p} = \sum_{k=1}^{3}\partial^k p^k = 0,$$

then the following condition holds:

$$\vec{p}(x_i)\cdot\vec{n_i} + \sum_{j\in s_i}\vec{p}_{f_{ij}}\cdot\vec{n}_{f_{ij}} = 0, \qquad i \in s_B,$$
$$\sum_{j\in s_i}\vec{p}_{f_{ij}}\cdot\vec{n}_{f_{ij}} = 0, \qquad i \notin s_B, \tag{4.4}$$

where $\vec{p}_{f_{ij}}$ is the value of the polynomial associated with the virtual face f_{ij}.

We shall prove the above theorem and corollary together. As one will see, (4.3) directly leads to (4.4), which in turn leads to condition (4.1) above that is required to complete the proof of Theorem 4.1.

Proof. Applying C-2 to $p \equiv 1$ leads to $a^k_{ii} + \sum_{j\in s_i} a^k_{ij} = 0$. Multiplying this by ϕ_i and adding the result to the definition of the first derivative operator (2.1), we have

$$\begin{aligned} m_i\delta^k\phi_i &= a^k_{ii}\phi_i + \sum_{j\in s_i}a^k_{ij}\phi_j + a^k_{ii}\phi_i + \sum_{j\in s_i}a^k_{ij}\phi_i \\ &= 2a^k_{ii}\phi_i + \sum_{j\in s_i}a^k_{ij}(\phi_i+\phi_j) \\ &= 2a^k_{ii}\phi_i + \sum_{j\in s_i}n^k_{f_{ij}}\frac{(\phi_i+\phi_j)}{2}. \end{aligned} \tag{4.5}$$

For interior points, $a^k_{ii} = 0$. For boundary points, $a^k_{ii} = \frac{n^k_i}{2}$. If we let $\phi_{f_{ij}} = \frac{\phi_i+\phi_j}{2}$, then we obtain (4.3).

To prove Corollary 4.2, let $\phi = p$ be a polynomial of total order L. Consistency of order L gives

$$m_i \partial^k p_i = \sum_{j \in s_i} p_{f_{ij}} n^k_{f_{ij}} + p_i n^k_i, \qquad i \in s_B,$$

$$m_i \partial^k p_i = \sum_{j \in s_i} p_{f_{ij}} n^k_{f_{ij}}, \qquad i \notin s_B. \tag{4.6}$$

If $\vec{p}$ is divergence-free, summing over (4.6) applied to each component of $\vec{p}$ results in (4.4).

To complete the proof of Theorem 4.1, it remains to show that volumes m_i and coefficients $\vec{a}_{ij}$ are consistent and do not lead to numerical sources. This can be seen through the following two properties:

1. Let $\phi = x^k$ (recall k denotes spatial dimension). Equation (4.3) becomes $\sum_{j \in s_i} n_{f_{ij}} \frac{x^k_i + x^k_j}{2} = m_i$ for an interior point i, and $\sum_{j \in s_i} n_{f_{ij}} \frac{x^k_i + x^k_j}{2} + n^k_i x^k_i = m_i$ for a boundary point.
2. Corollary 4.2 applied to $\vec{p} \equiv \vec{e}^k$ ($\vec{e}^k$ is the vector that has 1 as the kth component and all zeros otherwise) yields $\sum_{j \in s_i} n^k_{f_{ij}} = 0$ for an interior point i and $\sum_{j \in s_i} n^k_{f_{ij}} + n^k_i = 0$ for a boundary point.

These conditions exactly resemble (4.2). They guarantee that, around each solution point, a volume of size m_i is fully enclosed by its virtual faces, which include boundary elements if appropriate. Thus, no numerical sources arise from inconsistent definition of virtual normals and volumes. The scheme is locally conservative. ∎

Theorem 4.1 justifies the geometric interpretation of the coefficients a^k_{ij} and m_i as analogues of face areas and volumes. In close proximity to complex geometry, traditional meshes can contain warped cells and faces that sometimes lead to difficulty in satisfying the closure criteria (4.2). In this sense, Theorem 4.1 suggests that the current mesh-free scheme could actually enforce numerical conservation better than meshes. This brings further promise for using the current scheme in practice. In the next section, we further generalize the current scheme by using the above geometric interpretation to construct a generalized mesh-free framework.

5. A generalized framework. The reciprocity condition $a^k_{ij} = -a^k_{ji}$ in C-1 guarantees that the virtual face areas and normals are consistent, as seen by volumes i and j. With this reciprocity, the local conservation property resulting from the geometric interpretation of the coefficients a^k_{ij} and m_i mentioned in section 4 ensures that global conservation (3.1) holds regardless of the choice of interface flux formulation.

Therefore, the current formulation actually represents a more general conservative meshless framework. One can generate a set of m_i, n^k_i, and a^k_{ij} (using the algorithm in section 7, for example) that represent boundary faces, virtual cell volumes, and virtual interface areas. However, instead of the central average flux $\phi_f = \frac{\phi_i + \phi_j}{2}$, one can now apply more sophisticated interface fluxes while preserving numerical conservation.

More precisely, we can generalize (4.5) as

$$m_i \delta^k_f \phi_i = 2 a^k_{ii} \phi_i + \sum_{j \in s_i} 2 a^k_{ij} F_{ij}, \tag{5.1}$$

where the interface flux F_{ij} can be a function of ϕ_i, ϕ_j, the derivative of ϕ at the ith and jth points, and so on.

This freedom plays a vital role in allowing the current framework to handle nonlinearity, where one often needs to introduce extra stability through the choice of an appropriate flux scheme.

An example of the generalized derivative operator is the upwind scheme. In this scheme, the interface flux F_{ij} is defined as

$$F_{ij} = \begin{cases} \phi_i + \dfrac{x_j^k - x_i^k}{2}\delta_1^k\phi_i, & a_{ij}^k u^k > 0, \\ \phi_j + \dfrac{x_i^k - x_j^k}{2}\delta_1^k\phi_j, & a_{ij}^k u^k < 0, \end{cases} \tag{5.2}$$

where δ_1^k is a first order accurate reconstruction of the derivative in the kth spatial dimension, and $\vec{u}$ is the interface convection velocity. In the numerical examples detailed in this paper, δ_1^k is constructed as

$$\delta_1^k\phi_i = \sum_{j\in s_i} a_{1ij}^k(\phi_j - \phi_i),$$

where a_{1ij}^k are chosen by solving

$$\min \sum_{j\in s_i}(a_{1ij}^k)^2 \quad \text{such that} \quad \sum_{j\in s_i} a_{1ij}^k(x_j^{k'} - x_i^{k'}) = d_{kk'}$$

for each i and k, where $k' = I, II, III$, and d represents the Kronecker delta.

The second order accuracy of the upwinding flux F_{ij} leads to a first order accurate approximation of the derivative (5.1). In section 8, we compare this generalized scheme and the original scheme in numerical experiments, showing the high-quality results this generalization produces and the flexibility it allows.

6. Global divergence theorem and geometric conservation law. Before we discuss how to generate the coefficients a_{ij}^k and m_i, we first present some extra global properties of our scheme. Resembling (4.6) and Corollary 4.2, these properties provide important insight into further requirements for obtaining coefficients that satisfy C-1 and C-2.

THEOREM 6.1 (discrete divergence theorem). *If a_{ij}^k and m_i satisfy* C-1 *and* C-2*, then the following condition holds for all multivariate polynomials p of total order $2L$:*

$$\sum_{i\in s_B} p(x_i)n_i^k = \sum_{i=1}^{N} m_i\partial^k p(x_i). \tag{6.1}$$

Proof. It is sufficient to prove (6.1) for all multivariate monomials p of order less than or equal to $2L$. Let $p = p_1p_2$, where both p_1 and p_2 are monomials or order less than or equal to L, and thus satisfy condition C-2:

$$a_{ii}^k p_1(x_i) + \sum_{j\in s_i} a_{ij}^k p_1(x_j) = m_i\partial^k p_1(x_i), \tag{6.2}$$

$$a_{ii}^k p_2(x_i) + \sum_{j\in s_i} a_{ij}^k p_2(x_j) = m_i\partial^k p_2(x_i). \tag{6.3}$$

Multiplying (6.2) by $p_2(x_i)$ and (6.3) by $p_1(x_i)$, we add the results. Summing the added results over $i = 1, \ldots, N$, and using the fact, derived from $a_{ij} + a_{ji} = 0$ in condition C-1, that

$$\sum_{(i,j)\in E} a_{ij}^k \big(p_1(x_i)p_2(x_j) + p_1(x_j)p_2(x_i)\big) = 0,$$

where E is the set of all neighborhood pairs in the domain, we have

$$\sum_{i=1}^{N} 2a_{ii}^k p_1(x_i)p_2(x_i) = \sum_{i=1}^{N} m_i \partial^k \big(p_1(x_i)p_2(x_i)\big).$$

Inserting the definition of a_{ii}^k from condition C-1, we get (6.1) for $p = p_1 p_2$. ∎

In this proof, we took linear combinations of the constraints in C-2 and canceled out all a_{ij}'s to obtain (6.1). Therefore, this theorem shows that C-1 and C-2, as linear constraints for $\vec{a}_{ij}$, are *linearly dependent*. In other words, these conditions cannot be satisfied simultaneously unless m_i satisfies (6.1).

In addition, the following corollary shows that (6.1) as linear constraints for m_i are also dependent.

COROLLARY 6.2 (geometric conservation law). *If a_{ij}^k and m_i satisfy* C-1 *and* C-2*, and the vector valued multivariate polynomials $\vec{p}$ of order $2L$ satisfy the divergence-free condition*

$$\nabla \cdot \vec{p} = \sum_{k=1}^{3} \partial^k p^k = 0,$$

then the following condition holds:

$$\sum_{i\in s_B} \vec{p}(x_i) \cdot \vec{n}_i = 0. \tag{6.4}$$

Proof. Equation (6.4) is obtained by summing (6.1) over $k = I, II, III$ and using the divergence-free condition. ∎

Thus, to obtain a consistent set of a_{ij}^k and m_i, one must at least choose the boundary normals n_i^k appropriately according to (6.4). In section 7, we explore two different ways to construct the meshless coefficients satisfying C-1 and C-2 by enforcing these constraints.

7. Operator construction. In section 6, we listed necessary compatibility conditions resulting from the linear dependence of C-1 and C-2. In this section, we present two algorithms for generating coefficients that satisfy C-1, C-2, and these implied constraints.

While we have not shown the sufficiency of the implied compatibility conditions for the existence of coefficients a_{ij}^k and m_i that satisfy C-1 and C-2, solutions satisfying C-1 and C-2 do exist empirically. Numerical experiments in section 8 revealed that, by enforcing the compatibility conditions, the algorithms presented here produced compatible linear constraints (ones to which infinitely many solutions exist).

Many numerical simulations involve the solution of conservation laws in a domain enclosed by some discrete representation of the boundary geometry. Thus, both algorithms begin with a set of corrected boundary normals n_i^k that satisfy Corollary 6.2 (or (6.4)), which is one of the necessary conditions for the existence of compatible coefficients.

7.1. Segregated approach. In this approach, we first generate m_i that satisfy (6.1) and then solve for a_{ij}^k that satisfy C-1 and C-2. The algorithm is as follows:

1. Calculate estimates of $\vec{n}_i$ for all boundary points based on the geometry of the domain boundary. One can use various geometry-processing algorithms to obtain initial estimates of the boundary faces, (e.g., see Wang [27]).
2. Project the estimates of $\vec{n}_i$ into the linear subspace that satisfies (6.4).

   Specifically, letting $\mathbf{n}$ be a column vector that contains $\vec{n}_i$ for all boundary points i, the geometric conservation law (6.4) can be written as

   $$G^{\mathrm{T}}\mathbf{n} = 0. \tag{7.1}$$

   The number of columns of matrix G is equal to the number of linearly independent vector valued, *divergence-free* multivariate polynomials of maximum order $2L$. Each column of G contains the values of one of these polynomials at all boundary points. When L is small, G is a thin matrix.
   To ensure that the total volume enclosed by the boundary normals does not change during the projection process, we also enforce the constraint

   $$\frac{1}{n_d}\sum_{i\in s_B} x_i \cdot \vec{n}_i = m_0 \tag{7.2}$$

   for each closed boundary of the domain, where n_d is the number of spatial dimensions and $m_0 = \frac{1}{n_d}\sum_{i\in s_B} x_i \cdot \vec{n_0}_i$.
   Letting $\mathbf{n}_0$ be the initial estimate of $\mathbf{n}$ based on the geometry of the domain boundary, we calculate the change in $\mathbf{n}$ by solving

   $$\begin{aligned} R^T\mathbf{y} &= \mathbf{g} - G^{\mathrm{T}}\mathbf{n}_0, \\ \Delta\mathbf{n} &= Q\mathbf{y}, \\ \mathbf{n} &= \mathbf{n}_0 + \Delta\mathbf{n}, \end{aligned} \tag{7.3}$$

   where $\mathbf{g} = (0,\ldots,0,m_0)^T$ and $QR = G$ is the (thin) QR decomposition of G. The projected $\mathbf{n}$ satisfies the linear equation (7.1), which is equivalent to the geometric conservation law (6.4).
3. With initial estimates of $m_i = 0$, project m_i onto the linear manifold that satisfies (6.1).

   We denote $\mathbf{m}$ as a column vector that contains m_i for all points, and we write (6.1) as

   $$D^{\mathrm{T}}\mathbf{m} = E^{\mathrm{T}}\mathbf{n}. \tag{7.4}$$

   The matrices D and E have the same number of columns, which is equal to the number of linearly independent multivariate polynomials of maximum order $2L$. Each column of D contains the divergence of one of these polynomials at all points; the corresponding column of E contains the value of the polynomial at all boundary points. When L is small, both D and E are thin matrices.
   In order to compute $\mathbf{m}$ that satisfies (7.4), we perform the thin SVD $D = USV^{\mathbf{T}}$. Equation (7.4) becomes

   $$SU^{\mathrm{T}}\mathbf{m} = V^{\mathrm{T}}E^{\mathrm{T}}\mathbf{n}.$$

Due to the geometric conservation law (6.4), the matrix D is singular. The number of zero singular values of D is equal to the number of columns of matrix G. If $\mathbf{n}$ satisfies (7.1), the rows of $V^{\mathrm{T}}E^{\mathrm{T}}\mathbf{n}$ corresponding to the zero singular values are 0. Letting b_1 be the rows of $V^{\mathrm{T}}E^{\mathrm{T}}\mathbf{n}$ corresponding to the nonzero singular values, U_1 be the columns of U corresponding to the nonzero singular values, and S_1 be the square submatrix of S corresponding to the nonzero singular values, (7.4) becomes

$$S_1 U_1^{\mathbf{T}}\mathbf{m} = b_1,$$

which can be satisfied by

$$\mathbf{m} = U_1 S_1^{-1} b_1.$$

In addition, just like in finite volume schemes, we require $m_i > 0$, which is not guaranteed by SVD. In the event that some of the m_is are nonpositive, we invoke an optimization procedure that minimizes $\|\mathbf{m}\|_2$ subject to (7.4) and the positivity of m_i. To enforce the positivity constraint, we set $m_i > m_{min}$, where m_{min} is a user-selected parameter, typically on the order of $\sqrt{\epsilon_{mach}}$ (ϵ_{mach} is the machine zero) to avoid the virtual volume at any location to be arbitrarily close to zero. The resulting system is a quadratic program, so this part of the algorithm can be carried out using any solvers capable of handling quadratic programming problems or general convex optimization, such as `CVX` [11, 10] and CVXOPT [5].

4. Solve a constrained least squares problem for $\vec{a}_{ij}$ to enforce C-1 and C-2, while minimizing $\sum_{(i,j)\in E} \|\vec{a}_{ij}\|_2^2$, where E is the set of all neighborhood pairs $\{(i,j) \mid j \in s_i\}$.

We denote $\mathbf{a}$ as a column vector that contains a_{ij}^k for all neighborhood pairs. For each neighborhood pair (i,j), either a_{ij}^k or a_{ji}^k are stored, such that C-1 ($a_{ij}^k = -a_{ji}^k$) is automatically satisfied. We write C-2 in the linear form

$$C^{\mathrm{T}}\mathbf{a}^k = \mathbf{d}^k,$$

where $C^{\mathrm{T}}\mathbf{a}^k$ contains the terms $\sum_{j\in s_i} a_{ij}^k p(x_j)$, and $\mathbf{d}^k$ contains both the $a_{ii}^k p(x_i)$ and the $m_i \partial^k p(x_i)$ terms.

This system of constraints for the least squares problem can be quite large, especially if $L > 2$. In addition, the constraints must be satisfied to high accuracy to ensure numerical stability of the scheme. A number of tools are available for solving this problem for $L \leq 2$ (solving the system for large meshes and $L > 2$ is a challenging problem). In our case, we solve this system using the Krylov iterative method LSQR [22], which handles matrices of arbitrary ranks and dimensions. Note that although the system is singular due to the discrete divergence theorem (6.1), it has a compatible right-hand side constructed by choosing m_i and $\vec{n}_i$ that satisfy (7.4).

7.2. Coupled approach. In this approach, we simultaneously compute m_i and a_{ij}^k that satisfy C-1 and C-2 (and hence (6.1)). The coupled algorithm is as follows:

1. Calculate estimates of $\vec{n}_i$ for all boundary points based on the geometry of the domain boundary (same as in the segregated approach).
2. Project the estimates of $\vec{n}_i$ onto the linear subspace that satisfies (6.4) (also same as in the segregated approach).

3. Solve a quadratic program[2] for $\vec{a}_{ij}$ and m_i to enforce C-1 and C-2, while minimizing $\sum_{(i,j)\in E} \|\vec{a}_{ij}\|_2^2$, where E is the set of all neighborhood pairs $\{(i,j) \mid j \in s_i\}$.

   Again we denote $\mathbf{a}$ as a column vector that contains $\vec{a}_{ij}$ for all neighborhood pairs, and we write $\mathbf{m}$ as the vector containing m_i for all points. We write condition C-2 in the linear form

$$C^{\mathrm{T}}\mathbf{a} + P^k\mathbf{m} = \tilde{\mathbf{d}}^k,$$

   where, as before, $C^{\mathrm{T}}\mathbf{a}^k$ contains the term $\sum_{j\in s_i} a_{ij}^k p(x_j)$, but now $P^k\mathbf{m}$ contains the terms $m_i\partial^k p(x_i)$, and $\tilde{\mathbf{d}}^k$ contains boundary terms of the type $a_{ii}^k p(x_i)$.
   After introducing of $\mathbf{m}$ as unknowns, we obtain the system of constraints

$$\begin{bmatrix}\mathbf{C}^{\mathrm{T}} & \mathbf{P},\end{bmatrix}\mathbf{u} = \mathbf{d}, \tag{7.5}$$

$$\mathbf{C}^{\mathrm{T}} = \begin{bmatrix} C^{\mathrm{T}} & 0 & 0 \\ 0 & C^{\mathrm{T}} & 0 \\ 0 & 0 & C^{\mathrm{T}} \end{bmatrix}, \quad \mathbf{P} = \begin{bmatrix} P^I \\ P^{II} \\ P^{III} \end{bmatrix}, \quad \mathbf{u} = \begin{bmatrix} \mathbf{a} \\ \mathbf{m} \end{bmatrix} = \begin{bmatrix} \mathbf{a}^I \\ \mathbf{a}^{II} \\ \mathbf{a}^{III} \\ \mathbf{m} \end{bmatrix}, \quad \tilde{\mathbf{d}} = \begin{bmatrix} \tilde{\mathbf{d}}^I \\ \tilde{\mathbf{d}}^{II} \\ \tilde{\mathbf{d}}^{III} \end{bmatrix}.$$

   This system, along with the constraint that $m_i > m_{min}$, again can be solved using QP or convex optimization tools. Right preconditioning was applied by scaling the columns of C^{T} by $\|\vec{\Delta x}_{ij}\|_2$ when enforcing the constraints and scaling the objective function accordingly.
   Note that although the constraint is singular due to the discrete divergence theorem (6.1), we experienced no problem of infeasibility during the solution procedure after we had constructed the right-hand side by choosing $\vec{n}_i$ that satisfies (7.1).

Current experiments suggest the coupled algorithm gives better results, i.e., produces coefficients that lead to more accurate solutions. One can refer to section 8 for further details. There, one will also see results and analysis that justify the use of minimum norms as criteria for desirable solutions.

8. Numerical results. In this section, we present numerical results from applying the current framework to solving the advection equation in one and two dimensions.

8.1. Model problem—the advection equation. Consider the equation

$$\frac{\partial\phi}{\partial t} + \vec{u}\cdot\nabla\phi = 0 \tag{8.1}$$

with boundary condition $\phi(x,t) = \phi_B(x,t)$ at the inlet part of the boundary $\{x \in \partial\Omega \mid \vec{n}\cdot\vec{u} < 0\}$. The advection velocity $\vec{u} = (u^1, u^2, u^3)$ is constant. This equation is discretized as[3]

$$\frac{d\phi_i}{dt} + u^k\delta^k\phi_i = \begin{cases} 0, & i \notin s_B, \\ \dfrac{\phi_B - \phi_i}{m_i}[u_i^n]_-, & i \in s_B, \end{cases} \tag{8.2}$$

[2]For simple small problems, alternative algorithms such as SVD can be used. For instance, see section 8.2.

[3]Superscript k follows Einstein notation.

where $u_i^n = n_i^k u^k$ and $[u_i^n]_- = -\min(0, u_i^n)$ denotes the negative part. The penalty term in the discretization (8.2) is consistent with the continuous boundary condition.

The stability of this discretization can be proven using Corollary 3.3. We multiply (8.2) by $m_i\phi_i$ and sum over all i. Using (3.9), we have

$$\frac{de}{dt} = -\sum_{i\in s_B}\left(-\frac{\phi_i^2}{2}u_i^n - (\phi_i^2 - \phi_i\phi_B)[u_i^n]_-\right),$$

where the energy is defined by

$$e = \sum_{i=1}^{N}\frac{1}{2}m_i\phi_i^2.$$

By splitting u_i^n into positive and negative parts, $u_i^n = [u_i^n]_+ - [u_i^n]_-$, we then obtain

$$\frac{de}{dt} = \frac{1}{2}\sum_{i\in s_B}\left(-\phi_i^2[u_i^n]_+ - (\phi_i - \phi_B)^2[u_i^n]_- + \phi_B^2[u_i^n]_-\right).$$

The first two terms, which represent the energy convected out of the domain and the penalty term on the boundary, respectively, cannot increase the total energy. The third term corresponds to the energy convected into the domain and depends only on the boundary condition. Therefore, the energy cannot increase exponentially, and the semidiscrete scheme (8.2) is stable.

It is well known that the forward Euler scheme in time is unstable with a central-type scheme in space. To maintain stability of the discretization, we use the Crank–Nicolson scheme in time to obtain our results. We shall briefly show that Crank–Nicolson is unconditionally stable when applied to the linear advection equation with the current spatial discretization. The fully discrete scheme can be written as

$$\frac{\phi_i^{(t+1)} - \phi_i^{(t)}}{\Delta t} + u^k\delta^k\bar{\phi}_i^{(t)} = \begin{cases} 0, & i \notin s_B, \\ \dfrac{\bar{\phi}_B^{(t)} - \bar{\phi}_i^{(t)}}{m_i}[u_i^n]_-, & i \in s_B, \end{cases} \tag{8.3}$$

where

$$\bar{\phi}^{(t)} = \frac{1}{2}\left(\phi_i^{(t)} + \phi_i^{(t+1)}\right).$$

In a way similar to that in the semidiscrete case, we multiply (8.3) by $m_i\bar{\phi}_i^{(t)}$ and sum over i to obtain

$$\frac{e^{(t+1)} - e^{(t)}}{\Delta t} = -\sum_{i\in s_B}\left(-\frac{(\bar{\phi}_i^{(t)})^2}{2}u_i^n - (\bar{\phi}_i^{(t)})^2 - \bar{\phi}_i^{(t)}\bar{\phi}_B^{(t)})[u_i^n]_-\right)$$

and

$$\frac{e^{(t+1)} - e^{(t)}}{\Delta t} = \frac{1}{2}\sum_{i\in s_B}\left(-(\bar{\phi}_i^{(t)})^2[u_i^n]_+ - (\bar{\phi}_i^{(t)} - \bar{\phi}_B^{(t)})^2[u_i^n]_- + (\bar{\phi}_i^{(t)})^2[u_i^n]_-\right),$$

where the energy is now defined at each time step as

$$e^{(t)} = \sum_{i=1}^{N}\frac{1}{2}m_i\left(\phi_i^{(t)}\right)^2.$$

The arguments carry over from the semidiscrete case, showing that the fully discrete scheme is stable.

To show the ability of the generalized framework to accommodate more general numerical flux functions, we also discretize (8.1) with the upwinding derivative δ_f^k discussed in section 5:

$$\frac{d\phi_i}{dt} + u^k \delta_f^k \phi_i = \begin{cases} 0, & i \notin s_B, \\ \dfrac{\phi_B - \phi_i}{m_i} [u_i^n]_-, & i \in s_B. \end{cases} \tag{8.4}$$

Although this scheme does not conserve kinetic energy as the central flux scheme (8.2), it does maintain global and local conservation properties In section 8.3, one can also see the stability and convergence of this scheme.

8.2. one-dimensional results. In the one-dimensional (1D) examples, we apply our scheme to solve conservation laws in the domain $[0, 1]$ with uniformly distributed points indexed from left to right. We highlight the connection between the algorithms used to generate connecitivity and the quality of the numerical results.

8.2.1. Connectivity and coefficient generation. Each point i is connected to its four nearest neighbors plus point $i - 1$ or $i + 1$ if either of these is not already present in the set of nearest neighbors.

In this 1D example, we use both the segregated algorithm and the coupled algorithm to generate the meshless coefficients m_i and a_{ij}^k that satisfy linear consistency ($L = 1$), with the boundary normals set to ∓ 1 at the respective ends of the domain. For the segregated algorithm, the volumes were assigned to be uniform ($\frac{1}{N}$) across all points, satisfying the discrete divergence theorem. Because of the small sizes of the constraint matrices, we used SVD to solve the systems in both approaches, obtaining the minimum norm solution of the respective unknowns.

8.2.2. 1D advection equation. We take the advection velocity to be unity. Initially, $\phi = 0$ in the entire domain. The solution changes through the boundary condition $\phi(0, t) = \sin 2\pi t$ enforced using the penalty term in (8.2). Figure 8.1 shows the solutions at $t = 2$, obtained using Crank–Nicolson time stepping, which preserves the unconditional stability of the spatial scheme, with 400 points in the domain.

Both sets of coefficients satisfy the requirements of conservation and linear consistency. Both solutions also converge to the exact solution as the point density increases.

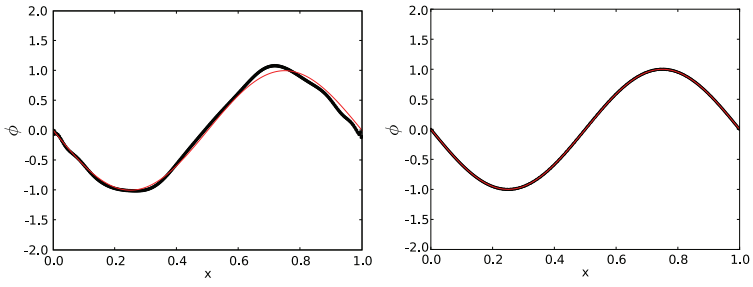

FIG. 8.1. *Solution to the 1D advection equation at $t = 2$. Left: using coefficients with uniform volumes. Right: using coefficients from the coupled approach. Red (thin): exact solution. Black (thick): numerical solution.*

However, one can see that the quality of the solution, even in this simple 1D case, depends on the meshless coefficients (and hence the method chosen to generate them).

To explain the reason behind such a phenomenon, we consider the Taylor expansion of the meshless operator (without loss of generality, we ignore the boundary terms, which will be cancelled in (8.6)). For $L = 1$,

$$\begin{aligned} m_i \delta^x \phi_i &= \sum_{j \in s_i} a_{ij} \phi_j \\ &= \sum_{j \in s_i} a_{ij} \left(\phi_i + \Delta x_{ij} \partial^x \phi_i + \frac{1}{2} \Delta x_{ij}^2 \partial^{xx} \phi_i + \frac{1}{6} \Delta x_{ij}^3 \partial^{xxx} \phi_i + \cdots \right) \\ &= m_i \partial^x \phi_i + \sum_{j \in s_i} a_{ij} \left(\frac{1}{2} \Delta x_{ij}^2 \partial^{xx} \phi_i + \frac{1}{6} \Delta x_{ij}^3 \partial^{xxx} \phi_i + \cdots \right). \end{aligned} \tag{8.5}$$

The global L_2 error in the derivative approximation can then be expressed by

$$\begin{aligned} \int_\Omega (\partial^x \phi - \delta^x \phi)^2 d\Omega &\approx \sum_i m_i (\partial^x \phi - \delta^x \phi)^2 \\ &= \sum_i \frac{1}{m_i} \left[\sum_j a_{ij} \left(\frac{1}{2} \Delta x_{ij}^2 \partial^{xx} \phi_i + \frac{1}{6} \Delta x_{ij}^3 \partial^{xxx} \phi_i + \cdots \right) \right]^2 \\ &= \mathbf{a}^{\mathrm{T}} \mathbf{R}_L^{\mathrm{T}} \mathbf{M}^{-1} \mathbf{R}_L \mathbf{a} \\ &= \| \mathbf{M}^{-\frac{1}{2}} \mathbf{R}_L \mathbf{a} \|_2^2, \end{aligned} \tag{8.6}$$

where

$$R_{L_{i,\hat{ij}}} = \sum_{r=L+1}^{\infty} \frac{\Delta x_{\hat{ij}}^r}{r!} \phi_{(r)i}$$

is the $N \times N_e$ matrix containing the Taylor remainder terms.

Since $\mathbf{R}_L$ depends on the solution and is not known a priori, the simplest way to reduce the size of the error expression is to reduce the norm of $\mathbf{a}$. When using SVD instead of convex optimization in the coupled algorithm, we actually minimize $\|\mathbf{a}\|_2^2 + \|\mathbf{m}\|_2^2$ (as opposed to $\|\mathbf{a}\|_2^2$). Recall that $\mathbf{m}$ can be related to $\mathbf{a}$ through the constraints for linear consistency such that $\mathbf{m} = C_1 \mathbf{a}$. Therefore, the SVD step actually replaces $\mathbf{R}_L^{\mathrm{T}} \mathbf{R}_L$ by $(I + C_1^{\mathrm{T}} C_1)$ and minimizes $\mathbf{a}^{\mathrm{T}} (I + C_1^{\mathrm{T}} C_1) \mathbf{a}$, which can be viewed as an alternative estimate of the global error.

To verify the above statements, we plot the leading error terms resulting from both sets of coefficients, normalized by the respective local volumes m_i, in Figure 8.2. One can see that the coupled algorithm produced a set of coefficients that led to much smaller leading error terms, consistent with the difference in accuracy of the numerical solutions in Figure 8.1.

The above results suggest that fixing m_i in the segregated algorithm essentially limits the attainable minimum norm of the vector $\mathbf{a}$ that satisfies the consistency constraints. In other words, when using the coupled algorithm instead of the segregated one, the introduction of m_i as unknowns reduces the norm of the coefficient vector.

As another numerical example, we solve the advection equation (8.1) with the same set of parameters, but now with the initial solution $\phi = \sin(2\pi x)$ and periodic boundary conditions. Figure 8.3 plots the numerical solutions at $t = 2$. Once again,

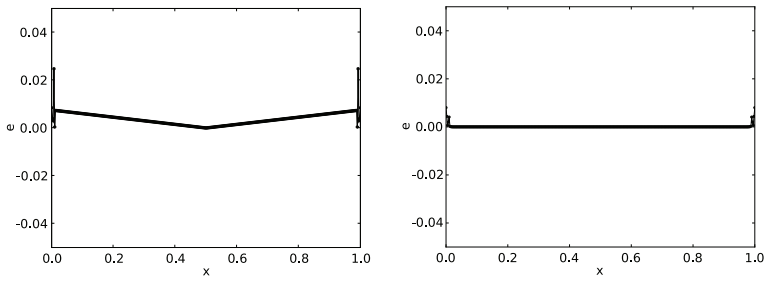

FIG. 8.2. *Representative error term* $|\frac{1}{m_i}\sum_j a_{ij}\Delta x_{ij}^2|$ *for each set of coefficients. Left: from coefficients with uniform volumes. Right: from coefficients from the coupled approach.*

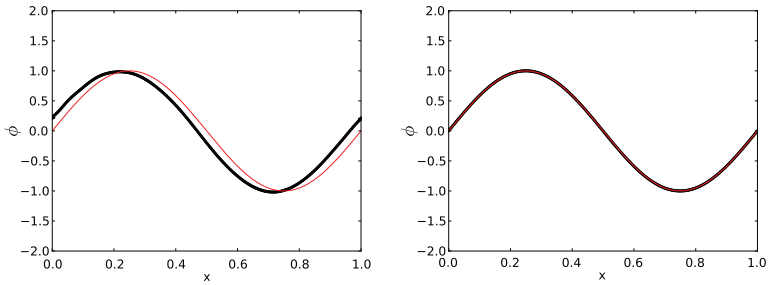

FIG. 8.3. *Solution to the* 1*D periodic advection equation at* $t = 2$*. Left: using coefficients with uniform volumes. Right: using coefficients from the coupled approach. Red (thin): exact solution. Black (thick): numerical solution.*

using uniform weights m_i and the segregated approach resulted in larger final numerical errors (dispersive in nature, as expected) than generating coefficients using the coupled approach, even though both solutions converged to the exact solution with point refinement.

8.3. Two-dimensional results. Here, we solve (8.1) in the two-dimensional (2D) domain $\Omega = \{x \mid 1 < \|x\|_2 < 5\}$. To demonstrate the flexibility of our formulation, we shall use both the original scheme (8.2) and the generalized framework with the upwind scheme (8.4).

8.3.1. Point cloud and neighborhood sets. We formed four sets of point clouds with 165, 563, 2060, and 7896 points, respectively. Each point cloud is the union of a Cartesian grid in the interior and evenly spaced points on the boundary. To generate the neighborhood set s_i for point i, we first set s_i to be the eight nearest points to i and enforce the reciprocal condition by setting $s_i = s_i \cup \{j \mid i \in s_j\}$. We perform a Cartesian subdivision of the domain and search the neighbors of each point only in its neighboring subdivisions. This results in a very efficient (with computational complexity of $O(N)$) algorithm for generating point clouds and neighborhood sets. Although the point clouds look relatively simple at first glance, we must point out that it is actually difficult to generate a traditional finite volume mesh using the current connectivity involving at least eight neighbors per point. Figure 8.4 shows the point cloud and neighborhood sets for $N = 165$.

8.3.2. Construction of n_i^k, m_i, and a_{ij}^k. In this example, we use the coupled algorithm to generate coefficients for polynomial order $L = 1$. At each boundary point

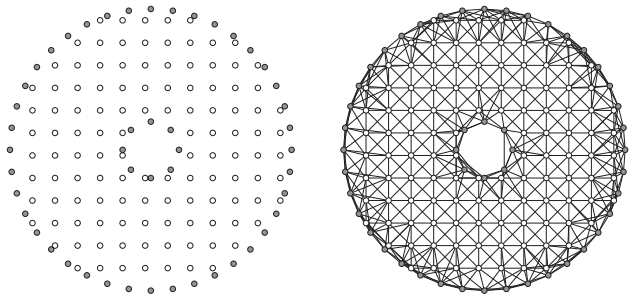

FIG. 8.4. *The point cloud and the neighborhood pairs with* 165 *points (the coarsest point cloud). Open circles: interior points. Filled circles: boundary points. A line connecting two points indicates that they are mutual neighbors.*

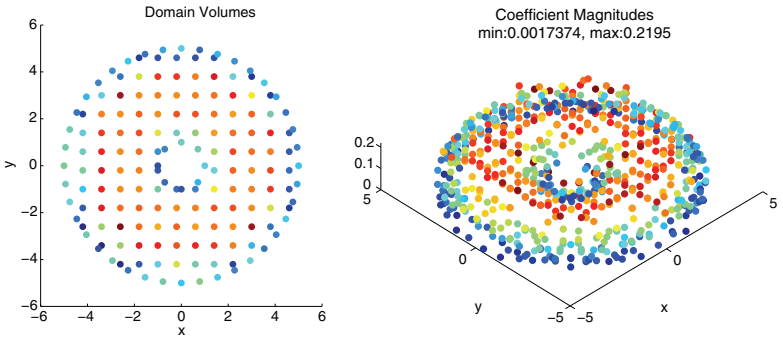

FIG. 8.5. *Virtual volumes and virtual face area magnitudes computed using the coupled algorithm. Left: virtual volumes. Right: virtual face areas.* $N = 165$.

i, we computed an initial estimate of n_i^k using the average of the normals of the two adjacent boundary faces. We corrected this estimate and used convex optimization to generate the remaining coefficients according to the algorithm in section 7. The objective function is $\|\mathbf{a}\|_2$, which is equivalent to replacing the 2D version of the error matrix $\mathbf{R}_L^{\mathrm{T}}\mathbf{M}^{-1}\mathbf{R}_L$ in (8.6) by the identity matrix.

In the present study, the optimization steps were carried out in MATLAB using CVX [11, 10]. For the largest point cloud, the coefficients were obtained in about 1 minute on a workstation with Intel Xeon 5160 processors in a quad-core configuration. All the pointwise consistency constraints were also satisfied to order 1×10^{-6} or better. For larger-scale problems, one can utilize other optimization software libraries [20, 9, 17]. Since the optimization problem is a quadratic programming problem, a QP-specific solver for better efficiency is currently being investigated. The employment of domain decomposition strategies to improve the efficiency of this optimization problem is also a topic of ongoing research.

Figure 8.5 shows scatter plots of the virtual volumes m_i and virtual face area magnitudes $\|\vec{a}_{ij}\|_2$ for $N = 165$. The semitoroidal shape formed by the coefficient magnitudes in Figure 8.5 shows that the constraints we enforce were enough to make the magnitudes of $\vec{a}_{ij}$ vary with local point spacing and connectivity. The virtual volumes m_i simply adjust accordingly to satisfy the consistency constraints. This was true for all current test cases and also in other unreported test problems. It is very encouraging that the unique solution generated by our algorithm appears to be very physically sensible.

8.3.3. Solution of the 2D advection equation. The advection equation (8.1) is solved with advection velocity $(u^1, u^2) = (-\sqrt{2}/2, \sqrt{2}/2)$. The initial and boundary conditions are

$$\phi(x,t) = \frac{\sqrt{2}}{20}(x+y) + 1, \qquad t = 0,$$
$$\phi(x,t) = \frac{\sqrt{2}}{20}(x+y) + \cos[t - 5 - u^k x^k]_+, \quad x \in \partial\Omega, \quad n^k u^k < 0.$$

The exact solution for this initial-boundary-value problem is

$$\phi(x,t) = \frac{\sqrt{2}}{20}(x+y) + \cos[t - 5 - u^k x^k]_+.$$

As mentioned, we computed the numerical solution using schemes (8.2) and (8.4). To assess the convergence of the scheme, we also computed the L_∞ and L_2 norms of the numerical error in the numerical solution against the analytic solution.

Figure 8.6 shows numerical solutions computed by the central flux scheme (8.2) at $t = 100$, again obtained using Crank–Nicolson time stepping. One can see that the resolution of the solution improves with increasing point cloud density, as expected. From Figure 8.7, one can also see that the scheme is roughly second order accurate, as one would expect from similar finite volume schemes with penalty boundary conditions.

Figures 8.8 and 8.9 show numerical solutions at $t = 100$, computed using the upwinding flux scheme (8.4), and the corresponding numerical errors. Aside from the

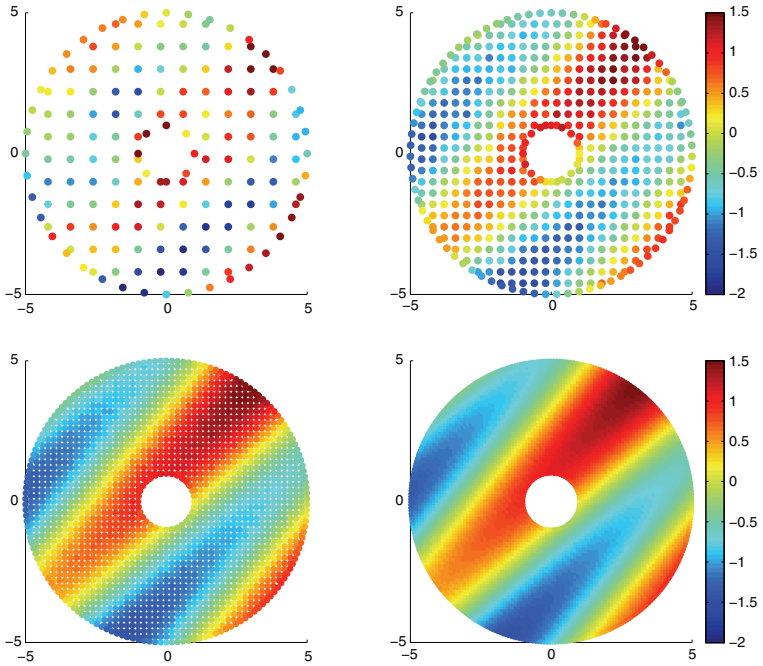

FIG. 8.6. *Solution of the advection equation at* $t = 100$ *using the central flux scheme* (8.2). *The upper left, upper right, lower left, and lower right plots correspond to point clouds of sizes* $N = 165$, 563, 2060, *and* 7896, *respectively.*

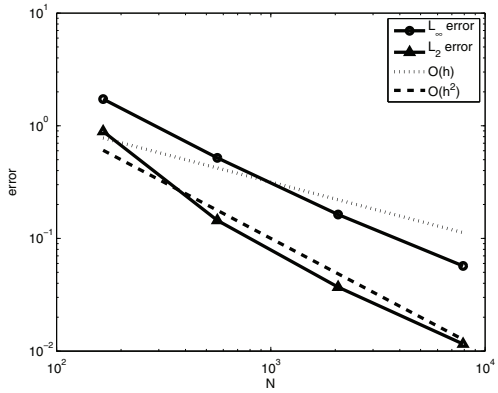

FIG. 8.7. *The numerical error of the central scheme* (8.2) *as a function of the point cloud size* N.

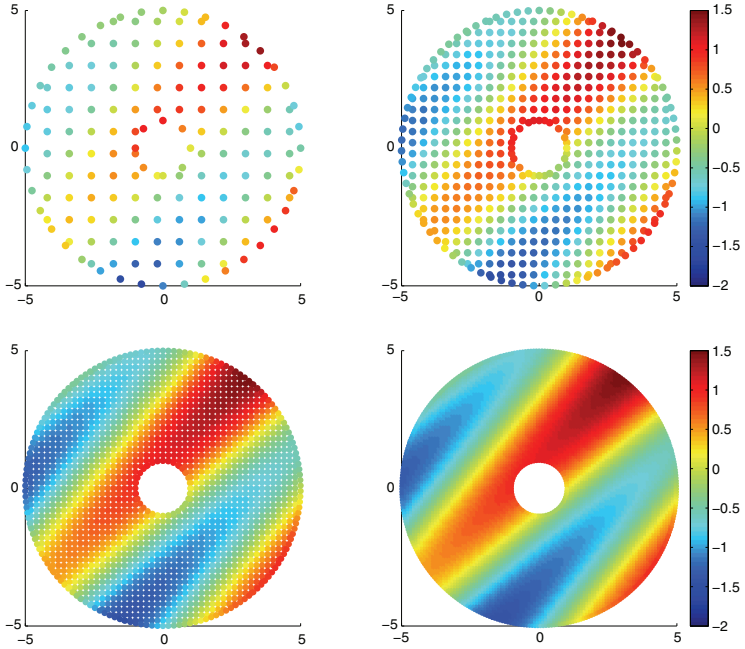

FIG. 8.8. *Solution of the advection equation at* $t = 100$ *using the upwinding scheme* (8.4). *The upper left, upper right, lower left, and lower right plots correspond to point clouds of sizes* $N = 165$, 563, 2060, *and* 7896, *respectively.*

additional smoothness in the solution profile due to upwinding, the solutions computed using both schemes are very similar, especially as the point density increases. The upwind scheme was also roughly second order accurate.

These results confirm the potential and flexibility of the current framework to harness well-proven schemes developed for conservation laws, greatly reducing the overhead for integrating it into existing solution procedures and hence making it a very attractive option. More importantly, other results show that the current framework can indeed handle nonlinear conservation laws trouble-free. Chiu, Wang, and Jameson [4] have applied it to numerically capture shockwaves in transonic flows by solving the Euler equations of gas dynamics.

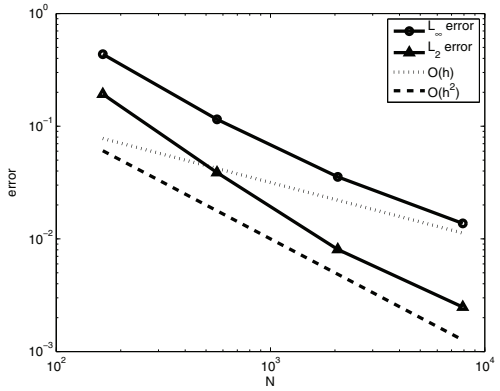

FIG. 8.9. *The numerical error of the upwinding scheme* (8.4) *as a function of the point cloud size* N.

9. Conclusion. We formulated a mesh-free derivative operator by enforcing reciprocity and polynomial consistency conditions C-1 and C-2, which guarantee that our operator is discretely conservative and has certain mimetic properties, such as summation by parts and kinetic energy conservation. Based on the local conservation properties of our meshless scheme, we showed that our scheme is a special case in a more general mesh-free framework that allows the use of existing numerical flux formulations. We presented a segregated algorithm and a coupled algorithm for constructing the mesh-free operator satisfying C-1, C-2, and necessary conditions derived therefrom. 1D results on the advection equation suggested that the coupled approach led to mesh-free operators that produce more accurate numerical solutions. 2D numerical results demonstrated numerical stability and convergence for both the original mesh-free scheme and the generalized framework with an upwind flux scheme, showing the potential of our current framework to serve as a natural extension of finite volume in the mesh-free context.

REFERENCES

[1] S. N. ATLURI AND T. ZHU, *A new meshless local Petrov-Galerkin (MLPG) approach in computational mechanics*, Comput. Mech., 22 (1998), pp. 117–127.

[2] J. T. BATINA, *A gridless Euler/Navier-Stokes solution algorithm for complex aircraft applications*, in proceedings of the 31st AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, 1993.

[3] T. BELYTSCHKO, Y. Y. LU, AND L. GU, *Element-free Galerkin methods*, Internat. J. Numer. Methods Engrg., 37 (1994), pp. 229–256.

[4] E. CHIU, Q. WANQ, AND A. JAMESON, *A conservative meshless scheme: General order formulation and application to Euler equations*, in proceedings of the 49th AIAA Aerospace Sciences Meeting, Orlando, FL, 2011.

[5] CVXOPT, http://abel.ee.ucla.edu/cvxopt/index.html.

[6] S. M. DESHPANDE, N. ANIL, K. ARORA, K. MALAGI, AND M. VARMA, *Some fascinating new developments in kinetic schemes*, in Proceedings of the Workshop on Modeling and Simulation in Life Sciences, Materials and Technology, A. Avudainayagam, P. Misra, and S. Sundar, eds., 2004, pp. 43–64.

[7] C. A. DUARTE AND J. T. ODEN, *H^p clouds—an h-p meshless method*, Numer. Methods Partial Differential Equations, 12 (1996), pp. 673–705.

[8] A. K. GHOSH AND S. M. DESHPANDE, *Least squares kinetic upwind method for inviscid compressible flows*, in proceedings of the 12th AIAA Computational Fluid Dynamics conferences, San Diego, CA, 1995.

[9] P. E. Gill, W. Murray, and M. A. Saunders, *SNOPT: An SQP algorithm for large-scale constrained optimization*, SIAM J. Optim., 12 (2002), pp. 979–1006.
[10] M. Grant and S. Boyd, *Graph implementations for nonsmooth convex programs,* in Recent Advances in Learning and Control, Springer, London, 2008, pp. 95–110.
[11] M. Grant and S. Boyd, *CVX: Matlab Software for Disciplined Convex Programming (web page and software)*. http://stanford.edu/∼boyd/cvx, 2009.
[12] E. J. Kansa, *Multiquadrics—a scattered data approximation scheme with applications to computational fluid-dynamics*. I. *Surface approximations and partial derivative estimates*, Comput. Math. Appl., 19 (1990), pp. 127–145.
[13] E. J. Kansa, *Multiquadrics—a scattered data approximation scheme with applications to computational fluid-dynamics*. II. *Solutions to parabolic, hyperbolic and elliptic partial differential equations*, Comput. Math. Appl., 19 (1990), pp. 147–161.
[14] A. Katz and A. Jameson, *Edge-based meshless methods for compressible flow simulations*, in proceedings of the 46th AIAA Aerospace Sciences Meeting, Reno, NV, 2008.
[15] R. Löhner, C. Sacco, E. Oñate, and S. Idelssohn, *A finite point method for compressible flow*, Internat. J. Numer. Methods Engrg., 53 (2002), pp. 1765–1779.
[16] J. M. Melenk and I. Babuska, *The partition of unity finite element method: Basic theory and applications*, Comput. Methods Appl. Mech. Engrg., 139 (1996), pp. 289–314.
[17] J. C. Meza, R. A. Oliva, P. D. Hough, and P. J. Williams, *Opt++: An object-oriented toolkit for nonlinear optimization*, ACM Trans. Math. Software, 33 (2007), article 12.
[18] J. J. Monaghan and R. A. Gingold, *Shock simulation by the particle method SPH*, J. Computat. Phys., 52 (1983), pp. 374–389.
[19] B. Nayorles, G. Touzot, and P. Villom, *Generalizing the finite element method: Diffuse approximation and diffuse elements*, Comput. Mech., 10 (1992), pp. 307–318.
[20] OBOE, https://projects.coin-or.org/OBOE.
[21] E. Oñate, O. C. Idelsohn, S. Zienkiewicz, and R. L. Taylor, *A finite point method in computational mechanics. Applications to convective transport and fluid flow*, Internat. J. Numer. Methods Engrg., 39 (1996), pp. 3839–3866.
[22] C. C. Paige and M. A. Saunders, *LSQR: An algorithm for sparse linear equations and sparse least squares*, ACM Trans. Math. Software, 8 (1982), pp. 43–71.
[23] V. Ramesh and S. M. Deshpande, *Euler computations on arbitrary grids using LSKUM*, in Computational Fluid Dynamics 2000: Proceedings of the First International Conference on Computational Fluid Dynamics, N. Satofuka, ed., Springer-Verlag, Berlin, 2000, pp. 783–784.
[24] C. Shu, H. Ding, H. Q. Chen, and T. G. Wang, *An upwind local RBF-DQ method for simulation of inviscid compressible flows*, Comput. Methods Appl. Mech. Engrg., 194 (2005), pp. 2001–2017.
[25] D. Sridar and N. Balakrishnan, *An upwind finite difference scheme for meshless solvers*, J. Comput. Phys., 189 (2003), pp. 1–29.
[26] P. V. TOTA, and Z. J. Wang, *Meshfree Euler solver using local radial basis functions for inviscid compressible flows*, in proceedings of the 18th AIAA Computational Fluid Dynamics Conference, Miami, FL, 2007.
[27] Z. J. Wang, *Improved formulation for geometric properties of arbitrary polyhedra*, AIAA J., 37 (1999), pp. 1326–1327.